SEM-HP: Un modelo Sistémico, Evolutivo y SEMántico para el desarrollo de Sistemas HiPermedia

Lina Guadalupe García Cabrera

TESIS DOCTORAL
JAÉN - 2001

Universidad de Granada

Escuela Técnica Superior de Ingeniería en Informática

Departamento de Lenguajes y Sistemas Informáticos

SEM-HP: Un modelo Sistémico, Evolutivo y SEMántico para el desarrollo de Sistemas HiPermedia

tesis doctoral

por

Lina Guadalupe García Cabrera

Memoria presentada para optar al grado de Doctor en Informática

Codirigida por los profesores:

Dr. José Parets Llorca

Profesor Titular de Universidad

Departamento de Lenguajes y Sistemas Informáticos

Universidad de Granada

Dra. M. José Rodríguez Fórtiz

Profesora Asociada

Departamento de Lenguajes y Sistemas Informáticos

Universidad de Granada

Resumen. Las principales actividades relacionadas con el uso y el desarrollo de los Sistema Hipermedia evolucionan a lo largo del tiempo. Los Sistemas Hipermedia necesitan modelos que respalden la naturaleza evolutiva de su construcción, mantenimiento y navegación. La Teoría del Sistema General, la Evolución del Software y los Modelos Cognitivos ofrecen una mejor perspectiva de los sistemas *web,* y consiguen salir triunfantes en la abstracción de su estructura, información y comportamiento. Un Sistema Hipermedia Evolutivo debe ser modelado mediante un conjunto de sistemas interrelacionados e interactivos que permitan: a) desde el punto de vista del autor, un control flexible y completo del desarrollo y mantenimiento del hiperdocumento; y b) desde la perspectiva del lector, una navegación que facilite la comprensión, el acceso y la selección de la información. En este trabajo, se propone el Modelo **Si**stémico, **E**volutivo y **SEM**ántico para el desarrollo de Sistemas **H**i**P**ermedia, **SEM-HP**, con las siguientes características:

- Admite una representación explícita del contenido semántico que permite:
  - realizar una estructuración del sistema de información, y
  - determinar sus posibilidades de cambio, actualización y evolución.
- Es lo suficientemente flexible como para ofrecer los mecanismos necesarios para incorporar y representar los dominios conceptuales del autor con el fin de lograr la caracterización de los dominios de información.

Palabras Clave: Sistemas de Información, *Software* Evolutivo, Sistemas Hipermedia Adaptativos, Estructuración basada en la Semántica, Sistemas Hipermedia Evolutivos.

Abstract. Main activities related to the use and development of Hypermedia-Systems evolve through time. Hypermedia-systems need models that support the evolutionary nature of their building, maintenance and navigation processes. The General System Theory, Evolutionary Software and Cognitive Models offer a better perspective of web-systems and succeed in abstracting their structure, information and behaviour. We assume that an Evolutionary Hypermedia System must be modelled by means of a set of interrelated and interacting systems that allow: a) from the author's viewpoint, a complete and flexible control of the development and maintenance of hyperdocuments; b) from the reader's perspective, an understandable navigation that allows access to and selection of information in an easy way. In this work, we propose a **S**ystemic, **E**volutionary and **SEM**antic Model to the development of **H**y**P**ermedia Systems, **SEM-HP**, with the following characteristics:

- It admits an explicit representation of the semantic content which allows us:
  - to structure the information-system, and
  - determines its possibilities of change, updating and evolution.
- It is flexible enough in offering the necessary mechanisms to incorporate and represent the author's conceptual-domains in order to characterise the information-domains.

Key Words: Information Systems, Evolutionary Software, Adaptive Hypermedia Systems, Semantic Structure, Evolutionary Hypermedia Systems.

EN MEMORIA

de Pepe, mi padre,

un hombre imprescindible, un verdadero entusiasta de la vida al que se le obligó precipitadamente a marchar. Lo suyo sí que fue integridad, amor propio y gran corazón. En estos momentos, más que nunca, siento su ausencia y su presencia. Va por ti, un beso.

Vivir es desatinar:
tejer para destejer:
aprender para olvidar;
amar para recordar
amor que no puede ser
a la alegría de ayer
que nunca vuelve a pasar.

Antonio Machado

A mi Madre, la mejor madre del mundo mundial y parte del extranjero.

A mis Hermanos, dos buenos hombres y todo unos caballeros.

A Pedro, todo un renacentista con un corazón de oro.

Tábula Gratulatoria

«La influencia que la vida de una persona ejerce sobre la otra es incalculable.»

Ralph Bunche.
Sociólogo y político estadounidense (1906-1971).

«Cualquiera que hayan sido nuestros logros alguien nos ayudó siempre a alcanzarlos.»

Althea Gibson.
La primera mujer afroamericana que ganó Wimbledon y Forest Hills (1927-).

El poder escribir, por fin, estas líneas representa para mí la culminación de un gran esfuerzo que no habría sido posible sin la contribución, material o espiritual, de muchas personas o entidades. A todas ellas debo un enorme agradecimiento. Disculpen si me extiendo pero son tantos y tan buenos los afectos que no puedo dejar de reconocer la labor de:

- D. José Parets Llorca y Dña. María José Rodríguez Fórtiz, mis mentores. A mi jefe por acoger con agrado el proyecto de investigar en educación cuya evolución a deparado en esta Tesis y por su acertada dirección. A mi jefa por no dejar de *dar luz* y por haber aportado unas sólidas bases formales de las que me he hecho eco.
- Mi Grupo de Investigación, cuyas reuniones siempre son fructíferas. Gracias a Ana, Jesús, Nuria, Mavi y Patricia, por sus comentarios, sugerencias y correcciones.
- Mi anterior Grupo de investigación «Procesamiento de Imágenes» en el que se me inició en el mundo de la investigación. De esos años conservo estupendas amistades con los que me une grandes dosis de humor llenas de chispa y una gran afición por celebrarlo todo con buenas viandas y caldos. Muchas gracias, pues, a D. Ramón Román Roldán, a D. José Martínez Aroza, a Vicente, a Kiko y a María José.
- A los compañeros del Departamento por sus muestras de apoyo y cariño, en especial, a José Ramón, Manuel Jaenes y Pedro González por auxiliarme en algún que otro desbarajuste técnico, a Paco Martínez por los quites docentes y a Rafael Segura por los consejos burocráticos.
- A Carmen, mi madre, ella sí que es una especie en extinción. Gracias por ser tan multidisciplinar, un gran consuelo y por ofrecer un respaldo constante. A mis hermanos, Rafael y Julián, por contribuir de manera decisiva a la culminación de este trabajo. Y al resto de la familia sobre todo a mis tías Lina y Pepita por su amor por la educación, la cultura y el arte.
- A Pedro por recoger la toalla en más de una ocasión y no dejar de darme con ella hasta llegar aquí. Fíjate, ya está.
- A mis amigos que desde el primer momento consideraron que el sitio más adecuado para mí era el universitario. Ahora, estoy dudando si esa recomendación fue con buena intención. En cualquier caso, gracias a mis *compas* de carrera y, sin embargo, muy buenos amigos: a Ramón, por su despiste y ese don para saber vivir, a Jose Antonio por ser feliz aunque no tenga un camión, a Antonio por seguir creyendo en las utopías, a Lola por dormir tanto y tan bien y por desdramatizar todas las cosas, a Javier por entretener tan bien a Pedro y a Miguel por ser tan guapo. También gracias a las niñas, por esos años tan buenos llenos de risas y despreocupaciones: a Mª José por saber lo que se sufre con esto, a MariMar por esos ratos de carcajada sin medida, a Rocío por estar siempre tan risueña y darnos cobijo en Londres (ahora su casa), a Nini por ser tan dulce y tan artista y a Graciela por ese derroche de energía. Por último a los amigos de mi amigo que, afortunadamente, también son los míos: a Juan Carlos por sus quejas (a las que me uno) y a Yoli por aguantarlo, a Lorenzo y a Lola por los descansos isleños, por cierto, ¿cuál es el próximo destino?.
- A Rao y Turoff por proporcionar a una perfecta desconocida las separatas de algunos artículos y a Balasubramanian, Bieber, De Bra y Salampasis por hacer públicos en la web sus trabajos y sus tutoriales que han sido mi principal fuente de formación.
- Finalmente, a Mozart por su «Concierto para clarinete y orquesta en La Mayor (K622)», a Bach por su «Aria para soprano, flauta y bajo continuo (*Las ovejas pastan seguras*) de la cantata de la caza (BWV 208)», a Jonathan Swift por «Los viajes de Gulliver» y a Eduardo Mendoza por «La aventura del tocador de señoras». A todos ellos gracias por proporcionarme esos momentos imprescindibles de evasión que llenan la vida.

Tabla de Acrónimos

A	Asociación
ACL	Access Control List
AN	Árbol de Navegación
C	Concepto
CERN	Centre Europeain de Recherche Nucleaire
DHTML	Dynamic HTML
DTD	Document Type Definition
EC	Estructura Conceptual
FRESS	File Retrieval and Editing System
GEDES	Grupo de Especificación, Diseño y Evolución de Software
gIBIS	graphical Issue Based Information System
HAM	Hypertext Abstract Machine
HDM	Hypermedia Design Methodology
HEDES	Herramienta para la Especificación, Diseño y Evolución de Software
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
II	Ítem de Información
KMS	Knowledge Management System
LD	Lógica Decriptiva
Lenguaje MN	Lenguaje de Meta-Navegación
Lenguaje N	Lenguaje de Navegación
MEDES	Método de Especificación, Diseño y Evolución de Software
MEMEX	MEMory eXtender
MS	Meta Sistema
NCSA	National Center or Supercomputing Application
NLS	oN Line System
OMT	Object Modelling Tool
OOHDM	Object Oriented Hypermedia Design Methodology
OWL	Office Workstation Ltd
R	Relación
RMDM	Relationship Management Data Model
RMM	Relationship Management Methodology
RP	Red de Petri
RT	ResTricciones

SA	Sistema de Aprendizaje
SC	Sistema de Conocimiento
SDS	Sistema de Desarrollo Software
SEM-HP	modelo Sistémico, Evolutivo y SEMántico para el Desarrollo de sistemas HiPermedia Systemic, Evolutionary, SEMantic Model for HyPermedia Systems
SM	Subsistema de Memorización
SN	Sistema de Navegación
SP	Subsistema de Presentación
SS	Sistema Software
UML	Unified Modeling Language
URL	Universal Resource Locator
VRML	Virtual Reality Markup Language
WWW	World Wide Web
Xlink	eXtensible Link
XML	eXtensible Markup Language
XPointer	eXtensible Pointer
XSL	eXtensible Style Language

Tabla de Símbolos

s	Concepto-sistema
r	Concepto-raíz
g	Relación genérica
i	Relación inicio
$\Gamma^+, \Gamma^-, \Gamma^{*+}, \Gamma^{*-}$	Funciones de correspondencia o alcanzabilidad
$\Gamma^{+n}(c_o, r_c)$	Conceptos-accesibles-desde
$\Lambda^n(c_o, o_d)$	Conceptos-Relacionados-mediante
$\Gamma^{-n}(r_c, c_d)$	Conceptos-alcanzan-a
Ac	Conjunto de asociaciones conceptuales A_c: C x R_c → C
AC_e	Acciones Evolutivas
AC_{em}	Acciones de Evolución de Memorización
AC_{md}	Acciones de Evolución de los Diccionarios
AC_{mec}	Acciones de Evolución relacionadas con la Estructura Conceptual del SM
AC_{mrt}	Acciones de Evolución relacionadas con las Restricciones del SM
A_f	Conjunto de asociaciones funcionales A_f: C x R_f → II
C	Conjunto de conceptos
c_d	Concepto destino
c_o	Concepto origen
D	Diccionario
DC	Diccionario de Conceptos
DR_c	Diccionario de Relaciones Conceptuales
DR_f	Diccionario de Relaciones Funcionales
EC_m	Estructura Conceptual de Memorización
II	Conjunto de ítemes de información
o	Objeto, un nodo concepto o un nodo ítem de información
R_c	Conjunto de relaciones conceptuales
R_f	Conjunto de relaciones funcionales
RT_a	Restricciones de Autor
RT_m	Conjunto de Restricciones de Memorización
RT_{ma}	Conjunto de Restricciones asociativas del Subsistema de Memorización
RT_{mc}	Conjunto de Restricciones sobre conceptos del Subsistema de Memorización
RT_{md}	Conjunto de Restricciones sobre los diccionarios del SM

$\mathbf{RT_{mii}}$	Conjunto de Restricciones sobre los ítemes de Información del Subsistema de Memorización
$\mathbf{RT_{mr}}$	Conjunto de Restricciones relacionales del Subsistema de Memorización
$\mathbf{RT_s}$	Restricciones de Sistema
SM	Subsistema de Memorización (EC_m, D, RT_m, AC_{em})
SP	Subsistema de Presentación (EC_p, EC^v_i, RT_p, AC_{ep})
SN	Sistema de Navegación (EC_n, RT_n, RP, AN, AC_{en})
$\mathbf{EC_p}$	Estructura Conceptual de Presentación
$\mathbf{EC^v_i}$	Estructuras Conceptuales Virtuales del SP
$\mathbf{RT_p}$	Conjunto de Restricciones de Presentación
$\mathbf{AC_{ep}}$	Acciones de Evolución de Presentación
$\mathbf{EC_n}$	Estructura Conceptual de Navegación
$\mathbf{RT_n}$	Conjunto de Restricciones de Navegación
$\mathbf{RT_{nb}}$	Conjunto de Restricciones de Navegabilidad de SN
$\mathbf{RT_o}$	Conjunto de Restricciones de Orden de SN
RP	Red de Petri

Lista de Figuras

Figura 0 Relaciones conceptuales entre los capítulosw de la memoria. XXV

Figura 1.1 Los Sistemas Hipermedia son multidisciplinares [Balasubramanian, 1993]. 4

Figura 1.2 Diferencias entre un texto y un hipertexto. 6

Figura 1.3 En un hipermedia se combinan todos los modos de expresión posibles. 6

Figura 1.4 En un hiperdocumento se puede usar tres elementos básicos: nodos, enlaces y anclajes. 7

Figura 1.5 Vannevar Bush. 10

Figura 1.6 Sistema Xanadu. 11

Figura 1.7 Andries van Dam. 12

Figura 1.8 Doug Engelbart. 12

Figura 1.9 Ben Shneiderman. 15

Figura 1.10 Tim Beners-Lee. 17

Figura 1.11 Michael Bieber. 17

Figura 1.12 Modelo básico hipermedia [Salampasis, 1997]. 18

Figura 1.13 Entorno conceptual de los Sistemas hipermedia [Salampasis, 1997]. 20

Figura 1.14 Arquitectura básica de un Sistema Hipermedia. 19

Figura 1.15 Vista detallada y resumida de un hipermedia representado mediante un grafo. 25

Figura 1.16 Paul De Bra. 31

Figura 1.17 Peter Brusilovsky. 31

Figura 2.1 Las distintas Generaciones de Sistemas Hipermedia [Salampasis, 1997]. 37

Figura 2.2 HAM: Hypertext Abstract Machine [Campbel, 1988]. 40

Figura 2.3 El Metamodelo Trellis [Furuta, 1990]. 42

Figura 2.4 El modelo Dexter [Halasz, 1990]. 43

Figura 2.5 El constructor Tower [De Bra, 1992]. 44

Figura 2.6 Un Modelo Sistémico, Semántico y Evolutivo para la construcción, mantenimiento y navegación de Sistemas Hipermedia. 52

Figura 3.1 El Proceso de Desarrollo Software [Rodríguez-Fórtiz, 2000] 60

Figura 3.2 Una arquitectura con Dos Niveles de Abstracción. 63

Figura 4.1 El Proceso de Desarrollo de Sistemas Hipermedia. 71

Figura 4.2 Arquitectura General del Modelo SEM-HP. 74

Figura 4.3 Estructura Conceptual —EC— sobre el Sistema de Solar. 76

Figura 4.4 Ítem de Información: Propiedades y Contenido. 78

Figura 4.5 Ejemplos Abstractos de Estructuras Conceptuales del Sistema de Conocimiento. 80

Figura 4.6 Composición de Ítemes de Información. 82

Figura 4.7 Una Subestructura Conceptual —ECp— seleccionada a partir de EC. 92

Figura 4.8 a) La Red de Petri que genera EC_p de la figura 4.7, b) La Red de Petri que genera EC_p a la que se ha añadido una restricción más de navegación. 94

Figura 4.9 Arquitectura detallada del Modelo SEM-HP. 98

Figura 4.10 El modelo SEM-HP: Una arquitectura con dos niveles de abstracción. 99

Figura 4.11 ESTRUCTURA y META-ESTRUCTURA de los Sistemas. 99

Figura 4.12 Evolución de SEM-HP: Propagación de los Cambios. 102

Figura 5.1 Un ejemplo genérico. 117

Figura 5.2 Ejemplo de la acción evolutiva asociativa: Borra-asociación-conceptual. 136

Figura 5.3 Ejemplo de la acción evolutiva asociativa: Marca-concepto-raíz. 138

Figura 5.4 Propagación Interna en el Subsistema de Memorización. 148

Figura 5.5 Ejemplo de composición de ítemes de información. 150

Figura 5.6 Un ejemplo de grafo al que queremos calcular su grafo condensado. 158

Figura 5.7 Grafo condensado del grafo de la figura 5.6. 159

Figura 5.8 Ejemplo de grafo al que se le calculan los centros para ayudar al autor en la creación de estructuras virtuales. 161

Figura 5.9 Ejemplo de la acción evolutiva: Oculta-AC. 164

Figura 5.10 Propagación Interna del Cambio en el Subsistema de Presentación. 168

Figura 5.11 Un ejemplo de Estructura Conceptual sobre la que quieren imponer Restricciones de Navegabilidad. 173

Figura 5.12 Recorridos posibles sobre la Estructura Conceptual antes y después de restringir la navegación como se indica en la tabla 5.60. 176

Figura 5.13 Operador **previo** aplicado a relación conceptual: ***visitar***(c2.f1)ß **previo**(***seguir***(r1)). 184

Figura 5.14 Operador **previo** aplicado a un ítem: ***visitar***(c2.f1) ß **previo**(***visitar***(c1.f7)). 184

Figura 5.15 Operador **antes** aplicado a un ítem: ***visitar***(c2.f1) ß **antes**(***visitar***(c5.f3)). 185

Figura 5.16 Operador **desde** aplicado a ítemes: ***visitar***(c2.f1) ß (***visitar***(c5.f3) **desde** ***visitar***(c4.f1)). 185

Figura 5.17 Cambiando las Restricciones de Navegabilidad en la Ec_n. 196

Figura 5.18 Propagación Interna del Cambio en el Sistema de Navegación. 197

Figura 5.19 Red de Petri para el operador lógico **and**. 202

Figura 5.20 Red de Petri para el operador lógico **or**. 203

Figura 5.21 Red de Petri para el operador temporal **desde**. 203

Figura 5.22 Red de Petri para los operadores temporales **previo/antes**. 204

Figura 5.23 Red de Petri para el operador lógico **not**. 205

Figura 5.24 Red de Petri para la primera cláusula simple del ejemplo. 206

Figura 5.25 Red de Petri para la segunda cláusula simple del ejemplo. 206

Figura 5.26 Red de Petri para la tercera cláusula simple del ejemplo. 206

Figura 5.27 Red de Petri compuesta por las redes de las figuras 5.24, 5.25 y 5.26. 207

Figura 5.28 Red de Petri compuesta por redes que tienen lugares en común. 208

Figura 5.29 Ejemplo de red de Petri con un marcado inicial. 209

Figura 5.30 Árbol de Alcanzabilidad de la red de Petri de la figura 5.29 después de tres disparos. 209

Figura 5.31 Árbol de Alcanzabilidad de la red de Petri de la figura 5.29. 211

Figura 5.32 Red de Petri con marcado inicial (1, 1, 0, 0, 0, 0, 0). 212

Figura 5.33 Árbol de Alcanzabilidad de la red de Petri de la figura 5.32. 213

Figura 5.34 Posibles navegaciones para el lector de acuerdo al árbol de Navegación de la figura 5.33. 213

Figura 5.35 Propagación externa del cambio entre los Sistemas de SEM-HP. 219

Figura 5.36 Flexibilidad y versatilidad de SEM-HP. 222

Figura 6.1 Diagrama de Clases del Sistema Hipermedia. 227

Figura 6.2 Diagrama de Clases del Modelo SEM-HP. 227

Figura 6.3 Diagrama de Clases del Subsistema de Memorización. 228

Figura 6.4 Diagrama de Clases del Subsistema de Presentación. 229

Figura 6.5 Diagrama de Clases del Sistema de Navegación. 230

Figura 6.6 Clase: Estructura Conceptual de Memorización. 231

Figura 6.7 Clase: Grafo. 231

Figura 6.8 Clase: Nodo Item. 231

Figura 6.9 Clases relacionadas con los diccionarios. 232

Figura 6.10 Clases relacionadas con los Conceptos y las Relaciones. 232

Figura 6.11 Clase: Estructura Conceptual de Presentación. 233

Figura 6.12 Clase: Estructura Virtual. 233

Figura 6.13 Clase: Estructura Conceptual de Navegación. 233

Figura 6.14 Clase: Red de Petri. 234

Figura 6.15 Clase: Diagrama de Secuencia del Sistema Hipermedia. 235

Figura 6.16 Diagrama de Secuencia del Subsistema de Memorización: Cambiando la Estructura de Memorización. 236

Figura 6.17 Diagrama de Secuencia del Subsistema de Memorización: Cambiando las Restricciones de Memorización. 237

Figura 6.18 Diagrama de Secuencia del Subsistema de Presentación: Cambiando la Estructura Conceptual de Presentación. 238

Figura 6.19 Diagrama de Secuencia del Subsistema de Presentación: Cambiando la Estructura Conceptual Virtual. 239

Figura 6.20 Diagrama de Secuencia del Subsistema de Navegación: Cambiando la Estructura Conceptual de Navegación. 240

Figura 6.21 Diagrama de Secuencia del Subsistema de Navegación: Cambiando las Restricciones de Navegación. 241

Figura 6.22 La herramienta SEM-HP: ejecución de todas sus aplicaciones. 243

Figura 6.23 El Sistema de Memorización: Creando la Estructura Conceptual. 244

Figura 6.24 El Sistema de Memorización: Editanto el contenido y las propiedades de los Ítemes de Información. 245

Figura 6.25 El Sistema de Memorización: Asociando Restricciones de Autor y creando los Diccionarios. 246

Figura 6.26 El Sistema de Presentación: Seleccionando la Estructura Conceptual de presentación. 249

Figura 6.27 El Sistema de Presentación: Creando Estructuras Conceptuales Virtuales. 250

Figura 6.28 El Sistema de Navegación: Ampliando y reducciendo la navegabilidad de las relaciones conceptuales. 251

Figura 6.29 El Sistema de Navegación: Generando las Restricciones de Navegación. 252

Figura 6.30 El Sistema de Navegación: Cambiando las Restricciones de Navegación y generando su red de Petri. 253

Figura 6.31 El Sistema de Navegación: Generando el Árbol de Navegación. 254

Figura 6.32 El Navegador: Tres posibles modos de interacción. 255

Figura 7.1 Arquitectura del modelo AHAM [Wu, 2001]. 267

Lista de tablas

Tabla 1.1 Acontencimientos más significativos de la Historia de los Sistemas Hipermedia. 10

Tabla 2.1 Requisitos para los Sistemas Hipermedia de la Próxima Generación. 39

Tabla 2.2 ¿Cómo descubrir la Semántica de Estructuración? 48

Tabla 2.3 Razones para la Adopción de Modelos Cognitivos. 51

Tabla 4.1 Roles que los Ítemes de Información pueden desempeñar en un contexto. 77

Tabla 4.2 Clasificación de las Restricciones. 87

Tabla 5.1 Funciones de correspondencia: Conceptos_accesibles_desde. 118

Tabla 5.2 Funciones de correspondencia: Conceptos_alcanzan_a. 118

Tabla 5.3 Funciones de correspondencia: Relacionados_mediante. 119

Tabla 5.4 Restricciones de las Relacionas Conceptuales. 123

Tabla 5.5 Operación Básica: Crea-nodo. 130

Tabla 5.6 Operación Básica: Borra-nodo. 130

Tabla 5.7 Operación Básica: Crea-arco. 130

Tabla 5.8 Operación Básica: Borra-arco. 131

Tabla 5.9 Operación Básica: Pon-etiqueta-nodo. 131

Tabla 5.10 Operación Básica: Pon-etiqueta-arco. 132

Tabla 5.11 Operación Básica: Quita-etiqueta-nodo. 132

Tabla 5.12 Operación Básica: Quita-etiqueta-arco. 132

Tabla 5.13 Operación Básica: Marca-nodo-raíz. 132

Tabla 5.14 Operación Básica: Quita-nodo-raíz. 132

Tabla 5.15 Operación Básica: Nodos-accesibles-desde. 132

Tabla 5.16 Operación Básica: Nodos-alcanzan-a. 133

Tabla 5.17 Operación Básica: Nodos-conectados-a. 133

Tabla 5.18 Operación Básica: Relacionados-mediante. 133

Tabla 5.19 Acciones Evolutivas Asociativas: Crea-concepto. 134

Tabla 5.20 Acciones Evolutivas Asociativas: Crea-asociación-conceptual. 134

Tabla 5.21 Acciones Evolutivas Asociativas: Crea-asociación-funcional. 134

Tabla 5.22 Acciones Evolutivas Asociativas: Borra-asociación-conceptual. 135

Tabla 5.23 Acciones Evolutivas Asociativas: Borra-asociación-funcional. 136

Tabla 5.24 Acciones Evolutivas Asociativas: Crea-A_c-y-concepto. 136

Tabla 5.25 Acciones Evolutivas Asociativas: Modifica-asociación-conceptual. 137

Tabla 5.26 Acciones Evolutivas Asociativas: Modifica-asociación-funcional. 138

Tabla 5.27 Acciones Evolutivas Asociativas: Marca-concepto-raíz. 138

Tabla 5.28 Acciones Evolutivas Asociativas: Cambia-etiqueta. 139

Tabla 5.29 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-ítem. 140

Tabla 5.30 Acciones Evolutivas para cambiar los Ítemes de Información: Edita-contenido-ítem. 140

Tabla 5.31 Acciones Evolutivas para cambiar los Ítemes de Información: Edita-propiedades-ítem. 141

Tabla 5.32 Acciones Evolutivas para cambiar los Ítemes de Información: Borra-ítem. 141

Tabla 5.33 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-variante. 141

Tabla 5.34 Acciones Evolutivas para cambiar los Ítemes de Información: Convierte-en-variante. 142

Tabla 5.35 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-ítem-copia. 142

Tabla 5.36 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-ítem-copia-contenido. 142

Tabla 5.37 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-ítem-copia-propiedades. 143

Tabla 5.38 Acciones Evolutivas para cambiar las Restricciones: Añade-RT_a. 143

Tabla 5.39 Acciones Evolutivas para cambiar las Restricciones: Borra-RT_a. 144

Tabla 5.40 Acciones Evolutivas para cambiar el Diccionario: Crea-etiqueta-concepto. 145

Tabla 5.41 Acciones Evolutivas para cambiar el Diccionario: Borra-etiqueta-concepto. 145

Tabla 5.42 Acciones Evolutivas para cambiar el Diccionario: Crea-etiqueta-Rconceptual. 146

Tabla 5.43 Acciones Evolutivas para cambiar el Diccionario: Borra-etiqueta-Rconceptual. 146

Tabla 5.44 Acciones Evolutivas para cambiar el Diccionario: Crea-etiqueta-Rfuncional. 146

Tabla 5.45 Acciones Evolutivas para cambiar el Diccionario: Borra-etiqueta-Rfuncional. 146

Tabla 5.46 Algoritmo 1 que redefine las Restricciones Relacionales Conceptuales. 148

Tabla 5.47 Acciones Evolutivas Asociativas: Crea-ítem-compuesto. 150

Tabla 5.48 Acciones Evolutivas para cambiar las Restricciones: Componer-R_c. 151

Tabla 5.49 Acciones Evolutivas para cambiar las Restricciones: No-Componer-R_c 151

Tabla 5.50 Algoritmo 2 que redefine las Restricciones Relacionales. 151

Tabla 5.51 Cálculo de las componentes fuertes mediante R Ä Q. 159

Tabla 5.52 Matriz de distancias D(G) para el grafo de la figura 5.8. 161

Tabla 5.53 Acciones Evolutivas para cambiar la Estructura Conceptual de Presentación: Oculta-AC. 163

Tabla 5.54 Acciones Evolutivas para cambiar la Estructura Conceptual de Presentación: Oculta-AF. 164

Tabla 5.55 Acciones Evolutivas para cambiar la Estructura Conceptual de Presentación: Muestra-AC. 165

Tabla 5.56 Acciones Evolutivas para cambiar la Estructura Conceptual de Presentación: Muestra-AF. 165

Tabla 5.57 Algoritmo 1: Redefinición de la Subestructura Conceptual de $EC_{p.}$ 167

Tabla 5.58 Restricciones de Orden que automáticamente genera el Sistema de Navegación a partir del grafo de la figura 5.11. 174

Tabla 5.59 Número de Restricciones de Orden que se generan a partir de la Estructura Conceptual de la figura 5.11. 175

Tabla 5.60 Imponiendo más Restricciones de Orden: Borrar reglas temporales. 176

Tabla 5.61 Imponiendo más Restricciones de Orden: Cambiar **or** por **and**. 177

Tabla 5.62 Imponiendo más Restricciones de Orden: Añadir operadores **antes** y **previo**. 177

Tabla 5.63 Sintaxis del Lenguaje **N**. 181

Tabla 5.64 Fórmula Temporal del Sistema: Cláusulas permitidas. 187

Tabla 5.65 Fórmula Temporal del Autor: Cláusulas permitidas y no permitidas. 188

Tabla 5.66 Los cambios realizados en la figura 5.17 suponen la activación —en color negro— y la desactivación —en color gris— de restricciones de orden. 196

Tabla 5.67 Modelado de la Navegación mediante una **RP**. 200

Tabla 5.68 Algoritmo 1 que redefine las Restricciones Relacionales Conceptuales. 218

Tabla 5.69 Formalismos empleados en la especificación de la ESTRUCTURA de los Sistemas. 220

Tabla 5.70 Formalismos empleados en la especificación de la Evolución de los Sistemas. 221

Tabla 6.1 Equivalencias entre las familias de lenguajes HTML, XML y SGML. 256

Tabla 7.1 Tipos de nodos, enlaces y reglas generales del Sistema RICH. 264

Tabla B.1 Inclusiones e Incompatibilidades entre las Restricciones sobre las Relaciones Conceptuales. 284

Tabla B.2 Reglas de inclusión o incompatibilidad entre las Restricciones sobre Relaciones Conceptuales. 286

Tabla B.3 Demol1: Demostración de la regla de Inclusión/Incompatibilidad 1. 286

Tabla B.4 Demol2: Demostración de la regla de Inclusión/Incompatibilidad 2. 287

Tabla B.5 DemoI3: Demostración de la regla de Inclusión/Incompatibilidad 3. 287

Tabla B.6 DemoI4: Demostración de la regla de Inclusión/Incompatibilidad 4. 288

Tabla B.7 DemoI5: Demostración de la regla de Inclusión/Incompatibilidad 5. 288

Tabla B.8 DemoI6: Demostración de la regla de Inclusión/Incompatibilidad 6. 289

Tabla B.9 DemoI7: Demostración de la regla de Inclusión/Incompatibilidad 7. 289

Tabla B.10 DemoI8: Demostración de la regla de Inclusión/Incompatibilidad 8. 290

Tabla B.11 DemoI9: Demostración de la regla de Inclusión/Incompatibilidad 9. 290

Tabla B.12 DemoI10: Demostración de la regla de Inclusión/Incompatibilidad 10. 291

Tabla B.13 Representantes de las Restricciones sobre las Relaciones Conceptuales. 292

Tabla B.14 DemoAs: Demostración del Representante As. 293

Tabla B.15 Consistencias entre las Restricciones sobre las Relaciones Conceptuales. 294

Tabla B.16 Axiomas de consistencia entre Restricciones sobre las Relaciones Conceptuales. 294

Tabla B.17 DemoC1: Demostración de la regla de consistencia 1. 295

Tabla B.18 DemoC2: Demostración de la regla de consistencia 2-6. 296

Tabla B.19 DemoC3: Demostración de la regla de consistencia 7-10. 297

Tabla B.20 DemoC4: Demostración de la regla de consistencia 11-12. 297

Tabla B.21 DemoC5: Demostración de la regla de consistencia 13. 298

Tabla B.22 DemoC6: Demostración de la regla de consistencia 14. 298

Tabla B.23 DemoC7: Demostración de la regla de consistencia 15. 298

Tabla B.24 DemoC8: Demostración de la regla de consistencia 16. 298

Tabla C.1 Salida del programa 1: Combinaciones 1-35. 306

Tabla C.2 Salida del programa 1: Combinaciones 36-70. 307

Tabla C.3 Salida del programa 1: Combinaciones 71-105. 308

Tabla C.4 Salida del programa 1: Combinaciones 106-140. 309

Tabla C.5 Salida del programa 1: Combinaciones 141-175. 310

Tabla C.6 Salida del programa 1: Combinaciones 176-210. 311

Tabla C.7 Salida del programa 1: Combinaciones 211-245. 312

Tabla C.8 Salida del programa 1: Combinaciones 246-280. 313

Tabla C.9 Salida del programa 1: Combinaciones 281-290. 314

Tabla C.10 Salida del programa 2: Combinaciones 1-40. 219

Tabla C.11 Salida del programa 2: Combinaciones 41-80. 320

Tabla C.12 Salida del programa 2: Combinaciones 81-120. 321

Tabla C.13 Salida del programa 2: Combinaciones 121-160. 322

Tabla C.14 Salida del programa 2: Combinaciones 161-200. 323

Tabla C.15 Salida del programa 2: Combinaciones 201-240. 324

Tabla C.16 Salida del programa 2: Combinaciones 241-280. 325

Tabla C.17 Salida del programa 2: Combinaciones 281-290. 326

Tabla C.18 Salida del programa 3: Consulta posible2(A, B). 328

Tabla C.19 Salida del programa 3: Consulta posible3(A, B, C). 329

Tabla C.20 Salida del programa 3: Consulta posible4(A, B, C, D). 330

Tabla C.21 Salida del programa 3: Consulta posible5(A, B, C, D, E) y posible6(A, B, C, D, E, F). 331

Tabla C.22 Comparación: Combinaciones con 1 Restricción. 332

Tabla C.23 Comparación: Combinaciones con 2 Restricciones (1-20). 332

Tabla C.24 Comparación: Combinaciones con 2 Restricciones (21-64). 333

Tabla C.25 Comparación: Combinaciones con 3 Restricciones (1-40). 334

Tabla C.26 Comparación: Combinaciones con 3 Restricciones (41-80). 335

Tabla C.27 Comparación: Combinaciones con 3 Restricciones (81-113). 336

Tabla C.28 Comparación: Combinaciones con 4 Restricciones (1-40). 337

Tabla C.29 Comparación: Combinaciones con 4 Restricciones (41-74). 338

Tabla C.30 Comparación: Combinaciones con 5 y 6 Restricciones. 339

Tabla D.1 Predicados del cuerpo de las cláusulas. 344

Tabla D.2 Restricción Ac1 expresada en el lenguaje de las Asociaciones. 345

Tabla D.3 Restricción Ac2 expresada en el lenguaje de las Asociaciones. 345

Tabla D.4 Restricción Ac3 expresada en el lenguaje de las Asociaciones. 345

Tabla D.5 Restricción At expresada en el lenguaje de las Asociaciones. 346

Tabla D.6 Restricción S1 expresada en el lenguaje de las Asociaciones. 346

Tabla D.7 Restricción S2 expresada en el lenguaje de las Asociaciones. 347

Tabla D.8 Restricción S3 expresada en el lenguaje de las Asociaciones. 347

Tabla D.9 Restricción Ll1 expresada en el lenguaje de las Asociaciones. 347

Tabla D.10 Restricción Ll2 expresada en el lenguaje de las Asociaciones. 348

Tabla D.11 Restricción Ll3 expresada en el lenguaje de las Asociaciones. 348

Tabla D.12 Restricción Tr1 expresada en el lenguaje de las Asociaciones. 348

Tabla D.13 Restricción Tr2 expresada en el lenguaje de las Asociaciones. 349

Tabla D.14 Restricción Tr3 expresada en el lenguaje de las Asociaciones. 349

Tabla D.15 Restricción As1 expresada en el lenguaje de las Asociaciones. 350

Tabla D.16 Restricción As2 expresada en el lenguaje de las Asociaciones. 350

Tabla D.17 Restricción As3 expresada en el lenguaje de las Asociaciones. 350

Preámbulo

«Hace tres años estábamos frente al abismo. Ahora hemos dado un paso al frente.»

Joan Pich i Pon.
Político catalán que alcanzó popularidad por sus pifias verbales (1878-1937).

«... a finales de la segunda guerra mundial, cuando la alianza de la ciencia y el horror había alcanzado cotas antes nunca vistas (de la experimentación médica nazi a la bomba atómica), un científico se plantea su misión en un mundo postbélico, y entre tantos sueños posibles escoge uno: una máquina que ponga cualquier publicación del mundo encima del escritorio.

¿Cuál era la finalidad de Bush?: Sencillamente: la creación intelectual»

Presentación del artículo «Como prodríamos pensar» de Bush en la Revista de Occidente por José Antonio Millán

Preámbulo

A pesar de la inutilidad que exageradamente se le atribuye a los cursos de doctorado, la presente memoria ratifica lo contrario. Gracias a ellos —bueno, al menos, a uno de ellos— descubrí de la diestra mano de D. José Parets Llorca al apodado 'Software Evolutivo'.

Este trabajo comenzó con el ánimo de aplicar *la evolución* al desarrollo e implantación de herramientas informáticas formativas. Sin embargo, este primer proyecto fue sufriendo más de un cambio de orientación en el transcurso de la indagación.

El análisis de las posibilidades de la tecnología informática en la educación [García-Cabrera, 1995, 1996] nos hizo identificar a los *Sistemas Hipermedia* como las aplicaciones más adecuadas por sus características de flexibilidad, adaptación y multifuncionalidad.

Parafraseando a Bieber [Bieber, 1995] se podría decir que *hipermedia* es la ciencia de las relaciones, que se concentra en estructurar, presentar y dar a los usuarios un acceso directo a los contenidos y a las interconexiones dentro de un dominio de información. En definitiva, la tecnología hipermedia asume como capacidades intrínsecas la composición, la lectura, la anotación, el enlace y la manipulación de productos de conocimiento. Además, la hipermedia ofrece la posibilidad de organizar un material de conocimiento multimediático procedente de diversas fuentes de muy distintos modos. Por otro lado, se ha implantado con éxito en los entornos de red, permitiendo no solo un acceso remoto sino también una colaboración entre diversos grupos de trabajo en la elaboración de estas aplicaciones.

Esto quiere decir, primero, que se cuenta con una tecnología de presentación y estructuración de dominios de información potente y flexible. Segundo que la hipermedia hace posible la construcción de herramientas abiertas con posibilidades de crecimiento y adaptación. Tercero, que admite combinar e integrar diversos documentos de cualquier tipo. Y cuarto que esta nueva tecnología acepta accesos simultáneos y desarrollos colaborativos.

Por estas razones, lo que inicialmente iba a ser una simbiosis entre evolución y educación, se convirtió en una reflexión sobre cómo conseguir crear hiperdocumentos que facilitasen y mejorasen el proceso enseñanza-aprendizaje. No obstante, el examen de las recientes investigaciones sobre la creación de documentos hipermedia, reveló que:

- las metodologías hipermedia no aseguran ni tan siquiera la construcción de buenos hiperdocumentos y
- al igual que el resto del software, los hipermedia no contaban con buenas herramientas que asistieran al autor en su proceso de construcción y, mucho menos, en su mantenimiento tan necesario en un sistema hipermedia.

De nuevo, la propia investigación encauzó nuestros pasos, había que saber por qué no era posible el diseño de buenos productos hipermedia. De modo, que se estuvo reflexionando sobre el proceso de desarrollo de Sistemas Hipermedia llegando a la conclusión de que éste era un proceso inherentemente evolutivo. En consecuencia, finalmente el propósito de esta tesis mutó para parecerse a la primera intención: aplicar *la evolución* al desarrollo e implantación de Sistemas Hipermedia.

Evolución e Hipermedia, pues, son los dos ámbitos en los que se enmarca este trabajo de cuya fuerte imbricación ya sospechó Vannevar Bush, su precursor, en el año 1945 [Bush, 1945]:

> «Para que un archivo resulte útil a la ciencia, ha de estar en continua ampliación, almacenado en algún lugar y, lo que es más importante, ha de poder ser consultado.»
>
> «Nuestra ineptitud a la hora de acceder al archivo está provocada por la artificialidad de los sistemas de indización . [...] La mente humana no funciona de esa manera.. La mente opera por medio de la asociación.»
>
> «Un memex es un aparato en el que una persona almacena todos sus libros, archivos y comunicaciones y que está memorizado de modo que puede consultarse con una gran velocidad y flexibilidad.»
>
> «el proceso de enlazar dos elementos distintos entre sí es lo que le otorga su verdadera importancia... pues cada uno de los elementos puede pertenecer, a su vez, a más de un sendero de información.»

Hemos hecho de estas palabras nuestras hipótesis de trabajo realizando en la presente Tesis Doctoral una revisión de los modelos empleados en el desarrollo de Sistemas Hipermedia para luego plantear una nueva perspectiva basada en la Teoría de Sistemas, la Evolución y la Semántica. Apoyándonos en estos pilares teóricos se ha ideado SEM-HP —un modelo **S**istémico, **E**volutivo y **SEM**ántico para el diseño de Sistemas **HiP**ermedia— con el que se pretende dotar al desarrollador con mecanismos para la especificación de Sistemas Hipermedia que respaldan todo su proceso de desarrollo evolutivo —construcción, navegación y mantenimiento—.

En un sentido más poético, pretendemos con este trabajo "*...bucear en los vastos océanos de la información y enhebrar un documento con otro, dejando una estela de significado entre las olas de ruido, contradicción y redundancia.*" [Candeira, 2001, página 90].

Objetivos

En definitiva, el presente trabajo pretende alcanzar, al menos, los siguientes objetivos:

- Revisar los Modelos de Referencia en los que se apoyan los Sistemas Hipermedia tradicionales. Este estudio pondrá de manifiesto hasta qué punto son capaces de dar soporte al proceso de desarrollo de un hiperdocumento y de hacerlo evolucionar.
- Hacer uso de la anterior revisión para tratar de:
  - definir claramente las etapas o fases que forman parte del proceso de desarrollo de Sistemas Hipermedia y
  - determinar las características que debe tener un modelo destinado a la construcción de hipermedias.
- Proponer una alternativa de solución, esto es, un modelo con el que se puedan construir sistemas de autor hipermedia que:

  - asistan al autor durante todo el proceso de desarrollo y de manera evolutiva, y
  - que faciliten el acceso y la navegación del lector.
- Identificar las bases teóricas en las que se apoyará este modelo.
- Proporcionar una formalización rigurosa que haga posible y viable su implementación.

Estructura de la Exposición

Esta memoria consta de este preámbulo, de siete capítulos y cuatro apéndices organizados en tres partes y de las referencias bibliográficas.

La primera parte, a la que se ha denominado "Contexto Hipermedia", se dedica a presentar los principales conceptos relacionados con los Sistemas Hipermedia y a realizar una revisión de este marco teórico. Está integrada por dos capítulos: "Marco Teórico" y "Análisis del Marco Teórico". En este segundo capítulo se hace un profundo estudio y crítica de los principales modelos hipermedia y se perfilan las características básicas del modelo que se propone en esta tesis.

En la segunda parte se hace una extensa y pormenorizada exposición conceptual y formal de la propuesta de solución: "El modelo SEM-HP". Incluye cuatro capítulos con los siguientes títulos: "Bases Teóricas de SEM-HP", "Descripción Conceptual de SEM-HP", "Formalización de SEM-HP" y "Consideraciones sobre Diseño e Implementación". Aunque la arquitectura del SEM-HP está integrada por tres Sistemas: el Sistema de Conocimiento —formado por los Subsistemas de Memorización y Presentación—, el Sistema de Navegación y el de Aprendizaje, esta memoria recoge la especificación y descripción formal de los dos primeros dejando el desarrollo del Sistema de Aprendizaje para el futuro debido a que:

- los Sistemas de Conocimiento y Navegación son los que están directamente involucrados en el proceso de desarrollo de Sistemas Hipermedia.
- el Sistema de Aprendizaje no es imprescindible aunque puede mejorar la funcionalidad del Hipermedia adaptándolo para que tenga en cuenta la interacción de lector.
- se considera fundamental desarrollar previamente una herramienta que aplique el modelo SEM-HP antes de ampliar o mejorar su funcionalidad.

Finalmente, la tercera parte engloba un capítulo y cuatro apéndices. Este último capítulo llamado "Recapitulación" se dedica a los trabajos relacionados, conclusiones y trabajos futuros. Con el fin de simplificar y facilitar la exposición se ha incluido la información adicional en los anexos. En el apéndice A, "Glosario de Términos de SEM-HP", se recoge la terminología empleada en el modelo SEM-HP propuesto. En los dos siguientes, en el B y en el C, se detalla la demostración de las Restricciones relacionadas con las Relaciones Conceptuales y los programas y resultados utilizados en su demostración. Por último, el apéndice D esboza una primera aproximación de uno de los problemas abiertos planteados en los trabajos futuros.

En el grafo de la figura 0 se representa las relaciones conceptuales entre los capítulos de esta memoria.

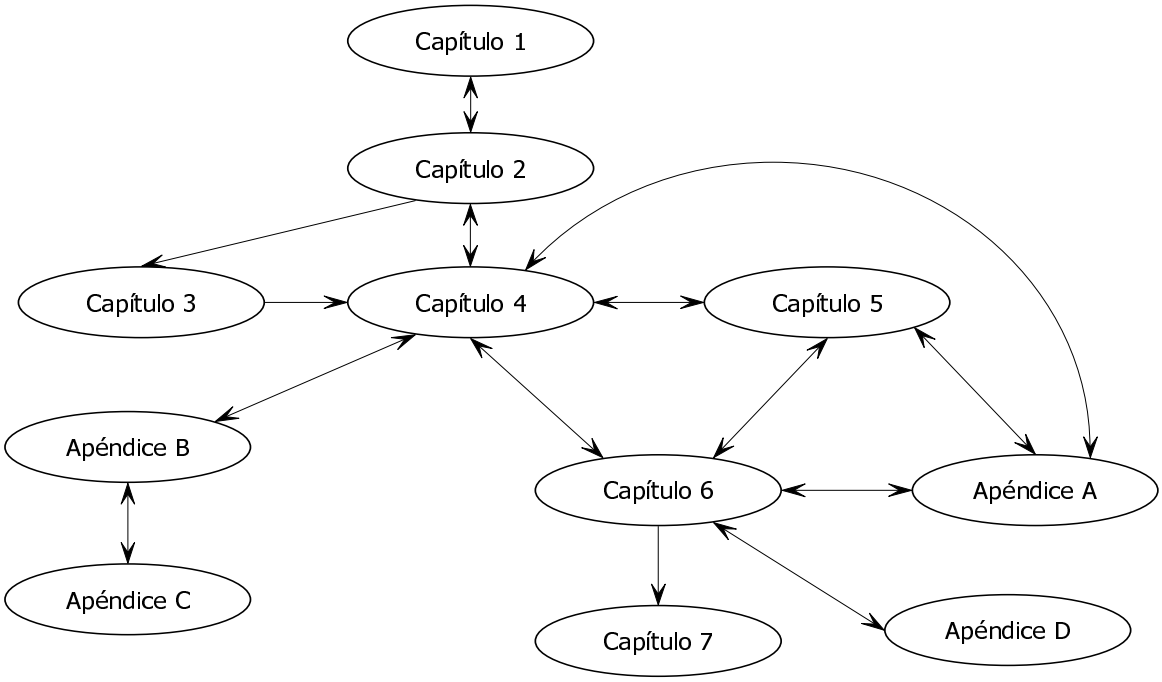


Figura 0 Relaciones conceptuales entre los capítulos de la memoria.

Las referencias se darán entre corchetes, y constarán del apellido del primer autor en letras mayúsculas y del año de publicación del libro o artículo citado. En caso de hacer referencia a dos trabajos del mismo autor y año, se diferenciarán mediante el uso de una letra minúscula añadida tras el año de publicación —a, b, ...—.

Este trabajo ha sido financiado en parte por los proyectos de investigación TIC97-0593-C05-04 y TIC2000-1673-C06-04, ambos concedidos por la Comisión Interministerial de Ciencia y Tecnología (CICYT).

Referencias

[Bieber, 1995] XXIII
[Bush, 1945] XXIV
[Candeira, 2001, página 90] XXIV
[García-Cabrera, 1995, 1996] XXIII

Índice General

Tábula Gratulatoria I
Tabla de Acrónimos III
Tabla de Símbolos V
Lista de Figuras VII
Lista de tablas XIII
Preámbulo XXI
Objetivos XXIV
Estructura de la Exposición XXV
Referencias XXVI

PARTE I: Contexto Hipermedia

CAPÍTULO I: Marco Teórico 1
1 Introducción 3
2 Definición de Hipermedia 5
2.1 Otras Definiciones de Hipermedia 8
3 Historia de la Tecnología Hipertexto/media 9
3.1 Personajes y Sistemas Hipermedia 9
3.1.1 Vannevar Bush y MEMEX (1945) 10
3.1.2 Ted Nelson y XANADU (1965) 11
3.1.3 The Hipertext Editing System (1967) y FRESS (1968) 11
3.1.4 AUGMENT/NLS (1968) 12
3.1.5 ZOG (1975) 13
3.1.6 NoteCards (1985) 13
3.1.7 Intermedia (1985) 14
3.1.8 KMS (1988) 14
3.1.9 Guide (1986) 15
3.1.10 HyperTIES (1986) 15
3.1.11 HyperCard (1987) 16
3.1.12 World Wide Web (1992) 16
4 Modelo Básico Hipermedia 17
5 Características Generales de los Sistemas Hipermedia 18
5.1 Arquitectura de los Sistemas Hipermedia 19

5.2 Navegando en Hipermedia ... 21
5.2.1 Estructura de los Hiperdocumentos ... 22
5.2.2 Ayudas para la Navegación ... 23
5.2.3 Estrategias de Búsqueda ... 25
5.2.4 Semánticas de Búsqueda ... 26
5.2.4 Navegación Inteligente ... 26
5.3 Nodos, Enlaces y otras Definiciones de Interés ... 27
5.3.1 Nodos ... 29
5.3.2 Enlaces ... 29
6 Hipermedia Adaptativos ... 30
7 Referencias ... 32

CAPÍTULO 2: Análisis del Marco Teórico ... 33
1 Introducción: Los Sistemas Hipermedia en Crisis ... 35
2 Los Modelos Hipermedia ... 36
2.1 Integración de Hipermedia en los Sistemas de Información ... 38
2.1.1 Requisitos para la Integración de la Funcionalidad Hipermedia ... 38
2.2 Modelos de Referencia Hipermedia ... 40
2.2.1 HAM, the Hypertext Abstract Machine ... 40
2.2.2 Meta-modelo de Trellis ... 41
2.2.3 El Modelo Dexter ... 42
2.2.4 El Modelo Formal de Hipertexto de Lange ... 43
2.2.5 El Modelo Tower ... 44
2.2.6 Evaluación de los Modelos de Referencia ... 45
2.3 Modelos Cognitivos ... 45
2.3.1 ¿Por qué un Modelo Cognitivo? ... 46
2.4 Evaluación de los Modelos Hipermedia ... 47
3 Descubriendo la Semántica de Estructuración de un Sistema de Información ... 48
3.1 Desde el Documento a la Estructura ... 48
3.2 Desde la Estructura al Documento ... 49
4 Construcción de un Modelo Hipermedia ... 50
5 Conclusiones ... 53
6 Referencias ... 54

PARTE II: El Modelo SEM-HP

CAPÍTULO 3: Bases Teóricas de SEM-HP ... 55
1 Introducción ... 57
2 Teoría de Sistemas ... 57
2.1 Conceptos Básicos ... 58
3 Evolución e Ingeniería del Software ... 59
3.1 El Desarrollo de Sistemas Hipermedia ... 59
3.2 Líneas de investigación ... 60
4 MEDES ... 62
4.1 Una Arquitectura con Dos Niveles de Abstracción ... 62
5 Semántica ... 64
5.1 Ontologías ... 64
6 Conclusiones ... 65
7 Referencias ... 66

CAPÍTULO 4: Descripción Conceptual de SEM-HP ... 67
1 Introducción ... 69
2 El Proceso General de Desarrollo de Hipermedia ... 70
2.1 Evolución durante el Proceso de Desarrollo ... 70
3 Características de SEM-HP ... 72
4 Arquitectura General del Modelo SEM-HP ... 73
5 Terminología de SEM-HP ... 74
5.1 Ítemes de Información ... 74

5.1.1 ¿Por qué Ítem? ... 75
5.1.2 Las Propiedades de los Ítemes de Información ... 75
5.1.3 El Contenido de los Ítemes de Información ... 77
5.1.4 Definiciones ... 78
5.2 Estructura Conceptual ... 79
5.2.1 Asociaciones Conceptuales ... 81
5.2.2 Prestaciones de las Asociaciones Conceptuales ... 81
5.2.3 Definiciones ... 83
5.3 Restricciones ... 84
5.3.1 Las Restricciones Estructurales ... 85
5.3.2 Las Restricciones de Navegación ... 85
5.3.3 Prestaciones de las Restricciones ... 86
5.3.4 Definiciones ... 87
5.4 Acciones Evolutivas y Meta-restricciones ... 88
5.4.1 Propagación del Cambio y Meta-Restricciones ... 88
5.4.2 Prestaciones de las Acciones Evolutivas ... 89
5.4.4 Definiciones ... 90
6 Arquitectura Detallada de SEM-HP ... 90
6.1 El Sistema de Conocimiento SC ... 91
6.1.1 El Subsistema de Memorización SM ... 91
6.1.2 El Subsistema de Presentación SP ... 92
6.1.3 Prestaciones del Sistema de Conocimiento ... 93
6.1.4 Definiciones ... 93
6.2 El Sistema de Navegación SN ... 93
6.2.1 Perspectiva del Autor ... 94
6.2.2 Perspectiva del Lector ... 95
6.2.3 Prestaciones del Sistema de Navegación ... 96
6.2.4 Definición ... 96
6.3 El Sistema de Aprendizaje SA ... 96
6.3.1 Definiciones ... 97
6.4 Sistema Hipermedia e Hiperdocumento ... 97
7 El Proceso de Desarrollo Detallado ... 98
7.1 Evolución del Subsistema de Memorización ... 100
7.2 Evolución del Subsistema de Presentación ... 101
7.3 Evolución del Sistema de Navegación ... 101
7.4 Evolución en el Modelo SEM-HP ... 101
8 Referencias ... 102

CAPÍTULO 5: Formalización de SEM-HP ... 103

Subcapítulo 5.1: Introducción ... 105
1.1 Planteamiento General ... 105
1.2 Formalismos ... 107
1.3 Definiciones Previas: Teoría de Grafos ... 108

Subcapítulo 5.2: Especificación Formal del Subsistema de Memorización 111
2.1 La Estructura Conceptual del Subsistema de Memorización ECm ... 113
2.2 El Diccionario del Subsistema de Memorización D ... 120
2.3 Las Restricciones del Subsistema de Memorización RTm ... 120
2.3.1 Restricciones de los Conceptos RTmc ... 121
2.3.2 Restricciones de las Relaciones Conceptuales RTmr ... 121
2.3.3 Restricciones Relacionadas con los Ítemes de Información RTmii ... 126
2.3.4 Restricciones Relacionadas con los Diccionarios RTmd ... 127
2.4 Acciones Evolutivas del Subsistema de Memorización ACem ... 128
2.4.1 Cambiando la Estructura Conceptual ECm ... 129
2.4.2 Cambiando las Restricciones de Autor ... 143
2.4.3 Cambiando el Diccionario D ... 145
2.5 Propagación Interna del Cambio ... 147

2.5.1 Propagación al Cambiar la Estructura Conceptual ECm 147
2.5.2 Propagación al Cambiar las Restricciones de Autor 147
2.6 Composición de Ítemes de Información 149

Subcapítulo 5.3: Especificación Formal del Subsistema de Presentación 153
3.1 La Estructura Conceptual del Subsistema de Presentación ECp 155
3.2 Las Restricciones del Subsistema de Presentación RTp 156
3.3 Estructuras Conceptuales Virtuales ECvi 157
3.3.1 Cálculo Automático de Componentes Fuertes y del Grafo Condensado 157
3.3.2 Cálculo Manual de los Componentes Fuertes y del Grafo Condensado 160
3.3.3 Cálculo Semiautomático 160
3.4 Las Acciones Evolutivas del Subsistema de Presentación ACep 162
3.4.1 Cambiando la Estructura Conceptual ECp 162
3.4.2 Selección del Subsistema de Información 165
3.4.3 Cambiando las Estructuras Conceptuales Virtuales ECpv 166
3.5 Propagación Interna del Cambio 167
3.5.1 Propagación al Cambiar la Estructura Conceptual ECp 167
3.5.2 Propagación al Cambiar las Estructuras Virtuales ECvi 168

Subcapítulo 5.4: Especificación Formal del Sistema de Navegación 169
4.1 Estructura Conceptual del Sistema de Navegación ECn 172
4.2 Las Restricciones del Sistema de Navegación RTn 172
4.2.1 Lógica Temporal para las Restricciones de Orden RTo 173
4.3 Definición del Lenguaje N 178
4.3.1 Sintaxis del Lenguaje N 178
4.3.2 Metalógica para el Lenguaje N 181
4.3.3 Semántica del Lenguaje N 183
4.4 Las Acciones Evolutivas del Sistema de Navegación ACen 187
4.4.1 Cambiando las Restricciones de Navegabilidad RTnb 188
4.4.2 Cambiando las Restricciones de Orden RTo 189
4.4.3 Propagación de los Cambios en el Sistema de Navegación 195
4.5 La Red de Petri RP 197
4.5.1 Definición Formal de una Red de Petri, RP 199
4.5.2 Transformación de Cláusulas en lenguaje N a RP 201
4.6 Árboles de Alcanzabilidad o de Navegación 209
4.6.1 Definiciones Previas 209
4.6.2 Aplicaciones del Árbol de Alcanzabilidad o Navegación 211

Subcapítulo 5.5: Consideraciones Generales 215
5.1 Propagación Externa del Cambio 217
5.1.1 SubSistema de Memorización: Propagación Externa del Cambio 217
5.1.2 SubSistema de Presentación: Propagación Externa del Cambio 218
5.1.3 Sistema de Navegación: Propagación Externa del Cambio 218
5.2 Prestaciones de la Formalización Usada 219
5.2.1 ESTRUCTURA y Formalismos 219
5.2.2 Evolución y Formalismos 220
5.3 Flexibilidad y Versatilidad de SEM-HP 221
5.4 Referencias 222

Capítulo 6: Consideraciones sobre Diseño e Implementación 223
1 Introducción 225
2 Especificación de SEM-HP mediante UML 226
2.1 Diagramas de Clases 226
2.2 Diagramas de Secuencia 234
3 Prototipos de las Interfaces de SEM-HP 242
3.1 El Sistema de Memorización 242
3.2 El Sistema de Presentación 247
3.3 El Sistema de Navegación 247
3.4 El Navegador 247

4 El Papel de XML en SEM-HP 256
4.1 XML 256
4.2 El lenguaje de enlazado de XML (XML Linking) 257
4.3 SEM-HP y XML 258
5 Referencias 259

PARTE III

Capítulo 7: Recapitulación 261
1 Trabajos Relacionados 263
1.1 El Sistema Hipertexto RICH 263
1.1.1 Representación de la Red Semántica 263
1.1.2 Comparativa 265
1.2 El Modelo de Referencia AHAM 266
1.2.1 Arquitectura de AHAM 266
1.2.2 Comparativa 268
1.3 Hipermedia como Autómatas 268
1.3.1 Comparativa 269
2 Conclusiones del trabajo 269
3 Problemas Abiertos 271
4 Referencias 273

Apéndice A: Glosario de Términos de SEM-HP 275

Apéndice B: Demostración de la Consistencia de RTa 281
1 Introducción 283
2 Combinaciones Posibles 284
3 Combinaciones Inconsistentes 285
3.1 Demostración de las Reglas de Inclusión e Incompatibilidad de la Tabla B.2 286
3.2 Programa1 : Elimina Combinaciones No Válidas 291
4 Combinaciones Consistentes 292
4.1 Demostración de los Representantes 293
4.2 Reglas Representantes 293
4.3 Demostración de las Combinaciones Consistentes 295
4.4 Programa 2: Comprueba que las Combinaciones Válidas cumplen los Axiomas 299
4.5 Programa 3: Genera a partir de los Axiomas las Combinaciones Válidas 299

Apéndice C: Programas y Resultados 301
1 Introducción 303
1.1 Programa 1: Elimina Combinaciones No Válidas 303
1.2 Resultados 1: Elimina Combinaciones No Válidas 305
1.3 Programa 2: Comprueba que las Combinaciones Válidas cumplen los Axiomas 314
1.4 Resultados 2: Comprueba que las Combinaciones Válidas cumplen los Axiomas . 318
1.5 Programa 3: Genera a partir de los Axiomas las Combinaciones Válidas 326
1.6 Resultados 3: Genera a partir de los Axiomas las Combinaciones Válidas 327
1.7 Comparación entre los Resultados del Programa 1 y 3 331

Apéndice D: Restricciones Asociativas 341
1 Introducción 343
1.1 Un Lenguaje para las Restricciones Asociativas RTa 343
2 Las Restricciones Relacionales expresadas en el Lenguaje de las Asociaciones 344

Bibliografía 351

Parte I: Contexto Hipermedia

CAPÍTULO I

Marco Teórico

«Hay que ser remero antes de llevar el timón, haber estado en la proa y observado los vientos antes de gobernar la nave»

Aristófanes.
Comediógrafo griego (450 a.C.-385 a.C).

«No es el haber recibido una lección lo que nos salva, sino el haber sabido aprovecharla.»

George Canning.
Político inglés (1770-1827).

Resumen

Este capítulo contiene los fundamentos teóricos necesarios para el desarrollo de los siguientes apartados de este trabajo. Evidentemente, la mayor parte del material aquí presentado no es original, aunque se ha adaptado con objeto de conseguir una unificación del contenido que provea a la presente tesis de cierta autosuficiencia.

Se mostrarán los conceptos básicos sobre sistemas hipermedia, se hará mención a su historia, se presentarán los distintos aspectos que son objeto de estudio en hipermedia, dedicando una mayor extensión a su funcionalidad básica, la navegación. El capítulo termina con una breve descripcion del campo de investigación más próximo a este trabajo, los denominados sistemas hipermedia adaptativos.

1 Introducción 3
2 Definición de Hipermedia 5
2.1 Otras Definiciones de Hipermedia 8
3 Historia de la Tecnología Hipertexto/media 9
3.1 Personajes y Sistemas Hipermedia 9
3.1.1 Vannevar Bush y MEMEX (1945) 10
3.1.2 Ted Nelson y XANADU (1965) 11
3.1.3 The Hipertext Editing System (1967) y FRESS (1968) 11
3.1.4 AUGMENT/NLS (1968) 12
3.1.5 ZOG (1975) 13
3.1.6 NoteCards (1985) 13
3.1.7 Intermedia (1985) 14
3.1.8 KMS (1988) 14
3.1.9 Guide (1986) 15
3.1.10 HyperTIES (1986) 15
3.1.11 HyperCard (1987) 16
3.1.12 World Wide Web (1992) 16
4 Modelo Básico Hipermedia 17
5 Características Generales de los Sistemas Hipermedia 18
5.1 Arquitectura de los Sistemas Hipermedia 19
5.2 Navegando en Hipermedia 21
5.2.1 Estructura de los Hiperdocumentos 22
5.2.2 Ayudas para la Navegación 23
5.2.3 Estrategias de Búsqueda 25
5.2.4 Semánticas de Búsqueda 26
5.2.4 Navegación Inteligente 26
5.3 Nodos, Enlaces y otras Definiciones de Interés 27
5.3.1 Nodos 29
5.3.2 Enlaces 29
6 Hipermedia Adaptativos 30
7 Referencias 32

1 Introducción

Las Ciencias de la Computación siguen el curso normal de ir ensanchando más y más sus límites. Se ha ido pasando desde las máquinas al software; de éste a la comunicación; de ésta al contenido y, posiblemente, de ahí al conocimiento. Esto significa que, en cierto sentido, las fronteras de la computación se están rompiendo. Según Terry Winograd [Winograd, 1997] las investigaciones de los próximos 50 años no se desarrollarán en las tradicionales áreas de *software* y *hardware*, sino que ambicionarán la posibilidad de comprender, analizar y crear espacios de interacción.

Según estas apreciaciones, la interacción hombre-máquina se convierte en uno de los campos de investigación de moda por su interés en el diseño de la interacción y, por ende, por su preocupación de estudiar las relaciones y comportamiento humano para conseguir una comunicación más flexible e inmediata con el computador.

La tecnología hipermedia precisamente se enmarca dentro de esta área de conocimiento, y se le asegura un próspero porvenir pero, ¿a qué debe su éxito? Parece ser que ha sabido aunar dos de los campos que, ahora y en el futuro, guiarán las investigaciones y la innovación dentro de las ciencias de la computación [Winograd, 1997]: la interacción humana y la comunicación.

Los sistemas hipermedia aparecen como una nueva clase de sistemas de administración de información compleja. Estos sistemas permiten crear, anotar, enlazar y compartir información de distintos medios como texto, gráficos, audio, vídeo, animación y programas. Proporcionan un acceso no secuencial, completo y uniforme a la información. De forma breve, un sistema hipermedia es un sistema de bases de datos que no tiene una estructura regular, y que proporciona un método único de acceso a la información. En realidad se trata de soportes de estructuración de información y acceso a esta información con una granularidad abstracta mayor a la de un sistema de bases de datos. La estructura general de los hipermedia permite aplicarlos a una gran variedad de dominios, tales como: manuales técnicos, diccionarios, enciclopedias, catálogos, periódicos y, en general, cualquier obra de referencia.

De esta primera aproximación a los sistemas hipermedia se desprende que su principal ventaja es la flexibilidad. No obstante, ésta parece no ser una razón suficiente para apostar por esta tecnología y uno se podría seguir preguntando, ¿por qué investigar en

Hipermedia

El término hipermedia es el sucesor del vocablo hipertexto que, por primera vez en 1965, utilizó Nelson. Los términos hipertexto e hipermedia se suelen emplear como sinónimos. La distinción entre hipertexto e hipermedia reside en el número de medios de comunicación empleados, por ejemplo Balasubramanian [Balasubramanian, 1993] define hipermedia como hipertexto con multimedia.

Bases de Datos

La mayoría de los sistemas hipermedia no usan un sistema de bases de datos para el almacenamiento de nodos y enlaces. Los sistemas de bases de datos relacionales no resultan adecuados para el almacenamiento y manipulación de la información basada en nodos debido a que no mantiene una estructura uniforme. Normalmente, un sistema de bases de datos no permite leer un dato mientras que está siendo actualizado. En un sistema hipermedia el tamaño de los distintos elementos es enorme, y la duración de las actualizaciones llevan demasiado tiempo como para excluir la realización de ambas operaciones simultáneamente. Sin embargo, los enlaces se pueden estructurar de forma adecuada para que se puedan incluir en una base de datos. Esto permitiría al sistema usar las funciones estándar de las bases de datos para asegurar que los enlaces siempre conducen a nodos existentes, y prohibir nodos aislados.

Los sistemas de bases de datos orientados a objetos parecen prometer una solución a estos problemas. Algunos intentos hay al respecto tales como el sistema hipermedia Multicard [Rizk, 1992].

este campo? Muchas son las razones que inducen a ello y como se puede observar de muy diversa índole:

- Esta nueva tecnología con apenas 25 años participa y reúne bajo sí a muchas de las áreas de conocimiento dentro de las ciencias de la computación: bases de datos, interfaz de usuario, recuperación de la información, comunicación, multimedia, ingeniería del software, colaboración en grupo, sistemas operativos y aspectos de integración —figura 1.1—.
- Sobrepasa las fronteras de las ciencias de la computación y necesita el auxilio de otras disciplinas de carácter humanístico y técnico como: psicología, comunicación, diseño gráfico, lingüística y pedagogía.
- Ha propiciado la aparición de los nuevos computadores de red —*Network Computer*—, el nuevo electrodoméstico de comunicación que pretende desbancar a los PCs. Esta nueva plataforma contará con un pequeñó sistema operativo de micronúcleo, un entorno de aplicación con capacidad para ejecutar Java, un navegador hipermedia y un protocolo estándar para comunicación [Wayner ,1996].
- Se está aplicando con éxito en entornos destinados a la divulgación de la información, y parece ofrecer interesantes características para el desarrollo de aplicaciones de enseñanza-apredizaje [García-Cabrera, 1996].

NC (Network Computer)

Un ordenador de red, en principio, puede ser cualquier cosa ya que es una plataforma única que no especifica el tipo de hardware, CPU o sistema operativo. En lugar de eso, define un modelo de cliente abierto, centrado alrededor de los estándares de Internet y de Java.

Aplicaciones de Enseñanza-Aprendizaje

La tecnología hipermedia asume como capacidades intrísecas la composición, lectura, anotación, enlace y manipulación de productos de conocimiento. Ofrece la posibilidad de organizar un material de información multimediático procedente de diversas fuentes y, de muy distintas maneras. Esto quiere decir, en primer lugar, que contamos con una tecnología de presentación y estructuración de dominios de información potente y flexible. En segundo lugar, que la hipermedia hace posible la construcción de herramientas abiertas con posibilidades de crecimiento y adaptación. Tercero, que admite combinar e integrar diversos documentos de cualquier tipo. Y por último, que esta nueva tecnología acepta accesos simultáneos y desarrollos colaborativos. Todas estas características hacen de la hipermedia una tecnología prometedora para la educación.

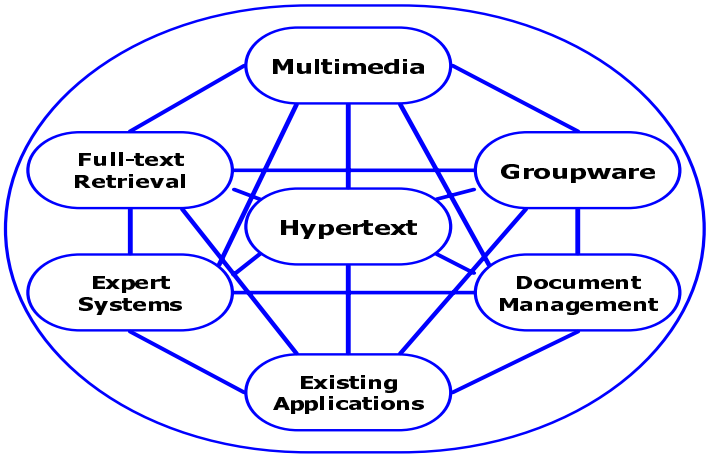


Figura 1.1 Los Sistemas Hipermedia son multidisciplinares [Balasubramanian, 1993].

A pesar del auge, prosperidad y amplio ámbito de aplicación de estos sistemas, las limitaciones y deficiencias de los actuales hipermedia parecen crecer. El modelo básico hipermedia no resulta suficientemente rico para organizar, estructurar, acceder y realizar algunas tareas de muchas aplicaciones. Problemas como la des-

orientación del usuario, el *conocimiento overhead* y la especificación de directrices para la construcción de redes de información, son objeto de estudio por los investigadores.

Se han propuesto algunas extensiones con el fin de enriquecer y ampliar la funcionalidad hipermedia haciéndola más integrable, abierta, dinámica y adaptable, por una lado, a nuevas aplicaciones y, por otro, al comportamiento del usuario.

En el siguiente apartado se definirán algunos conceptos clave sobre estos sistemas. A continuación —apartado 3—, se hará un breve repaso de la reciente y corta historia de los sistemas hipermedia haciendo hincapié en los sistemas y personajes que han contribuido especialmente en este campo. Posteriormente, en el apartado 4, se presenta el modelo básico al que responde, en líneas generales, cualquier sistema hipermedia. En el apartado 5, se ampliarán las primeras definiciones para, luego, describir la funcionalidad básica de los hipermedia: la navegación. Este capítulo terminará —apartado 6— presentando una visión general sobre las líneas de investigación en las que podría inscribirse este trabajo.

Seguramente ya habrá notado algo raro en el formato de este escrito. Con objeto de imbuirle de este mundo de los hipermedia, se presenta únicamente este capítulo como si fuera un hipermedia. Algunas de las palabras o frases que aparecen en este documento tienen un color distinto al negro y se muestran subrayadas. Estos son los enlaces, que en un hipermedia real le llevarían de forma automática a una nueva página o párrafo. Aquí se va a emplear tres tipos distintos de enlaces:

- Los marcados en color rojo serán enlaces internos, esto es, en el mismo capítulo, más adelante o atrás, existe más información sobre ese concepto.
- Los marcados en color azul son enlaces que en un hipermedia se denominan de referencia trasversal. Se interpretan como comentarios, aclaraciones o anotaciones relativos a una palabra o frase marcada. En este escrito estos enlaces aparecen en la zona de la derecha de la página.
- Por último, los marcados con color verde son enlaces que le llevan a otro capítulo distinto del actual como, por ejemplo, el apéndice que reúne todas las referencias bibliográficas.

2 Definición de Hipermedia

Hipermedia es un acrónimo que combina las palabras hipertexto y multimedia. Hipertexto e hipermedia se suelen utilizar como sinónimos a pesar de que, evidentemente, el primer vocablo únicamente sugiere que toda la información se ofrece en forma de texto. En esta memoria siempre se utilizará la palabra hipermedia porque prácticamente la totalidad de estos documentos incluyen gráficos y, muchos de ellos, también sonido y vídeo —figura 1.2 y 1.3—.

Shneiderman [Shneiderman, 1989a, 1989b] define hipermedia como *«una base de datos con referencias cruzadas activas que le permiten al lector saltar a otras partes de un documento siempre que así lo desee»*. Esta definición pone de manifiesto algunas cuestiones interesantes sobre hipermedia:

- Un sistema hipermedia es una base de datos. La información no es simplemente un cubo lleno de *bytes*, sino que está estructurada. Aunque esta organización de la información se hace de modo distinto al de la mayoría de las bases de datos tradicionales.
- La operación típica que realiza el usuario es saltar entre las partes de esta base de datos para consultar la información.

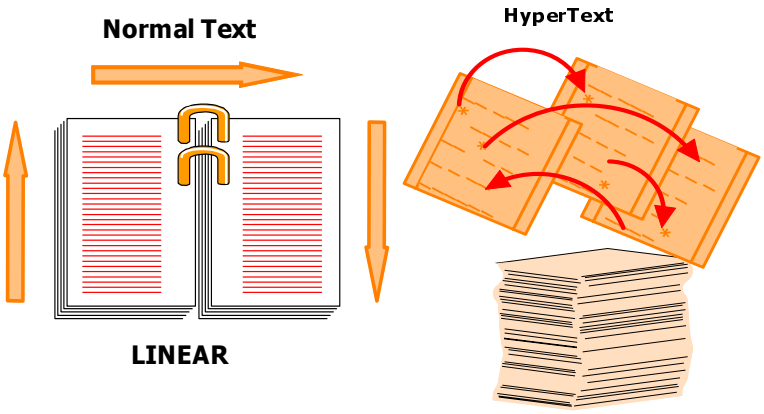


Figura 1.2 Diferencias entre un texto y un hipertexto.

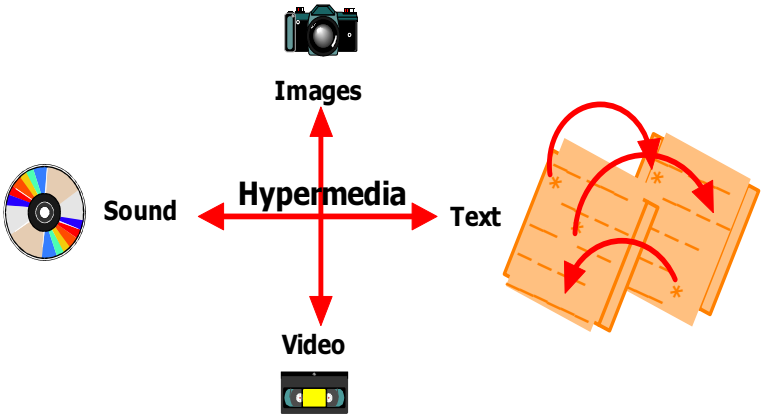


Figura 1.3 En un hipermedia se combinan todos los modos de expresión posibles.

- Además de las distintas piezas de información —textual u otras—, la base de datos contiene conexiones entre trozos de información, de algún modo relacionadas, que van a guiar al usuario en su exploración a través de la información.

Aunque se podría usar una base de datos suficientemente sofisticada para cargar y recuperar la información, las interfaces de usuario típicas deberían ser reemplazadas por nuevas interfaces que soportaran interacción hipermedia, a las que se denomina normalmente buscadores y navegadores —*browsing*—.

Hiperdocumento

En la literatura el término *hipermedia* se usa para hacer referencia a un sistema hipermedia pero también para hablar de la información que contiene o a ambos. Todo esto induce a confusión, aquí emplearemos tres palabras distintas:

- un hiperdocumento es la información que contiene el sistema hipermedia, incluyendo los ítemes de información, o *nodos*, y las conexiones entre ellos, o *enlaces*.
- un sistema hipermedia es una pieza software que deja al lector leer -y a veces escribir- hiperdocumentos y que no contiene un hiperdocumento.
- un hipermedia es un sistema hipermedia que contiene un hiperdocumento.

Las distintas piezas de información, o partes de la base de datos, se llaman nodos; las conexiones entre los distintos nodos, enlaces. Ambos forman lo que se llama un hiperdocumento. Enlaces y nodos adoptan la forma de un grafo arbitrariamente complejo.

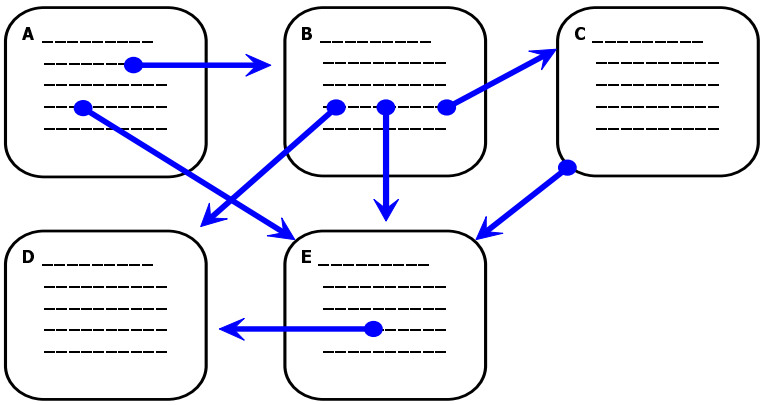


Figura 1.4 En un hiperdocumento se puede usar tres elementos básicos: nodos, enlaces y anclajes.

En la figura 1.4 se puede ver una representación de un hiperdocumento muy simple que tiene tan sólo cinco nodos y siete enlaces. Esta figura también muestra un nuevo elemento —en el dibujo está representado por un punto— que enlaza y apunta a un punto específico —palabra o región— dentro de un nodo, es lo que se conoce con el nombre de anclaje —*anchor*—.

En un documento escrito en papel también se tiene algo parecido a los enlaces. El índice es un conjunto de enlaces, pero en este caso no es posible a partir de una palabra del libro que se indica en el índice saltar a otra página, sin previamente acudir primero al índice para saber de qué página se trata. Otros ejemplos de enlaces serían las referencias bibliográficas o las notas al pie. En realidad lo que han hecho los hiperdocumentos es generalizar este último concepto de nota al pie. Si se fija, en este elemento se puede observar que se indica siempre algún símbolo que advierte de la presencia de un enlace, y que el destino del enlace es un trozo de información independiente.

Algunos autores no consideran suficiente la presencia de nodos y enlaces para llamar hiperdocumento a un documento. No existe un consenso en cuanto a las características que un sistema debe incorporar para identificarlo como hipermedia. Frank Halasz [Halasz, 1988b] y otros han formulado unos *requisitos adicionales* para que se pueda decir de un sistema que es un hipermedia —definición estricta de hipermedia—.

En este escrito se dará una perspectiva computacional de hipermedia, algunas veces desde las bases de datos, y otras desde una perspectiva orientados a objetos. Existen otras definiciones de hipermedia que arrancan desde la literatura o las interfaces de usuario, o la filosofía.

Definición Estricta de Hipermedia

Muchos de los autores e investigadores sobre sistemas hipertexto consideran que un sistema que despliega información a base de nodos y nos permite enlazarlos, no merece necesariamente la etiqueta de hipermedia. Cada sistema hipermedia debe proporcionar algunas herramientas para encontrar un camino a través del hiperdocumento, y no sólo seguir simples enlaces.

Existen diferentes herramientas para ayudar al usuario en la búsqueda de una ruta sin perderse en el camino, como vistas dinámicas de la estructura de red del documento, enlaces bidireccionales o representaciones históricas.

Nielsen [Nielsen, 1990] es un poco menos preciso, pero mucho más intuitivo, a la hora de juzgar a un sistema como hipermedia o no: debe parecer y notarse -*look and feel*- como un hipermedia.

2.1 Otras Definiciones de Hipermedia

Se reúnen en este apartado todas las definiciones de hipermedia que se han encontrado en la literatura técnica:

- √ Ted Nelson [Nelson, 1967] definió hipermedia como *«a combination of natural language text with the computer's capacity for interactive branching or dynamic display (...) of a nonlinear text (...) which cannot be printed on a conventional page».*
- √ Nelson [Nelson, 1987] también escribió: *«Hypertext, or non-sequential writing with free user movement along links, is a simple and obvious idea. It is merely the electronification of literary connections as we already know them».*
- √ Landow y Delany [Landow, 1991] ven a los hipermedia como una nueva forma de literatura: *«We can define hypertext as the use of the computer to transcend the linear, bounded and fixed qualities of the traditional written text».*
- √ J. D. Bolter [Bolter, 1989] también tiene un punto de vista literario: *«A hypertext consists of topics and their connections, and (...) the topics can be paragraphs, sentences, or individual words. A hypertext is like a printed book that the author has himself attacked with a pair of scissors and cut into convenient verbal sizes. The difference is that the electronic hypertext does not simply dissolve into a disordered bundle of slips; the author defines its structure by establishing electronic connections among the slips».*
- √ Jeff Conklin [Conklin 1987] da una definición más informática: «*Mechanisms are being devised which allow direct machine-supported references from one textual chunk to another; new interfaces provide the user with the ability to interact directly with these chunks and to establish new relationships between them. These extensions of the traditional text fall under the general category of hypertext —also known as nonlinear text—*».
- √ J. Fiderio [Fiderio, 1988] da la siguiente definición en la conocida revista Byte: *«Hypertext, at its most basic level, is a DBMS that lets you connect screens of information using associative links. At its most sophisticated level, hypertext is a software environment for collaborative work, communication, and knowledge acquisition. Hypertext products mimic the brain's ability to store and retrieve information by referential links for quick and intuitive access».*
- √ En ciencia cognitiva, hipertexto se percibe de forma muy distinta [Rao, 1990]: *«It is appropriate to view hypertext as a method of supporting the expression of relationships among objects in a database. Hypertext should be treated as a general-purpose tool with approaches to handling nodes, links, and retrieval, that fits within the context of any application and convey common meanings to the users».*
- √ J. McDaid [McDaid, 1991] enfatiza los aspectos multimedia de hipermedia: *«Hypermedia is Theodore Nelson's term for computer-mediated storage and retrieval of information in a nonsequential fashion. An extension of Nelson's earlier coinage, "hypertext" (for non-sequential writing), hypermedia implies linking and navigation through material stored in many media: text, graphics, sound, music, video, etc. But the ability to move through textual information and images is only half the system: a true hypermedia environment also includes tools that enable readers to rearrange the material».*

3 Historia de la Tecnología Hipertexto/media

Si se revisa la literatura relacionada con los avances en hipermedia se puede comprobar que los sistemas hipermedia son prácticamente de ayer. Su invención data de finales de los 80, y han sido popularizados en los 90. La introducción de Guide en 1986, de HyperCard en 1987, y la primera Conferencia sobre Hipermedia de ACM en ese mismo año, son claros signos de que una nueva área de investigación y desarrollo estaba naciendo. Word Wide Web, una iniciativa de Tim Berners Lee en el CERN dos años más tarde, y que divulgó Marc Andreessen y Eric Bina a partir de 1992 con el navegador Mosaic, convirtió a la tecnología hipermedia en una realidad para el sector comercial. Sin embargo, las primeras ideas acerca de esta nueva forma de concebir un documento tienen un origen mucho anterior, más incluso que el gran fenómeno de las Ciencias de la Computación.

Halasz en 1989 divide a los sistemas hipermedia en dos generaciones. La primera incluye a sistemas como Xanadu, ZOG, NLS/Augment, Hypertext Editing System, FRESS o Dynabook. Todos ellos son sistemas hipertexto basados en mainframes multiusuario que comparten un sistema de información en red. La segunda generación comienza en los 80 con la llegada de las sistemas orientados a la investigación —desarrollados en estaciones de trabajo— como Notecards, Neptune, Intermedia, KMS, Writing Environment, Emacs/INFO, Document Examiner. Estos sistemas son similares en su concepción, pero añaden nodos con gráficos o animación e interfaces de usuario más avanzadas. Además se diseñan tanto para sistemas monousario como multiusuario. Estos logros se fueron incorporando posteriormente a otras plataformas, como PC o Macintosh, dando lugar a sistemas como Guide, HyperTies o HyperCard.

En la tabla 1.1 se puede apreciar los hechos históricos más relevantes que han propiciado e impulsado el desarrollo de lo que hoy se conoce como sistemas hipermedia. Como puede verse, la tabla acaba en el año 1987. Los acontecimientos y desarrollos después de esta fecha son demasiado numerosos para nombrarlos.

No obstante, se puede decir que a partir de esa fecha se inicia una tercera generación, en la que las investigaciones están orientadas al desarrollo de prototipos que intentan conceptualizar los sistemas hipermedia mediante modelos abstractos. Entre ellos se pueden citar los modelos HAM, Dexter, y Trellis. Canto, Nested Context Model, HyperSet, InterSet e HyperNet son modelos hipermedia y sistemas prototipo que tienen como objetivo eliminar algunos de los problemas hipermedia proporcionando sistemas con mecanismos de estructuración más elaborados que permitan componer entidades. Harmony es un sistema hipermedia orientado a objetos. Asimismo, no es posible dejar en el olvido, la introducción del ya famoso World Wide Web como hipermedia distribuido de libre acceso en el que conviven varios tipos de interfaces de usuario.

3.1 Personajes y Sistemas Hipermedia

Como se ha visto, el concepto de hipermedia se ha ido desarrollando a lo largo del tiempo gracias a la creación de muchas aplicaciones y a las aportaciones de un gran número de investigadores. Se descri-

Historia de los Sistemas Hipermedia	
1945	Vannevar Bush propuso Memex en su artículo "As We May Think".
1965	Ted Nelson introduce Xanadu y acuña el término 'hipertexto'.
1967	Adries van Dam desarrolla Hypertext Editing System en la Universidad de Brown, al que siguió la introducción de FRESS en 1968.
1968	Doug Engelbart da una aplicación de demostración de NLS como parte del proyecto Augment que comenzó en 1962.
1975	Un equipo de la Universidad de Carnegie-Mellon, dirigido por Robestson, desarrolla el sistema ZOG, que más tarde se convierte en KMS.
1978	Un equipo del MIT encabezado por Andrew Lippman desarrolla el Aspen Movie Map, el primer ejemplo de aplicación multimedia que incluye videodiscos.
1985	Janet Walker desarrolla el Symbolics Document Examiner, el primer hipermedia utilizado por clientes.
	Se desarrollan varios sistemas hipermedia entre los que se incluyen NoteCards de Xerox e Intermedia de la Universidad de Brown.
1986	OWL introduce Guide para el Machintosh, el primer sistema hipermedia ampliamente difundido, basado en el sistema Guide de Unix, desarrollado por Peter Brown en la Universidad de Kent.
1987	Apple entrega HyperCard junto a cada Macintosh.
	ACM organiza la primera Conferencia sobre Hipermedia.

Tabla 1.1 Acontencimientos más significativos de la Historia de los Sistemas Hipermedia.

be aquí las implementaciones y las personas que han contribuido significativamente en este progreso.

3.1.1 Vannevar Bush y MEMEX (1945)

Vannevar Bush (1890-1974) fue uno de los consejeros científicos del presidente Roosevelt. A principios de 1932 desarrolló algunas ideas sobre MEMEX —MEMory EXtender— que dejó escritas en un borrador en 1939. Este manuscrito se publicó en 1945 en el Atlantic Monthly bajo el título «As We May Think» [Bush, 1945].

MEMEX nunca se construyó, pero pretendía ser un dispositivo mecanizado que permitiera consultar cualquier clase de material escrito y organizarlo de cualquier modo que quisiera el usuario, con la posibilidad de añadir comentarios y enlaces a otros documentos.

Figura 1.5 Vannevar Bush.

MEMEX debería almacenar toda la información escrita en microfilm y dar la imagen al usuario de un escritorio. Este puesto de trabajo tendría muchos proyectores de microfilm para que el usuario pudiera ver varios documentos al tiempo. El usuario podría escribir comentarios y notas en el margen para luego *escanearlos* y añadirlos al sistema.

La característica más innovadora de MEMEX fue la posibilidad de crear enlaces asociativos entre ítemes de información o documentos. Estos enlaces se combinarían para formar rutas de viaje a través de los conceptos más importantes de la información. Incluso llegó a sugerir el nacimiento de una nueva profesión, la de los guías, que podrían establecer caminos útiles a través de esa gran masa de información, esto es, las personas que hoy se dedican a estructurar la WWW [Berners-Lee, 1992] creando los denominados *WebRings*.

Vannevar Bush fue uno de los científicos pioneros en el desarrollo de los computadores. Inventó en 1931 el Differential Analyzer en el MIT. Su conocimiento sobre los enormemente costosos que eran entonces los computadores hizo que no llegara a proponer a MEMEX como un producto para la computadora.

3.1.2 Ted Nelson y XANADU (1965)

La idea que tenía Ted Nelson [Nelson, 1987] cuando imaginó Xanadu fue crear un entorno literario unificado a escala global. Un almacén para cualquier escrito de quien fuera. De ahí el término Hipertexto que introdujo este autor en 1965.

En Xanadu se hizo una fuerte separación entre la interfaz de usuario y la base de datos. El énfasis se puso en esta última. Esta base de información se implementó en UNIX. En este sistema se puede direccionar cualquier subcadena de cualquier documento a partir de cualquier escrito. Este mecanismo requiere de un esquema de direccionamiento mucho más fuerte que el URL de World Wide Web debido a que cada carácter de cada documento necesitaba una única dirección. En Xanadu nunca se borra texto. Se mantiene un registro de todas las versiones de cada ítem de información. Se desarrolló originalmente en la Universidad de Brown, pero más tarde lo distribuyó la empresa Autodesk.

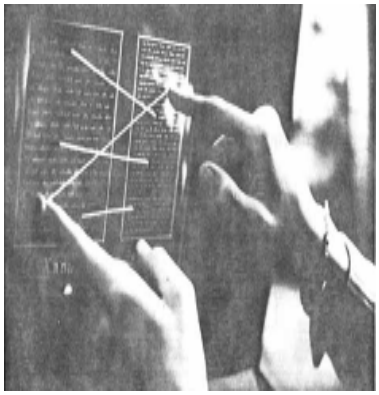


Figura 1.6 Sistema Xanadu.

3.1.3 The Hipertext Editing System (1967) y FRESS (1968)

El primer sistema hipertexto que llegó a funcionar fue el desarrollado por la Universidad de Brown, por el equipo dirigido por Andries van Dam. Este sistema se ejecutaba en 128KB de memoria sobre un mainframe IBM/360. Financiado por IBM, más tarde lo vendió

a Houston Manned Spacecraft Center, donde se usó para producir la documentación del programa espacial Apollo.

Figura 1.7 Andries van Dam.

En 1968 van Dam desarrolló FRESS —File Retrieval and Editing System—. Se trataba de una versión mejorada para sistemas multiusuario y de tiempo compartido para el mismo *mainframe* de IBM. Más tarde se reimplementó y lo comercializó Philips. FRESS fue empleado por cientos de miembros y estudiantes de las facultades a lo largo de una década. Esta herramienta tenía una estructura jerárquica dinámica con enlaces bidireccionales. En terminales gráficas, FRESS ofrecía múltiples ventanas y gráficos vectorizados. A esta aplicación le siguió el sistema Electronic Document que soportaba gráficos a color y ayudas de navegación. Alrededor de 1985, la Universidad de Brown anunció Intermedia, un potente sistema hipermedia para Macintosh sobre A/UX.

3.1.4 AUGMENT/NLS (1968)

A pesar de que en aquellos tiempos los computadores resultaban demasiado caros para ser dedicados a trabajos interactivos y para computaciones no numéricas, Engelbart inició su proyecto en 1962 con el propósito de desarrollar herramientas que aumentaran las capacidades y productividad humanas. En 1963 escribió un artículo titulado: «*A Conceptual Framework for the Augmentation of Man's Intellect*» [Engelbart, 1963]. El sistema que propuso, H-LAM/T —Human using Language, Artifacts, and Methodology, in which he is Trained— puso al usuario como núcleo central. Para él, el usuario y el computador debían cambiar componentes dinámicamente en una simbiosis que tuviera el efecto de ampliar la inteligencia del mismo. El equipo de investigación de Engelbart en el SRI —Stanford Research Institute— creció hasta un total de 45 personas que trabajaban en un conjunto de herramientas que hoy se denomina aplicaciones ofimáticas.

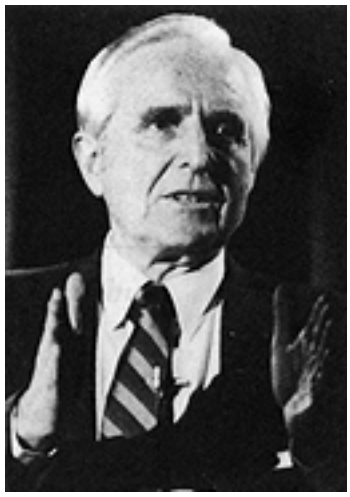


Figura 1.8 Doug Engelbart.

En concreto, en 1968, Engelbart introduce NLS —oN Line System—, una herramienta experimental para almacenar todas las especificaciones, planos, diseños, programas, documentación, informes, etc. Los documentos poseían una sofisticada estructura jerárquica sobre la que se implantaban una red de enlaces referenciales. El usuario podía establecer el nivel de detalle así como el número de elementos por nivel que deseaba consultar. Además, también podía modificar el contenido de los documentos, introduciendo sus propios comentarios. A lo largo de los años llegó a tener alrededor de 100.000 ítemes de información. Las terminales eran muy sofisticadas, incluyendo proyectores de vídeo, teclados especializados y ratón.

3.1.5 ZOG (1975)

ZOG es un proyecto que inició CMU —Carnegie Mellon University— alrededor de 1975. La palabra ZOG no significa nada. Se eligió porque era corta y fácil de pronunciar y recordar. Este sistema se desarrolló para que se utilizara en modo multiusuario. Cada segmento de la base de datos de ZOG se llamó marco —*frame*—, y consiste en un título, una descripción, una línea con las órdenes estándares de ZOG, y un conjunto de ítemes en un menú que conducen a otros marcos. La estructura es jerárquica, aunque puede incluir enlaces de referencia trasversal. La implementación original se ejecutaba en *mainframes*, pero más tarde se portó a estaciones de trabajo PERQ.

En 1982 ZOG se instala y usa como sistema de administración de información para portaaviones USS Carl Vision. Este aplicación es más una base de datos textual, o una herramienta de instrucción, que un sistema hipertexto interactivo. Su interfaz puramente textual, con pequeños nodos, conduce fácilmente a la desorientación del usuario.

Dos de sus principales diseñadores, Donald MacCraken y Robert Akscyn, crean la empresa Knowledge Systems en 1981 y construyen KMS —el sucesor de ZOG para estaciones de trabajo Sun y Apollo—.

3.1.6 NoteCards (1985)

Franks Halasz [Halasz, 1988a] diseñó NoteCards en Xerox PARC como sistema de investigación, pero más tarde se comercializó. Se desarrolló para máquinas Xerox Lisp usando el entorno de programación InterLisp. En esta plataforma la aplicación es parte del entorno, por tanto, los usuarios pueden fácilmente implementar tipos de nodos especializados si conocen el lenguaje Lisp.

Cada nodo es un sólo *notecard* que contiene una cantidad arbitraria de información. Al igual que Guide e Intermedia, utiliza *scrolling* para mostrar las distintas partes de un nodo. Cuando se sigue un enlace, el nodo destino se despliega en una nueva ventana. Los usuarios pueden abrir tantas ventanas como quieran.

NoteCards permite unos 50 tipos de tarjetas especializadas para aplicaciones que necesitan datos estructurados concretos. Los enlaces son conexiones tipificadas entre tarjetas. El tipo es una etiqueta elegida por el creador del enlace, que puede indicar el tipo de relación entre el nodo origen y el destino.

El navegador puede mostrar con este sistema una vista gráfica panorámica de la estructura de enlaces del hiperdocumento. Los enlaces de distinto tipo se muestran con un patrón de línea diferente. Las tarjetas se pueden anidar jerárquicamente usando el llamado *FileBox*. Los *FileBoxes* son tarjetas especiales —similares a los directorios— que pueden contener tanto *FileBoxes* como otras tarjetas.

Frank Halasz

Frank Halasz trabajó con Randy Trigg y Tom Moran en Xerox —Palo Alto—. Entre 1983 y 1986 desarrolló el sistema NoteCards. En 1986 se traslada a MCC para trabajar en el proyecto HyperActivity, que consigue la integración del hipermedia con la ingeniería del software.

Por su formación en psicología, Halasz se interesó por los aspectos relacionados con los usuarios y, en general, con la interacción hombre-máquina. En la conferencia de ACM Hipertexto-87 presenta su famoso artículo [Halasz, 1988b], en el que se indentifica siete requisitos básicos para la siguiente generación de sistemas hipermedia: mejorar las capacidades de búsqueda, habilidad para definir com-

posiciones, estructuras virtuales —dinámicas—, computación en redes hipermedia, control de versión, soporte para la colaboración y, finalmente, extesibilidad y personalización. En el 1991 Halasz revisa y rehace estos siete aspectos y propone: acabar con la tiranía del enlace, sistemas abiertos, soporte para trabajo en grupo, interfaces de usuario para grandes espacios de información, hipermedias de gran complejidad y extensibilidad y personalización.

Halasz también es reconocido por ser junto a Mayer Schwartz el autor del Modelo de Referencia Dexter [Halasz, 1990], el cual recoge algunas de las estructuras y características estándar para sistemas hipermedia.

3.1.7 Intermedia (1985)

Intermedia [Meyrowitz, 1986] [Yankelovich, 1988] es un sistema hipermedia más avanzado y fue desarrollado por la Universidad de Brown. Como producto comercial presenta el inconveniente de haberse implementado para el sistema operativo A/UX —UNIX— para Macintosh. Al igual que Guide y NoteCards, Intermedia soporta ventanas con *scrolling*, por tanto, el tamaño de un nodo no está limitado por las dimensiones de la ventana.

En este hipermedia destaca el protocolo de enlace, que permite integrar otras aplicaciones dentro del documento. Los enlaces son siempre bidireccionales, y conectan anclajes —*anchors*— en vez de nodos. Ofrece dos tipos de vistas generales para facilitar la navegación del usuario:

- Nodos panorámicos —*Overview nodes*—: estos nodos muestran una parte fija de la estructura del documento. Se podría comparar con un menú en el que se indican los componentes más significativos que están relacionados con el nodo, por ejemplo, los enlaces y vecinos más próximos. Estos nodos se construyen manualmente.
- La vista web: consiste en una visión general gráfica que se crea automáticamente a partir de la estructura de enlaces del hiperdocumento completo.

3.1.8 KMS (1988)

Donald McCracken y Robert Akscyn, los principales diseñadores del sistema ZOG de la CMU, crearon la compañía Knowledge Systems en 1981 y produjeron el KMS —Knowledge Management System— [Akscyn, 1988]. KMS se ejecuta en estaciones Sun y Apollo. Emplea la misma interfaz basada en marco —*frames*— que ZOG, con el mismo peligro de desorientación. Sin embargo, KMS fue mejorado de tal forma que, según sus desarrolladores, este problema se reduce debido a la rapidez con la que el usuario se podía mover entre los marcos.

Este sistema distinguía entre dos tipos de enlaces: jerárquicos y de referencia trasversal. La diferencia entre ambos está en que el seguimiento de los segundos significa saltar a otra parte del hiperdocumento. KMS se diseñó para la colaboración entre usuarios. No hay distinción entre lectores y autores, y todos pueden realizar cambios que son inmediatamente visibles al resto de los usuarios. Este sistema ignora el riesgo de que dos usuarios quieran al tiempo cambiar el mismo marco, esto es, no prohíbe el acceso a un marco cuando éste se está editando. La aparencia de esta aplicación difiere de la del resto de los sistemas debido a la estructura casi estrictamente jerárquica de los hiperdocumentos, de modo que, los enlaces son presentados separados del cuerpo del marco.

3.1.9 Guide (1986)

Peter Brown comenzó Guide [Brown, 1991] como un proyecto de investigación en hipermedia en la Universidad de Kent y Caterbury en 1982. La primera versión se ejecutó sobre una estación PERQ en 1983. La empresa Office Workstation Ltd —OWL— se interesó por Guide y se comenzó a desarrollar una versión para Macintosh en 1984.

OWL-Guide se revisó en 1986 y se convirtió en el primer sistema hipermedia comercial de propósito genérico. Más tarde se portó también al IBM-PC y fue el primer sistema disponible tanto para Mac como para PC.

Este sistema no requiere ninguna programación para utilizarlo, y se basa en cuatro tipos de botones: anotación, referencia, orden y reemplazo. Los botones de anotación crean una pequeña ventana con un corto mensaje temporal; los de referencia establecen un enlace entre dos documentos a partir de un elemento común; los de orden permiten ejecutar un programa externo y los de reemplazo provocan la sustitución de un texto o gráfico por otro más detallado. La mayoría de los sistemas hipermedia usan *paginación* cuando se sigue un enlace, esto quiere decir que el nodo que se está mostrando se sustituye por el nodo destino. En Guide el mecanismo de enlace se basó en el *reemplazamiento*. Esto es, cuando se sigue un enlace se abre un hueco para ubicar ahí el nuevo nodo. Por tanto, el anclaje se sustituye por el contenido del nodo destino. También se puede cerrar el nodo destino reemplazándolo de nuevo por el anclaje. En consecuencia, la estructura del hiperdocumento debe ser estrictamente jerárquica.

3.1.10 HyperTIES (1986)

HyperTies se inició como un proyecto de investigación en la Universidad de Maryland por Ben Shneiderman [Shneiderman, 1987] alrededor de 1983. Tenía como objetivo construir The Interactive Encyclopedia System —TIES—, pero más tarde le cambió el nombre por HyperTies por un conflicto de marcas.

Figura 1.9 Ben Shneiderman.

La característica clave de HyperTies es la simplicidad [Shneiderman, 1987]. Desde 1987 ha sido distribuido comercialmente por IBM-PC. Este sistema emplea una pantalla plana de texto y no necesita el ratón para la interacción. Los anclajes están resaltados y el usuario puede moverse entre ellos mediante las teclas de los cursores. Cuando se sigue un enlace se sustituye el nodo actual por el destino. Cada nodo llena por completo la pantalla, con la excepción de unas pocas líneas finales, que se utilizan para información adicional, y botones de paginación. Un nodo puede estar formado por muchas páginas de texto que se pueden ver moviendo

hacia arriba o abajo el cursor. Al situarlo sobre un anclaje, HiperTies muestra una breve descripción del destino. Esto le permite al usuario evaluar si es de su interés el contenido del nodo destino.

En HyperTies todos los enlaces llevan a la primera página del nodo destino. Cuenta con una tabla —base de datos— con los pares anclaje-destino. En consecuencia la primera frase de un nodo puede aparecer como anclaje en distintos nodos, y siempre conduce a la primera página del mismo. Esta mecanismo facilita la edición pero lo hace menos flexible.

3.1.11 HyperCard (1987)

En 1987 Apple introduce HyperCard [Apple, 1989] y lo distribuye libremente con cada Macintosh, haciéndose uno de los sistemas hipermedia más populares. Sin embargo, esta aplicación no fue diseñada originalmente como hipermedia.

La unidades básicas en HyperCard son la tarjeta —*card*— y la pila. Este sistema permite establecer enlaces entre las tarjetas y las pilas sin ningún tipo de restricción de jerarquía o estructura. En una tarjeta pueden aparecer tanto texto como botones. Quizás, la principal contribución de esta herramienta es la posibilidad de construir estos botones rectangulares sobre la pantalla y asociarlos a un *script* escrito en HyperTalk, un lenguaje de programación inspirado en SmallTalk. En la mayoría de los casos el programa únicamente consiste en una instrucción *goto*.

El lenguaje HyperTalk es de propósito general y muy fácil de aprender. Estaba dirigido a realizar prototipos de interfaces gráficas de usuario. El destino de un enlace es simplemente el resultado de ejecutar uno de estos programas. Esto significa que es posible enlazar un programa que conduzca a diferentes destinos dependiendo del contexto del usuario. Por último, HyperTalk contaba con órdenes de visibilidad y ocultamiento que permiten hacer a los campos de una tarjeta visibles o invisibles.

3.1.12 World Wide Web (1992)

World Wide Web —o WWW o Web— [Berners-Lee, 1992] [Berners-Lee, 1994] es un hipermedia enorme que está compuesto por miles de servidores de información distribuida por todo el mundo en máquinas Internet. El World Wide Web Consortium —W3C— proporciona una gran cantidad de información sobre la tecnología WWW y sus últimos desarrollos. Aunque no supone ningún avance teórico, sin duda alguna, ha contribuido decisivamente en la divulgación mundial de las técnicas hipermedia.

WWW comenzó como un proyecto en el CERN —*Centre Européenne pour la Recherche Nucléaire*— en Suiza. Tim Berners Lee inventó el Web como medio de comunicación para los físicos a través de Internet. La idea era almacenar en los servidores los artículos científicos y otros documentos con el fin de facilitar el trabajo en común.

La interfaz de usuario original de Web fue desarrollada para el computador Next. La Web se hizo popular cuando Marc Andreesen, estudiante graduado en el NCSA —National Center or Supercomputing Application— de la Universidad de Illinois, crea una nueva interfaz denominada Mosaic. Parte del equipo de desarrollo de Mosaic dejó la universidad para crear su propia empresa Netscape Communications. Su nuevo navegador, Netscape Navigator, rápidamente superó a Mosaic.

La Web está basada en una arquitectura cliente-servidor sobre Internet, mediante el uso del protocolo HTTP —*HyperText Transfer Protocol*— [Fielding, 1997]. Con este protocolo, los clientes presentan al usuario documentos —páginas Web— basados en el lenguaje de marcas HTML —*HyperText Markup Language*—, que obtienen de los servidores. Por tanto, para ofertar la informa-

ción sobre Web se debe arrancar un servidor HTTP. Los documentos se enlazan entre sí mediante la inclusión de anclajes embebidos en la página. Estos anclajes contienen la dirección URL —*Universal Resource Locator*— del documento o página destino o un llamada parametrizada a una rutina CGI que produce en tiempo de ejecución el documento destino. URL es un nombre único que identifica una parte de un nodo independientemente del lugar del mundo en el que se encuentre. Consiste en cinco partes: el protocolo de red que está usando para recuperar el nodo, la dirección completa Internet de la máquina servidora —*host*— en la que reside el nodo, el número de puerto que usa, el nombre —el camino— completo del nodo y una indicación opcional de la parte del nodo sobre la que está interesado. La sintaxis de URL en WWW es: *protocol:// host:port/nodename#anchorname*.

Figura 1.10 Tim Beners-Lee.

Desde su aparición, la comunidad hipermedia viene manifestando las lagunas e incosistencias teóricas y prácticas de la web. Autores como Bieber [Bieber, 1997b] demandan a los desarrolladores de WWW que incorporen las características de alto nivel que contemplan lo que ellos denominan los hipermedia de cuarta generación; «*environments on the Web that would support collaborative authoring [Schackelford, 1993], authoring and guiding users through multitask processes [Noll, 1996] [Scacchi, 1989] [Tanik, 1996], group decision making and collaborative work [Hiltz, 1996] [Streitz, 1992] [Turoff, 1991], personal and group organizing tools [Marshall, 1995], and enterprise coordination [Engelbart, 1990]. While each of these environments has its own core set of functionality, the same types of hypermedia features, such as annotation, overviews, paths and guided tours, would serve users in all of them*».

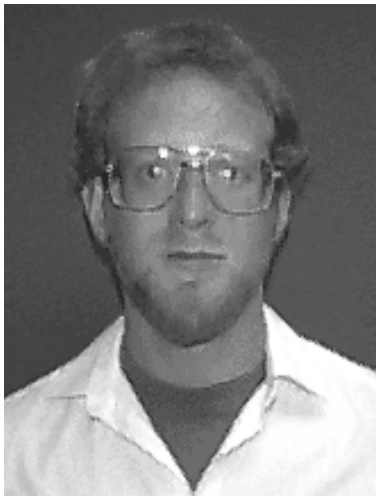


Figura 1.11 Michael Bieber.

4 Modelo Básico Hipermedia

Desde un punto de vista abstracto, todo sistema hipermedia se puede estudiar bajo un modelo básico [Rivlin, 1994]. Éste se puede dividir en dos submodelos distintos pero interdependientes —figura 1.12—.

El primero es el submodelo de datos en el que los nodos se interconectan con enlaces directos, formando una estructura de grafo dirigido. Este subsistema admite como operaciones válidas la adición, borrado, y actualización de nodos y enlaces. El submodelo de proceso es el segundo elemento del sistema que se está describiendo. La principal preocupación de este submodelo será proporcionar mecanismos de acceso a la información.

Este modelo que se acaba de presentar se caracteriza por su generalidad, flexibilidad e incompletitud. Por ejemplo, E.F. Codd [Codd, 1981] define un modelo de datos como una combinación de estructura de datos, operaciones y reglas de integridad. En este sentido, el modelo de datos hipermedia es incompleto, pues no define ninguna limitación para determinar la consistencia de la información de la red —por ejemplo, es válida la inserción de un enlace sin asociarlo a un nodo destino—. Por otro lado, el submodelo de proceso es muy primitivo, no se detallan especificaciones para definir la navegación de acceso —como el modo de activar un enlace por parte del usuario, por ejemplo—.

Submodelo de Datos

Como se ha visto, los nodos son las unidades primitivas que organizan la información en una red hipermedia. Los nodos funcionan como colecciones de datos no estructurados que se unen con el objetivo de crear una entidad lógica. El propósito de esta entidad lógica debería proporcionar una abstracción de información coherente dentro del espacio de información existente. Por ejemplo, en un sistema hipermedia sobre la vida de Leonardo de Vinci, un nodo puede combinar una descripción en texto, un relato hablado, una imagen y un video sobre su lugar de nacimiento, Vinci -cerca de la ciudad de Florencia-. Los enlaces son otras de las unidades fundamentales de submodelo de datos. Éstos implementan las interconexiones directas entre nodos. Aunque algunos sistemas soportan también enlaces bidireccionales, no es lo más normal.

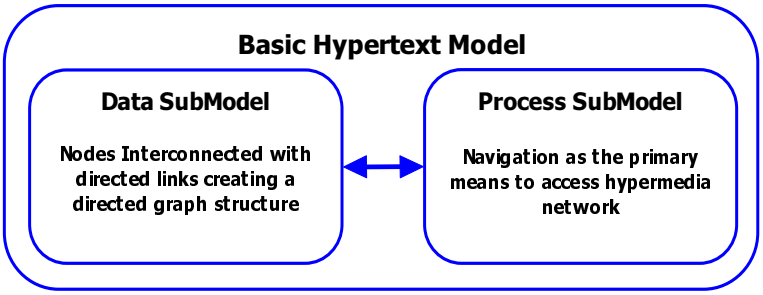


Figura 1.12 Modelo básico hipermedia [Salampasis, 1997].

Submodelo de Proceso

El concepto de hipermedia no se reduce únicamente a organizar los datos multimedia interconectándolos con nodos. La navegación, que es el medio que permite al usuario moverse a través de la red hipermedia activando y siguiendo los enlaces, es otra de las características que definen un hipermedia. La navegación es el medio fundamental de acceso a la información en una red hipermedia, y supone el aspecto esencial del submodelo de proceso dentro del modelo básico de hipermedia. En principio este mecanismo de acceso debería ser lo suficientemente rico e intuitivo para que el usuario pueda navegar según su propia motivación, sin necesidad de ninguna ayuda externa. Algunos sistemas incluyen dentro de la funcionalidad de navegación la posibilidad de regresar al nodo visitado anteriormente, o moverse al primero de los nodos. Ambas características forman parte del submodelo de proceso.

5 Características Generales de los Sistemas Hipermedia

Casi todos los sistemas hipermedia que se construyen obedecen, de un modo u otro, al modelo básico que se acaba de presentar. Sin embargo, a medida que el concepto hipermedia se ha hecho popular, se han ido proponiendo extensiones que enriquecen y mejoran este modelo rudimentario. En esta sección se intentan identificar algunos de estos nuevos aspectos básicos relacionados con los sistemas hipermedia poniendo especial énfasis en la navegación.

En el esquema —figura 1.13 [Salampasis,1997]— de la siguiente página se hace una representación de este marco conceptual. Se puede ver cómo el proceso de diseño depende mucho del dominio al que se aplique el hipermedia, de la funcionalidad que le exija y, por último, de las herramientas de edición con las que se cuente. Por otro lado, tal y como se comentará en los siguientes apartados, dentro del modelo básico hipermedia el concepto de navegación ocupa un papel primordial.

5.1 Arquitectura de los Sistemas Hipermedia

«Un sistema hipermedia es un entorno que permite el desarrollo de aplicaciones hipermedia basándose en un modelo hipermedia dado» [Usandizaga, 1998], que caracteriza la información y determina cómo se puede organizar, presentar y navegar. En la arquitectura general de un sistema hipermedia se pueden identificar las siguientes características básicas:

- Un **Sistema de Almacenamiento** que guarda todas las piezas de información o nodos y, en algunos casos, los enlaces que los relacionan. Este depósito de información puede ser un sistema de ficheros o una base de conocimiento, o un sistema de gestión de bases de datos relacionales, o un sistema de gestión de bases de datos orientadas a objetos.
- Un **Sistema de Autor**, con herramientas para crear y administrar nodos —de múltiples medios— y enlaces y, por tanto, para construir los hiperdocumentos.
- Un **Motor Hipermedia** para administrar la información sobre nodos y enlaces.
- Una **Intefaz Gráfica de Usuario**, con la ayuda de navegadores/buscadores —*browsers*— y diagramas generales —*overview diagrams*—. Permiten al usuario navegar a través de grandes cantidades de información, activando enlaces y animando el contenido de los nodos.
- **Mecanismos de recuperación** de información tradicionales tales como búsquedas con palabra clave, búsquedas de autor, etc. También se incluyen preguntas sobre la estructura junto a las preguntas sobre el contenido, recuperando parte de la red hipermedia basada en algunos criterios fijados por el usuario.

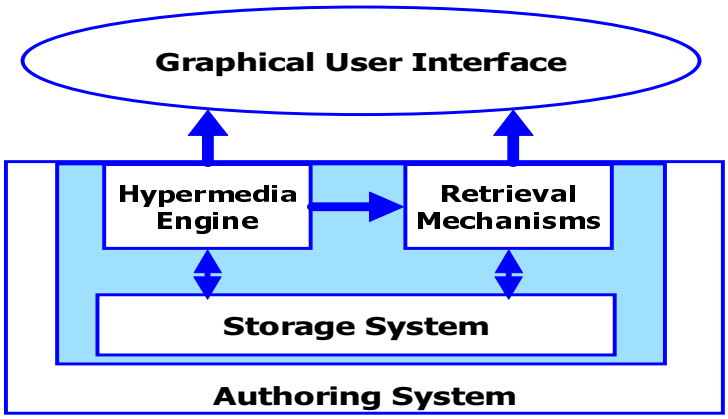


Figura 1.14 Arquitectura básica de un Sistema Hipermedia.

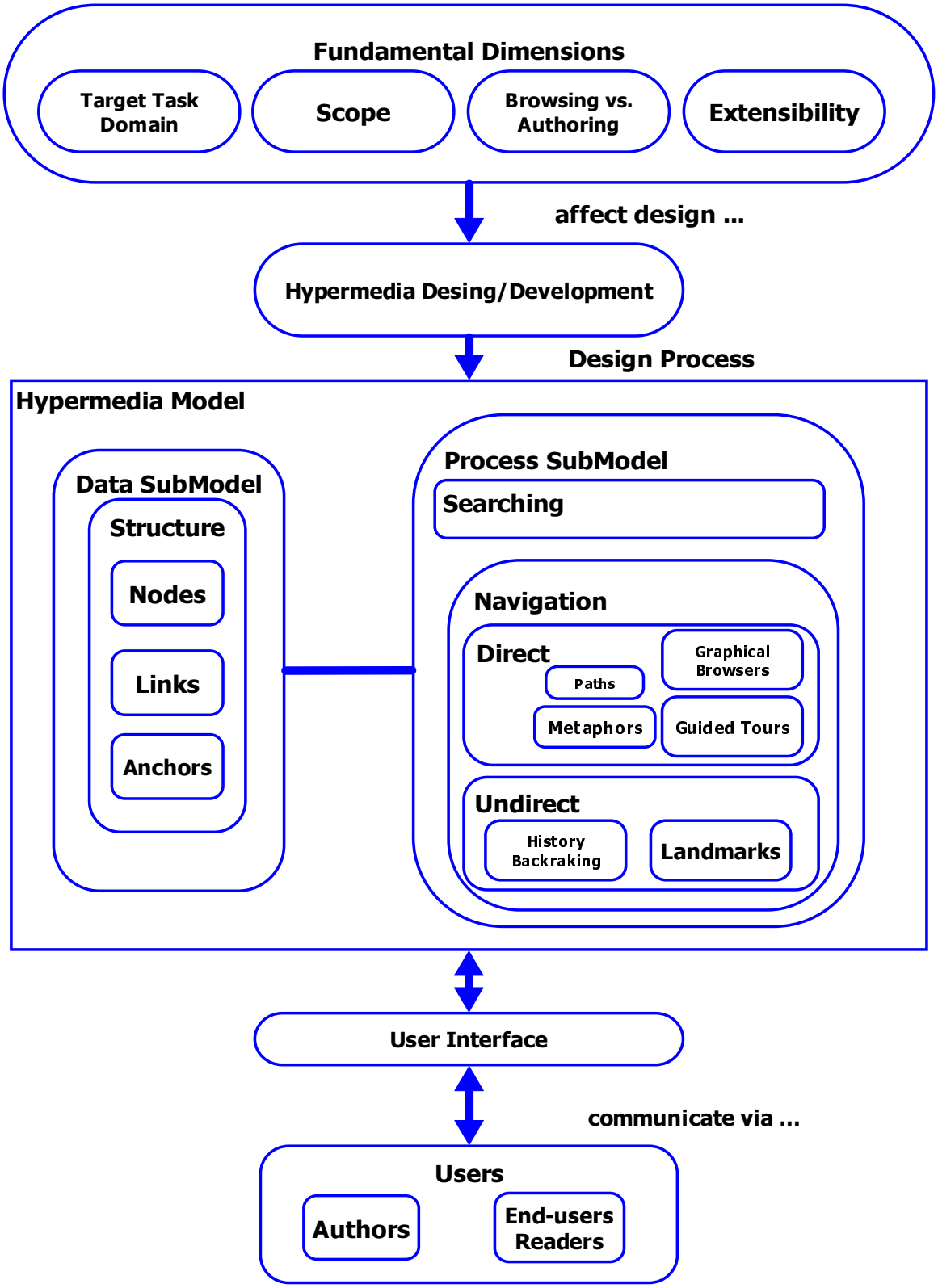


Figura 1.13 Entorno conceptual de los Sistemas hipermedia [Salampasis, 1997].

Como se ve en la figura 1.14, el motor hipermedia y el mecanismo de recuperación se dedican a mostrar y recorrer la información a través de la interfaz gráfica de usuario. El sistema de autor permite construir el sistema de almacenamiento, el motor hipermedia y el mecanismo de recuperación. En realidad, los sistemas de autor suelen ser mucho más rígidos: sólo permiten construir una estructura a partir de sistemas de almacenamiento, motor y recuperación ya predefinidos. En consecuencia, la mayoría de los sistemas de autor no consideran el mantenimiento de los hiperdocumentos ni tampoco preven la posibilidad de que el autor quiera diseñar y cambiar su representación.

5.2 Navegando en Hipermedia

Según Ziegler [Ziegler, 1997] el término navegación se podría definir como *la interacción que realiza el usuario con el fin de moverse entre distintas partes de un documento, conjunto de datos o piezas de información enlazadas arbitrariamente*. Por tanto, ésta es la característica idiosincrática por excelencia de un sistema hipermedia. La navegación hace de los sistemas hipermedia herramientas poderosas para la administración de la información. Sin embargo, este tipo de interacción no es algo que ocurre sin problemas. Por el contrario, muchas de las dificultades a las que se enfrenta el sistema hipermedia están relacionadas con esta característica, como son la desorientación y el conocimiento *overhead*. En esta sección se verá que si bien la navegación es la fuente de todas sus cualidades, paradójicamente también es el origen de muchas barreras e interesantes desafíos que están por superar.

En un libro es posible saltar aleatoriamente las páginas y leer el material en cualquier orden. Por supuesto, siempre se sabe dónde se está más o menos, porque se puede ver por completo el libro. Además, las páginas están numeradas, e indican exactamente donde se encuentra. Sin embargo, seguramente uno no entenderá lo que está leyendo a no ser que siga un orden normal de lectura secuencial. Los autores asumen que al llegar a esa página ya leyó, y recuerda, las páginas precedentes.

Por el contrario, en un hipermedia se pretende facilitar una lectura exploratoria. Se pueden seguir los enlaces que el lector considere oportunos, para que siempre tenga la posibilidad de informarse de algo que no se explica en la página que está consultando. Desgraciadamente existe el riesgo de que el usuario se desoriente. Algunas investigaciones experimentales demuestran que hasta en un documento de pequeño tamaño el lector acaba perdiéndose en el hiperespacio [Nielsen, 1990].

Conocimiento *Overhead*

En la legibilidad de un hiperdocumento, el conocimiento *overhead* influye de forma negativa. Se puede definir el conocimiento *overhead* como el esfuerzo adicional y la concentración necesaria para mantener muchas tareas o seguir rastros al mismo tiempo. Cada esfuerzo adicional del lector reduce los recursos mentales utilizables para la comprensión. Este trabajo añadido está relacionado con la *orientación*, *navegación* y *ajuste a la interfaz de usuario*, cuando el lector trabaja con el hiperdocumento.

Perdidos en el Hiperespacio

Existen numerosas investigaciones sobre el fenómeno de la desorientación [Nielsen, 1990]. Este síndrome es una combinación de dos problemas:

- cuando uno se siente interesado por un nodo, rápidamente decide pasar a leerlo porque teme no encontrarlo más tarde,
- mientras se navega -realizando seguramente varios saltos entre nodos- pierde la noción de en qué parte o punto de un documento se encuentra.

Ambos problemas son muy parecidos a lo que nos pasa en la vida diaria cuando exploramos una determinada área geográfica como, por ejemplo, visitar una ciudad. Pero en este medio es más fácil volver a orientarse porque nos encontramos en un espacio bidimensional, y todas la distancias son euclídeas. En teoría siempre es posible mantener la pista de dónde se está y de cómo volver al punto de partida. En un hiperdocumento las distancias no son *normales*: todos los enlaces son de igual longitud. No existen ángulos o esquinas y, además, un nodo puede tener muchos enlaces de salida. No hay modo de saber si un enlace está cerca de un nodo que has visitado anteriormente. Si se intenta dibujar la estructura de un hiperdocumento como un grafo en el que todos los enlaces son de la misma longitud, llegará un momento en el que dos dimensiones serán insuficientes. Para paliar estos problemas se han ideado algunos mecanismos de ayuda.

La navegación del usuario a través de un hiperdocumento se ve influenciada por tres aspectos fundamentales:

- la estructura del hiperdocumento,
- las ayudas para la navegación que proporciona el sistema hipermedia, y
- la estrategia de búsqueda que emplee el usuario.

Todos estos factores han sido estudiados en la literatura, en concreto los experimentos de navegación de De Vocht [De Vocht, 1994] estudian la interacción entre los tres.

Una aproximación más formal al problema de la desorientación consiste en estudiar los posibles caminos que se pueden seguir en un hiperdocumento. Esto se conoce con el nombre de semánticas de búsqueda —*browsing semantics*—.

5.2.1 Estructura de los Hiperdocumentos

La navegación se puede diseñar de acuerdo a un número básico de estructuras que determinan los caminos de diálogo potencial que son posibles en un determinado sistema.

La forma más flexible de navegación es la que proporcionan las estructuras en red, las cuales representan el principal paradigma para los sistemas hipermedia [Nielsen, 1990]. Pero toda red arbitraria de información termina por ser confusa, incluso cuando cuenta con un número reducido de nodos. Se han estudiado varios mecanismos para proporcionar estructuras adicionales, como por ejemplo agrupar nodos o distinguir entre varios tipos de enlaces. Otros mecanismos que facilitan la compresión y mejoran la orientación son las jerarquías, las agregaciones, las redes semánticas y las redes de Petri.

La aproximación que más se emplea es imponer sobre una red de enlaces libres, una estructura regular, como una jerarquía o una pila de secuencias preestablecidas. Proporcionar estructuras transparentes es la manera más adecuada de prevenir la desorientación del lector. En este sentido, las denominadas ayudas de navegación mejoran el diseño y la comprensión de la estructura del documento. Los experimentos de De Vocht [De Vocht, 1994] investigan hasta qué punto un usuario llega a conocer la estructura de un hiperdocumento mientras navega. El resultado de los mismos demuestra que, depende mucho de la estrategia de navegación que use el usuario, y de las ayudas que oferta el sistema, al menos para muchas de las estructuras actuales.

Kibbly [Kibbly, 1990] y Mayes [Mayes, 1994] identifican dos estructuras paralelas en una red hipermedia. Una de ellas proporciona el conjunto de nodos y enlaces explícitos. A esta estructura

Experimentos de Navegación

De Vocht ha realizado un número de experimentos relacionados con la navegación, en los que se combinan diferentes estructuras de hiperdocumento, ayudas y estrategias de navegación. El propósito de estos experimentos es comprobar qué estructuras, ayudas y estrategias son las mejores.

Aunque el rango de posibles combinaciones es bastante amplio, los experimentos llegan a conclusiones bastante claras, útiles para autores, lectores y constructores de sistemas hipermedia:

- Las estructuras que mejor trabajan para todas las combinaciones de ayudas y estrategias de navegación, tanto en sesiones de búsqueda largas como cortas, son aquellas que emplean muchos enlaces de referencia trasversal -*cross-reference*-, o que tienen una estructura jerárquica.
- Los *highlighting links* o enlaces resaltados facilitan el trabajo porque aumentan la eficiencia de la búsqueda en todos los casos.
- La estrategia de navegación *primero-profundidad* es la mejor en todos los casos, excepto para sesiones de búsqueda cortas en documentos con pocos enlaces de referencia trasversal.
- La ayuda *fish-eye* no se consideró en los experimentos porque el objetivo era intentar descubrir la estructura de un hiperdocumento distribuido, en los que esta facilidad no es posible -únicamente se conoce la estructura de un lugar-.
- La ayuda de las *migajas* tampoco se estudió, porque estos experimentos asumen que el lector reconoce un nodo que ya ha sido visitado sin necesidad de dejar marcas.

Estructuras Adicionales

Nodos y enlaces forman una red que es un grafo dirigido. Dependiendo del dominio y ámbito de información que queramos representar en un sistema hipermedia, esta estructura tan simple podría ser inadecuada para la navegación, orientación y comprensión del documento. Estructurar el hiperespacio con algo más que grafos puede afectar, directa o indirectamente, a la navegación, mejorar la orientación, y ayudar a los usuarios a asociar y comprender mejor el hiperdocumento.

se le denomina *hiperespacio*. La otra hace referencia al *espacio conceptual* que se bosqueja a partir del conocimiento explícito e implícito presente en los nodos, enlaces y todas las posibles asociaciones entre la información. Las herramientas de navegación como los *browsers* ayudan al usuario a explorar el hiperespacio, pero ignoran el espacio conceptual que se desarrolla en paralelo. El espacio conceptual es mucho más complejo que el hiperespacio porque se basa en el conocimiento implícito. El hiperespacio tan sólo tiene que buscar y localizar información, mientras que el espacio conceptual tiene que conseguir la comprensión de esta información. Esta visión de los hipermedia, que ahora mismo es objeto de investigación —por autores como Hammond [Hammond, 1987] y Kibby [Kibby, 1990]—, puede representar un factor crucial en sistemas relacionados con la enseñanza-aprendizaje.

5.2.2 Ayudas para la Navegación

Se han desarrollado varias formas de ayuda para la navegación que asisten a los usuarios en la tarea de encontrar la información que están buscando, procurando evitar su desorientación dentro del hiperdocumento. Algunas de estas son:

Marcha atrás —*Backtracking*—. En la mayoría de los sistemas hipermedia los enlaces son unidireccionales. Esto significa que tienen un origen y un destino. Los anclajes origen son visibles, y se usan para activar, a través del enlace, el destino. Dentro del nodo destino el anclaje final no es visible, y no se puede usar para volver hacia atrás. Para subsanar este problema casi todos los navegadores ofrecen un botón de vuelta atrás —*back*— que permite regresar al último nodo que se visitó. También es útil un botón de marcha hacia adelante —*forward*— que devuelva al lugar de procedencia.

Chivatazo —*Sneak preview*—. El modo usual de activar un enlace consiste en *disparar* con el botón del ratón un anclaje. Estos anclajes son palabras o frases y ofrecen muy poca información sobre el destino. Algunos sistemas anticipan algún tipo de información antes de seguir el enlace. En HyperTies, por ejemplo, se muestra una pequeña descripción del nodo destino cuando se posa el cursor sobre el anclaje. En Mosaic se ofrece la URL —*Universal Resource Locator*— completa del destino.

Enlaces resaltados —*Highlighting links*—. Los navegadores muestran de forma diferente los enlaces que apuntan a nodos que ya han sido visitados a los enlaces que aún están sin estrenar. El medio para indicar esto puede ser emplear un color distinto o algún tipo diferente de estilo como subrayar la palabra.

Estructuras Jerárquicas

Es la forma más común de organizar la información. Los nodos se estructuran en árboles jerárquicos y proporcionan al usuario un conjunto de enlaces semi-explícitos [Botafogo, 1992]. El propósito de esta agrupación es suministrar a los lectores un conocimiento de la estructura del hiperdocumento, para que consigan un acceso más rápido a la información. Este mecanismo tan simple facilita la orientación, y mejora las tareas de escribir y leer el documento.

Estructuras basadas en Agregaciones o Conjuntos

La agregación es una operación que agrupa un conjunto de nodos relacionados para formar un objeto de más alto nivel. Este objeto se puede manejar como un único nodo. El uso de agregaciones [Parunak 1991] ayuda a los usuarios a tener un conocimiento a base de abstracciones sobre el hiperdocumento. Organizar de esta forma la información permite asistir al lector de acuerdo a intereses particulares. Por ejemplo, en un hipermedia sobre geografía europea, se puede comunicar al sistema cierto interés por los países del Este.

Redes Semánticas

Un hipermedia se puede ver como una red semántica [Wang, 1995]. Un enlace entonces debe suministrar información acerca de la pareja de nodos que interconecta, y por tanto, un nodo puede tener diferentes contenidos cuando participa de relaciones diferentes.

Redes de Petri

Aplicar el modelo de redes de Petri a un hipermedia permite desarrollar los sistemas de navegación y edición que pueden incorporar técnicas analíticas. Por otro lado, las redes de Petri permiten una especificación poderosa sobre cómo se explora el hiperdocumento. Estas nuevas habilidades incluyen sincronización de caminos simultáneos a través del documento considerando control de acceso y seguridad. El sistema más conocido basado en redes de Petri es el Trellis [Stotts, 1991].

Anclajes únicos —*Unique anchors*—. En un hipermedia se puede emplear el mismo anclaje —la misma palabra o frase— para apuntar a distintos nodos. Esto crea una enorme confusión. Algunos sistemas como HyperTies prohíben el uso de la misma palabra situada en distintos puntos del hiperdocumento para apuntar a diferentes nodos.

Migajas —*Bread crumbs*—. Cuando se está navegando a través de un entorno desconocido siempre se tiene el temor de perderse. Una técnica bastante común presente en muchos cuentos e historias de intriga, que consiste en dejar marcas tras de sí, para luego encontrar el camino de regreso. Esta técnica asume que el usuario cuenta con escasa memoria para recordar dónde estuvo. Por otro lado, algunos experimentos demuestran que aquellos nodos que se ilustran con figuras son más fáciles de reconocer y recordar sin necesidad de emplear esta facilidad.

Lista histórica —*History list*—. Aunque los sistemas ofrecen enlaces para retornar al nodo anterior, siempre es bueno poder volver a repetir algún camino realizado; entre otras cosas porque, de esta manera, el lector comprenderá en qué punto del hiperdocumento se encuentra. Esta es la razón que lleva a los navegadores a ofrecer una lista con una entrada, junto a su dirección, por cada nodo visitado.

Señaladores —*Bookmarks*—. Mientras se lee un documento se pueden señalar algunas páginas para volver posteriormente sobre ellas. En un hiperdocumento esta necesidad se acrecienta por dos razones, porque:

- en el empeño por localizar algo muy concreto, rápidamente se deja sin consultar otro tipo de información que, seguramente, se desea volver a visitar,
- muchas veces la búsqueda obedece a un tanteo sobre algún asunto; con este vistazo se pretende reunir nodos en principio interesantes que posteriormente pueden formar parte de una *hiperteca* en particular.

Vistas —*Fish-eye views*—. Este tipo de herramientas intentan visualizar la estructura del hiperdocumento. Se pueden ofrecer dos tipos de representaciones de esta organización:

- Una que genera un grafo general de todo el hiperdocumento. Este esquema normalmente será demasiado grande para poderlo mostrar en una pantalla. Una ventana con *scrolling*, o un *zoom*, serán algunas de las técnicas que se emplearán para moverse por toda la estructura del documento. A este tipo de vistas generales se les denomina **birds-eye views**.
- Otra posibilidad, más razonable, será mostrar un grafo plano que muestre con detalle la estructura del nodo actual, y de forma somera el resto, dependiendo de la distancia a este nodo. A esta vista, un poco distorsionada, se le conoce con el nombre de **fish-eye view** [Furnas, 1986]. El principal problema que plantea esta ayuda es decidir qué se debe mostrar con detalle y qué se debe suavizar, esto es, qué nodos forman parte de un supernodo o nodo compuesto, y cuáles no.

  En la figura 1.15 se muestra un ejemplo de un mismo documento representado con estas dos tipos de técnicas. En la figura de la izquierda se detalla todo el grafo mientras que en la figura de la derecha se simplica agrupando nodos que no son relevantes en ese momento.

Rutas guiadas —*guided tours*—. La navegación a través del hiperdocumento puede conducir fácilmente a la desorientación. Cuando se está leyendo sobre algún tema, normalmente uno se siente interesado por informarse sobre algunos aspectos relacionados. Pero también es muy importante encontrar un orden lógico que dé sentido al conjunto de nodos que se está consultando.

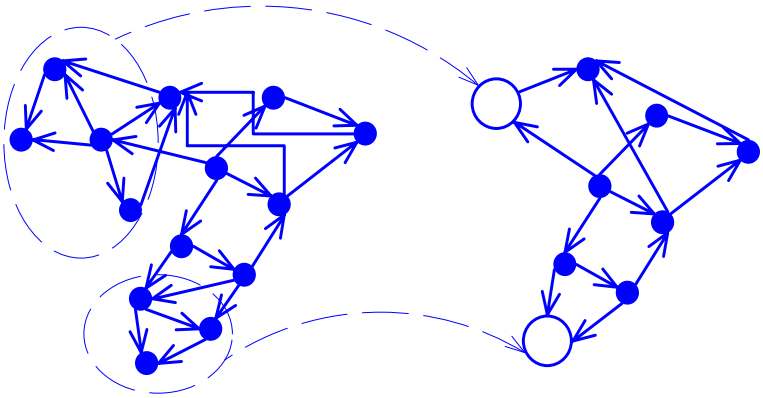


Figura 1.15 Vista detallada y resumida de un hipermedia representado mediante un grafo.

Estas rutas, sugeridas por Bush [Bush, 1945], se pueden ver como una clase de superenlace que conecta una serie de nodos. Cuando uno decide seguir uno de estos caminos guiados, tan sólo debe activar el anclaje *del siguiente nodo*.

Las rutas guiadas son muy útiles para los sistemas que proporcionan información sobre diferentes temas como por, ejemplo, la que ofrece un museo a sus visitantes.

Sin embargo, este tipo de mecanismo es difícil de mantener en hiperdocumentos que están sujetos a cambios. Y, por otro lado, son insensibles a los deseos de los usuarios, que pueden manifestar interés por algún tema sobre el que el autor no ha creado ningún recorrido. Guinan y Smeaton [Guinan, 1992] diseñaron un algoritmo que crea dinámicamente rutas guiadas que responden a los intereses de los lectores. Cuando se pregunta sobre alguno de los nodos que contiene información sobre algún tema, el algoritmo genera un viaje que contiene todos los nodos en orden lógico que responde a esa inquietud.

La mayoría de los sistemas hipermedia ofrecen unas cuantas de estas facilidades. El conocimiento de las ayudas para la navegación que están disponibles puede determinar el número y tipo de enlaces que los autores van a construir en un hipermedia.

5.2.3 Estrategias de Búsqueda

El comportamiento de los usuarios frente a un hiperdocumento no sólo depende de su interés y de la información que se encuentra en los nodos. Individuos diferentes siguen estrategias distintas. Por ejemplo, hay unos que adoptan una actitud más prudente y otros más desafiante. Los primeros son aquellos que exploran los vecinos de un nodo antes de comenzar a moverse en profundidad. A este tipo de navegación se le denomina *primero-anchura —breadth-first—*. La conducción atrevida dentro de un hiperdocumento se caracteriza por avanzar cada vez más lejos siguiendo todos los enlaces que encuentra a su paso, únicamente regresa cuando se topa con un nodo sin enlaces de salida. A esta estrategia se la conoce como *primero-profundidad —depth-first—*.

De Vocht [De Vocht, 1994] distingue, además de estás, otras posibles estrategias de búsqueda:

- Aleatoria: Cuando los nodos se visitan en cualquier orden. Esto significa que el usuario está construyendo una posible lista de nodos interesantes para luego visitar y juzgar más despacio.
- Unas cuantas variantes basadas en las anteriores estrategias, que se diferencian en el número de enlaces que siguen por nodo y en qué dirección:

- Estrategia 1: utiliza una proporción fija del número de enlaces de salida por nodo.
- Estrategia 2: utiliza un número fijo de enlaces de salida por nodo.
- Estrategia 3: utiliza un número aleatorio de nodos de salida por nodo.
- Estrategia 4: combina las estrategias de primero-profundidad y primero-anchura, de tal forma que para cada nodo avanza cierta distancia a partir de él y, luego, explora en anchura desde ahí.

5.2.4 Semánticas de Búsqueda

Con el fin de determinar los posibles problemas que el usuario puede experimentar durante la navegación, se han estudiado las semánticas de búsqueda formalmente [Stotts, 1989] [Aalst, 1993]. Esta investigación estudia todos los caminos que son posibles a través de un hiperdocumento.

Si se parte de un modelo hipermedia simple que cuenta únicamente con nodos y enlaces, entonces los posibles caminos serán tan sólo unas secuencias de nodos. Pero si asociamos nombres a los nodos, las secuencias forman un lenguaje simple que se puede analizar formalmente. Este lenguaje se puede expresar con un autómata finito, cuyos estados y transiciones se corresponden uno a uno con los nodos y los enlaces del hiperdocumento. El estado del autómata representa el nodo actual.

Los problemas de navegación pueden entonces reducirse a las propiedades de un autómata. Un interbloqueo en un autómata se corresponde con un nodo que no tiene nodos de salida. Una cadena infinita de caracteres en un autómata supone un bucle en un hiperdocumento. Cuando se considera además del hiperdocumento, el propio sistema hipermedia, la situación se complica: las ayudas de navegación ofrecen al sistema la posibilidad de generar más caminos a través del documento. Algunos sistemas son capaces de desplegar más de un nodo al tiempo, y seguir nodos desde ambos simultáneamente. En estos casos se necesitan otros modelos para describir estas situaciones, como las redes de Petri.

5.2.4 Navegación Inteligente

En la mayoría de los hipermedia la navegación, aunque facilitada por herramientas de ayuda, está guiada por el usuario. Cuando ésta se convierte en una responsabilidad distribuida entre el usuario y el sistema hipermedia se denomina *navegación inteligente*. En este caso, se usa un mecanismo de inferencia para crear caminos de acuerdo al comportamiento del usuario. A los sistemas que incorporan este tipo de navegación de les conoce como sistemas dinámicos.

Sistemas Dinámicos

La mayor parte de los hipermedia de la segunda generación implementan un modelo estático y explícito. Los nodos, enlaces y anclajes se declaran explícitamente, y deben enumerarse durante el proceso de creación en vez de establecerse dinámicamente y generarse bajo demanda. Sin embargo, los sistemas de información tales como Sistemas de Soporte de Decisiones —DSS— y Sistemas Expertos necesitan una implementación dinámica del hipertexto, una que dependa principalmente de estructuras virtuales y computadas para generar una red hipertexto en tiempo de navegación.

Para solucionar este problema, Bieber [Bieber, 1992] ha desarrollado una shell que soporta múltiples aplicaciones DSS a través de una interfaz de usuario hipertexto. Ésta, apoyándose en el motor hipertexto proporcionado por las aplicaciones DSS dota al sistema de una rica funcionalidad hipertexto

consistente en navegación, estructuras virtuales, computación y presentaciones a medida. Dado que muchos de los componentes que conforman el DSS son generados en tiempo de navegación como resultado de la interacción del usuario, era difícil predefinir todos los nodos, enlaces y anclajes en tiempo de creación.

Este mismo autor [Bieber, 1992] propone el mecanismo denominado reglas puente —*bridge laws*— para determinar los enlaces y anclajes apropiados de forma automática y empotrarlos en la aplicación interactiva. Estos anclajes proporcionan acceso a informes, operaciones —órdenes DSS— y otros componentes de las aplicaciones DSS. En consecuencia, las reglas puente son rutinas de traducción que proporciona la aplicación a la interfaz hipertexto. Éstas hacen corresponder los elementos definidos en la aplicación original sin datos hipertexto o base de conocimiento, con las entidades del motor hipertexto, sin alterar los datos de la aplicación o bases de conocimiento.

Stotts y Furuta [Stotts, 1991] también han contribuido significativamente al campo de los sistemas dinámicos. Proponen el uso de estructuras virtuales que son descritas como adaptaciones dinámicas de la estructura hipertexto. Para calcularlas, se debe recoger la información de interacción del usuario con el sistema hipertexto. Con esta información se hacen inferencias y decisiones y se crean los cambios físicos apropiados en el documento en el momento adecuado. Para ellos, un documento hipertexto puede considerarse que tiene dos niveles: una *estructura fija* subyacente de información estructurada que es creada por el autor del hipertexto, y una *estructura flexible* que se superpone sobre la anterior y que se adapta al comportamiento de cada usuario. El nivel flexible se puede generar dinámicamente. La forma en la que la información se organiza y presenta puede ser alterada sin cambiar la información actual relacionada contenida en los enlaces originales. Esto es parecido al concepto de Bieber de las reglas puente que simplemente hacen corresponder los datos no hipertexto con la interfaz hipertexto sin cambiar los datos subyacentes. Esta técnica de adaptación dinámica se ha implementado en el sistema Trellis de Stotts y Furuta basándose en redes de Petri.

5.3 Nodos, Enlaces y otras Definiciones de Interés

La información en un hiperdocumento se divide en piezas que se llaman *nodos*. Cada nodo debería formar una unidad. Decidir qué información constituye un nodo cuando se convierte un documento escrito en papel en un hipermedia es quizás la operación más dificil y especial. No hay reglas, aunque sí multitud de recomendaciones, siempre ambiguas, para determinar el tamaño ideal de un nodo. Algunos sistemas proporcionan nodos de un tamaño fijo; otros variable, formados por varias páginas, que pueden mostrarse en ventanas distintas, o en una única ventana con *scrolling*. Los nodos que están formados por varias páginas se llaman pilas —*stacks*— en algunos sistemas.

Un enlace conecta dos nodos, y es normalmente dirigido, esto es, tiene un nodo origen y un nodo destino. Usualmente, un enlace se asocia con una parte específica del nodo origen, como una palabra, frase o dibujo. A esta zona se le llama *anclaje* —*anchor*— del enlace.

El destino es, en principio, el nodo completo, pero en algunos sistemas hipermedia es posible designar una parte del nodo destino. Un ejemplo típico de esto podría ser un enlace desde la llamada de una función en un programa en C a la definición de esta función dentro del módulo de código que contiene un conjunto de funciones relacionadas. Otro ejemplo es el uso de enlaces a un documento de bibliografía: siguiendo una referencia bibliográfica se acaba en la reseña bibliográfica correspondiente que es la que se quiere consultar, y no al principio del documento que contiene todas las reseñas bibliográficas.

Como puede observarse, los anclajes designan una parte del nodo origen. No obstante, en algunos sistemas hipermedia no es posible hacer una conexión entre una parte de un nodo y la representación visual del anclaje. En HiperCard por ejemplo, los anclajes son tan sólo las formas dibujadas —rectángulos normalmente— del fondo. Al autor le corresponde el trabajo de asegurarse de que la zona activa y el texto de un anclaje coincidan en pantalla. No obstante, esta aproximación no suele ser usual porque en un sistema flexible se podrá cambiar el tamaño de las ventanas y, por tanto, se modificarán las posiciones relativas de las palabras en la pantalla.

Aunque es relativamente fácil unir anclajes al texto, en algunos casos es imposible: precisamente cuando el nodo no es un texto. Piense en un dibujo representado en GIF dentro de un fichero de imagen JPG. No se podría crear un anclaje que conectara un objeto en una imagen. El problema está en que no se tiene una forma estándar de estructurar y, por esta razón, de manipular directamente parte de esta imagen. El problema normalmente se soluciona, —tal como hace HyperCard—, definiendo zonas como anclajes. Algunos formatos gráficos ofrecen el concepto de objeto como VRML —*Virtual Reality Markup Language*—.

Como se dijo en la definición de hipermedia la gran novedad de estos sistemas es que no se impone un orden predeterminado a la hora de leer un texto. Se acaba de ver que son los anclajes los que permiten escapar precisamente de esta limitación. El usuario tan sólo tiene que seguir uno de estos elementos con un *clic* de ratón para encontrar una información más amplia o clara sobre un determinado término. Ahora bien, no todas las conexiones entre nodos que se implementan mediante enlaces y anclajes son de este tipo, esto es, no son *referencias-trasversales —cross-references—*. La mayoría de los hiperdocumentos tienen una estructura parcialmente jerárquica, y muchos de estos enlaces obedecerán a esta organización que se le ha dado al documento. Conceptualmente se puede distinguir ambos tipos de enlaces, pero de forma visual no existe diferencia entre ambos. En algunos sistemas se suelen proporcionar mapas conceptuales que representan esquemáticamente la organización de la información que se está visitando, y el lugar en el que se encuentra el lector en cada momento.

La creación de enlaces es una tarea difícil para el autor del hiperdocumento. Hay dos peligros concretos:

- Si de un nodo no salen enlaces, el lector queda bloqueado. Afortunadamente la mayoría de los sistemas hipermedia tienen una ayuda de vuelta atrás que le permite retornar a los nodos previamente consultados. Esta idea se traduce en un simple botón de *Back*, o en un complicado y completo mecanismo de historia de navegación.
- Si el grafo de nodos y enlaces es muy denso entonces pueden producirse ciclos que conducen y obligan al lector a revisitar los mismos nodos. Algunos sistemas hipermedia proporcionan una indicación visual que informan al lector de esta situación —por ejemplo, cambiando el color del anclaje—.

El autor tiene que encontrar un compromiso entre proporcionar pocos enlaces, que limitan la iniciativa del usuario, y suministrar enlaces en abundancia, que no hacen distingos entre caminos más o menos interesantes en un hiperdocumento.

5.3.1 Nodos

Los nodos, junto con los enlaces, son las dos elementos fundamentales que hacen posible la organización la información. Se puede decir que los nodos son almacenes de piezas de información que están directamente interrelacionadas. En el modelo básico de hipermedia que se ha descrito, los nodos son un conjunto de primitivas de abstracciones no tipificadas en un espacio de información. Este modelo tan simple resulta inadecuado en muchas situaciones. Por esta razón se han ido ampliando la concepción y funcionalidad de estos elementos:

Mecanismos de Composición. El modelo básico carece de mecanismos de composición para estructurar, representar y gestionar grupos de nodos como unidades únicas separadas del resto de sus componentes. La tecnología hipermedia no es capaz de representar, al menos de forma natural, relaciones completas o parciales. Los requisitos para trabajar con mecanismos de composición son especialmente críticos, ya que se está asumiendo que los sistemas hipermedia manejan una gran cantidad de información. Muchas cuestiones se deben resolver, como: ¿Un nodo compuesto puede incluir composiciones?; cuando se establecen enlaces, ¿se puede hacer referencia al propio nodo, o se debe identificar al nodo dentro del contexto de una composición?; las composiciones, ¿se deben implementar con nodos y enlaces especializados, o son por completo un nuevo mecanismo?

Tipificados o No Tipificados. Otro importante asunto relacionado con los nodos es la posibilidad de considerar a éstos como entidades tipificadas o no tipificadas. La mayoría de los sistemas hipermedia actuales trabajan con nodos no tipificados. Se caracterizan únicamente por la primitiva de información, o por combinaciones de éstas —por ejemplo, texto o texto y gráficos—. Los autores de hipermedia no pueden definir tipos de nodos pertenecientes a tipos que el sistema no es capaz de reconocer.

Contenido y Estructura sin Posibilidad de Adaptación. Los nodos son almacenes de información pasivos. Se están proponiendo mecanismos que adapten el contenido y la estructura de un nodo al contexto actual. Los nodos adaptativos podrían significar en algunos dominios, como en los sistemas educacionales, un avance significativo. Por ejemplo, un nodo puede incluir dos textos diferentes que expliquen la energía nuclear dependiendo del tipo de audiencia al que vaya dirigido. Del mismo modo, la estructura mediante la que organiza un nodo su información podría adaptarse al contexto en el que se estuviera trabajando —con más o menos comentarios o ramificaciones—.

5.3.2 Enlaces

Los enlaces son la otra unidad fundamental con la que se puede construir una red hipermedia. Normalmente se añade a cada enlace una etiqueta explicativa. Ésta es usualmente una cadena de caracteres, y sirve para identificar el lugar de destino. Los enlaces son habitualmente unidireccionales a pesar de que la implementación de enlaces bidireccionales no presenta ningún problema. En general, se pueden describir los enlaces como binarios, de una sóla dirección, etiquetados pero no tipificados, explícitamente definidos y fijos. Esta perspectiva puede ser enriquecida considerando:

Enlaces Virtuales o Computados. Los enlaces virtuales o computados son aquellos que se determinan dinámicamente a través del uso del hiperdocumento. Por ejemplo, en un sistema que soporte este tipo de enlaces se puede definir el destino de uno como aquel nodo que contenga la mejor imagen que responda a la solicitud del usuario X. Conseguir esto no será nada fácil, y dependerá de la posibilidad de que el sistema sea capaz de reorganizar la estructura de la red hipermedia en tiempo de ejecución.

Enlaces Multidestino. Una de las posibles mejoras que se pueden añadir a la noción básica de enlace es permitirle más de un destino. Cuando se activa ese enlace, el sistema suministrará los posibles destinos y esperará a que el usuario seleccione uno de ellos. Una manera alternativa de tratar esta opción es mostrar todos los nodos destino, o bien mostrar aquel nodo que mejor encaja con el usuario.

Enlaces *Span* to *Span*. Los enlaces no sólo pueden establecerse entre dos nodos diferentes. También se pueden establecer entre dos partes distintas del mismo nodo. En este caso, junto al nodo destino se proporciona además la parte específica de éste. A esta clase de enlaces se les denomina *span-to-span*.

Enlaces con Procedimientos Adjuntos. Algunos sistemas admiten que se adjunte a un enlace un determinado procedimiento. Éste se activará cuando el usuario siga ese enlace. Este tipo de enlaces consigue entornos de información mucho más potentes y con mayores posibilidades.

6 Hipermedia Adaptativos

No se ha encontrado en la literatura técnica ningún grupo de investigación o trabajo que se dedique a contemplar todo el proceso de diseño de sistemas hipermedia desde una perspectiva evolutiva. El campo de investigación más próximo, que guarda ciertas similitudes en sus objetivos, modelos y aplicaciones —sistemas educativos y sistemas de recuperación de información— son los denominados sistemas hipermedia adaptativos.

Los hipermedia adaptativos constituye una reciente área de investigación —principios de los noventa, breve historia de los Sistemas hipermedia adaptativos— que nace de la encrucijada entre sistemas adaptativos y sistemas de usuario inteligentes. Los hipermedia adaptativos tratan de ajustar la información al usuario y guiarlo en el hiperespacio presentándole únicamente el material que pueda resultarle más interesante teniendo en cuenta sus objetivos y preferencias. Es decir, los sistemas hipermedia deben incorporar mecanismos que sean capaces de adaptarlos a las necesidades de cada uno de sus posibles lectores. Para alcanzar este propósito, este tipo de sistemas «*reflects some features of the user and/or*

Breve Historia de los Sistemas Hipermedia Adaptativos

A principios de los noventa, algunos grupos de investigación empezaron a interesarse por modelar a los usuarios de los sistemas hipermedia. Como respuesta a esta preocupación se empezaron a celebrar algunos talleres sobre esta temática [UM, 1994], [UM, 1997], [Hypertext, 1998] y [UM, 1999].

La proliferación de investigación en este área ha provocado la celebración de la primera Conferencia Internacional [AH, 2000] en el pasado verano del 2000. La segunda conferencia está por determinar pero tendrá lugar posiblemente en el verano del 2002.

characteristic of his system usage in a user model, and utilizes this model in order to adapt various behavioral aspects of the system to the user» [AH, 1998].

Figura 1.16 Paul De Bra.

Un Sistema Hipermedia Adaptativo ofrece tres funcionalidades básicas [De Bra, 1999]:

- Se registran todas las acciones que realiza el usuario cuando navega a través del hiperdocumeto. Basándose en estas observaciones estos sistemas mantienen un modelo del conocimiento que tienen los usuarios sobre cada concepto del dominio.
- El modelo de usuario se aplica para clasificar todos los nodos —páginas— en grupos de acuerdo a los conocimientos, intereses y metas del usuario. El sistema manipula los anclajes de los nodos para que apunten a la información que es relevante para el usuario. Brusilovsky [Brusilovsky, 1996] denomina a esto *navegación adaptativa*. Dependiendo de la clasificación que tenga el nodo, el enlace puede ser activado o desactivado.
- Para asegurar que el contenido de una página contiene la información apropiada, el sistema muestra, oculta, resalta o difumina algunos fragmentos que son opcionales a la hora de presentar una página. De esta manera, en cada momento a cada usuario se le ofrece únicamente la información adecuada.

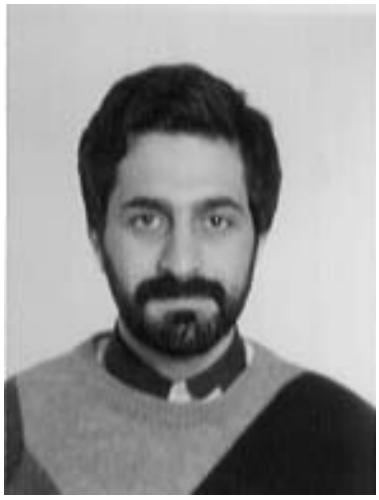


Figura 1.17 Peter Brusilovsky.

Algunos de los aspectos en los que se está trabajando son [Jameson, 1998]:

- el propósito de la adaptación, esto es, en qué sentido la adaptación que ofrece el Sistema beneficia al usuario,
- qué tipo de información acerca del usuario se debe representar,
- cómo debe interpretarse la información que suministra el lector,
- qué tipo de métodos o técnicas de la Inteligencia Artificial han de incorporarse para identificar un determinado tipo de usuario,
- cómo el Sistema puede adelantarse a las reacciones del lector a partir del modelo de usuario y, por último,
- qué tipo de evaluación empírica se debe realizar para saber si los métodos que emplea el Sistema son válidos y útiles para el usuario.

7 Referencias

[Aalst, 1993] 26
[AH, 1998] 31
[AH, 2000] 30
[Aksyn, 1988] 14
[Apple, 1989] 16
[Balasubramanian, 1993] 3, 4
[Berners-Lee, 1992] 11, 16
[Berners-Lee, 1994] 16
[Bieber, 1992] 26
[Bieber, 1997b] 17
[Bolter, 1989] 8
[Botafogo, 1992] 23
[Brown, 1991] 15
[Brusilovsky, 1996] 31
[Bush, 1945] 10, 25
[Codd, 1981] 18
[Conklin 1987] 8
[De Bra, 1999] 31
[De Vocht, 1994] 22, 25
[Engelbart, 1963] 12
[Engelbart, 1990] 17
[Fiderio, 1988] 8
[Fielding, 1997] 16
[Furnas, 1986] 24
[García-Cabrera, 1996] 4
[Guinan, 1992] 25
[Halasz, 1988a] 13
[Halasz, 1988b] 7, 13
[Halasz, 1990] 14
[Hammond, 1987] 23
[Hiltz, 1996] 17
[Hypertext, 1998] 30
[Jameson, 1998] 31
[Kibbly, 1990] 22
[Kibby, 1990] 23
[Landow, 1991] 8
[Marshall, 1995] 17
[Mayes, 1994] 22
[McDaid, 1991] 8
[Meyrowitz, 1986] 14
[Nelson, 1967] 8
[Nelson, 1987] 8, 11
[Nielsen, 1990] 7, 21, 22
[Noll, 1996] 17
[Parunak 1991] 23
[Rao, 1990] 8
[Rivlin, 1994] 17
[Rizk, 1992] 3
[Salampasis, 1997] 18, 20
[Salampasis,1997] 19
[Scacchi, 1989] 17
[Schackelford, 1993] 17
[Shneiderman, 1987] 15
[Shneiderman, 1989a, 1989b] 5
[Stotts, 1991] 23, 27
[Streitz, 1992] 17
[Sttots, 1989] 26
[Tanik, 1996] 17
[Turoff, 1991] 17
[UM, 1994] 30
[UM, 1997] 30
[UM, 1999] 30
[Usandizaga, 1998] 19
[Wang, 1995] 23
[Wayner ,1996] 4
[Winograd, 1997] 3
[Yankelovich, 1988] 14
[Ziegler, 1997] 21

Capítulo 2

Análisis del Marco Teórico

«El pasado es una especie de lámpara puesta a la entrada del porvenir para disipar una parte de las tinieblas que lo envuelven.»

Robert de Lamennais.
Filósofo y sacerdote francés (1782–1854).

«El tiempo gastado en reflexionar es una economía de tiempo.»

Publio Siro.
Poeta latino (siglo I a. C.).

«Es más fácil juzgar el talento de un hombre por sus preguntas que por sus respuestas.»

Duque de Levis.
Soldado francés (1720–1787).

Resumen

En este capítulo se coteja la crisis en la que se encuentra la creación de documentos hipermedia. La constatación de esta situación lleva a plantearse la siguiente cuestión: ¿Por qué los sistemas hipermedia no consiguen diseñar hiperdocumentos que cumplan con los requisitos de calidad reconocidos en los métodos de concepción hipermedia?

El dar respuesta a esta pregunta, implica un análisis crítico del marco teórico hipermedia con el que se pretende:

a) definir cuál debería ser la funcionalidad básica que deben proporcionar los hipermedia,

b) reflexionar sobre el proceso de concepción y desarrollo de un sistema hipermedia y,

c) estudiar los modelos hipermedia propuestos en la literatura técnica.

Todo este examen y meditación permitirá asentar las características básicas que, desde nuestro punto de vista, debe incorporar un modelo hipermedia. Las herramientas, que se basen en un modelo con esas cualidades, facilitarán y mejorarán el diseño de los sistemas de información hipermedia.

1 Introducción: Los Sistemas Hipermedia en Crisis 35
2 Los Modelos Hipermedia 36
2.1 Integración de Hipermedia en los Sistemas de Información 38
2.1.1 Requisitos para la Integración de la Funcionalidad Hipermedia 38
2.2 Modelos de Referencia Hipermedia 40
2.2.1 HAM, the Hypertext Abstract Machine 40
2.2.2 Meta-modelo de Trellis 41
2.2.3 El Modelo Dexter 42
2.2.4 El Modelo Formal de Hipertexto de Lange 43
2.2.5 El Modelo Tower 44
2.2.6 Evaluación de los Modelos de Referencia 45
2.3 Modelos Cognitivos 45
2.3.1 ¿Por qué un Modelo Cognitivo? 46
2.4 Evaluación de los Modelos Hipermedia 47
3 Descubriendo la Semántica de Estructuración de un Sistema de Información 48
3.1 Desde el Documento a la Estructura 48
3.2 Desde la Estructura al Documento 49
4 Construcción de un Modelo Hipermedia 50
5 Conclusiones 53
6 Referencias 54

1 Introducción: Los Sistemas Hipermedia en Crisis

El estudio y análisis de los conceptos, áreas de investigación y requisitos en los sistemas hipermedia señala a la *integración* como factor clave para conseguir herramientas de tercera y cuarta generación. Una interpretación más amplia sobre el término, más cercano a lo que se ha dado en llamar *la funcionalidad hipertexto*, pone de manifiesto que este avance depende, en gran medida, de que estas aplicaciones incorporen el *dinamismo* como un requisito necesario e imprescindible. Sin embargo, ninguno de los modelos de referencia propuesto en la literatura se hace eco de esta concepción —al menos en su dimensión más general—.

Son muchos los autores que vienen denunciando una cierta crisis en el proceso de desarrollo de sistemas hipermedia. El artículo "*As We Should Have Thought*" [Nürnberg, 1997] es bastante elocuente en este sentido con frases como, "*linking is harmful* [De Young, 1990], [Meyrowitz, 1991] *and hypermedia is dead*" .

A nuestro juicio, el estancamiento de la investigación en este campo nace de la tenue línea divisoria entre el modelo y la presentación o edición en los sistemas hipermedia. Por un lado, los modelos de referencia existentes [Campbell, 1988], [Furuta, 1990], [Halasz, 1990], [Lange, 1990], [De Bra, 1992], tal y como nos dice Schnase [Schnase, 1993], *"tend to focus on abstracting the connectivity of hypermedia —links— from its underlying information —nodes— rather than abstracting structure from functionality"*. Por otro lado, la representación estática de la red de información los invalida para incorporar la mayoría de las características de sistemas y entornos de autor que pretenden avanzar en la estructuración, presentación y acceso a la información interrelacionada.

La mayor parte de los actuales modelos de sistemas hipermedia se centran más en la edición, y en la documentación del documento a través de enlaces prefijados, que en la construcción dinámica del documento. En ellos, no existe una definición explícita de la semántica que determine las posibilidades de dinamismo y de navegación de la red hipermedia. A nuestro juicio, los sistemas hipermedia deberían ser consideramos sistemas de información que facilitan los procesos de estructuración y acceso de dicha información teniendo en cuenta las *asociaciones conceptuales* que se pueden establecer entre los distintos ítemes de información. Dicho de otro modo, el proceso de construcción del hipermedia debe estar basado necesariamente en un *modelo cognitivo*. Este modelo nos posibilita la representación del contenido semántico y éste, a su vez, nos permite desarrollar herramientas que respaldan todo el ciclo de vida del hipermedia —la construcción, el mantenimiento y la navegación—. A lo largo de este capítulo intentaremos justificar estas hipótesis mediante el análisis de los objetivos y de los modelos proporcionados por la literatura técnica.

Para superar tal anquilosamiento necesitamos de modelos en los que las técnicas semánticas permitan abstraer la estructura, la información y el comportamiento de un hipermedia, porque "*hypermedia is just a special case of a general philosophy of computing in which structure is more important than data*" [Nürnberg, 1997]. Además, debemos pensar en un modelo donde tengan cabida todas las características actuales y previsibles de un sistema hipermedia de alto nivel —lo que

Bieber y otros [Bieber, 1997a] [Bieber, 1997b] llaman los sistemas hipermedia de tercera y cuarta generación—. Una arquitectura que realmente permita al autor del documento un control completo y flexible de su construcción y mantenimiento y, al lector, una navegación que facilite la comprensión, el acceso y la selección del material expuesto.

Estas son algunas de las conclusiones que se extraerán en los siguientes apartados del capítulo. En él se realiza un análisis y una crítica de los modelos formulados en la literatura para la creación de sistemas hipermedia. En la apartado 2, se describe, en primer lugar, el objetivo que se marcaron la mayoría de las propuestas: "la integración de la funcionalidad hipermedia en los sistemas de información". Posteriormente —apartado 3—, se presentan las diferentes alternativas a la hora de estructurar sistemas hipermedia. Se discuten sus ventajas e inconvenientes. En la apartado 4 aplicamos las conclusiones extraídas del anterior estudio y establecemos cuáles son las características que debe tener un modelo para la construcción de un buen Sistema Hipermedia. Por último, en la apartado 5, se resumen las conclusiones sobre el análisis y crítica realizados.

2 Los Modelos Hipermedia

Como ya se vio en el capítulo 1, los sistemas hipermedia son anteriores al nacimiento de World Wide Web —WWW— y, a lo largo de su corta pero intensa historia, se han propuesto modelos, metodologías y herramientas. De hecho, se pueden distinguir tres generaciones hipermedia [Halasz, 1988b], [Salampasis, 1997] —figura 2.1—:

- La primera está representada por sistemas hipermedia basados en *mainframes*. Estos sistemas eran capaces de proporcionar soporte para trabajo en grupo pero, al mismo tiempo, presentaban serias limitaciones en cuando a facilitar la navegación y ofrecer buenos entornos gráficos.
- La segunda generación comprende sistemas basados en estaciones de trabajo y computadores personales, para usuarios individuales o pequeños grupos, incluyen avanzadas interfaces de usuario, información multimedia y ayudas gráficas para la navegación. Los actuales sistemas hipermedia pertenecen a esta segunda generación. No obstante, resultan ser sistemas cerrados con mecanismos de almacenamiento pero escasa interoperatividad.
- En 1987 comienza la tercera generación, en la que la investigación se orienta hacia el desarrollo de prototipos que intentan incluir la conceptualización de sistemas hipermedia mediante modelos abstractos. Los modelos de referencia HAM, Dexter y Trellis pertenecen a esta tentativa. En esta etapa, los prototipos son creados con el objetivo de soportar mecanismos de estructuración mediante nodos compuestos.

El principal objetivo de la última generación, en la que ahora estamos inmersos, es lograr la incorporación de las características hipermedia en los sistemas software y de información para proporcionar a los usuarios de estas aplicaciones mecanismos asociativos de acceso, análisis y organización de la información —esto es, la integración de la funcionalidad hipermedia en los Sistemas de Información [Ashman, 1996]—.

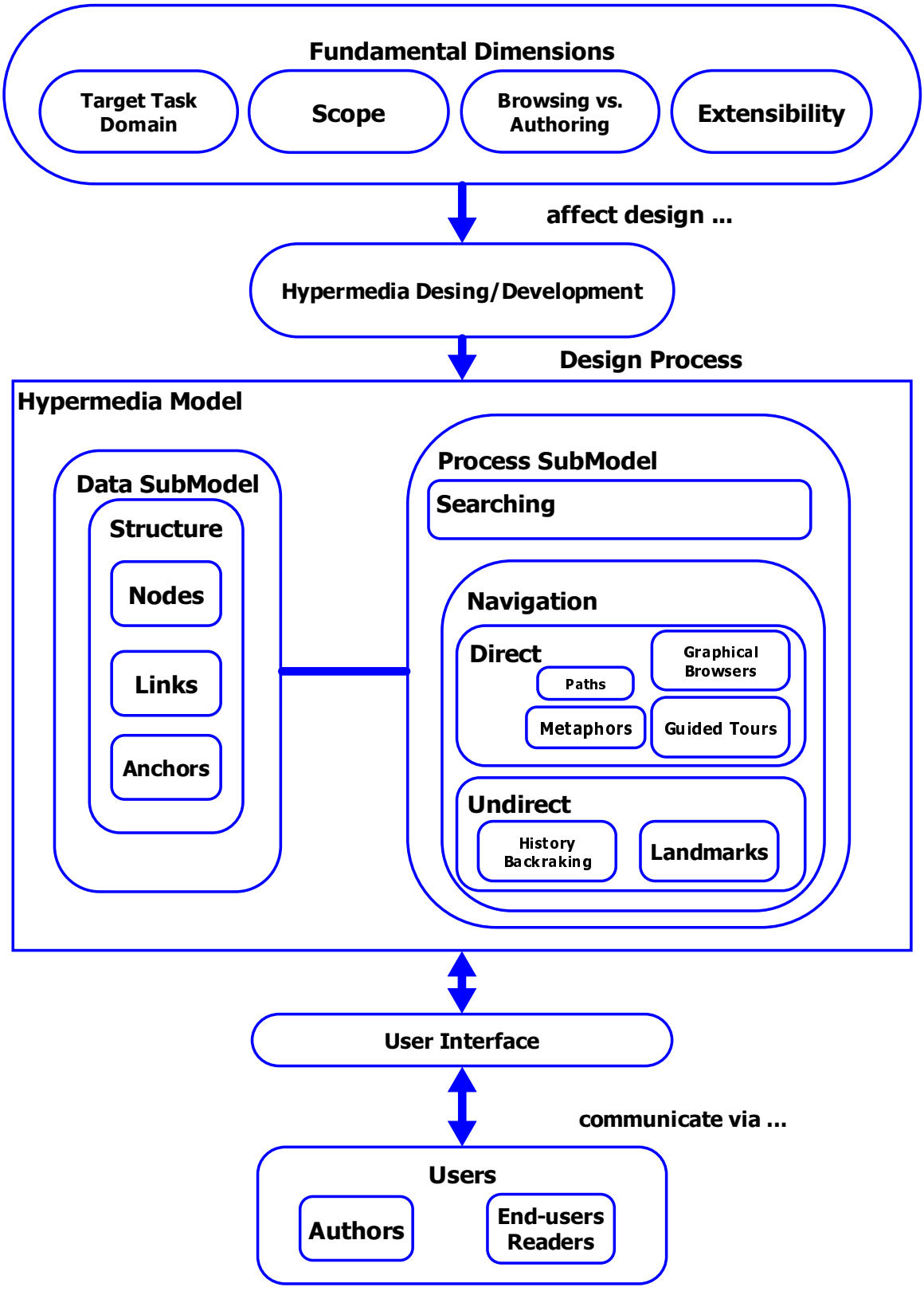


Figura 2.1 Las distintas Generaciones de Sistemas Hipermedia [Salampasis, 1997].

2.1 Integración de Hipermedia en los Sistemas de Información

En 1990, Jakob Nielsen [Nielsen, 1991] propone como objetivo el conseguir una completa integración de hipermedia en los sistemas de información. La sombra del término integración es bastante alargada. Según Balasubramanian [Balasubramanian, 1993], hacer de los sistemas hipermedia sistemas abiertos e integrables significa tener en cuenta cuestiones tan diversas como: interoperabilidad, programabilidad, tipificación de nodos y enlaces, enlaces distribuidos, control de concurrencia para el acceso multiusuario en entornos compartidos, mantenimiento de enlaces públicos y privados, soporte de sistemas operativos, redes, reglas de traducción —*bridge laws*— entre las aplicaciones y sus representaciones en hipermedia, protocolos de enlace, soporte multimediático, interfaces de usuario consistentes y control de versiones.

El recetario anterior ofrece una idea clara de cómo han ido creciendo las expectativas sobre los sistemas hipermedia. Lo que en un momento se reducía a ofrecer un simple sistema *on-line* de ayuda, un divertimento en algunos casos, ha ido tomando nuevas y más profundas dimensiones. Para Frank Halasz [Halasz, 1988b] y otros autores como Lange [Lange, 1990] y [Garzotto, 1995] los sistemas hipermedia de la próxima generación deben abarcar requisitos tan diversos y ambiciosos como los que aparecen reflejados de forma resumida en la tabla 2.1 de la página siguiente.

Según vemos, todos estos aspectos refuerzan esa idea de considerar a los hipermedia como sistemas abiertos e integradores; no sólo capaces de soportar una navegación sobre una red de información multimediática sino con serias posibilidades de ofrecer adaptación al comportamiento de usuarios al navegar, de incorporar nuevos contenidos y recorridos de navegación e incluso de rediseñar la propia red de información.

2.1.1 Requisitos para la Integración de la Funcionalidad Hipermedia

La integración de sistemas hipermedia debería ir más allá de la simple interoperabilidad en Sistemas de Información. Por ejemplo, para Bieber [Bieber, 1993] todos los sistemas software deberían beneficiarse de la funcionalidad hipermedia [SIGLINK, 1994] "*La funcionalidad hipermedia añade valor a estos sistemas al permitir a los usuarios caminos alternativos para acceder a la información, a través de relaciones explícitas e implícitas de estos sistemas... Suministrando estructuras de información asociativas se podría ayudar a los sistemas a ser rediseñados de forma más flexible*." Este mismo autor considera los sistemas hipermedia como una herramienta capaz de incorporar una interfaz dinámica a los sistemas de información.

Haciéndonos eco de estas ideas creemos que la funcionalidad de hipermedia se ve enormemente devaluada si se prescinde de su capacidad dinámica. De hecho la integración de los Sistemas Hipermedia en los Sistemas de Información pasa, desde nuestro punto de vista, por:

1. Dotar de contenido semántico, interpretable automáticamente, a la interacción de los componentes del Sistema de Información y del Hipermedia.
2. Permitir la adaptabilidad del Sistema Hipermedia a las nuevas y cambiantes necesidades del Sistema de Información.

Ello implica que los *sistemas de autor* deben permitir tanto la captura y definición de estas interacciones como la posibilidad de hacerlas evolucionar a lo largo del tiempo. De hecho, los autores

[Bieber, 1992] [Stotts, 1991] que trabajan en este campo entienden el dinamismo como adaptación del hipermedia en un doble sentido:

1. A los nuevos cambios estructurales y de contenido que el/los diseñador/es, a través de sistemas de autor, le imprimen al hiperdocumento y

<table>
<tr><td>a. Apertura y Distribución

Un sistema hipermedia abierto es un sistema que puede conectar información de otros sistemas (tanto hipermedia como no-hipermedia). Debería ser posible el libre intercambio de datos. La distribución permite al sistema almacenar en lugares geográficamente dispersos de forma transparente al usuario.</td><td>b. Soporte para trabajo colaborativo o compartido

Esto incluye acceso multi-usuario a la red hipermedia, mecanismos de control de concurrencia robustos, propagación hacia los usuarios de cualquier cambio hecho en la red por otros usuarios y seguimiento de las contribuciones hechas por cada miembro de un equipo.</td><td>c. Integridad de Datos/Corrección

Debería prohibir referencias peligrosas a objetos o atributos y prevenir una inclusión recursiva de objetos compuestos. Debería soportar tanto la noción de reglas de las bases de datos relacional extendida como el concepto de semántica en una base de datos orientada a objetos.</td></tr>
<tr><td>j. Extensibilidad y Personalización

Supone la habilidad para manejar extensiones del modelo de datos existente (esquema de evolución) de forma flexible y segura. El nivel de base de datos debería no solo ser capaz de manejar la parte estructural del modelo de datos hipermedia sino también la parte semántica.</td><td rowspan="2">Requisitos para los Sistemas Hipermedia de la Próxima Generación</td><td>d. Dinamismo

Los sistemas futuros necesitan ser capaces de reconfigurar dinámicamente la estructura del sistema hipermedia en respuesta a los cambios hechos en la red o en sus contenidos. Las estructuras virtuales serán posibles solo cuando los modelos actuales sean actualizados para adaptarse a los cambios de información.</td></tr>
<tr><td>i. Soporte Multimedia

El nivel de base de datos debería ser capaz de almacenar y recuperar eficientemente múltiples medios. Debería ser transparente con independencia del tipo de material.</td><td>e. Mecanismos de búsqueda y consulta

Además de la navegación, se debería proporcionar mecanismos de búsqueda y consulta eficientes para solucionar parcialmente los problemas de desorientación. Existen dos tipos de consulta: consultas estructurales para recuperar parte de la red y consultas de contenido que recuperan un nodo específico.</td></tr>
<tr><td>h. Control de versión

Se requiere controlar las versiones con objeto de mantener la pista de los cambios de la red. Se puede incluir en entidades tales como nodos y enlaces pero también a nivel de la red hipermedia como un todo.</td><td>g. Composición

Con el incremento del uso de hipermedia en aplicaciones sofisticadas es necesario tratar con grupos de nodos y enlaces. Los nodos compuestos deben ser primitivas de construcción en el modelo básico hipermedia. Las relaciones de inclusión (o parte-de) deben añadirse a los enlaces.</td><td>f. Computación

La incorporación de la IA dentro de los motores hipermedia será un área interesante a explorar. Se podría incluir herencia, mantenimiento de certeza, búsquedas basadas en reglas y motores de inferencia. Las inferencias computacionales pueden hacer más eficientes los accesos a la información de la red.</td></tr>
</table>

Tabla 2.1 Requisitos para los Sistemas Hipermedia de la Próxima Generación.

2. al comportamiento y preferencias exhibidas por los lectores durante navegación de los documentos que se traducen en cambios en la presentación del hiperdocumento.

A las que se podría añadir una tercera:

3. a las aplicaciones que incorporan la funcionalidad hipermedia y que necesitan explicitar en la interfaz el estado de la ejecución pudiendo modificar en un momento dado la propia red hipermedia.

Expuestos los objetivos, a continuación se estudian los modelos de Referencia y Cognitivos planteados para comprobar si responden a las anteriores demandas. En los siguientes apartados —2.2 y 2.3— se presentan y analizan dos maneras de concebir un modelo hipermedia. La primera, adopta una postura más clásica, y modela un sistema hipermedia como una jerarquía de niveles. La segunda, más ambiciosa, recoge la preocupación desde la segunda generación de sistemas hipermedia por lograr una interpretación de la estructura de estos sistemas.

2.2 Modelos de Referencia Hipermedia

Desde la tercera generación de sistemas hipermedia —finales de los 80— se vienen proponiendo Modelos de Referencia con la intención de convertir los sistemas hipermedia en sistemas abiertos e integrar su funcionalidad en cualquier entorno o aplicación. Estos modelos describen todos los elementos conceptuales que, a juicio de sus autores, intervienen en un sistema hipermedia. A continuación se describirán brevemente las cinco maneras más difundidas de entender y modelar estos sistemas. Después se hará una crítica de los mismos.

2.2.1 HAM, the Hypertext Abstract Machine

HAM o the Hypertext Abstract Machine —1987— [Campbell, 1988] —figura 2.2— fue el primer intento de llegar a expresar un sistema hipermedia mediante un modelo abstracto. HAM no describe un sistema hipermedia completo; tan sólo los objetos HAM y las operaciones que se les pueden aplicar. HAM es el nivel de abstracción que se sitúa encima del sistema de almacenamiento, y que administra y suministra toda la información hipermedia a las aplicaciones y a las interfaces de usuario.

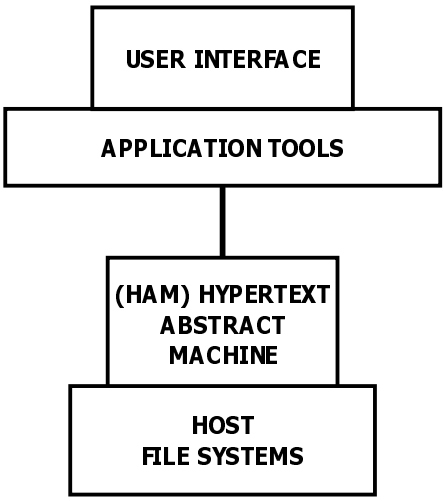


Figura 2.2 HAM: Hypertext Abstract Machine [Campbel, 1988].

El modelo HAM se basa en cinco tipos de objetos: *grafos*, *contextos*, *nodos*, *enlaces* y *atributos*. HAM mantiene la historia de estos objetos, permite seleccionarlos mediante mecanismos de filtrado —por medio de expresiones basadas en los atributos de los objetos y en sus valores— y un acceso restringido a través de un mecanismo de seguridad basado en ACL —*Access Control List*—

Un grafo es el objeto de más alto nivel y está formado por uno o más contextos. Estos, constituyen un subconjunto de *nodos* relacionados mediante *enlaces* de un hiperdocumento, es decir, son subgrafos. Los *atributos* pueden asociarse a contextos, nodos y enlaces representando propiedades de los objetos específicos de una aplicación, o bien contienen información que describen más detalladamente un objeto. El modelo únicamente describe dos de estos atributos: el identificador y la versión basada en el tiempo de creación o actualización del objeto.

2.2.2 Meta-modelo de Trellis

Richard Furuta and P. David Stotts deducen un meta-modelo a partir del *Sistema Trellis* [Furuta, 1990] —figura 2.3— basado en Redes de Petri. Este modelo distingue cinco niveles lógicos: el Nivel de Componente Abstracta —*Abstract Component Leve*l—, el Nivel de Hipertexto Abstracto —*Abstract Hypertext Level*—, el Nivel de Contexto Concreto —*Concrete Context Level*—, el Nivel de Hipertexto Concreto —*Concrete Hypertext Level*— y el Nivel de Hipertexto Visible —*Visible Hypertext Level*—. Dentro de cada nivel se realiza la representación de una o más partes del hipertexto. A diferencia de HAM, los niveles describen niveles de abstracción y no componentes del sistema.

El primer nivel describe los distintos elementos: estructura —*structure*—, contenidos abstractos —*abstract contents*—, botones abstractos —*abstract buttons*— y contenedores abstractos —*abstract containers*—, que se asociarán en el segundo nivel para formar el hipermedia. Además de los tradicionales nodos —*abstract contents*— y enlaces —*abstract buttons*—, el sistema Trellis incorpora dos elementos más: la estructura y los contenedores —*structure* and *containers*—. La primera, únicamente describe el esquema —*placeholders*— del grafo en el que se podrán ubicar los contenidos y las relaciones abstractas. Los contenedores son una abstracción de una agrupación de distintas piezas de información que se combinan para ser visualizadas conjuntamente.

Las asociaciones entre los elementos del Nivel de Componente Abstracto se realiza en el Nivel de Hipertexto Abstracto pudiéndose realizar las siguientes asociaciones: contenido-estructura, botones-estructura y contenedores-estructura. El Nivel de Hipertexto Abstracto, por tanto, describe las asociaciones, pero no la manera en las que se presentarán en la visualización del hipertexto. La correspondencia entre el hipertexto abstracto y las ventanas en dónde se desplegará una determinada información, o los detalles de operación derivados del seguimiento de un enlace, se realizarán en el Nivel de Contexto Concreto. Por último, el cuarto y el quinto nivel se ocuparán de la apariencia final del documento que mostrará una determinada interfaz de usuario.

La separación en este modelo entre componentes y asociaciones entre componentes, por un lado, y su implementación mediante redes de Petri, por otro, consiguen una adaptación dinámica de la apariencia y el comportamiento del hiperdocumento cuando es navegado. Para Stotts y Furuta, un documento hipermedia tiene dos niveles: una estructura fija subyacente de información estructurada que es creada por el autor del hipermedia —primer y segundo nivel—, y una estructura flexible que se superpone sobre el anterior y que se adapta al comportamiento de cada usuario —el resto de los niveles—.

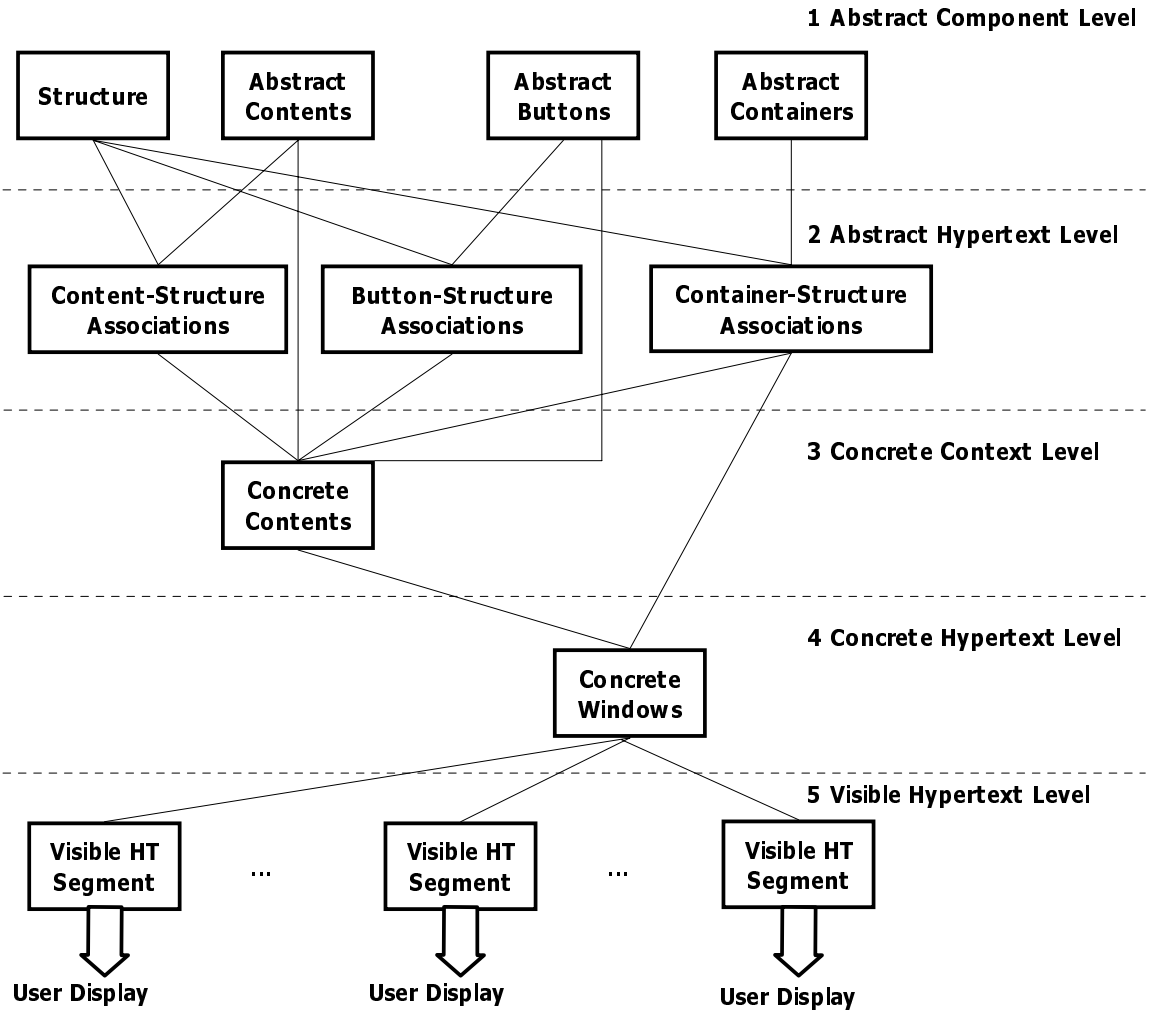


Figura 2.3 El Metamodelo de Trellis [Furuta, 1990].

2.2.3 El Modelo Dexter

Quizás el modelo más referenciado es el modelo Dexter [Halasz, 1990] —figura 2.4—. El objetivo de este modelo "*is to provide a principled basis for comparing systems as well as for developing interchange and interoperability standards*".

Divide un sistema hipermedia en tres capas: capa de ejecución —*run-time layer*—, capa de almacenamiento —*storage layer*— y capa intra-componente —*within-component layer*—. El modelo se centra en la capa de almacenamiento, que describe la estructura como un conjunto finito de componentes.

Un componente puede ser tanto un nodo, un enlace o una entidad compuesta formada por otros componentes —en realidad, grafos dirigidos acíclicos—. Sus estructuras internas son responsabilidad de la capa intra-componente. Para el modelo Dexter es como si este nivel estuviera fuera del modelo hipermedia por definición —cuya interpretación corresponde a las aplicaciones que los suministran—. Sí se preocupa por definir la interfaz entre esta capa y la de almacenamiento denominada *anclaje*

—*anchoring*—. Este intermediario es el encargado de direccionar indirectamente ítemes dentro del contenido de un componente.

Por último, establece una interfaz entre la capas de almacenamiento y la de ejecución: las denominadas especificaciones de presentación —*presentation-specifications*—. Éstas informan del modo en el que el componente se presenta al usuario. El cual no sólo depende de la herramienta específica que hace la presentación, sino también de las propiedades del propio componente y/o del camino que nos lleva a ese componente.

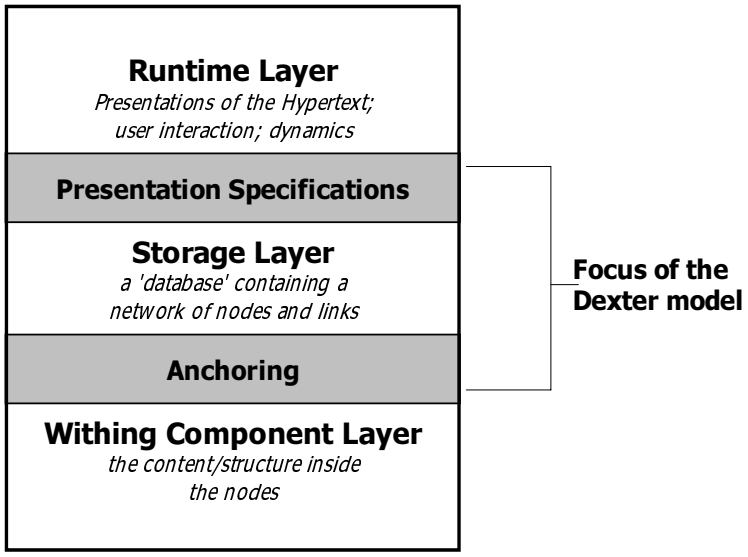


Figura 2.4 El modelo Dexter [Halasz, 1990].

2.2.4 El Modelo Formal de Hipertexto de Lange

El *modelo Formal de Hipertexto* de Danny B. Lange [Lange, 1990], al igual que el modelo Dexter, pone el énfasis en el modelo de datos. En él se definen nodos, enlaces y estructuras arborescentes de nodos.

A diferencia del modelo Dexter, sí se ocupa de la estructura interna de los nodos de un hiperdocumento. Define los *slots* —que pueden ser comparados a registros de datos de un lenguaje de programación— y, también, los *campos* —secuencias de caracteres dentro de un *slot*—. De esta forma, el modelo admite la posibilidad de referenciar un nodo, un *slot* dentro de un nodo o un campo dentro de un *slot* de un nodo. El modelo admite la posibilidad de distinguir entre tipos de nodos y tipos de enlaces añadiendo atributos y valores a estos atributos.

Finalmente, este modelo no se ocupa ni de la representación ni de la interpretación de la navegación. A juicio del autor, esto es tarea de las aplicaciones que operan con el hiperdocumento a través de las operaciones especificadas y de los objetos de datos que suministra el modelo. El modelo de datos se define como una base de datos orientada a objetos persistentes, resolviéndose en este ámbito aspectos como la distribución, administración de versión y control de acceso.

2.2.5 El Modelo Tower

De Bra, Houben y Kornatzky proponen un modelo más general orientado a objetos que llaman *Extensible Data Model for Hyperdocuments* [De Bra, 1992] o *Modelo Tower*. Este modelo "*is based on separation of concerns between the fixed aspects of a hyperdocument that would always be present, and the variable part that is extensible*". Los autores distinguen dos niveles, el nivel más bajo está formado por objetos de primera clase como son los nodos, enlaces y *anclajes*. El segundo nivel, es el que se encarga de modelar constructores que fabrican representaciones más complejas de información. Distingue tres formas de hacerlo:

- El constructor de objetos compuestos —*composite object constructor*— que modela unidades de información a partir de colecciones de nodos, enlaces y *anchors*. Esencialmente construye grafos y, "*intuitively, a composite object can be viewed as a template containing holes into which the components are plugged"*.
- El constructor de torre —*tower constructor*, figura 2.5—, que agrupa los múltiples niveles que describen un objeto. Por ejemplo, un nodo tendría una dimensión estructural que consiste en su contenido —el texto— y las operaciones que lo manipulan —el editor de textos—, y un nivel de representación que describe su apariencia en pantalla. Otro nivel sería el papel semántico de sus atributos. El número y la naturaleza de los niveles de un objeto torre es arbitrario dependiendo de la información que suministren las aplicaciones externas al hiperdocumento.
- El constructor ciudad —*city constructor*—, que mantiene unidas todas las posible vistas o perspectivas de un objeto. Esto es, las vistas se corresponden con los diferentes papeles de un mismo objeto. La diferencia entre torre y ciudad es que la primera proporciona información diferente sobre el objeto, mientras que la segunda da diferentes descripciones de, básicamente, la misma información.

Con estos constructores, el modelo ofrece un nivel de interfaz que permitiría la integración de diferentes fuentes de información, así como la creación de estructuras virtuales o computadas a partir de una descripción de sus componentes.

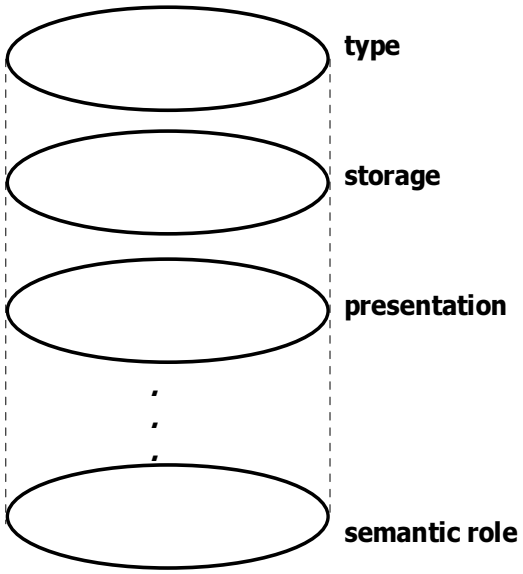


Figura 2.5 El constructor Tower [De Bra, 1992].

2.2.6 Evaluación de los Modelos de Referencia

A pesar de las buenas intenciones, estos modelos consiguen apenas una sistematización a la hora de implementar los distintos elementos que nos llevan a la construcción de un sistema hipermedia. Como mucho, logran una estandarización de los componentes que permitirían una actualización manual o estática de la red hipermedia. En este sentido los modelos reseñados hacen una interpretación del concepto integración un tanto simplista. Su actuación se reduce a facilitar la incorporación de una estructura rígida para luego olvidarse de nuevos cambios y evoluciones a las que debe someterse la estructura hipermedia.

Como se puede observar, todos estos modelos describen los posibles elementos conceptuales que nos podemos encontrar en un sistema hipermedia. Consideran distintos niveles de abstracción aunque muchos de ellos únicamente describen por completo el nivel de almacenamiento. En general, casi todos ellos coinciden en definir:

a) una primera capa de almacenamiento en la que se representan nodos, enlaces y *anclajes*. Algunos de ellos incluyen también composiciones de nodos, estructuras y composiciones de estructuras;

b) un segundo nivel en donde se hace una correspondencia entre los elementos *front-end*, o del sistema hipermedia con los modelos, variables y cálculos de los objetos *back-end* o de la aplicación; y

c) por último, el nivel de presentación, o más bien de edición, de la estructura hipermedia mediante una interfaz de usuario.

En nuestra opinión se pueden extraer dos consecuencias claves del anterior análisis:

1. Los modelos de referencia hipermedia están más enfocados a la edición o construcción estática y al recorrido del documento mediante enlaces prefijados que a la construcción dinámica del documento.

2. La mayoría de estos modelos admiten tímidamente la posibilidad de definir algún tipo de atributo asociado a nodos y enlaces que de alguna forma describa su comportamiento. Sin embargo, o no se establece claramente cuáles deben ser —HAM, Trellis, Modelo Tower—, o bien asumen que esa descripción es responsabilidad de las aplicaciones que suministran los objetos de información —Dexter, Modelo Formal de Lange—. En el mejor de los casos, definen atributos que están directamente vinculados a la presentación del documento —un enlace al principio de una página, o dentro de la página—.

2.3 Modelos Cognitivos

Como alternativa a los Modelos de Referencia están aquellos en los que nodos y enlaces son tipificados semánticamente para organizar la información de manera más efectiva y conducir así mejor la navegación de los lectores. No existe casi ninguna propuesta teórica de este tipo, aunque sí muchos entornos o aplicaciones concretas en las que se incluyen categorías de nodos y enlaces.

El único modelo genérico que se ha encontrado en la literatura especializada es la propuesta de Rao y otros [Rao, 1990] basada en la Teoría del Intelecto de Guilford —*Guilford's Structure of Intellect model*—. Su morfología general para sistemas hipermedia incluye seis tipos de nodos y doce tipos de enlaces. Los primeros representan conocimientos o ideas —p. ej. *detail*, *issue*, *proposition*—;

los segundos, relaciones entre estas ideas. Los enlaces se clasifican como convergentes o divergentes. Los primeros —p. ej. *specification*, *inference*— son aquellos que se centran o profundizan en un determinado pensamiento, mientras que los segundos —p. ej. *elaboration*, *opposition*— expanden o inician nuevas ideas.

Este modelo presenta tres grandes inconvenientes:

a) esta taxonomía únicamente beneficia a los autores o desarrolladores de sistemas [Bieber, 1997b] y no a los lectores;

b) una jerarquía es del todo artificial para representar un espacio conceptual, y

c) el mundo conceptual es tan amplio que se duda de la utilidad de una semántica tan general que al final siempre resultará demasiado ambigua.

Estos problemas no los presentan aquellas aplicaciones que incluyen modelos cognitivos concretos como, por ejemplo, el sistema gIBIS —*graphical Issue Based Information System*— [Conklin, 1989] para debates entre un equipo de diseñadores de software.

Su modelo conceptual emplea tres tipos de nodos: *issues*, *positions* y *arguments* y también los tipos de relaciones que pueden conectar determinados tipos de nodos —por ejemplo, entre *issue* y *position* pueden establecerse las relaciones *is_suggested_by* o *questions*—. De esta manera, el sistema, además de transcribir las decisiones de diseño, guarda también el *design rationale* subyacente [Shipman, 1997]. Esto es, se hace una representación de todo el proceso de razonamiento y decisión sobre un conjunto de temas relacionados.

El modelo gIBIS, sin duda, es muy práctico y efectivo, pero sólo para el dominio conceptual —coordinación de debates técnicos— para el que ha sido diseñado. No se puede aplicar a la mayoría de las áreas de aplicación tales como la enseñanza, museos, manuales técnicos, catálogos, obras de referencia, etc.

Los sistemas hipermedia adaptativos [AH, 1998] —ver capítulo 1— también se podrían catalogar como cognitivos aunque, en general, únicamente se ocupan de modelar al usuario y no al diseño y mantenimiento de sistemas hipermedia. En consecuencia, el modelo conceptual suele estar supeditado:

- a los posibles intereses, preferencias, actitudes o metas que pueda comunicar el lector,
- al comportamiento que se pueda llegar a observar del mismo cuando navega —por ejemplo, la selección de determinados enlaces, el tiempo que se pasa en una determinada página, etc— y, por último,
- a facilitar la recuperación y presentación de la información al usuario de acuerdo a los gustos e intenciones que haya elegido de manera explícita —informando directamente— o implícita —registrando su interacción—.

2.3.1 ¿Por qué un Modelo Cognitivo?

Tal y como se ha visto, los Modelos de Referencia modelan la edición final del hipermedia y algunos la navegación llevada a cabo por el lector, pero no consideran suficientemente los procesos de diseño, construcción y el mantenimiento del hipermedia realizados por el autor. Sin embargo, ésta construcción es muy importante porque implica un proceso de estructuración que queda implícito en el documento, aunque diluido.

Además, la integración de la funcionalidad hipermedia —apartado 2.1— [Ashman, 1996], [García-Cabrera, 1997], objetivo principal de los Modelos de Referencia, no es posible sin una comprensión y un control del sistema de información. Sin un consenso sobre cómo se pueden identificar los objetos de un sistema, y cómo pueden combinarse, no es posible conseguir una integración con otras aplicaciones, porque no sabrán cómo dirigirse a ellos ni qué hacer con ellos.

De hecho, algunos autores como Bieber [Bieber, 1991] consideran los sistemas hipermedia como una interfaz idónea para integrarla en sistemas de información dinámicos, tales como los sistemas de decisión basados en el conocimiento. ¿Por qué no se hace un razonamiento inverso? ¿Por qué no incorporamos un sistema de conocimiento que nos permita controlar y estructurar inteligentemente un sistema hipermedia?

Pensamos que los modelos cognitivos pueden llegar a expresar el *design rationale* haciendo explícita la semántica que nos permita estructurar el sistema de información. Semántica que al hacerse explícita en su construcción también pueda emerger, si así se desea, en su navegación. Esa dotación de semántica es la que va hacer posible el control del sistema de información y, por tanto, su mantenimiento y su integración en cualquier entorno informatizado. Sobre este razonamiento —esta hipótesis— se desarrollará, en gran medida, el presente trabajo.

2.4 Evaluación de los Modelos Hipermedia

Si bien varios autores proponen sistemas hipermedia con posibilidades de cambio, muy pocos [Rao, 1990] consideran que el proceso de construcción necesariamente debe estar basado en un modelo cognitivo. Y, como vimos anteriormente, ninguno de los modelos enunciados se ocupa de la representación de la actividad del hipermedia ni del proceso de diseño que nos lleva a estructurar la red de información de algún modo en especial. En particular, nos llama la atención la ausencia de una sistematización en cuanto al significado de:

a) las unidades de información y

b) las relaciones que se manejan.

En definitiva, no existe una explicitación de la semántica que determinen las posibilidades de dinamismo y de navegación de la red hipermedia. Si queremos que el sistema de representación hipermedia sea un reflejo de la red de información del sistema deberíamos tipificar, en suma, dotar de significado a los elementos que lo componen.

Bajo la luz de este prisma semántico y evolutivo se hacen mayores y más interesantes las posibilidades de esta nueva tecnología, de tal forma que podemos hablar de:

- hipermedia como interfaz para poder aplicarse a sistemas dinámicos o adaptativos,
- hipermedia como tecnología que dota de dinamismo a las aplicaciones existentes e
- hipermedia como herramienta de construcción y mantenimiento de software que permite cambios en el proceso de creación de diseño y, por extensión, a toda aplicación cuya concepción, diseño y construcción denote un proceso creativo: interfaces de usuario, sistemas educacionales, etc.

3 Descubriendo la Semántica de Estructuración de un Sistema de Información

En el estudio anterior, hemos llegado a la conclusión de que los sistemas hipermedia necesitan incorporar un *Modelo Cognitivo*. Pero, *¿cómo* se puede llegar a descubrir la semántica de estructuración de un Sistema de Información? En la literatura técnica encontramos dos posturas a la hora de responder esta pregunta —ver tabla 2.2—. La primera supone que la estructura conceptual de un documento se puede descubrir explorándolo y analizándolo, esto es, el proceso de estructuración se logra examinando los documentos. Por el contrario, se puede considerar que el autor es el responsable de representar dicho proceso y que los documentos se deben obtener mediante el modelo cognitivo proporcionado. En los siguientes párrafos se examinarán con detalle ambas propuestas.

Desde el Documento a la Estructura	**Desde la Estructura al Documento**
Análisis Espacial/Aspecto Visual	Ontología Universal
Análisis del Contenido/Lenguaje Natural	Ontologías Poliédricas Multi-locales

Tabla 2.2 ¿Cómo descubrir la Semántica de Estructuración?

3.1 Desde el Documento a la Estructura

Dos son las posibilidades que se plantean a la hora de buscar la semántica de los hiperdocumentos:

a) El analizar los aspectos visuales o espaciales de los documentos [Marshall, 1994] y,

b) El analizar su contenido.

La primera propuesta implica asumir que el significado se encuentra embebido en la sintaxis. Esto significa que las propiedades de edición de un hiperdocumento —el estilo, las fuentes, los párrafos, los enlaces, etc— reflejan los esquemas cognitivos y las asociaciones conceptuales de un determinado conocimiento.

Aunque esta suposición pueda llegar a ser válida para documentos que obedezcan a patrones bien conocidos y más o menos estándares —informes, horarios, programas, organigramas, etc ...—, en la mayoría de los casos, las distintas piezas de información son parciales, ambiguas y borrosas —*fuzzy*—. Tal y como dice Kaplan [Kaplan, 1994] *"...visual or graphical representations only show one aspect of a virtual space. Hypermedia allows us to operate in a space different from the geometric space: the semantic space. The architectural space of the documents and the semantic space managed during their conceptual development are weak correspondences*".

La explicación de esta conclusión es obvia. La mayor parte de las presentaciones obedecen a aspectos puramente estéticos, pedagógicos, culturales y lingüísticos, y en el peor de los casos, a limitaciones técnicas o de implementación. El análisis de gran número de hiperdocumentos complejos muestra que este proceso es imposible para una computadora e incluso puede llegar a serlo para los humanos[1] . Esto es debido a que el proceso de estructuración no ha quedado registrado en ningún sitio; se tienen las consecuencias del proceso de estructuración pero no las causas que lo han provocado.

La segunda alternativa presupone la comprensión del lenguaje natural endulzado o enriquecido con características gráficas. Aunque se puede considerar esta solución, pensamos que conseguir una completa representación del conocimiento en todos los dominios humanos es imposible, al menos, en un futuro próximo.

3.2 Desde la Estructura al Documento

Otra posibilidad es que los autores de los documentos construyan y estructuren sus propios dominios conceptuales. Basándose en algunos tipos de redes conceptuales, los autores pueden caracterizar sus dominios de información y confeccionar sus documentos. Esto significa que los sistemas de autor deben proporcionar ontologías [Uschold, 1996] abiertas que permitan la descripción y estructuración semántica de un sistema de información hipermedia.

Según esto, ¿cuál debería ser el ámbito de una de estas ontologías? ¿Se debería apostar por una conceptualización formal, estándar y universal que integrase a todas las ontologías posibles? Desgraciadamente, *"only a local context can motivate a given ontology which aims at schematizing a conceptualization"*. "*Knowledge in a domain could be extracted by understanding and generalizing the conceptualizations inherent in the lexical repositories of a domain*". "*Taxonomic sources and context, select some issues of the global conceptualization, like pointing the finger to a site in the mind of an ideal, intersubjective expert*". "*Put differently, we need to move from local heterogeneity of intended meanings to multi-local, polyhedric intended meanings*" [Gangemi, 1995].

La conceptualización de un dominio de información es sólo posible en un contexto concreto. Además de aspectos lingüísticos y culturales, un dominio de conocimiento depende del uso que se haga de la información. Al igual que Hendriks y Vriens [Hendriks, 1999], creemos que "*due to the dynamic nature of knowledge assets we must take into account tacit knowledge, involving such intangible factors as personal beliefs, perspectives and values, and embedded in individual experience"*. Incluso la misma información organizada de diferentes modos o usada con objetivos distintos requiere estructuras conceptuales diferentes. En concreto, el uso de la información determinará:

- El conjunto de relaciones y conceptos que se han de registrar.
- El grado de estructuración de estos conceptos.

---

1 Se ha estudiado la red de enlaces del Virtual Memory Tutorial —CNE Modules Tutorial Central, http://cne.gmu.edu/modules/modules.html— con el objetivo de interpretar las relaciones entre las distintas páginas de información. Nos ha resultado imposible porque: a) la única información que se guarda sobre las páginas es el nombre del fichero donde se almacena sin describir en absoluto su contenido; y b) las relaciones, muchas veces, atañen a parte de una página y no todo el contenido de la página.

- El tipo de comprobaciones de integridad que se deben llevar a cabo en un determinado espacio semántico.
- Y, como consecuencia, las representaciones —más o menos formales—, que se requieren en un dominio de conocimiento concreto.

4 Construcción de un Modelo Hipermedia

El primer paso para construir un modelo es saber qué es y qué no es. Según el diccionario de la Real Academia Española [REAL, 1992] un modelo es: *un esquema teórico de un sistema o de una realidad compleja, que se elabora para facilitar su comprensión y el estudio de su comportamiento*. En el diccionario de lengua inglesa Collins Cobuild [Collins, 1992] encontramos una definición similar: "*a model of a system or process is a theoretical description that can help you understand how the system or process works, or how it might work*". Según esta definición, en primer lugar debemos tener muy claro cuál es el sistema o la realidad compleja de la que queremos realizar una abstracción y, en segundo lugar, ese modelo deberá describir su comportamiento y hacerlo comprensible.

Evidentemente, el fin último de este estudio, en una ingeniería, será ofrecer una metodología y una herramienta que nos permita construir y emular ese modelo. Pero esta realización práctica no tendrá sentido ni éxito —como viene sucediendo— si nos olvidamos de cuál es la materia prima con la que trabajamos, y de las características que describen su comportamiento y que nos ayudan a comprenderla.

Un sistema hipermedia es un sistema de información que ofrece soportes para estructurarlo y acceder al mismo, de acuerdo a las asociaciones conceptuales que se pueden establecer entre sus distintos ítemes de información. Luego, tal como enunció y anunció Vannevar Bush, «*the process of tying two items together is the important thing*» [Bush, 1945]. Desgraciadamente, muchas de las herramientas actuales, y un buen número de autores hacen caso omiso de este pilar básico. Maravillados con el mecanismo —establecer y seguir enlaces— se han olvidado de la política —del proceso de estructuración—. *«The primacy of structure over link»* [Nürnberg, 1997] transformaría lo que ahora son sistemas de información basados en enlaces entre trozos de información, en verdaderos sistemas de conocimiento basados en ítemes de información estructurados [Nanard, 1991].

Pero no se puede estructurar y, en su caso, reestructurar una determinada información si no existe una identificación de los elementos que la componen y unas reglas que nos digan cómo han de organizarse. El proceso de construcción de un hipermedia necesariamente debe estar basado en un modelo cognitivo tal y como se ha visto. Únicamente la semántica nos va a permitir, por ejemplo, establecer enlaces entre las distintas piezas de información de una manera consciente —y no irreflexiva o caprichosa—, hacer explícito el contexto y las relaciones de un ítem de información evitando la *desorientación* y el *cognitive overhead* [Conklin, 1987], y realizar actualizaciones y adaptaciones que respeten y hagan comprender la estructura conceptual subyacente.

Tareas	Un Modelo Cognitivo	Usuarios
Desarrollo	Proceso de Diseño Incremental Colaboración entre Autores	Autor/a
Mantenimiento	Representación del *Design Rationale*	Autor/a
Navegación	Proporciona Acceso Contextualizado Facilita la comprensión	Lector/a

Tabla 2.3 Razones para la Adopción de Modelos Cognitivos.

Desde un punto de vista mucho más tangible, pensemos en los usuarios de un sistema hipermedia y escuchemos cuáles son sus pretensiones y demandas —ver tabla 2.3—:

a) Para el autor los hiperdocumentos se elaboran poco a poco, se modifican, se abandonan, se retoman y, a menudo, esta tarea cambiante se realiza en colaboración con otros autores. Por tanto, el modelo, como nos dice Nanard [Nanard, 1995] "*must support the designer's incremental and opportunistic activity all along the design process*". Un buen mantenimiento o evolución de los sistemas de información sólo es posible si guardamos información sobre cómo se ha concebido el documento y de qué manera puede ser recorrido de acuerdo a los propios criterios de los autores. Por tanto, se necesita representar el *Design Rationale* [Shipman, 1997] o proceso de razonamiento y decisión subyacente que se ha llevado a cabo.

b) Para el lector los hiperdocumentos son sistemas de información susceptibles de ser navegados. A los lectores les interesa saber en cada momento qué tipo de información se les está ofreciendo, qué se les ha ofrecido hasta el momento, y qué se les puede llegar a ofrecer. Para ellos los hiperdocumentos son, o deberían ser, fuentes de información y de representación de conocimiento que facilitan su comprensión.

Ambas posturas refuerzan la idea de que un modelo hipermedia debe ser *dinámico* o *evolutivo*, esto es, debe ser capaz de representar el proceso de diseño y la actividad del hipermedia que nos lleva a estructurar la red de información de algún modo en especial. Pero este dinamismo no es posible sin una explicitación de la semántica de estructuración que determine las posibilidades de *evolución* [Parets-Llorca, 1996] y de *navegación* de un sistema hipermedia.

Ahora bien, huyendo de la permisividad absoluta para establecer enlaces entre dos nodos no se debe caer en el otro extremo, la rigidez absoluta. Idear constructores de estructuras fijas que respondan a un modelo cognitivo concreto —gIBIS [Conklin, 1989], SEPIA [Streitz, 1992]— que impongan severas restricciones hacen inviable su uso por parte del autor. Parafraseando a Marshall y otros [Marshall, 1994], consideramos que es mucho más interesante ofrecer un sistema descriptivo en el que el autor caracterice sus dominios de interés y use sus propias abstracciones para limitar y estructurar sus sistemas de información. En consecuencia, el sistema debe proporcionar mecanismos que permitan incorporar y representar las ontologías o dominios conceptuales [Uschold, 1996] que el autor considere oportunos.

Del anterior análisis se deducen las siguientes conclusiones —figura 2.6—:

- Para construir un buen modelo hipermedia debemos saber qué elementos lo integran, cuál es la funcionalidad o los tipos de servicios que deben ofrecer cada uno de ellos y, por último, qué comunicación y material deben fluir entre cada uno de estos elementos.
- El modelo debe facilitar y hacer flexible la construcción, el mantenimiento y la navegación de un sistema hipermedia. Estos tres aspectos clave están sujetos a continuos cambios y adaptaciones que el modelo debe ser capaz de incorporar.
- La semántica debe impregnar el modelo, ya que las posibilidades de estructuración y de realizar cambios, adaptaciones o evolución dependerán del nivel de explicidad que se haya logrado de esta representación semántica.
- El modelo debe ofrecer una semántica flexible con la que el autor pueda caracterizar sus propios dominios de información.

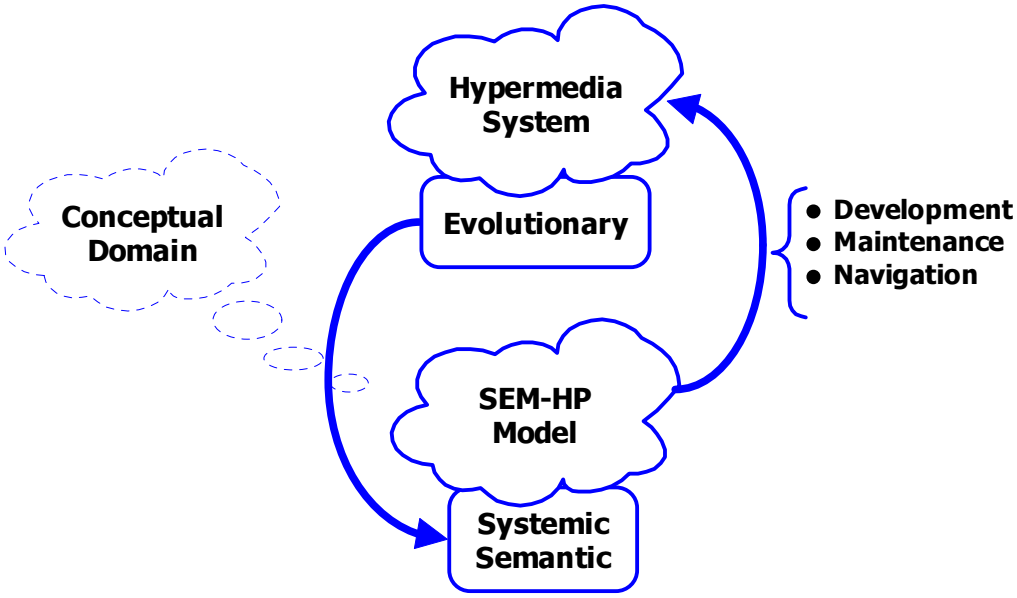


Figura 2.6 Un Modelo Sistémico, Semántico y Evolutivo para la construcción, mantenimiento y navegación de Sistemas Hipermedia.

En definitiva, un buen modelo hipermedia debe tener, al menos, dos características:

a) debe estar basado en un modelo cognitvo, esto es, debe hacer explícita la semántica de estructuración; y

b) debe ser evolutivo.

Un tercer postulado hasta ahora no debatido es considerar a los hipermedia desde una perspectiva puramente sistémica. El esquema tradicional de los modelos basados en una jerarquía de niveles traducibles a una metodología más o menos secuencial y estática nos parece impropio para representar una realidad compleja y cambiante, en donde la construcción, el mantenimiento y el uso —la navegación— se confunden por sus fuertes interdependencias. En estos modelos basados en capas no existe una correspondencia entre estructura y funcionalidad. Un sistema hipermedia puede ser concebido como un conjunto de sistemas interrelacionados en continua evolución.

La explicación y el amparo se puede encontrar en la definición de sistema de Le Moigne [Le Moigne, 1990]: "*Un sistema es algo (no importa qué, si se presume identificable) que dentro de algo (entorno) para algo (finalidad o proyecto) hace algo (actividad = funcionamiento) por algo (estructura = forma estable) que se transforma en el tiempo (evolución).*" Considerar más de un sistema es fundamental para ofrecer diferentes representaciones de una misma realidad compleja. Cada una de ellas —cada sistema— recogerá algunas de las características del modelo y, por tanto, ofrecerá ciertas funciones al resto de los sistemas y, en consecuencia, a los usuarios —autor y lector— que interactúan con los mismos.

De lo anteriormente expuesto se concluye que un Sistema Hipermedia se puede modelar mediante una serie de sistemas interrelacionados y en interacción. Cada uno de estos sistemas registrará algunas de las características del modelo y, por tanto, ofrecerá ciertas funciones al resto de los sistemas y, en última instancia, a los usuarios —autores y lectores— que interactúan con él. Este modo de entender y modelar los sistemas hipermedia hace factible, comprensible y flexible la construcción, mantenimiento y la navegación de un sistema de información en constante evolución. Será la semántica, tal y como se describe en los siguientes capítulos, la que haga esto posible.

5 Conclusiones

El análisis del contexto hipermedia da cuenta de sus pretensiones y deseos, al tiempo que desencanta por la confusión que campea desde su concepción hasta las realizaciones prácticas. El estudio de los distintos modelos propuestos para el diseño de sistemas hipermedia, pone de manifiesto su incapacidad para representar los procesos de desarrollo, mantenimiento y navegación de un sistema información en constante evolución.

Los modelos hipermedia de referencia se centran más en la edición y navegación del documento a través de enlaces prefijados que en la construcción dinámica del mismo. La mayoría de los modelos tímidamente admiten la posibilidad de definir algún tipo de atributo que se le pueda asociar a nodos y enlaces, a fin de que se pueda describir su comportamiento. Además, ninguno de ellos establece, de forma clara, mecanismos para definir y representar la semántica de estructuración de los sistemas hipermedia.

Consideramos que la piedra angular de todo este enjambre es la integración. Ahora bien, en nuestra opinión: *la integración en hipermedia es (una) cuestión de semántica* [García-Cabrera, 1997]. Esta frase encierra un doble sentido:

- la integración hay que entenderla en su sentido más amplio, abierto, adaptativo y dinámico.
- la integración y, por tanto, el dinamismo pasa por concebir una red semántica que ponga de manifiesto el tipo de relaciones estructurales y las posibilidades de remodelación de la red.

Por estas razones, los sistemas hipermedia deberían estar basados en *modelos cognitivos* en donde se hace una representación explícita del conocimiento que aportan. De esta manera, el sistema es capaz de hacer una interpretación de la estructuración del sistema de información. Esto es clave, porque permite al autor un control completo y flexible de su construcción y mantenimiento, y al lector, una navegación que facilita la comprensión, el acceso y la selección de material expuesto. Además,

estos modelos basados en la semántica de estructuración, facilitan la incorporación y representación, de manera flexible, de las ontologías definidas por el autor, a la vez que, apoyan la evolución y adaptación del sistema hipermedia.

En definitiva, un Sistema Hipermedia debe estar basado en un modelo Semántico y Evolutivo formado por un conjunto de sistemas interrelacionados y en interacción.

6 Referencias

[Ashman, 1996] 36, 47
[Balasubramanian, 1993] 38
[Bieber, 1991] 47
[Bieber, 1992] 39
[Bieber, 1993] 38
[Bieber, 1997a] 36
[Bieber, 1997b] 36, 46
[Bush, 1945] 50
[Campbel, 1988] 40
[Campbell, 1988] 35, 40
[Collins, 1992] 50
[Conklin, 1987] 50
[Conklin, 1989] 46, 51
[De Bra, 1992] 35, 44
[DeYoung, 1990] 35
[Furuta, 1990] 35, 41, 42
[Gangemi, 1995] 49
[García-Cabrera, 1997] 47, 53
[Garzotto, 1995] 38
[Halasz, 1988b] 36, 38
[Halasz, 1990] 35, 42, 43
[Hendriks, 1999] 49
[Kaplan, 1994] 48
[Lange, 1990] 35, 38, 43
[Le Moigne, 1990] 53
[Marshall, 1994] 48, 51
[Meyrowitz, 1991] 35
[Nanard, 1991] 50
[Nanard, 1995] 51
[Nielsen, 1991] 38
[Nürnberg, 1997] 35, 50
[Parets-Llorca, 1996] 51
[Rao, 1990] 45, 47
[REAL, 1992] 50
[Salampasis, 1997] 36, 37
[Schnase, 1993] 35
[Shipman, 1997] 46, 51
[SIGLINK, 1994] 38
[Streitz, 1992] 51
[Sttots, 1991] 39
[Uschold, 1996] 49, 51

Parte II: El Modelo SEM-HP

CAPÍTULO 3

Bases Teóricas de SEM-HP

«Convierte tu muro en peldaño.»

Rainer Maria Rilke.
Escritor checo en lengua alemana (1875-1926).

«El suceso futuro proyecta su sombra.»

Thomas Campbell
Poeta escocés (1777-1844).

«Su Majestad Imperial me hablaba con frecuencia y yo, por mi parte, le respondía, aunque ninguno de los dos entendíamos palabra.»

Jonathan Swift, en los 'Viajes de Gulliver', PRIMERA PARTE, capítulo II.
Escritor irlandés (1667-1745).

Resumen

El modelo —denominado SEM-HP— que se propone en esta tesis se fundamenta en tres bases teóricas: la Teoría de Sistemas, la Evolución del Software y la representación de la Semántica de Estructuración.

En este capítulo se hace, en primer lugar, una sucinta revisión de las distintas perspectivas que se pueden adoptar en cada uno de esos campos de investigación. Posteriormente, se hará hincapié en nuestra particular interpretación de los mismos con objeto de hacer más nítida la descripción, en los siguientes capítulos, del modelo SEM-HP.

1 Introducción 57
2 Teoría de Sistemas 57
2.1 Conceptos Básicos 58
3 Evolución e Ingeniería del Software 59
3.1 El Desarrollo de Sistemas Hipermedia 59
3.2 Líneas de investigación 60
4 MEDES 62
4.1 Una Arquitectura con Dos Niveles de Abstracción 62
5 Semántica 64
5.1 Ontologías 64
6 Conclusiones 65
7 Referencias 66

1 Introducción

Hasta el momento, se ha puesto de manifiesto la incapacidad de las arquitecturas existente para ofrecer un modelo en el que se puedan reflejar las connotaciones dinámico-semánticas de los sistemas hipermedia. Este análisis sirve de punto de partida para emprender un estudio en el que se contemplan a los sistemas hipermedia desde una perspectiva sistémica. Tres van a ser, por tanto, las asunciones teóricas en las que se apoyará la propuesta que se presenta en esta tesis:

- El proceso de desarrollo de los sistemas hipermedia debe hacer posible la representación y estructuración de un dominio de información. Además debe ofrecer distintas vistas del mismo dominio y diferentes formas de recorrerlo o navegarlo. Esto es, el sistema hipermedia debe soportar su triple funcionalidad, y para ello deberá ser concebido como un conjunto de sistemas interrelacionados y en interacción en constante evolución: el modelo es *Sistémico*.
- El proceso de desarrollo debe de poder integrar los continuos cambios y actualizaciones que sufrirán los sistemas de información, dominios conceptuales y rutas de navegación: el modelo debe ser *Evolutivo*.
- Con el objeto de controlar este proceso de desarrollo, el modelo deberá respaldar los siguientes elementos: mecanismos para la representación de un sistema de información, esto es, para la representación de un dominio conceptual u ontología a la que pertenece el dominio de información; utilidades que nos permitan recorrer o rememorar el conocimiento almacenado. En definitiva, el modelo debe ser capaz de representar la *Semántica* del proceso de desarrollo, esto es, debe estar basado en un *modelo cognitivo* [Rao, 1990] [García-Cabrera, 2000].

Resumiendo, el modelo hipermedia que se sugiere debe integrar distintos sistemas. Cada uno de ellos representará su propio proceso de construcción y controlará su propia evolución y hará evolucionar al resto de los sistemas.

Dado que los términos sistémico, evolutivo y semántico puede llevar a distintas interpretaciones y que éstas pueden dar lugar a distintos esquemas de modelización, se dedicará este capítulo a aclarar qué se entiende por cada uno de esos vocablos y las teorías en las que se enmarca esta aportación. En particular, el apartado 2 se dedica a describir la Teoría de Sistemas y el tercero a la evolución y su papel dentro de la Ingeniería del Software. A continuación, en el apartado 4, se explica cómo se interpretan los conceptos de la Teoría de Sistemas y la evolución del software en MEDES. Finalmente, en el quinto se recogen brevemente las conclusiones.

2 Teoría de Sistemas

El modelo que se propone en esta tesis adopta muchos de los principios de la denominada Teoría de Sistemas. En particular, se enmarca en la línea de investigación traducida como Teoría del Sistema

General [Bertalanffy, 1968, 1977] [Le Moigne, 1977, 1990] [Morin, 1977] que se ocupa de la elaboración de un modelo y no de descubrir las leyes que rigen los sistemas.

En la Teoría del Sistema General el concepto de Sistema no es una entidad que se pueda describir enumerando sus componentes; es en palabras de Morin el "*concepto complejo más simple*" [Morin, 1977, p. 177]. Pero además, el sistema no es algo ajeno y alejado del modelador. Desde el punto de vista sistémico, el observador forma parte también de la definición del sistema observado, y el sistema observado forma parte a su vez del intelecto y de la cultura del observador-sistema.

En consecuencia, la modelización sistémica es constructivista; es un proceso, una actividad de desarrollo en la que los distintos componentes o elementos de un sistema están activos, esto es, ofrecen un funcionamiento en un determinado entorno que viene determinado por su estructura. Los conceptos de sistema y evolución están bastante imbricados, de modo que el cambio de la estructura del sistema hará que éste se comporte de forma diferente.

2.1 Conceptos Básicos

A continuación describimos los conceptos de la Teoría del Sistema General que han sido recogidos en el modelo SEM-HP. El primero de ellos no podría ser otro que la definición de modelo, que se entiende como: "*una construcción conceptual que refleja clara y esquemáticamente ciertos aspectos de un fenómeno natural y que permite hacer deducciones y predicciones comprobables*"[Bertalanffy, 1975, p. 95].

Ahora bien, para los autores sistémicos, la operación o proceso de construcción del modelo y su producto forman un todo inseparable [Le Moigne, 1990]. "*El sujeto interviene en la definición del sistema en y por sus intereses, sus selecciones y finalidades, es decir, que aporta al concepto de sistema, a través de una sobredeterminación subjetiva, la sobredeterminación cultural, social y antropológica*"[Morin, 1977, p.l67]. La intervención del modelador o creador es fundamental, porque sin su actividad y su particular perspectiva del objeto abstracto que trata de modelar, no se llegaría a contar con un modelo.

Esto es, la actividad del modelador es crucial porque es él quién elabora el modelo a partir de la realidad modelable, y porque sin su particular visión no existiría siquiera dicho modelo. Por otro lado, los sistemas interactúan entre sí mediante acciones que no sólo hacen cambiar a un determinado componente del sistema sino que desencadenan reacciones, transacciones y retroacciones. Las acciones son, por tanto, el mecanismo mediante el cual los componentes del sistema interactúan unos con otros, y también el modo en el que el Sistema Software se comunica con el entorno.

Las acciones que se realizan en el sistema llevarán a éste a un determinado estado que Le Moigne entiende como una instantánea fotográfica externa de los distintos componentes que se han modelado [Le Moigne, 1977, p. 155].

Tal y como se dijo anteriormente, la Teoría del Sistema General asume que "un todo es más que la suma de sus partes" [Bertalanffy, 1968, p.53]. Dicho de otro modo, un sistema presenta unas cualidades emergentes que no se explican a partir de las características de las partes.

El concepto de emergencia trae consigo el de restricción o constreñimiento debido a que cada una de las partes ejercerá algún tipo de influencia sobre las otras. Así mismo las restricciones en las partes suponen limitaciones en el todo y los constreñimientos del todo tendrán su repercusión en las partes [Morin, 1977, p.136]. Estos dos últimos conceptos son en realidad los que provocan la evolución del sistema.

En todo sistema se podrá distinguir un doble aspecto, el estructural y el funcional. La estructura —la forma— es "el conjunto de reglas de ensamblaje, de unión, de interdependencia, de transformaciones, que se concibe bajo el nombre de estructura, y ésta, en el límite, tiende a identificarse con la invariante formal de un sistema" [Morin, 1977, p. 159]. En cambio, la función —la acción, la dinámica— es producto y consecuencia de la estructura. Esto quiere decir que si cambia la estructura cambiará su función, y que ambas irán evolucionando juntas.

3 Evolución e Ingeniería del Software

La evolución nace como respuesta a las limitaciones que ha demostrado la Ingeniería del Software en la producción de software. Parets-Llorca [Parets-Llorca, 1995b] considera que son tres las principales carencias que se le pueden achacar a los actuales métodos comerciales de desarrollo de software:

1. No recogen la interactividad, recursividad y creatividad del proceso de desarrollo.
2. Tratan de definir etapas perfectamente enlazadas sin contemplar la vuelta atrás.
3. No contemplan la capacidad evolutiva del sistema ni el papel del modelador.

En definitiva, la evolución debe ser una de las características básicas de los Sistemas Software. Los modelos y herramientas que se dediquen a la construcción de software deben asumir de manera explícita el proceso evolutivo e incluir mecanismos y conceptos que permitan abordar los cambios que puedan sufrir los entornos en los que se hallan inmersos.

3.1 El Desarrollo de Sistemas Hipermedia

El diseño e implementación de Sistemas Hipermedia no es más que un proceso de desarrollo software con ciertas connotaciones especiales [Murugesan, 1999]:

- Se centra en la representación de sistemas de información en los que es posible asociar cualquier tipo de contenido y en facilitar el diseño de su estructuración y navegación.
- Está mediatizado por el contenido —el mundo conceptual— que se presenta.
- Está dirigido a usuarios con perfiles muy diversos.
- Requiere de un desarrollo relativamente rápido y de un mantenimiento continuo que impide el seguimiento riguroso de cualquier metodología.
- Los creadores de estos sistemas no siempre cuentan con una profunda formación técnica.

Estas circunstancias respaldan aún más la clásica premisa: la calidad del producto obtenido depende de la calidad de su proceso de desarrollo. Paradójicamente, la mayor parte de las aplicaciones hipermedia siguen la estrategia ad-hoc —como por ejemplo, *code-and-fix* o *just-do-it* [Goldberg, 1995] —, afortunadamente ya superada por la Ingeniería del Software.

El desarrollo de sistemas hipermedia debería tener más en consideración el modelo *proceso-producto* —figura 3.1— de Conradi [Conradi, 1993]. En él se recalca la idea de que el producto obtenido es responsabilidad del Proceso de Producción Software. Evidentemente, éste debe contar con todos los elementos necesarios para llevar a cabo el proceso de producción, es decir, debe proporcionar la tecnología y los modelos adecuados. Por último, el Metaproceso se ocupará de hacer evolucionar los dos anteriores procesos. Sin duda, dos son las aportaciones más significativas de este entorno [Rodríguez-Fórtiz, 2000b]:

- la existencia de una férrea interdependencia entre los tres procesos y los productos que se elaboran, y
- la naturaleza evolutiva del proceso de desarrollo dirigida por el Metaproceso.

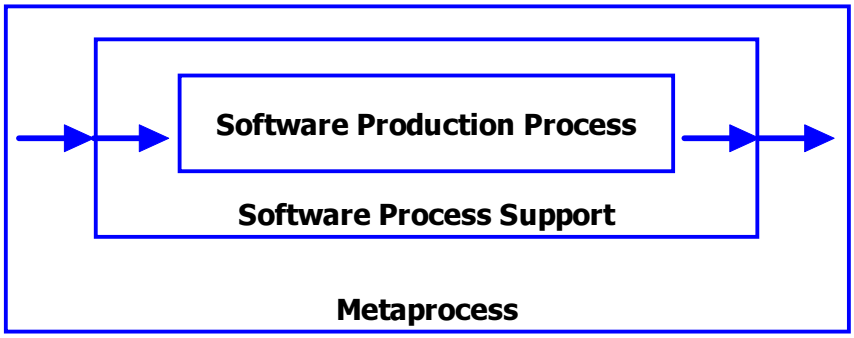


Figura 3.1 El Proceso de Desarrollo Software [Rodríguez-Fórtiz, 2000]

Con objeto de mejorar este proceso de desarrollo se han ido proponiendo un buen número de metodologías y herramientas que se centran en la elicitación de requisitos y en la sistematización del proceso de diseño. Siguiendo los pasos de la Ingeniería del Software también se han propuesto metodologías para el diseño de aplicaciones hipermedia que, en general, se basan en el modelo entidad-relación —es el caso de HDM, *Hypermedia Design Methodology* [Garzotto, 1993] y RMM, *Relationship Management Methodology* [Isakowitz, 1995]— o en el modelo orientado objetos —dos claros ejemplos son OOHDM, *Object Oriented Hypermedia Design Methodology* [Schwabe, 1996] y EORM, *Enhanced Object-Relationship Model* [Lange, 1996]—.

Sin embargo, las metodologías abandonan el proceso de desarrollo cuando ya han elaborado la aplicación y no consideran la posibilidad de modificar los modelos que se han construido. Normalmente, las herramientas tienden a ofrecer heurísticas que intentan verificar que el producto es correcto, pero en absoluto incorporan estrategias para transformarlo y propagar su impacto en el modelo.

El proceso de construcción de Software y, en particular, el de Sistemas Hipermedia, no debe desdeñar su natural tendencia al cambio. Esta modificación se podrá efectuar, durante el desarrollo o a lo largo de la vida del sistema, motivado por la adaptación o mejora del producto y, en cualquier caso, será efectuado desde un Sistema de Desarrollo por los creadores del hipermedia.

3.2 Líneas de investigación

El estudio de la Evolución Software se ha instalado definitivamente en la investigación con diversas perspectivas y objetivos. En líneas generales, se pueden identificar seis tendencias distintas:

- Belady y Lehman [Belady, 1976] consideran la evolución del software como un proceso dinámico en el que se deben establecer reglas cuantitativas que revelen el comportamiento y grado de desarrollo del Sistema Software. En su trabajo más reciente, Lehman y Ramil [Lehman, 2000], se centran en dar respuesta a *qué* y *por qué* evoluciona un software en vez de *cómo*. Estos autores proponen el uso de teorías formales que modelen dinámicamente los sistemas que expliquen las causas y las razones que los hacen cambiar.
- Durante los ochenta, se sugieren nuevos modelos de ciclo de vida que intentan incorporar la iteración en el proceso de desarrollo del software. El modelo de prototipos [Agresti, 1986], el modelo espiral de Boehm [Boehm, 1986] y el modelo de Henderson [Henderson-Sellers, 1990] pertenecen a esta etapa. Todos estos autores establecen la base para la automatización del proceso del software, la gestión del flujo de trabajo y el control de versiones.
- Por otro lado, [Aoyanne, 2000] [Niertrasz, 2000] [Amano, 2000] plantean el diseño de patrones como mecanismos capaces de hacer segura la evolución, controlando el impacto del cambio y gestionando las restricciones.
- Otra de las aproximaciones considera la evolución como la sucesión de los distintos estados que se dan en un sistema, y las acciones que producen estos cambios. El objetivo en este caso es proporcionar un modelo de la funcionalidad del sistema. Algunos ejemplos de esta propuesta son los denominados diagramas de transición de estados de OMT —Object Modelling Tool— [Rumbaugh, 1991] y UML —Unified Modeling Language— [Booch, 1999], y los modelos de tratamiento de características de MERISE [Tardieu, 1983].
- La evolución también puede estar centrada en la fase de implementación; los programas se transforman para generar nuevas versiones del mismo software. Estos cambios pueden realizarse en la especificación de requisitos y automatizar los cambios en el software [Kozaczynski, 1992], o preservar el significado original pero intentar mejorar algunos aspectos como la eficiencia del código [Berzins, 1993]. Otros autores [Said, 2001] se dedican a llevar a cabo *transacciones* automáticas basándose en factores de calidad.
- Otra forma de percibir un Sistema Software es considerar que éste tiene una ESTRUCTURA que evolucionará mediante un proceso de maduración. En este caso la evolución implica una sucesión de transiciones o cambios de estructura mediante la actividad del equipo de desarrollo. En este sentido se han propuesto modelos que acometen la construcción de estas transiciones de estructuras. [Banerjee, 1987] y [Casais, 1990] se enmarcan en esta línea y han incorporado conceptos clave como la historia de la evolución de una estructura descrita como una secuencia de transiciones de evolución. Además de esto, [Heckel, 2001], [Wermenlinger, 2001] y [Mens, 2001] proponen el uso de la reescritura de grafos como herramienta formal fundamental de un meta-modelo que posibilite la modificación de la estructura de los programas y así resolver los problemas de evolución.

Este trabajo se centra precisamente en la última perspectiva, que coincide con la propuesta por los autores de la Teoría del Sistema General y por MEDES —*cf.* el siguiente apartado—. El proceso de evolución se concibe como un proceso de maduración que comienza cuando se engendra el sistema y que se lleva a cabo durante toda la vida del sistema. La evolución se puede definir como "*una transformación de la estructura a lo largo del tiempo, producida por el desarrollador*" [Parets-Llorca, 1995a]. Esto es, la evolución del Sistema Software —SS— es la capacidad funcional principal del modelador ya que éste no puede modificarse a sí mismo. El desarrollador trabajará en un nivel de abstracción superior, al que se denomina MetaSistema —MS—, que le ofrecerá un conjunto de formalismos y herramientas con las que podrá modelar el Sistema Software situado en un nivel más bajo de abstracción.

4 MEDES

La visión y aplicación de los conceptos de la Teoría de Sistemas y de la evolución que se usará en el modelo SEM-HP procede de las aportaciones de [Parets-Llorca, 1993, 1995a, 1995b, 1996a, 1996b, 1999] y de su metodología MEDES, un Método de Especificación, Diseño y Evolución de Software.

[Parets-Llorca, 1995a] recoge las siguientes conclusiones con respecto a la Teoría del Sistema General:

1. Los sistemas software son desarrollados por equipos de analistas, diseñadores y programadores —a los que se denomina desarrolladores— que elaboran modelos sucesivos finalmente implementables en un computador. Por tanto, el desarrollo de software se lleva a cabo mediante funciones recursivas de concepción/representación hasta la obtención del modelo final.
2. El objetivo principal del desarrollador es realizar modelos mediante un Sistema de Desarrollo Software —SDS— cuya finalidad será la de producir Sistemas Software isomorfos con el SDS.
3. Los modelos elaborados deberían permitir la concepción, re-concepción, representación y re-representación, esto es, la aplicación de una actividad recurrente durante el proceso de modelado o desarrollo de Software.
4. Durante el proceso de concepción y desarrollo —de modelización—:
   4.1. El observador reflexiona sobre sus percepciones y concibe un sistema que deberá representar mediante algún lenguaje.
   4.2. El observador necesita de algún mecanismo con el que pueda reconcebir lo ya concebido y a re-representar lo ya representado.
5. El Sistema de Desarrollo Software es un METASISTEMA de los sucesivos Sistemas Software que se construirán mediante el mismo. Esto quiere decir, que en última instancia es el desarrollador el que concibe y construye el modelo pero auspiciado por el SDS que proporciona los mecanismos formales y de representación suficientes para que se pueda concebir, desarrollar y hacer evolucionar el Sistema Software.

4.1 Una Arquitectura con Dos Niveles de Abstracción

En consecuencia, desde el punto de vista de MEDES, un Sistema Software consiste en un conjunto de sistemas que interactúan entre ellos y con el entorno. Estos sistemas consisten en una ESTRUCTURA que va madurando a lo largo de tiempo. El equipo que los desarrolla es el responsable de modificar esta ESTRUCTURA con objeto de ir evolucionando su funcionalidad. Por tanto, en cada uno de los sistemas se puede distinguir dos niveles de abstracción: el Sistema Software —SS— y Metasistema —MS—:

1. El primer nivel especifica la Estructura del SS. El usuario únicamente interactúa con el Sistema Software ejecutando los componentes de su estructura. Ésta puede ir cambiando, evolucionando y madurando en el transcurso del tiempo y durante su uso. De acuerdo a la Teoría de Sistemas, la funcionalidad del Sistema depende de su estructura. En consecuencia, si se modifica la estructura cambiará la funcionalidad de los sistemas. Cuando el usuario decida cambiar la funcionalidad del Sistema Software deberá informar al desarrollador para que éste modifique el SS mediante MS.
2. El segundo nivel es el MS, que se dedica a realizar estas acciones estructurales que hacen posible la construcción y re-construcción de los distintos componentes del Sistema Software, cambiando así su funcionalidad. El desarrollador interactúa a este nivel, especificando y diseñando y, por tanto, creando y modificando el SS.

En resumen, un Sistema Software consiste en un conjunto de componentes que llevan a cabo *acciones* —las que demanda el usuario durante su interacción—. A estas acciones se les puede asociar un conjunto de *restricciones* —pre o post condiciones— que determina cuándo se pueden realizar. En otras palabras, si para una determinada acción se cumplen sus restricciones asociadas, la acción se llevará a cabo. Por tanto, las restricciones se basan en cada momento en el estado previo del sistema y garantizan la integridad del mismo.

El MS también es un sistema software pero, a diferencia del primer nivel, su estructura no cambia y su funcionalidad es siempre la misma. El desarrollador hará uso de este Meta-nivel —mediante una herramienta CASE— que proporciona *acciones evolutivas* con las que podrá ir creando y modificando la estructura de cualquier Sistema Software y, en consecuencia, su funcionalidad.

No se podrán realizar todas las modificaciones que se deseen en el Sistema Software porque, tal y como se dijo, la consistencia debe mantenerse durante su evolución. Para ello, tanto el Meta-Sistema como el propio desarrollador suministran *invariantes* que se establecen en forma de *restricciones* y *meta-restricciones* que se asociarán a las acciones evolutivas que se desencadenan desde el Meta-nivel. Cada vez que el diseñador, a lo largo del proceso de desarrollo SS, intente hacerlo evolucionar, el MS velará por garantizar su integridad.

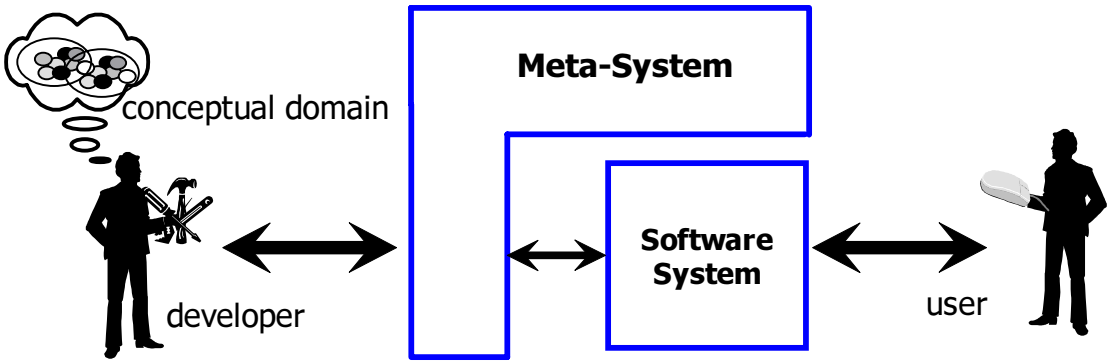


Figura 3.2 Una arquitectura con Dos Niveles de Abstracción

5 Semántica

Tal y como se concluyó en el anterior capítulo los autores de los documentos deben construir y estructurar sus propios dominios conceptuales. Basándose en algunos tipos de redes conceptuales u ontologías, los autores pueden caracterizar sus dominios de información y confeccionar sus documentos. Esto significa que los sistemas de autor deben proporcionar mecanismos flexibles y abiertos para la representación del conocimiento [Brachman, 1985b] que permitan la descripción y estructuración semántica de un sistema de información hipermedia. Sin embargo, quedan algunas preguntas sin respuesta: ¿qué se entiende por ontología?, ¿cómo se representará formalmente esta ontología?

5.1 Ontologías

Una ontología [Uschold, 1996] [Guarino, 1995] "*is an explicit specification of a conceptualization*" [Gruber, 1995], "*is an explicit knowledge level specification of a conceptualization, (...) which may be affected by the particular domain and task it is intended for*" [Van Heijst, 1997], "*representating knowledge for the purpose of solving some problem is strongly affected by the nature of the problem and the inference strategy to be applied to the problem*". "*Firstly, the application task determines to a large extent which kinds of knowledge should be enconded. (...) Secondly, the knowledge must be encoded in such a way that the inference strategy used can reason efficiently*" [Bylander, 1988]. En suma, una descripción de los términos mediante las relaciones que se pueden establecer entre ellos.

Sin duda, esta primera aproximación hace de la ontología un mecanismo adecuado para la representación de la semántica de estructuración. No obstante, esta propuesta es demasiado débil para el modelo SEM-HP porque en él no sólo se quiere dar cuenta de lo que existe, sino que también se debe poder especificar su estructura. El cómo se organice el conocimiento supone el uso de algún lenguaje. Si se prescinde de las reglas sintácticas y semánticas, el término 'ontología' será simplemente sinónimo de 'terminología'.

Por esta razón, otros autores dan una definición mucho más concreta de ontología; para ellos "... *describes a taxonomy of concepts for that task or domain that define the semantic interpretation of the knowledge*" [Alberts, 1993]. Basándose en esta interpretación se han propuesto lenguajes declarativos y gráficos de representación del conocimiento. Estos lenguajes incorporan una sintaxis, unas reglas de inferencia, un vocabulario de símbolos no-lógicos y, algunas veces, una representación gráfica —mapas de conceptos [Sowa, 1991]— que restringen la interpretación de los símbolos en el vocabulario. La mayoría de ellos se pueden clasificar en alguno de los siguientes formalismos:

- Sistemas basados en Lógica Descriptiva —LD— [Devanbu, 1994]. Este tipo de sistemas adopta una perspectiva centrada en objetos, donde el mundo es modelado mediante individuos unidos con relaciones binarias y agrupados en clases —denominados conceptos—. En cada sistema basado en LD, los conceptos del dominio de aplicación se definen mediante descripciones en las que se usan otros conceptos. Éstas se construyen a partir de conceptos atómicos, roles y constructores que proporciona el propio lenguaje. Los Sistemas basados en LD ofrecen una gran variedad de mecanismos de inferencia, tales como la inclusión —

subsumption—, detección de inconsistencias, pertenencias, etc. Son formalismos de la inteligencia artificial que hacen posible la representación de dominios de conocimiento centrándose en las clases de objetos y sus relaciones, y ofreciendo inferencias sobre la estructura de clases.

- Sistemas basados en conocimiento, descendientes de KL-ONE [Brachman, 1985a, 1990] [Woods, 1992] denominados Sistemas Terminológicos —tales como, BACK [Peltason, 1989], CLASSIC [Borgida, 1989, 1992], Kris [Baader, 1991], LOOM [McGregor, 1988]— Estos sistemas pretenden hacer explícita la terminología de un dominio de aplicación de forma similar a los sistemas basados en LD. Pero, además, clasifican automáticamente estas definiciones y consultas basadas en taxonomías, de acuerdo a las relaciones semánticas tales como la inclusión y la equivalencia.

A pesar de ser buenos mecanismos de representación del conocimiento y razonamiento de amplio uso en áreas como la inteligencia artificial, la lingüística y sistemas basados en conocimiento presentan serias limitaciones:

- La interpretación del domino conceptual es categórico, en el sentido de que estas lógicas consideran que el mundo está formado por objetos elementales —agrupados en conceptos— y relaciones binarias entre ellos. Esto quiere decir que las relaciones entre los conceptos generan una estructura estrictamente jerárquica, obligando al autor, en la mayoría de los casos, a realizar una representación del conocimiento impropia o artificial.
- Siguen sin proporcionarnos mecanismos que nos permitan añadir restricciones sobre las relaciones que se pueden llegar a establecer entre los conceptos. En nuestra opinión una ontología debería ser algo más que una simple taxonomía de términos.

Aunque estos inconvenientes son importantes, se pueden solventar con el uso de algunas lógicas descriptivas especiales propuestas por [Nebel, 1990, 1991], [Baader, 1990] y [De Giacomo, 1994]. Pero existe una tercera traba crucial que hace imposible el uso de estos formalismos en el modelo SEM-HP: no ofrecen ningún mecanismo para hacer evolucionar a estos lenguajes. En resumen, "*aunque los LDs aparentemente están equipados con procedimientos completos y sólidos de razonamiento y terminación, adolecen de bastantes limitaciones no sólo a la hora de representar dominios completos sino también para representar mecanismos de evolución eficientes*" [Hurtado-Torres, 2001]. Como alternativa se propone el uso de grafos y restricciones sobre este formalismo. La explicación detallada del mismo y su uso se hará en el capítulo 5 dedicado a la formalización del modelo SEM-HP.

6 Conclusiones

La Sistémica, la Evolución del Software y la Semántica son las bases teóricas en las que se fundamenta el modelo SEM-HP. La Teoría de Sistemas —en particular la Teoría del Sistema General— y la evolución entendida como proceso de maduración y, en concreto, la perspectiva adoptada por MEDES y su arquitectura de dos niveles de abstracción —Sistema Software y Metasistema— se adoptan en esta propuesta. Para representar la semántica de estructuración se hará uso de ontologías. No obstan-

te, se descartan los mecanismos de representación basados en Lógicas Descriptivas y KL-ONE lenguajes porque no incluyen ningún tipo de soporte para hacerlos evolucionar.

7 Referencias

[Agresti, 1986] 61
[Alberts, 1993] 64
[Amano, 2000] 61
[Aoyanne, 2000] 61
[Baader, 1990] 65
[Baader, 1991] 65
[Banerjee, 1987] 61
[Belady, 1976] 61
[Bertalanffy, 1968, 1977] 58
[Bertalanffy, 1968, p.53] 58
[Bertalanffy, 1975, p. 95] 58
[Berzins, 1993] 61
[Boehm, 1986] 61
[Booch, 1999] 61
[Borgida, 1989, 1992] 65
[Brachman, 1985, 1990] 65
[Brachman, 1985] 64
[Bylander, 1988] 64
[Casais, 1990] 61
[Conradi, 1993] 60
[De Giacomo, 1994] 65
[Devanbu, 1994] 64
[García-Cabrera, 2000] 57
[Garzotto, 1993] 60
[Goldberg, 1995] 59
[Gruber, 1995] 64
[Guarino, 1995] 64
[Heckel, 2001] 61
[Henderson-Sellers, 1990] 61
[Hurtado-Torres, 2001] 65
[Isakowitz, 1995] 60
[Kozaczynski, 1992] 61
[Lange, 1996] 60
[Le Moigne, 1977, 1990] 58
[Le Moigne, 1977, p. 155] 58
[Le Moigne, 1990] 58
[Lehman, 2000] 61
[McGregor, 1988] 65
[Mens, 2001] 61
[Morin, 1977, p. 159] 59
[Morin, 1977, p. 177] 58
[Morin, 1977, p.136] 58
[Morin, 1977, p.l67] 58
[Morin, 1977] 58
[Murugesan, 1999] 59
[Nebel, 1990, 1991] 65
[Niertrasz, 2000] 61
[Parets-Llorca, 1993, 1995a, 1995b, 1996a, 1996b, 62
[Parets-Llorca, 1995a] 61, 62
[Parets-Llorca, 1995b] 59
[Peltason, 1989] 65
[Rao, 1990] 57
[Rodríguez-Fórtiz, 2000b] 60
[Rumbaugh, 1991] 61
[Said, 2001] 61
[Schwabe, 1996] 60
[Sowa, 1991] 64
[Tardieu, 1983] 61
[Uschold, 1996] 64
[Van Heijst, 1997] 64
[Wermenlinger, 2001] 61
[Woods, 1992] 65

Capítulo 4

Descripción Conceptual de SEM-HP

«Hay ocasiones en la vida en las que para salir airosamente hace falta estar un poco loco.»

François de la Rochefoucauld.
Escritor francés (1613-1680).

«Donde truena un hecho, ten la certeza de que ha relampagueado una idea.»

Ippoleto Nievo.
Escritor iltaliano (1831-1861).

«Un vieaje de diez mil kilómetros empieza por un solo paso.»

Proverbio Chino.

Resumen

Del análisis de los distintos modelos de sistema hipermedia propuestos en la literatura técnica se deduce que son incapaces de representar los procesos de desarrollo, mantenimiento y navegación de un sistema de información en continua evolución.

Por esta razón, se sugiere un modelo cognitivo compuesto por tres Sistemas interrelacionados y en incesante evolución: el *Sistema de Conocimiento*, *el Sistema de Navegación* y el *Sistema de Aprendizaje*. En cada uno de estos Sistemas se hace una representación explícita del conocimiento suministrado. De esta forma, el modelo ofrece la posibilidad de realizar una interpretación de la estructura semántica de los distintos dominios de información.

La representación explícita dota al Sistema Hipermedia de una funcionalidad añadida; el autor puede controlar de manera flexible los procesos de desarrollo, mantenimiento y evolución del mismo.

1 Introducción 69
2 El Proceso General de Desarrollo de Hipermedias 70
2.1 Evolución durante el Proceso de Desarrollo 70
3 Características de SEM-HP 72
4 Arquitectura General del Modelo SEM-HP 73
5 Terminología de SEM-HP 74
5.1 Ítemes de Información 74
5.2 Estructura Conceptual 79
5.3 Restricciones 84
5.4 Acciones Evolutivas y Meta-restricciones 88
6 Arquitectura Detallada de SEM-HP 90
6.1 El Sistema de Conocimiento SC 91
6.2 El Sistema de Navegación SN 93
6.3 El Sistema de Aprendizaje SA 96
6.4 Sistema Hipermedia e Hiperdocumento 97
7 El Proceso de Desarrollo Detallado 98
7.1 Evolución del Subsistema de Memorización 100
7.2 Evolución del Subsistema de Presentación 101
7.3 Evolución del Sistema de Navegación 101
7.4 Evolución en el Modelo SEM-HP 101
8 Referencias 102

1 Introducción

Las principales actividades relacionadas con el uso y el desarrollo de los Sistemas Hipermedia evolucionan a lo largo del tiempo. Los Sistemas Hipermedia necesitan modelos que respalden la naturaleza evolutiva de su construcción, mantenimiento y navegación. La Teoría del Sistema General [Le Moigne, 1990], la Evolución del Software [Parets-Torres, 1996] y los modelos cognitivos ofrecen una mejor perspectiva de los sistemas *web* y consiguen salir triunfantes en la abstracción de su estructura, información y comportamiento.

Un Sistema Hipermedia Evolutivo debe ser modelado mediante un conjunto de sistemas interrelacionados e interactivos que permitan:

a) desde el punto de vista del autor, un control flexible y completo del desarrollo y mantenimiento del hiperdocumento;

b) desde la perspectiva del lector, una navegación que facilite la comprensión, el acceso y la selección de la información.

En este capítulo, se propone el Modelo SEM-HP para la construcción de Sistemas Hipermedia Evolutivos con las siguientes características:

- Admite una representación explícita del contenido semántico que permite:
  - realizar una estructuración del sistema de información, y
  - determinar sus posibilidades de cambio, actualización y evolución.
- Es lo suficientemente flexible como para ofrecer los mecanismos necesarios para incorporar y representar los dominios conceptuales del autor con el fin de lograr la caracterización de los dominios de información.

Se dedica este capítulo, pues, a la descripción conceptual del Modelo SEM-HP, esto es, a presentar sus Sistemas, a detallar su ESTRUCTURA —apartado 4.1 del capítulo 3—, a determinar la funcionalidad de los mismos y de sus componentes y, por último, a explicar en qué consiste y cómo se consigue hacer evolucionar a los Sistemas.

Para ello, se comienza por detallar, de manera general, las etapas y características del proceso de desarrollo de un Sistema Hipermedia. A continuación, en el apartado 3, se pormenorizan las características básicas del Modelo SEM-HP, haciendo hincapié en las peculiaridades que hacen de él un Modelo útil. En el apartado 4 se introduce la arquitectura general del Modelo, definiendo someramente los Sistemas que lo conforman. Después, en el apartado 5, se definen y explican los conceptos más importantes del Modelo: Ítemes de Información, Estructura Conceptual, Restricciones, Acciones Evolutivas y Meta-Restricciones. El apartado 6 proporciona una extensa descripción del Sistema de Conocimiento y del Sistema de Navegación. Finalmente, en el apartado 7, se explica detalladamente el proceso de desarrollo de Sistemas Hipermedia cuando se aplica el Modelo SEM-HP.

En consecuencia, a grades rasgos, la exposición se centra en la descripción del proceso de desarrollo de Sistemas Hipermedia —apartado 2—, del Modelo SEM-HP —apartados 3-6—, y de cómo el Modelo SEM-HP respalda todo el proceso de desarrollo de Sistemas Hipermedia —apartado 7—.

2 El Proceso General de Desarrollo de Hipermedias

Desde nuestro punto de vista, los Sistemas Hipermedia no son más que Sistemas de Información especiales que se basan en un domino conceptual. El autor del mismo lo representará de acuerdo a los conocimientos que tenga sobre el tema y a la finalidad que le quiera imprimir. Un Sistema Hipermedia, en última instancia, se dedica a capturar, presentar y recorrer los esquemas cognitivos del autor sobre un determinado asunto. En consecuencia, en el proceso de desarrollo de Sistemas Hipermedia se pueden distinguir tres fases de diseño interrelacionadas:

fase 1. Creación de mundos conceptuales y de información,

fase 2. Selección de presentaciones o vistas de éstos, y

fase 3. Navegación de estas presentaciones.

El desarrollador, que en el contexto de los sistemas hipermedia se denomina *autor*, se encarga precisamente de diseñar, construir y mantener el hipermedia durante estas fases; mientras que la interacción del *lector*, el usuario final de los sistemas hipermedia, estará centrada principalmente en la navegación que debe estar apoyada por mecanismos que faciliten la comprensión, el acceso y la selección de la información.

2.1 Evolución durante el Proceso de Desarrollo

El proceso de desarrollo de Sistemas Hipermedia, al igual que cualquier otro proceso de desarrollo de software, es iterativo y evolutivo —apartado 3.1 del capítulo 3—. Su proceso de creación estará sujeto a continuos cambios y afectará a diversos aspectos, tales como:

- los conceptos que se quieren ofrecer —fase 1—,
- el modo en el que se pretende relacionar o estructurar estos conceptos —fase 1—,
- los documentos o ítemes de información que explican estos conceptos —fase 1—,
- la manera en la que se quiere presentar o ver esta información —fase 2— y, finalmente,
- la forma en la que se pretende que el lector recorra esta información —fase 3—.

Esto quiere decir que, para mejorar la calidad de los Sistemas Hipermedia, es imprescindible que el Sistema de Desarrollo asista al autor durante el proceso de construcción, de adaptación y de mejora de los mismos, en cada una de sus etapas. Para ello, el Sistema de Desarrollo cuenta con cuatro mecanismos: *Acciones Evolutivas*, *Restricciones*, *Propagación de los Cambios* y *Meta-Restricciones*.

Mediante las *Acciones Evolutivas*, el desarrollador, se encarga del diseño y de la construcción de mundos conceptuales y de información, presentaciones y navegaciones. En todo momento, el Sis-

tema Hipermedia debe estar en un estado coherente. Para ello, tanto el Sistema de Desarrollo como el autor especifican y asocian *Restricciones* a las Acciones de Evolución que garantizan la integridad del mismo de dos formas distintas:

- Llevando a cabo únicamente aquellas modificaciones que desembocan en estados consistentes.
- Propagando los cambios con objeto de que el resto de las fases del Hipermedia evolucionen de acuerdo a la modificación realizada.

Puesto que los creadores también pueden cambiar las Restricciones, el Sistema de Desarrollo deberá garantizar también la consistencia de esa modificación. Para ello fija un conjunto de *Meta-Restricciones* que regulan el modo en que se pueden cambiar dichas Restricciones.

Las Acciones Evolutivas y las Meta-Restricciones forman parte del Sistema de Desarrollo o Meta-Sistema —apartado 4.1 del capítulo 3— cuya misión es realizar cambios estructurales en el Sistema Hipermedia manteniendo su *integridad* y haciéndolo evolucionar.

Como puede observarse en la figura 4.1, el autor interacciona con el Meta-Sistema para diseñar y construir la ESTRUCTURA de cada fase del Sistema Hipermedia. Para ello hará uso de las Acciones Evolutivas que estarán controladas por las restricciones que se establezcan por el Meta-Sistema y por el mismo. La consistencia del cambio de las restricciones definidas por el autor se garantiza con las Meta-Restricciones. Finalmente, el autor puede comunicarse también con el Sistema Hipermedia para consultar y comprobar su desarrollo. Por su parte, el lector lo único que puede hacer es recorrer las presentaciones. Sin embargo, se beneficia de este modo de entender los hipermedia al proporcionarle una navegación contextualizada.

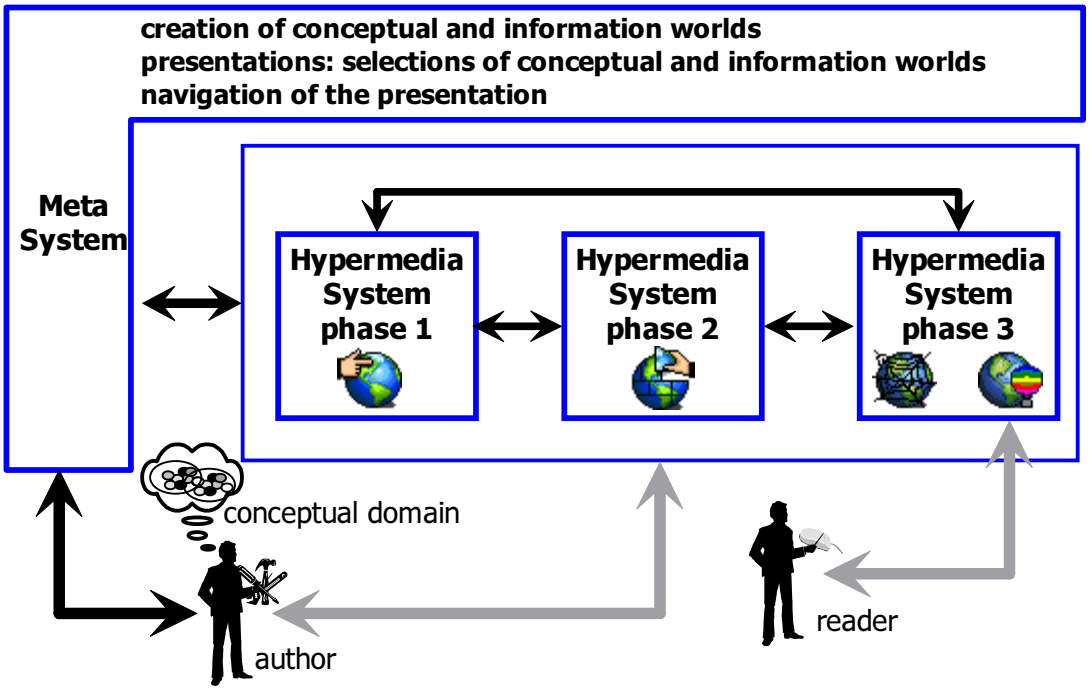


Figura 4.1 El Proceso de Desarrollo de Sistemas Hipermedia.

3 Características de SEM-HP

El esquema tradicional de los modelos basados en una jerarquía de niveles traducibles a una metodología más o menos secuencial y estática —capítulo 2— parece impropio para representar una realidad compleja y cambiante, en donde la construcción, el mantenimiento y el uso —la navegación— se confunden por sus fuertes interdependencias. En estos modelos no existe una correspondencia entre Estructura y Funcionalidad —apartado 4.1 del capítulo 3—, y no asisten al autor en el proceso de desarrollo de Sistemas Hipermedia.

Como alternativa se propone un Modelo **S**istémico —basado en la Teoría de Sistemas—, **E**volutivo y **SEM**ántico para el diseño de Sistemas **HiP**ermedia, **SEM-HP**, con las siguientes características:

1. El modelo facilita y hace flexible la construcción, el mantenimiento y la navegación de los sistemas hipermedia. Estos tres procesos clave estarán expuestos a continuos cambios y adaptaciones, y el modelo es capaz de integrarlos.
2. La representación explícita de la semántica de estructuración impregna el modelo, ya que es la que posibilita su estructuración y la que determina su posible cambio, adaptación y evolución. Esto es, el proceso de diseño y construcción está basado en un modelo cognitivo [García-Cabrera, 2000] [García-Cabrera, 2001b].
3. Beneficia a sus usuarios —autor y lector—, durante el desarrollo y actividades de uso: construcción, mantenimiento y navegación, puesto que:
   - El autor puede realizar un proceso de diseño incremental de sus hiperdocumentos.
   - La colaboración entre autores se convierte en una realidad.
   - Se puede realizar un buen mantenimiento debido a que el proceso de diseño y razonamiento subyacente o *Desing Rationale* [Shipman, 1997] llevado a cabo puede ser representado.
   - El lector cuenta con un acceso contextualizado que facilita la comprensión del hiperdocumento.
4. El modelo ofrece una representación semántica flexible con la que el autor puede caracterizar sus propios dominios de información mediante sus propias ontologías.
5. Cada uno de los Sistemas sigue la arquitectura de dos niveles de abstracción de MEDES —apartado 4 del capítulo 3—: a) un Meta-Sistema o Sistema de Desarrollo que se dedica a construir la ESTRUCTURA del Sistema Hipermedia y b) el Sistema Hipermedia que ofrece una determinada FUNCIONALIDAD mediante la ejecución de los componentes de su ESTRUCTURA.

4 Arquitectura General del Modelo SEM-HP

SEM-HP modela un Sistema Hipermedia mediante una serie de sistemas interrelacionados y en interacción. Cada uno de estos sistemas registra algunas de las características del modelo y, por tanto, ofrece ciertas funciones al resto de los sistemas y, en última instancia, a los usuarios —autores y lectores— que interactúan con él.

Tres son los Sistemas que integran este modelo: el *Sistema de Conocimiento*, el *Sistema de Navegación* y el *Sistema de Aprendizaje*:

1. El **Sistema de Conocimiento** tiene como objetivo principal el almacenamiento, la estructuración y el mantenimiento de las distintas piezas de información. Permite el almacenamiento del conocimiento adquirido sobre el dominio de información que se quiere representar. Este conocimiento guiará el proceso de diseño y de estructuración del sistema de información, al mismo tiempo que determina las posibilidades de transformación y cambio de esa estructura a lo largo del tiempo. Desde el punto de vista del proceso de desarrollo de Sistemas Hipermedia, el Sistema de Conocimiento soporta el diseño, construcción y mantenimiento de las fases 1 y 2.

2. El **Sistema de Navegación** asiste al lector en su proceso de interacción con el sistema de información. En cada instante de tiempo, de manera dinámica, basándose en la base de conocimiento y la propia actividad del lector, este sistema determina:

   - en primer lugar, el dominio de información al que tiene acceso, y
   - en segundo lugar, sus posibilidades de interacción.

   Desde la perspectiva del proceso de desarrollo, este Sistema permite el diseño, construcción y mantenimiento de la fase 3.

3. Por último, el **Sistema de Aprendizaje** optima el proceso de adquisición de conocimiento a partir del sistema hipermedia adaptando la navegación a las necesidades de información y a los conocimientos que va logrando del lector. De manera dinámica contribuye a mejorar la navegación a partir de un determinado modelo de usuario.

Cada uno de estos Sistemas redefine, y en algún caso extiende, la base de conocimiento proporcionada por el *Sistema de Conocimiento*, que es estable para el lector pero dinámica para el/la diseñador/a o herramienta de autor. El lector interactúa únicamente con el Sistema de Navegación, mientras que el autor puede hacer uso tanto del Sistema de Conocimiento como del propio Sistema de Navegación. En consecuencia, para el primero el sistema de información no cambia; sin embargo, para el segundo, el sistema de información está en un continuo proceso de maduración —apartado 3.2 del capítulo 3—.

Cada Sistema se sustenta a sí mismo y contribuye con información adicional. Esta información determinará qué piezas de información se pueden consultar y bajo qué prisma. Los distintos Sistemas

interactúan entre sí y su interacción produce, de manera dinámica o evolutiva, adaptaciones dentro y fuera de los mismos.

Como puede observarse en la figura 4.2, los distintos sistemas interactúan entre ellos —flechas negras—. El Sistema de Conocimiento con los Sistemas de Navegación y Aprendizaje, y el Sistema de Aprendizaje con el de Navegación. El lector únicamente se comunica con el Sistema de Navegación y el de Aprendizaje, mientras que el autor interactúa con los tres Sistemas —flechas grises—.

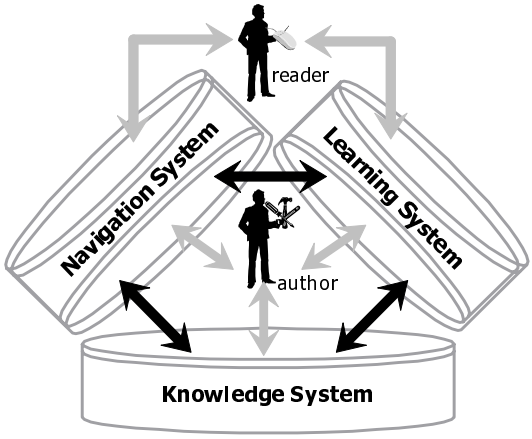


Figura 4.2 Arquitectura General del Modelo SEM-HP.

5 Terminología de SEM-HP

En este apartado se presentan las bases conceptuales y la definición de los términos que se utilizan en el Modelo SEM-HP. La exposición se centra en los conceptos más importantes de nuestro modelo: *los Ítemes de Información* —II—, la *Estructura Conceptual* —EC—, las *Restricciones* —RT—, las *Acciones Evolutivas* —AC_e— y las *Meta-Restricciones* —M-RT—.

En su explicación irán apareciendo otros conceptos básicos para el Modelo, al tiempo que se desvelan las interrelaciones entre los mismos. El objetivo principal será mostrar cómo la Estructura Conceptual, las Restricciones y las Acciones Evolutivas consiguen desempeñar un papel destacado en los aspectos cognitivos y evolutivos de un sistema hipermedia.

5.1 Ítemes de Información

Un Sistema Hipermedia es un sistema de información que está formado por distintos elementos o piezas de información a los que se puede hacer referencia, usar y componer. A estas piezas de información se las llama *Ítemes de Información*.

5.1.1 ¿Por qué Ítem?

Se ha preferido el término "ítem" en lugar de "*chunk*" de información más extendido en la literatura hipermedia. Dos razones justifican esta decisión:

a) por reconocimiento y fidelidad histórica —es el vocablo que usó Vannevar Bush en su memorable y precursor artículo "*As We May Think*" [Bush, 1945]—; y

b) porque, realmente, se considera que el término "ítem de información" remarca más la idea de que esa pieza de información tiene una entidad propia y constituye una unidad conceptual.

Se rechaza la posibilidad de tomar como unidad de información, una página o un documento porque:

a) sin duda, estos vocablos tienen más que ver con la edición y la presentación visual de una determinada información y,

b) realmente, en la práctica, en una página o documento puede aparecer más de un trozo de información con entidad conceptual; esto es, una página suele estar formada por varios *chunks* o ítemes.

5.1.2 Las Propiedades de los Ítemes de Información

El autor del Sistema Hipermedia puede asociar *Propiedad*es a estos ítemes de información. Éstas dicen, de algún modo, el tipo de información con la que se está trabajando y la función que puede desempeñar un ítem en un determinado contexto.

La propiedad más importante de un ítem de información es el conjunto de *Concepto*s a los que está asociado; esto es, un ítem de información será identificado por uno o más conceptos o ideas etiquetadas. Estos conceptos formarán parte de un determinado *Dominio Conceptual* creado por el autor durante el proceso de desarrollo y mantenimiento del Sistema Hipermedia.

El conjunto de ítemes de información etiquetados o identificados mediante los conceptos de un Dominio Conceptual pertenecen al denominado *Dominio de Información*. Un ítem puede estar asociado a más de un concepto de un mismo Dominio Conceptual pero también podría pertenecer a más de un Dominio Conceptual. Esto es, los Dominios Conceptuales determinan subconjuntos no disjuntos de ítemes de información.

Para ilustrar las definiciones, explicaciones y descripciones del modelo se va a utilizar en lo sucesivo un dominio conceptual y de información concreto sobre el *Sistema Solar* —figura 4.3—.

En el ejemplo de la figura 4.3 se puede observar que el dominio conceptual "Sistema Solar" cuenta con 14 conceptos, representados mediante elipses, como *Planets*, *Stars*, etc y con 17 ítemes, representados por cuadrados, como *Su1*, *C2*, etc, que forman parte en este momento del Dominio de Información de este Dominio Conceptual.

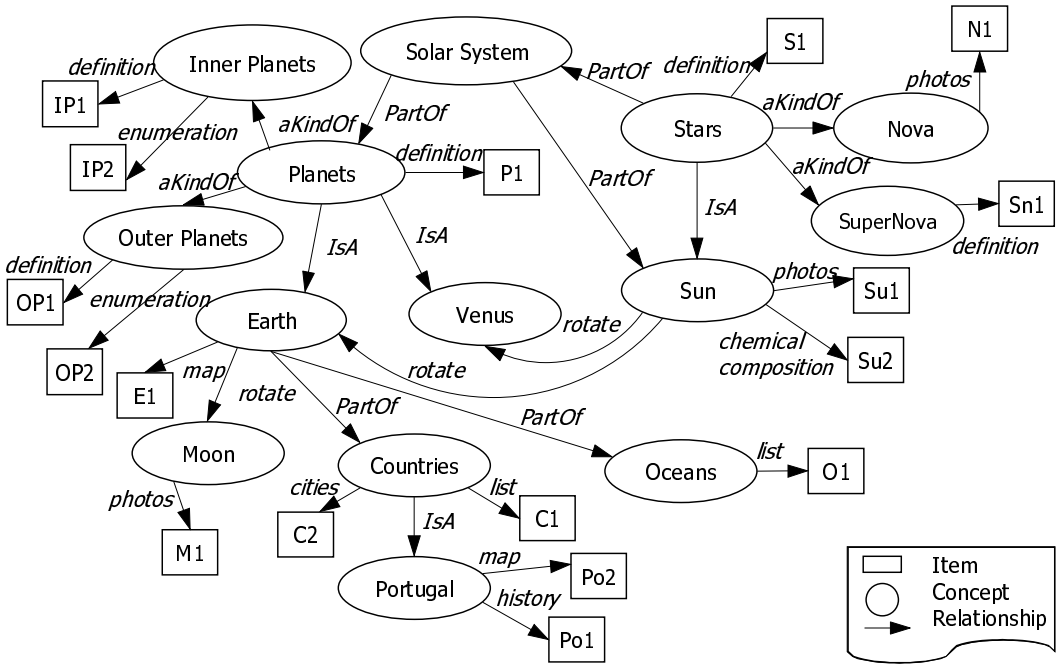


Figura 4.3 Estructura Conceptual —EC— sobre el *Sistema de Solar*.

Tal y como puede observarse, los conceptos son dependientes del Dominio Conceptual, esto quiere decir que deben ser creados por los autores expertos en la materia sobre la que trata el documento. Estos conceptos se establecen a lo largo del proceso de elaboración y mantenimiento del documento.

Ahora bien, sobre un concepto concreto se pueden elaborar más de un ítem de información. Cada uno, desempeñará un papel distinto dentro de un determinado contexto. A esta función o intención del ítem de información se le llama *Rol*. Los ítemes de información, por tanto, se asocian a los conceptos mediante una determinada función. Esta propiedad le será de gran utilidad:

- al autor, porque le proporciona información sobre la función del ítem; y
- también para el lector, porque le revelará la intención de una determinada asociación contextualizando su navegación.

El Modelo proporciona algunos de los roles que desempeña una pieza de información —tabla 4.1 —. Se ha intentado proporcionar las funciones más habituales. Dado que este conjunto de funciones puede resultar incompleto o inadecuado en algún dominio conceptual, el modelo admite la incorporación de tantas funciones como el autor considere oportunas al diccionario de roles.

Por ejemplo, con el concepto *Sun* —uno de las *Stars*— se identifica más de un ítem de información, sobre este concepto el Sistema Hipermedia puede ofrecer *photos* o *chemical-composition.* En este caso el autor ha considerado oportuno incorporar dos nuevas funciones.

una **INTRODUCCIÓN** o exordio.	una **DEFINICIÓN**	un **EJEMPLO**
una **ACLARACIÓN** (nota al pie)	una **BIBLIOGRAFÍA** relacionada	una RECETA o **ALGORITMO**
una **PROFUNDIZACIÓN** (algo más formal relacionado con el tema, una navegación vertical)	unas **RECOMENDACIONES** (advertencias o sugerencias).	un RAZONAMIENTO o **EXPLICACIÓN**.
una **OPINIÓN** (dictamen, juicio o parecer que se forma de una cosa cuestionable).	unos **ANTECEDENTES** (acción, dicho o circunstancia anterior que sirve para juzgar hechos posteriores).	una **COMPARACIÓN** (confrontar uno o más métodos, líneas de pensamiento, autores, etc.).
un **ANÁLISIS** de un tema o problema (distinción y separación de las partes de un todo hasta llegar a conocer sus principios o elementos).	una **AMPLIACIÓN** DE INFORMACIÓN (algo más extenso, más detalles, una navegación horizontal)	una **JUSTIFICACIÓN** (las causas, razones que nos llevan a tratar un determinado tema, aspecto o problema).
otros **ENFOQUES** o puntos de vista del mismo tema o problema (dirigir la atención o el interés hacia un asunto o problema desde unos supuestos previos, para tratar de resolverlo acertadamente).	una **VALORACIÓN** objetiva o subjetiva (reconocer, estimar o apreciar el valor o mérito de un persona o cosa).	

Tabla 4.1 Roles que los Ítemes de Información pueden desempeñar en un contexto.

Además de conceptos y roles, un ítem de información tiene otras propiedades adicionales:

- el *Tipo de Lenguaje* en que se expresan sus contenidos —texto, sonido, gráfico, imagen, animación, ejecución o hipermedia—,
- las relacionadas con los *Aspectos de Edición* —autor/es, fecha de edición, última fecha de revisión, organismo, calidad, número de visitantes, tipo de visitantes, organismos que lo recomiendan, personas que lo recomiendan, idioma—, y
- la versión o *Nivel de Especialización*. Esta propiedad es interesante, pues se puede llegar a tener más de una definición de un concepto —por ejemplo *Stars*— según el grado de especialización del lector al que va dirigida. De esta manera, el sistema hipermedia ofrece la posibilidad de seleccionar —tal y como se verá en el apartado dedicado a las restricciones— un determinado material para un lector o grupo de lectores específicos.

Siguiendo con el ejemplo —figura 4.4—, el ítem de información *N1* podría estar caracterizado con las siguientes propiedades y valores: Concepto = *Nova*, Función = *photos*, Lenguaje-expresión = *image*, Organismo = *PlanetEducation*, fecha-creación = *26feb00*, Nivel = *teenagers*.

5.1.3 El Contenido de los Ítemes de Información

El contenido de los ítemes de información será creado y editado por el autor. Desde el *Sistema de Conocimiento* se facilitará la construcción, la reutilización y la composición de estas piezas de información. Dado que el contenido y las propiedades, son informaciones que se mantienen por separado, los contenidos podrán ser suministrados y editados por alguna otra aplicación *software* o base de datos

de información. El autor haciendo uso de la herramienta de desarrollo se dedicará a incorporar ítemes y a decidir:

- qué informaciones constituyen una unidad conceptual, y
- qué propiedades caracterizan esa pieza de información.

La composición no es algo nuevo, muchos de los modelos semánticos de la literatura técnica incorporan este tipos de mecanismos. Sin embargo, se dedican a construir únicamente documentos que son una composición de datos representados y presentados según una plantilla de estilo, a modo de un formulario de una base de datos. En nuestra opinión esto no es suficiente porque:

- no todos los escritos responden a este esquema tan sencillo,
- esta forma de proceder tiende a la creación de hiperespacios y no de verdaderos espacios conceptuales —apartado 5.2.1 del capítulo 1—. Los datos de un texto no tienen porqué reflejar su temática ni su función.
- no son capaces de incorporar cualquier tipo de información —expresado con cualquier medio como imágenes, sonido, animaciones, etc—.

El modelo que aquí se presenta admite cualquier pieza de información que constituya una entidad conceptual, provenga del propio sistema de autor, de una aplicación o base de datos de información.

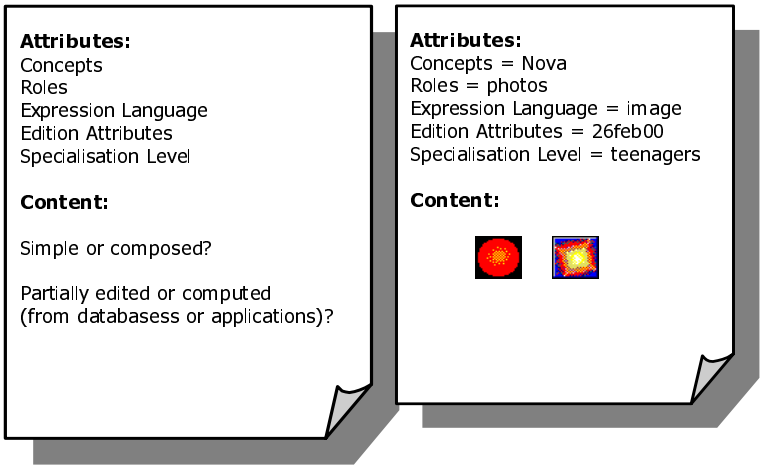


Figura 4.4 Ítem de Información: Propiedades y Contenido.

5.1.4 Definiciones

Un **Concepto** es una idea, pensamiento o abstracción que puede ser etiquetado por el autor con el fin de hacer explícito su conocimiento y hacerlo comprensible. De manera breve, es una idea etiquetada.

Un **Ítem de Información** es cualquier trozo de información identificable en un sistema hipermedia y que representa una unidad conceptual del sistema de información. Cada uno de estos ítemes de información tiene una serie de propiedades que describen el tipo y funcionalidad de la información que contienen.

El **Nivel de Especialización** o versión es una de las propiedades asociadas a un ítem de información que determina el grado o nivel de especialización con el que se expresa o presenta la información contenida en el ítem.

Una **Propiedad** de un ítem de información es aquella información que está asociada a un ítem de información y que describe el tipo de información que contiene, su función y su comportamiento.

Un ítem de información puede desempeñar diferentes ***Roles*** o **Funciones** en el contexto de un sistema de información. Desde el punto de vista del autor, un ítem puede realizar o desempeñar un determinado *rol* en el contexto. Desde la perspectiva del lector, éste sigue un enlace con la intención de alcanzar un determinado tipo de información sobre un concepto concreto.

5.2 Estructura Conceptual

El conjunto de los conceptos pertenecientes a un Dominio Conceptual constituye un grafo, en el que los nodos y los enlaces son etiquetados con un significado. Este grafo o red semántica [Sowa, 1991] representa el Dominio Conceptual de un determinado Dominio de Información.

En consecuencia, el *Dominio Conceptual* estará formado por los conceptos y también por las asociaciones que el autor defina entre los mismos. A estas asociaciones se las denomina *Asociaciones Conceptuales*. Por su parte, el *Dominio de Información* estará integrado por todos aquellos ítemes de información y por las asociaciones que les unen a uno o más conceptos del Domino Conceptual. Estos ítemes se asocian con los conceptos mediante los roles, o funciones a las que se llama *Asociaciones Funcionales*. Dado que una asociación es una correspondencia entre dos nodos se puede decir de forma breve que el conjunto de Asociaciones Conceptuales es el Dominio Conceptual y el de Asociaciones Funcionales el Dominio de Información. Con objeto de permitir un desarrollo provisional e incompleto, el Modelo también admite la posibilidad de incluir ítemes que, por el momento, no estén asociados a ningún concepto.

Por ejemplo, en la figura 4.5 se muestra un ejemplo abstracto de dos Estructuras Conceptuales. Tres de los ítemes —representados por rectángulos— no están asociados en esta instantánea a ninguno de los conceptos.

El Dominio Conceptual —las Asociaciones Conceptuales— junto al Dominio de Información asociado mediante las funciones —las Asociaciones Funcionales— forman un grafo al que se denomina *Estructura Conceptual* —EC—. Este grafo tiene dos tipos de nodos: conceptos e ítemes de información. Los conceptos pueden estar asociados entre sí pero los ítemes únicamente puede ser asociados a los conceptos.

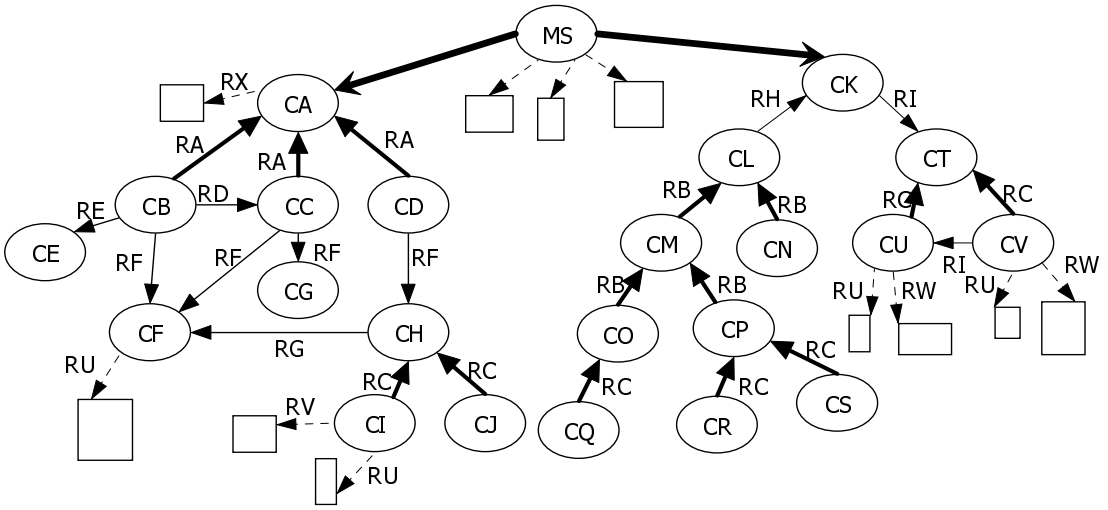


Figura 4.5 Ejemplos Abstractos de Estructuras Conceptuales del Sistema de Conocimiento.

Por tanto, la Estructura Conceptual se puede definir más formalmente como: EC = (C, II, A_c, A_f), donde

- C es el conjunto de conceptos pertenecientes a un Dominio Conceptual,
- II es el conjunto de ítemes de información que forma parte del Dominio de Información asociado a ese Dominio Conceptual,
- A_c es el conjunto de asociaciones entre conceptos o *Asociaciones Conceptuales* etiquetadas o Dominio Conceptual y
- A_f es el conjunto de asociaciones entre conceptos e ítemes de información o *Asociaciones Funcionales* etiquetadas —el Dominio de Información—.

En la figura 4.3 tenemos un ejemplo concreto de Estructura Conceptual cuyo grafo está formado por los siguientes conjuntos de nodos y asociaciones:

C= {Solar-System, Planets, Inner-Planets, Outer-Planets, Stars, Nova, Supernova, Earth, Venus, Sun, Moon, Countries, Oceans, Portugal}

II={P1, IP1, IP2, OP1, OP2, E1, M1, C1, C2, Po1, Po2, O1, Su1, Su2, S1, N1, Sn1}

A_c= {<Earth, rotate, Moon>, <Earth, PartOf, Countries>, <Earth, PartOf, Oceans>, <Sun, rotate, Earth>, <Sun, rotate, Venus>, <Countries, IsA, Portugal>, <Solar_System, PartOf, Planets>, <Solar_System, PartOf, Sun>, <Stars, IsA, Sun>, <Stars, PartOf, Solar_System>, <Stars, aKindOf, Nova>, <Stars, aKindOf, Supernova>, <Planets, IsA, Earth>, <Planets, IsA, Venus>, <Planets, aKindOf, Inner-Planets>, <Planets, aKindOf, Outer-Planets> }

A_f= {<Planets, definition, P1>, <Inner-Planets, definition, IP1>, <Inner-Planets, enumeration, IP2>, <Outer-Planets, definition, OP1>, <Outer-Planets, enumeration, IP2>, <Earth, map, E1>, <Moon, photos, M1>, <Countries, list, C1>, <Countries, cities, C2>, <Portugal, map, Po2>, <Portugal, history, Po1>, <Oceans, list, O1>, <Sun, photos, Su1>, <Sun, quimical composition, Su2>, <Stars, definition, S1>, <Nova, photos, N1>, <Supernova, definition, Sn1>}

5.2.1 Asociaciones Conceptuales

Antes de seguir con la explicación, se quiere aclarar el porqué se ha optado por usar el vocablo "asociación" en lugar de "enlace" más difundido en la literatura técnica. En nuestra opinión, la palabra "enlace" tiene una clara connotación estática y ambigua en los actuales modelos. El término asociación refleja el hecho de que la conexión entre dos ítemes de información obedece a la relación que puede existir entre el/los conceptos que cada uno representa, y no a razones circunstanciales —de edición o estética—, tal y como ahora ocurre con los enlaces.

Continuando con la exposición, en la Estructura Conceptual se distinguen dos tipos de Asociaciones Conceptuales a las que se llama: *Asociaciones Conceptuales de Referencia* y *Asociaciones Conceptuales de Dependencia*.

Las primeras, al igual que los conceptos, están supeditadas al Dominio Conceptual y, por tanto, deben ser definidas por el autor para cada dominio particular. El creador del Sistema Hipermedia tiene que proporcionar sus propias ontologías [Uschold, 1996]. Estas ontologías —conceptos y relaciones entre conceptos— definen un conjunto o diccionario de palabras clave que el autor usará para estructurar y mantener sus documentos hipermedia, y que el lector empleará para seleccionar un determinado ítem de información.

Volviendo al ejemplo de la figura 4.3, las siguientes son Asociaciones Conceptuales de Referencia:

{<Earth, rotate, Moon>, <Sun, rotate, Earth>, <Sun, rotate, Venus>}

En cuanto a las Asociaciones Conceptuales de Dependencia, son aquellas relaciones de conexión o subordinación que son independientes del Dominio Conceptual y que tienen un carácter genérico. Se distinguen las siguientes: agregación —*ParteDe* o *PartOf*—, la instanciación —*EsUn* o *IsA*— y la especificación —*UnTipoDe* o *aKindOf*—.

La dependencia *ParteDe* permitirá establecer una cierta jerarquía entre algunos de los conceptos. La dependencia *UnTipoDe* da la posibilidad de distinguir entre distintas modalidades de un mismo concepto. Por último, la dependencia *esUn* permite definir un concepto usando otros más genéricos.

En la Estructura Conceptual sobre el Sistema Solar —figura 4.3— se encuentran las siguientes Asociaciones Conceptuales de Dependencia:

{<Earth, PartOf, Countries>, <Earth, PartOf, Oceans>, <Countries, IsA, Portugal>, <Solar_System, PartOf, Planets>, <Solar_System, PartOf, Sun>, <Stars, IsA, Sun>, <Stars, PartOf, Solar_System>, <Stars, aKindOf, Nova>, <Stars, aKindOf, Supernova>, <Planets, IsA, Earth>, <Planets, IsA, Venus>, <Planets, aKindOf, Inner-Planets>, <Planets, aKindOf, Outer-Planets>}

5.2.2 Prestaciones de las Asociaciones Conceptuales

Las Asociaciones Conceptuales nos permiten estructurar el conocimiento. Esta semántica de estructuración va a ser clave a la hora de construir y mantener el Sistema de Información porque explicita las posibles asociaciones que se pueden establecer entre la información.

Además, ofrecen otro tipo de prestaciones como la posibilidad de componer ítemes de información de una forma lógica; es decir, pueden agruparse los ítemes asociados a conceptos que mantienen un mismo tipo de relación con otro concepto.

Por ejemplo —figuras 4.3 y 4.6—, los conceptos *Inner Planets* y *Outer Planets* mantienen una asociación en la que participa la misma relación, *aKindOf*, con *Planets*. En consecuencia, se puede construir un ítem compuesto agrupando todos los ítemes asociados con los conceptos hijos. El nuevo concepto será etiquetado con el concepto más general, *Planets*, y adquiere las funciones de todos los ítemes que se han reunido, *definition* y *Enumeration*.

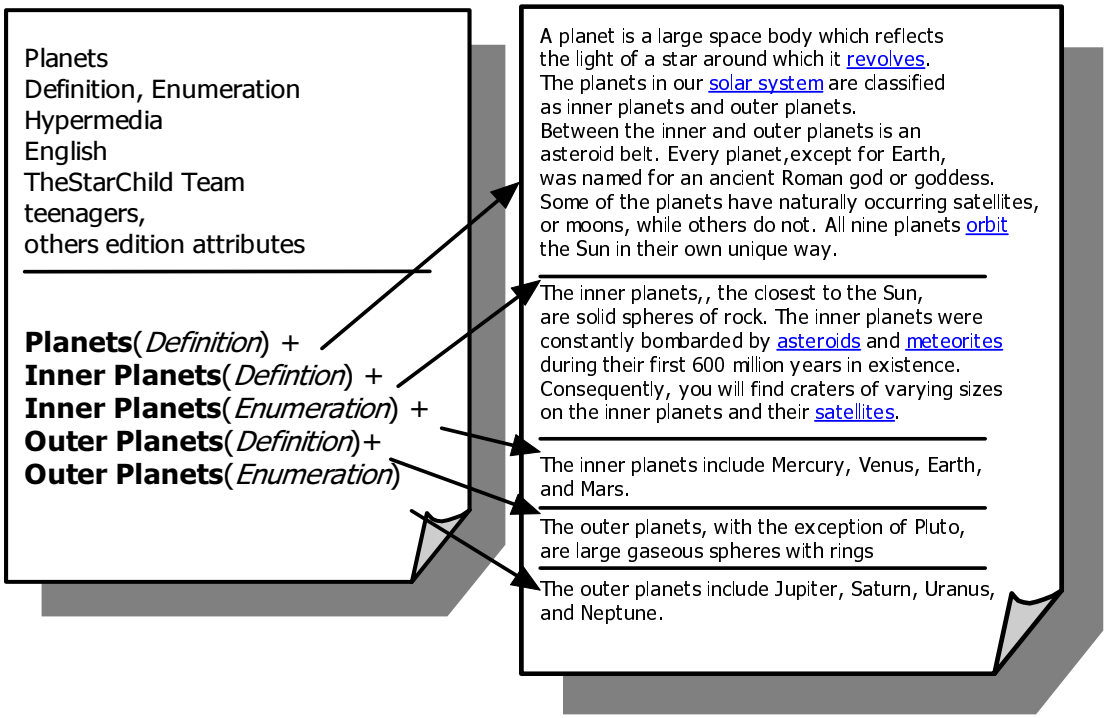


Figura 4.6 Composición de Ítemes de Información.

Por último, las Asociaciones Conceptuales conceden la posibilidad de definir lo que se denomina, *Entorno Conceptual*, esto es, el conjunto de conceptos que están relacionados con otro concepto. Dicho de otro modo, el conjunto de conceptos que alcanzan a un concepto y los que pueden ser alcanzados desde el mismo.

En el ejemplo de la figura 4.3, en el entorno conceptual más inmediato —que alcanzan o son alcanzados de forma directa— de *Stars* se encuentran los conceptos *Solar-System*, *Nova*, *Supernova* y *Sun*.

La noción de Entorno Conceptual incorpora al Modelo una serie de operaciones muy interesantes conocidas en la literatura técnica como preguntas basadas en la estructura —*queries based on the structure* [Bieber, 1997b]—. Estas operaciones se convierten en un mecanismo de control que

facilita el proceso de desarrollo al autor ya que con ellas puede comprobar si ha organizado su conocimiento como deseaba. También mejoran la interacción del lector proporcionándole una estrategia de recuperación de información contextualizada. Algunos ejemplos de estas operaciones son:

- Qué conceptos complementan —añaden información— a uno dado.
- Qué conceptos se derivan de uno dado.
- Qué conceptos producen o causan uno dado.
- Cuáles son los conceptos que se encuentra un nivel arriba o abajo en la Estructura Conceptual.
- Qué conceptos distan una distancia *d* de uno dado, esto es, alcanzan o pueden ser alcanzados recorriendo antes o después *d* conceptos.
- Qué documentos están relacionados con algún domino conceptual. En este caso, la herramienta de autor debe desplegar una aplicación de búsqueda que seleccione aquellos documentos con un entorno conceptual semejante.

Resumiendo, las asociaciones entre los conceptos facilitan la creación dinámica y flexible del Sistema Hipermedia porque establecen espacios conceptuales, proporcionan la posibilidad de construir nuevos ítemes y, finalmente, asisten al autor y al lector en su interacción con el entorno conceptual.

Las *Restricciones* —tal y como luego se verá en el apartado 5.3— sobre las asociaciones conceptuales guían el proceso de construcción y el mantenimiento de la Estructura Conceptual, pues aconsejan o prohíben determinadas estructuraciones y asociaciones en un dominio de información determinado.

5.2.3 Definiciones

Una **Asociación Conceptual** es una relación etiquetada entre dos conceptos.

Una **Asociación Conceptual de Dependencia** es una asociación etiquetada entre dos conceptos que es independiente del Dominio Conceptual considerado {*PartOf*, *aKindOf*, *IsA*}.

Una **Asociación Conceptual de Referencia** es una asociación etiquetada entre dos conceptos, dependiente de un Dominio Conceptual y, por tanto, son definidas por el autor.

Una **Asociación Funcional** es una relación etiquetada con un *rol* o función entre un concepto y un ítem de información. De este ítem se dirá que pertenece al Dominio de Información que determina el Dominio Conceptual que incluye a ese concepto.

Un **Dominio Conceptual** es el conjunto conceptos con los que se puede identificar los distintos ítemes de información de un sistema hipermedia y el conjunto de asociaciones que se pueden establecer entre estos conceptos, esto es, el conjunto de Asociaciones Conceptuales de una Estructura Conceptual.

Un **Dominio de Información** es el conjunto de ítemes de información identificados con conceptos que pertenecen a un Dominio Conceptual concreto, esto es, el conjunto de Asociaciones Funcionales de una Estructura Conceptual.

Un **Entorno Conceptual** es el conjunto de conceptos relacionados con un determinado concepto, esto es, que alcanzan o son alcanzados por el mismo.

La **Estructura Conceptual** de un Dominio Conceptual es una red o grafo etiquetado cuyos nodos son conceptos e ítemes y, en la que se mantiene información sobre:

a) las asociaciones entre conceptos, y

b) las asociaciones entre conceptos e ítemes de información o asociaciones funcionales.

5.3 Restricciones

Las *Restricciones*, RT, guían la construcción, el mantenimiento y la navegación de los Sistemas Hipermedia —el proceso de desarrollo—. Son aportadas desde los distintos Sistemas de SEM-HP, y todas ellas son aplicadas, tal y como se verá más adelante, por ellos mismos. Fundamentalmente, se dedican a:

a) delimitar las asociaciones entre conceptos que se pueden efectuar en la Estructura Conceptual. Esto es, el autor o el sistema pueden imponer algún tipo de limitación estableciendo qué conceptos se pueden asociar y mediante qué asociaciones.

b) restringir las asociaciones entre ítemes de información que nos podemos encontrar durante la navegación. De manera dinámica, para un determinado ítem de información se cumplirán un conjunto de restricciones que limitarán el conjunto de ítemes asociados. Se denomina a este conjunto *Entorno de un Ítem de Información*.

A las restricciones que pertenecen al primer conjunto se les denomina *Restricciones Estructurales* mientras que a las del segundo se las llama *Restricciones de Navegación*.

Independientemente del grupo al que pertenezcan estas restricciones, habrá restricciones básicas impuestas desde el Sistema de Desarrollo que no cambian y a las que se denomina *Restricciones de Sistema*, RT_s. El autor también puede incluir restricciones adicionales que puede modificar en cualquier momento, y que reciben el nombre de *Restricciones de Autor*, RT_a.

5.3.1 Las Restricciones Estructurales

Las Restricciones Estructurales son aquellas que se basan en la semántica de estructuración del sistema de información representada mediante la Estructura Conceptual. Estas restricciones se especifican desde el *Sistema de Conocimiento* y son aplicadas para restringir la construcción de la Estructura Conceptual.

Algunas de estas restricciones son de sistema, RT_s, y otras son de autor, RT_a. Las restricciones de sistema establecen invariantes básicos que no pueden ser transgredidos si se quiere mantener la integridad del Sistema.

Algunos ejemplos de este primer conjunto de restricciones son:

- Cada asociación de la Estructura Conceptual debe conectar dos conceptos, o un concepto y un ítem.
- Cada arco y nodo de la Estructura Conceptual debe estar etiquetado.
- No puede existir en una Estructura Conceptual dos nodos con la misma etiqueta.

El autor también puede especificar restricciones adicionales. Éstas determinan qué conjunto de asociaciones entre conceptos son posibles en esa Estructura Conceptual. Son descritas mediante algún formalismo y le permiten al Sistema comprobar si la Estructura Conceptual es válida, durante el proceso de creación de la misma, de acuerdo a las restricciones establecidas.

Algunos ejemplos pueden ser los siguientes:

- El concepto *Star*s puede ser conectado con el concepto *Planets* mediante la asociación *rotate*.
- La asociación *rotate* debe ser acíclica.
- Un concepto X puede ser conectado con el concepto *Countries* si el concepto *Countries* previamente tiene una asociación con el concepto *Earth*.

5.3.2 Las Restricciones de Navegación

Este grupo de restricciones está destinado a diseñar y controlar la propia navegación. Las restricciones de navegación ofrecen la posibilidad de realizar distintos *estilos* de navegación sobre una misma estructura semántica. Mediante estas restricciones será factible tener en cuenta:

- El tipo de navegación, estableciendo un grupo de restricciones que limiten más o menos los caminos que pueden ser recorridos o navegados dentro de una Estructura Conceptual.
- La navegación llevada a cabo por el usuario a lo largo del tiempo o historia funcional, que puede definirse como el conjunto de operaciones realizadas por el usuario durante una sesión de trabajo. Esto es, los ítemes de información seleccionados por el lector, y el orden en el que los ha visitado.

- Consideraciones sobre la seguridad y el control de acceso. La identificación del usuario o la procedencia de la conexión restringen la Estructura Conceptual a la que se puede acceder, fija los roles a las que no se puede acceder, y las versiones que no son accesibles para ese lector.

Obviamente, la navegación, en última instancia, estará restringida por el *mundo ontológico* que haya concebido y diseñado el autor. No se pueden establecer restricciones por parte del autor que, de algún modo, contradigan la Estructura Conceptual que se pretende recorrer.

Por ejemplo, para la estructura conceptual de la izquierda de la figura 4.5, el autor no podría fijar como restricción de navegación: *antes de visitar el ítem que está unido a CI con RV se debe visitar el ítem que está asociado a CF mediante RU*. Es imposible llegar a cumplirla porque no es posible alcanzar CI desde CF.

Estas restricciones serán descritas formalmente mediante teoría de grafos y un lenguaje basado en lógica temporal —una descripción detallada del uso de estas formalizaciones se verá en el siguiente capítulo— que admite expresiones como: "*si el lector ha visitado algún ítem del concepto H podrá ver todos los ítemes de J*", "*teniendo en cuenta que el lector conoce tal y tales ítemes, que están relacionados con estos conceptos y que ha recorrido tal.... entonces se puede mostrar tal y tal ítem*".

Al igual que Stotts y Furuta [Stotts, 1998], consideramos que un hipermedia es un *documento interactivo* que proporciona una *estructura dinámica*. De esta afirmación se deduce que se hace necesario expresar en lógica temporal las secuencias de enlaces que se pueden seguir durante la navegación. Efectivamente, estos autores proponen este formalismo para comprobar las propiedades del hipermedia; esto es, si la navegación diseñada se comporta tal y como espera el diseñador. En la propuesta que aquí se está presentando también haremos uso de lógica temporal como mecanismo inherente para expresar las restricciones de navegación.

En consecuencia, este tipo de reglas condiciona, en cada instante, las piezas que se pueden visitar en ese momento y desde que ítemes de información pueden ser alcanzados. Estas restricciones son incorporadas por el autor del hipermedia, y elegidas indirectamente por el lector, al especificar un tipo de navegación, por ejemplo, o al navegar a través del hipermedia.

Volviendo al ejemplo, el creador del Sistema Hipermedia podría especificar mediante estas restricciones que no se muestren los conceptos *Nova* o *Supernova* y sus ítemes asociados hasta que el usuario no conozca —no haya visitado— la *definición* del concepto *Stars*.

5.3.3 Prestaciones de las Restricciones

Las restricciones juegan un papel fundamental en la evolución del Sistema Hipermedia participando activamente en el proceso de desarrollo del mismo —apartado 2—. En cada una de sus fases se proporcionan operaciones —a las que se denomina Acciones Evolutivas, apartado 5.4— para su diseño y construcción.

A estas operaciones se les asocian las Restricciones definidas con objeto de guiar este desarrollo. Es decir, las Restricciones son las precondiciones y postcondiciones de estas acciones de evolu-

ción. Únicamente si se cumple la operación será llevada a cabo. De este modo, se consigue que el cambio de cualquier componente del Sistema Hipermedia lo mantega íntegro.

Como ya se comentó anteriormente los autores también pueden cambiar las restricciones de autor. En consecuencia, el Meta-Sistema especifica un conjunto de Meta-Restricciones que se asocian a estas Restricciones para garantizar de nuevo que el conjunto de Restricciones no contiene ningún tipo de contradicción. Las Acciones Evolutivas y las Meta-Restricciones serán explicadas con mayor detalle en el apartado dedicado a tal efecto —apartado 5.4—.

En la tabla 4.2 se resumen los tipos de restricciones descritas caracterizándolas mediante el Sistema desde el que se establecen, los responsables de definirlas y las limitaciones a las que están sujetas.

Tipo de Restricción	Sistema	Definidas por	Respetan
Restricciones Estructurales	Conocimiento	Sistema y Autor	Las de Autor deben respetar las de Sistema
Restricciones de Navegación	Navegación	Autor	Restricciones Estructurales

Tabla 4.2 Clasificación de las Restricciones.

5.3.4 Definiciones

El **Entorno de un Ítem de Información** es el conjunto de restricciones que se cumplen en el momento de alcanzar dicho ítem y que delimita o restringe el conjunto de ítemes de información que se pueden alcanzar desde el mismo.

Las **Restricciones** son el conjunto de condiciones o reglas que limitan las asociaciones conceptuales y de navegación. Garantizan la consistencia de los distintos componentes de los Sistemas y realizan la función de precondiciones para las acciones de evolución.

Las **Restricciones de Autor**, RT_a, son el subconjunto de restricciones que son establecidas por el autor del Sistema Hipermedia y que pueden ser modificadas en cualquier momento.

Las **Restricciones Estructurales** son aquellas restricciones que delimitan las asociaciones entre conceptos que se pueden efectuar en la Estructura Conceptual. Pueden ser impuestas tanto por el sistema como por el autor desde el Sistema de Conocimiento.

Las **Restricciones de Navegación** son aquellas restricciones que restringen las asociaciones entre ítemes de información que nos podemos encontrar durante la navegación. Pueden ser especificadas por el autor desde el Sistema de Navegación y deben respetar la Estructura Conceptual en la que se basa.

Las **Restricciones de Sistema**, RT_s, son el subconjunto de restricciones impuesto a priori y que no cambia.

5.4 Acciones Evolutivas y Meta-restricciones

Hasta ahora se han descrito las piezas de información con las que se construye un documento con el modelo SEM-HP, las propiedades que los caracterizan, los contenidos de esos ítemes, y la estructura conceptual y demás restricciones que controlan el proceso de estructuración y navegación.

Un cuarto elemento es clave en el modelo: las operaciones que permiten soportar de manera consistente todos y cada uno de sus componentes a las que se denominan *Acciones Evolutivas* o *de Evolución* —AC_e —. Todos los Sistemas incluyen un conjunto de acciones de evolución mediante las cuales se podrá cambiar y propagar, de manera dinámica, el desarrollo del sistema hipermedia.

Con estas operaciones el diseñador que interactúa con el Sistema Hipermedia podrá construir y mantener —hacer evolucionar— cada uno de los Sistemas. En general, estas acciones ocasionan un registro del cambio efecturado y desencadenan comprobaciones, basadas en algunas de las restricciones, sobre la repercusión de esa modificación en cada uno de los Sistemas. Dicho de otro modo, estas operaciones deben verificar un conjunto de Restricciones —RT — que, por supuesto, también pueden cambiar, aunque únicamente aquellas que hayan sido definidas por el autor.

5.4.1 Propagación del Cambio y Meta-Restricciones

Como ya se ha apuntado, estas Restricciones impiden la violación de la integridad y garantizan la consistencia de los Sistemas. Para ello, siempre que se lleve a cabo un determinada *Acción de Evolución* perteneciente a uno de los Sistemas, se deberán acometer las siguientes tareas:

1. En primer lugar, se deberá comprobar que el elemento que se ha cambiado conserva su integridad. Obviamente, las *Restricciones de Sistema* que representan aquellos aspectos que son básicos o fundamentales, no admiten ningún tipo de modificación.
2. A continuación, el Sistema deberá propagar el cambio al resto de los elementos del Sistema. Esto quiere decir, que se está haciendo una propagación del cambio dentro del propio Sistema con el fin de garantizar su consistencia.
3. Por último, el Sistema debe hacer partícipes del cambio realizado dentro del Sistema al resto de los Sistemas de SEM-HP con el fin de que se adapten al mismo. Esta propagación externa asegura la integridad de todos los Sistemas y, por tanto, la del Sistema Hipermedia.

Para hacer referencia de manera general a estas tres actuaciones, que se desencadenan por haber realizado una Acción Evolutiva, se usará la expresión: *Propagación del Cambio*.

El sistema también provee al creador de operaciones para modificar las Restricciones de Autor. En este caso, es necesario asegurar que el conjunto de Restricciones definidas por el autor no contenga contradicción alguna. Para ello, se suministran un conjunto de *Meta-Restricciones* que se encargan de comprobar en todo momento que las acciones de evolución efectuadas para modificar las Restricciones de Autor son consistentes. Como ya se dijo, las Acciones Evolutivas y las Meta-Restricciones forman parte de lo que en MEDES —apartado 4 del capítulo 3— se llama Meta-Sistema.

5.4.2 Prestaciones de las Acciones Evolutivas

La herramienta de autor construida con este modelo es capaz de advertir sobre las consecuencias de una Acción Evolutiva, prohibiéndola cuando su aplicación haga imposible el funcionamiento del sistema de información. Concretamente, este modelo fácilmente incorpora algunas *comprobaciones de Validez* [Bieber, 1997b] sobre la posibilidad de establecer una determinada asociación entre ítemes, tales como:

- *¿pertenece ese concepto al Dominio Conceptual del documento?*,
- *¿existe un ítem de información etiquetado de ese modo?*, o
- *¿se puede añadir esta nueva restricción sin contradecir las anteriores?*.

Y también *pruebas de Relevancia*, tales como:

- *¿esa asociación es posible desde ahí?*,
- *¿es recomendable que se ofrezca toda la información etiquetada bajo ese concepto, o no?*,
- *el sistema de conocimiento, de navegación o, incluso, el de aprendizaje, ¿imponen algún tipo de restricción?, ¿de qué tipo?, ¿por qué?*

No obstante, normalmente el sistema puede trabajar con cierto grado de inconsistencia, lo que posibilitará una construcción y mantenimiento flexible del sistema de información.

> Por ejemplo, ítemes de información sin etiquetar o parcialmente etiquetados, conceptos no aplicados a ningún ítem de información, conceptos no asociados al grafo conceptual, ausencia de restricciones estructurales o de navegación, etc.

El sistema puede detectar estas incoherencias, pero no siempre será capaz de realizar un tratamiento automático. Esto será posible cuando en el sistema tengamos restricciones que determinen la interpretación de esa información y, por tanto, las acciones de evolución a las que puede ser sometida. Esto es, si se dispone de una buena semántica cognitiva, es posible un control computado de esa información. Dicho de otro modo, una estructura conceptual bien especificada, unos ítemes de información caracterizados con sus propiedades, un conjunto de restricciones de estructuración y de navegación, aseguran la asistencia mediante el Sistema de Desarrollo.

En ausencia de una representación semántica completa el sistema no es inútil porque admite, al menos, un mantenimiento semiautomático o manual bajo la interpretación del autor. El desarrollador puede seguir haciendo consultas sobre la semántica de estructuración —apartado 5.2.2— para tomar decisiones de diseño.

En consecuencia, la herramienta que implementa el modelo ofrece un mecanismo flexible que trabaja en un mundo imperfecto, que actúa y ayuda de forma automática cuando es posible y que, siempre, facilita el mantenimiento manual haciéndolo lo más llevadero posible.

5.4.4 Definiciones

Las **Acciones Evolutivas** son el conjunto de operaciones que puede cambiar los artefactos y restricciones de los distintos Sistemas. Una acción de evolución debe verificar un conjunto de Restricciones y, de esta forma, garantizar la consistencia del Sistema y llevar a cabo la propagación del cambio dentro y fuera del Sistema.

Las **Meta-Restricciones** son restricciones definidas a priori y que garantizan la consistencia de las Acciones Evolutivas que cambian las Restricciones de Autor.

Se utiliza la expresión **Propagación del Cambio** para hacer referencia a los tres tipos de actuaciones —comprobación de la consistencia del cambio, propagación interna y propagación externa— que se desencadenan cuando el autor realiza una Acción Evolutiva.

6 Arquitectura Detallada de SEM-HP

Cada uno de los Sistemas de SEM-HP se puede definir mediante:

a) uno o más artefactos que representan su particular visión del Dominio Conceptual y de Información,

b) un conjunto de Restricciones, RT, que controlan la construcción y garantizan la consistencia de cada uno de estos artefactos, y

c) un conjunto de Acciones Evolutivas, AC_e, que nos permiten hacer cambios y propagarlos en el sistema hipermedia.

A continuación se definen y describen de forma más detallada los distintos Sistemas y sus posibles componentes. No obstante, el acercamiento que se hace en este apartado es parcial, pues tiene como único objetivo presentar de manera general el Modelo y aclarar la funcionalidad de los distintos Sistemas. La descripción estrictamente formal y completa de SEM-HP se aborda en el siguiente capítulo de esta memoria. Con objeto de facilitar la comprensión de la exposición se han reunido las anteriores definiciones del apartado 5 en el apéndice A.

6.1 El Sistema de Conocimiento SC

El objetivo principal del *Sistema de Conocimiento* —SC— es el almacenamiento, la estructuración y el mantenimiento de las distintas piezas de información. Está formado por un *Subsistema de Memorización* y un *Subsistema de Presentación*.

6.1.1 El Subsistema de Memorización SM

El *Subsistema de Memorización* —SM— permite el almacenamiento del conocimiento adquirido sobre cada *Dominio de Información —Asociaciones Funcionales—*. Memoriza toda la información relacionada con un *Dominio Conceptual* concreto —Asociaciones Conceptuales— al que pertenecerá el Dominio de Información que queremos presentar. Por tanto, los elementos que se manejan desde este Subsistema son estos mismos:

1. El Dominio Conceptual bajo el cual debe ser concebido el hipermedia y con el que se catalogarán los Ítemes de Información.
2. El Dominio Funcional al que pertenecen los Ítemes de Información, esto es, las distintas piezas de información que se pueden utilizar a la hora de construir los hipermedia. Estos ítemes de información estarán expresados en algunos de los posibles lenguajes —tales como texto, sonido, gráfico, ejecución, imagen, animación o hipermedia— y podrán estar catalogados con uno o varios de los conceptos del dominio. Finalmente, podrán estar etiquetados con una o más funciones en un contexto determinado. A estos ítemes se les podrán asociar, asimismo, determinadas propiedades de edición y el nivel de especialización.

Ambos dominios conforman la *Estructura Conceptual*, EC_m, que se formaliza —tal y como más tarde se verá— mediante un grafo dirigido, $EC_m = (C, II, A_c, A_f)$, donde

- C es el conjunto de conceptos,
- II es el conjunto de ítemes de información,
- A_c es el conjunto de asociaciones conceptuales etiquetadas y
- A_f es el conjunto de asociaciones funcionales etiquetadas.

De manera resumida, se puede decir que el *Subsistema de Memorización* establece la materia prima con la que se debe construir el Sistema Hipermedia, sus posibles componentes y la naturaleza o comportamiento de los mismos.

La Estructura Conceptual será construida por los autores de manera dinámica, poco a poco, mediante un proceso iterativo de maduración. Para ello, el Subsistema incorpora algunas acciones evolutivas AC_{em} de creación, borrado y modificación, tales como: *añadir-concepto*, *borrar-asociación*, *añadir-ítem* etc.

Estas acciones deben verificar un conjunto de Restricciones, RT, a las que se encomienda el mantenimiento de la consistencia de la EC_m. Estas restricciones podrán ser básicas, RT_s, en cuyo caso forman parte funcional del *Subsistema de Memorización;* o bien pueden estar definidas por el autor, RT_a —como se describió en el apartado 5.3—. Por tanto, el *Subsistema de Memorización* se define como $SM = (EC_m, RT, AC_{em})$, donde

- EC_m es el grafo anteriormente definido que representa el dominio conceptual y de información del sistema hipermedia,

- RT es el conjunto de restricciones, y
- AC_{em} es el conjunto de acciones de evolución.

6.1.2 El Subsistema de Presentación SP

El *Subsistema de Presentación* —SP— determina el conjunto de las posibles vistas o presentaciones para un determinado Dominio Conceptual y de Información. De alguna manera, establece los subdominios conceptuales y de información potenciales que se pueden llegar a construir con los ítemes del *Subsistema de Memorización*.

El *Subsistema de Presentación*, partiendo de la Estructura Conceptual que le ofrece el *Subsistema de Memorización*, le permite al autor del hipermedia seleccionar un subconjunto de los conceptos y de las asociaciones de EC_m. Este nuevo grafo —EC_p, un subgrafo de EC_m— será el que se presentará al lector. Por tanto, la Estructura Conceptual del Subsistema de Presentación se define como $EC_p = (C_p, II_p, A_{cp}, A_{fp})$ donde,

- C_p, II_p, A_{cp} y A_{fp} son subconjuntos de C, II, A_c y A_f, respectivamente.

La Subestructura Conceptual elegida debe respetar, por supuesto, todas las Restricciones —RT— establecidas en el *Subsistema de Memorización*. Cada vez que el creador/a cambie esta subestructura, el sistema deberá comprobar que el nuevo Subdominio Conceptual y de Información seleccionado verifica estas Restricciones.

A modo de ejemplo, suponga que del dominio conceptual sobre el Sistema Solar —figura 4.3— el autor decide mostrar únicamente el subgrafo formado por el concepto *Earth* y todos los conceptos y asociaciones que se pueden alcanzar desde este concepto —figura 4.7—.

Como resultado, el *Subsistema de Presentación* se define como $SP = (EC_p, RT, AC_{ep})$, donde

- EC_p es una subestructura de la Estructura Conceptual original EC_m,
- RT es el mismo conjunto de Restricciones del *Subsistema de Memorización* y
- AC_{ep} es el conjunto de acciones evolutivas que le permiten al autor limitar o reducir la EC_m.

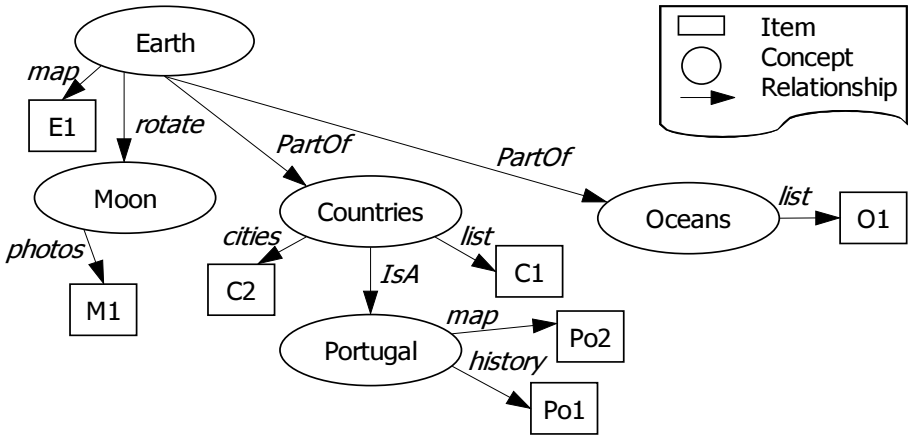


Figura 4.7 Una Subestructura Conceptual —ECp— seleccionada a partir de EC.

6.1.3 Prestaciones del Sistema de Conocimiento

Como vemos, el *Sistema de Conocimiento* almacena las distintas piezas de conocimiento de los mundos conceptuales sobre los que versan los documentos del autor. También se encarga de estructurar semánticamente estos ítemes de información.

Además, este Sistema le ofrece al autor la posibilidad de establecer Restricciones de Autor, RT_a. Haciendo uso de estas Restricciones el sistema puede ayudar al autor en la creación y mantenimiento —garantizando su consistencia— de sus dominios conceptuales y de información. Esto quiere decir que el modelo admite una construcción flexible, parcialmente computada, dependiendo de la información con la que se cuente.

Una representación más exhaustiva y formal hará posible su interpretación por el sistema; una representación más informal e incompleta será interpretada por el autor pero, en cualquier caso, el sistema lo asiste en su proceso de creación y de mantenimiento mediante la aplicación de ese conocimiento desde los dos Subsistemas.

6.1.4 Definiciones

El **Sistema de Conocimiento** es uno de los Sistemas del modelo SEM-HP que modelan un sistema hipermedia. Proporciona los ítemes de información, su catalogación y las reglas básicas que establecen sus posibles asociaciones. Está formado por los *Subsistemas de Memorización* y *Presentación*.

El **Subsistema de Memorización** es uno de los dos subsistemas del *Sistema de Conocimiento*. Establece la materia prima con la que se ha de construir el sistema hipermedia con un componente fundamental: la Estructura Conceptual.

El **Subsistema de Presentación** es uno de los dos subsistemas que modelan el *Sistema de Conocimiento* de un sistema hipermedia. Permite seleccionar un subconjunto de una Estructura Conceptual con el fin de determinar el documento hipermedia que se mostrará mediante el *Sistema de Navegación*.

6.2 El Sistema de Navegación SN

El Modelo no está completo si nos olvidamos de la funcionalidad básica de un documento hipermedia: su navegación. Por tanto, el modelo cuenta también con un *Sistema de Navegación* que permite recorrer o *rememorar* el conocimiento almacenado y adaptarlo a las características y a la interacción del lector. El *Sistema de Navegación* nos posibilita ordenar, de algún modo, la Estructura Conceptual asociada al mismo, proporcionada por el *Subsistema de Presentación*.

En consecuencia, con este Sistema interactúan tanto el autor como el lector, pero con distintos niveles de abstracción. El desarrollador podrá comunicarse con el Meta-Sistema o Sistema de Desa-

rrollo para especificar la navegación, y también con el Sistema Software o Navegador para comprobar que su diseño ha sido correcto. Sin embargo, el lector únicamente hace uso del navegador para explorar la información que le ha seleccionado el autor y en el modo que éste haya determinado.

6.2.1 Perspectiva del Autor

Desde el *Sistema de Navegación*, basándose en la Subestructura Conceptual, EC_p, proporcionada por el *Subsistema de Presentación*, el autor puede añadir más restricciones —si lo considera oportuno— con objeto de establecer caminos más restringidos en el subgrafo.

Por ejemplo, la Subestructura EC_p seleccionada de la figura 4.7 establece que si el lector se encuentra visitando alguno de los ítemes de información —C1 ó C2— asociados al concepto *Countries*, entonces podrá alcanzar todos los ítemes relacionados con el concepto *Portugal*—Po1 y Po2—. Esto es lo que determina la EC_p; sin embargo, el autor podría decidir que esta forma de navegar no es la apropiada. Así, por ejemplo, puede limitar estos recorridos de forma que:

- si se ha visitado el ítem asociado mediante *cities* a *Countries*, sí puede visitar el mapa y la historia de Portugal.
- si se ha visitado el ítem asociado mediante *list* a *Countries* solamente se puede mostrar el mapa de Portugal.

Estas Restricciones o reglas de navegación, RT_n, se expresarán formalmente mediante un lenguaje basado en lógica temporal. A partir de la EC_p y de las restricciones temporales se puede construir una red de Petri de forma automática. Las redes de Petri [Peterson, 1981] dotan de una semántica operacional a las fórmulas en lógica temporal, al tiempo que ofrecen una navegación ejecutable. El lenguaje lógico para describir restricciones de navegación, así como el algoritmo que construye la red de Petri a partir de las fórmulas en lógica temporal, se describirá más adelante —en el capítulo 5—.

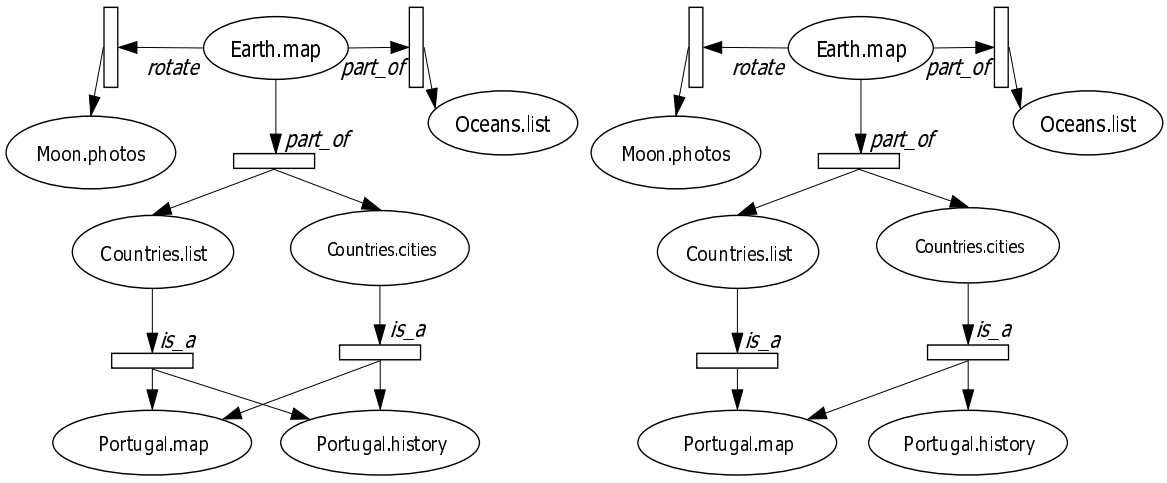


Figura 4.8 a) La Red de Petri que genera EC_p de la figura 4.7, b) La Red de Petri que genera EC_p a la que se ha añadido una restricción más de navegación.

En la figura 4.8 se muestra, a la izquierda, la red de Petri que se genera a partir de la EC_p de la figura 4.7 sin ninguna restricción de navegación adicional. A la derecha, se puede ver la red de Petri que se calcularía a partir de la EC_p y de la restricción de navegación establecidas por el autor descritas dos párrafos más arriba.

En consecuencia, el *Sistema de Navegación* se define como SN = (EC_p, RT_n, RP, AC_{en}), donde

- EC_p es una presentación determinada;
- RT_n es el conjunto de restricciones especificada por el autor mediante lógica temporal;
- RP es la red de Petri y AC_{en} es el conjunto de acciones de evolución que nos permiten añadir, borrar o modificar restricciones de navegación.

6.2.2 Perspectiva del Lector

Desde la perspectiva del lector, la navegación puede verse como la ejecución de una presentación particular. Con el *Sistema de Navegación* no sólo interacciona el autor sino también, con igual intensidad, el lector. De hecho, algunas de las propiedades de edición serán actualizadas desde este sistema, como el número de visitantes, tipo de visitantes, y personas u organismos que recomiendan el documento.

El usuario, en su interacción, se dedica a visitar ítemes de información mediante el *disparo* de sus enlaces o asociaciones. En estas circunstancias, el Sistema de Navegación tiene que considerar, en cada momento:

- Primero, el ítem de información en el que se encuentra el lector del documento: desde qué lugar se pide saltar hacia otro ítem de información, o lo que es lo mismo, seguir una asociación.
- Segundo, el Entorno Conceptual de dicho ítem de información. En concreto, debe tener en cuenta el concepto al que está asociado ese ítem de información y, también, los conceptos con los que está asociado ese concepto.
- Tercero, el Entorno del Ítem de Información, esto es, el conjunto de Restricciones de navegación que se cumplen en ese momento para ese ítem de información.

Volviendo al caso del ejemplo que se está exponiendo en este apartado —figura 4.7 y 4.8—, el Sistema de Navegación durante el proceso de navegación determina que:

- El lector se encuentra situado, por ejemplo, en el ítem C1.
- Como está asociado al concepto *Countries* y éste está asociado a *Portugal*, este usuario puede seguir dos enlaces posibles: *history* y *map*.
- Sin embargo, existe una restricción de navegación que dice que si el lector se encuentra en el ítem que está asociado al concepto *Countries* mediante *list*, únicamente puede saltar hacia el ítem unido a *Portugal* con la función *map*.

Por tanto, desde el navegador mostrará al usuario una sola posibilidad de navegación, la que lleva le lleva al ítem Po2.

6.2.3 Prestaciones del Sistema de Navegación

El *Sistema de Navegación* le permite al autor cambiar las restricciones de navegación. Estas restricciones pueden ser añadidas, borradas y modificadas. Al igual que el *Sistema de Conocimiento*, la consistencia debe estar garantizada durante el proceso de desarrollo del *Sistema de Navegación*.

Pero para modelar esta evolución se debe apoyar en un nivel superior de abstracción al que se utiliza para especificar las restricciones de navegación, la lógica temporal proposicional. La lógica temporal de predicados proporciona ese meta-nivel con el que será posible mantener y cambiar las restricciones de navegación.

Por tanto, las acciones evolutivas que modifican las restricciones de navegación tendrán asociadas una serie de Meta-Restricciones. En este Sistema, los cambios pueden producirse en las restricciones de navegación, RT_n, definidas por el autor y, por tanto, en la red de Petri, RP, obtenida a partir de ellas.

Insistiendo otra vez en el ejemplo, a partir de la subestructura conceptual —EC_p, figura 4.7— el *Sistema de Navegación* puede generar la red de Petri de la figura de la izquierda. Sin embargo, el autor puede añadir una restricción más: "*el lector sólo puede alcanzar el ítem Portugal.map si ha visitado el ítem Countries.cities*". Luego, el Sistema de Navegación se ve forzado a producir una nueva red de Petri —la de la derecha de la figura 4.8—.

Por supuesto, todas las limitaciones no son admisibles y, para ello, —como ya se ha mencionado— se hace uso de las *Meta-Restricciones* que garantizan en todo momento la integridad del cambio. Por ejemplo, todos los ítemes seleccionados mediante EC_p deben poder ser alcanzados y todas las asociaciones conceptuales deben poder ser disparadas. Esto quiere decir que el Sistema debe verificar, al menos, estas dos Meta-Restrictiones.

6.2.4 Definición

El **Sistema de Navegación** es uno de los Sistemas del Modelo SEM-HP que modelan un sistema hipermedia. Restringe o filtra el conjunto de las presentaciones posibles para optar por un subconjunto de las mismas haciendo uso de las restricciones aportadas por el *Subsistema de Presentación* y las suministradas por el propio *Sistema de Navegación*.

6.3 El Sistema de Aprendizaje SA

Por último es útil, aunque no imprescindible para modelar un sistema hipermedia, un *Sistema de Aprendizaje* que evalúe y modifique la navegación. Para ello se tendrá en cuenta:

- el tipo de información que pretende alcanzar el lector, esto es, sobre qué tema, aspectos, enfoques, nivel, etc, está interesado —a esto se le ha denominado *meta*—,
- y/o los conocimientos que quiere adquirir, o más bien aprehender —denominado *logro*—. En este caso, el Sistema debería construir un modelo del conocimiento que tienen los usuarios sobre cada concepto del dominio.

El estudio conceptual y especificación formal de este Sistema queda fuera de los objetivos planteados en la presente memoria, tal y como se justificó en el Preámbulo.

6.3.1 Definiciones

El **Sistema de Aprendizaje** es uno de los Sistemas del Modelo SEM-HP que modelan un sistema hipermedia. Permite la evaluación y la modificación del Sistema de Navegación teniendo en cuenta, en cada instante, las metas o los logros que fijan los lectores.

Un **Logro** es el conjunto de conocimientos que quiere adquirir o aprehender un lector y que pueden ser definidos a partir de la Estructura Conceptual.

Una **Meta** es el conjunto de ítemes de información que pretende alcanzar el lector.

6.4 Sistema Hipermedia e Hiperdocumento

Una vez presentado todo el modelo, la estructura y funcionalidad de todos y cada uno de sus Sistemas, estamos en condiciones de definir qué son, en el ámbito de SEM-HP, tanto un *Sistema Hipermedia* como un *Documento Hipermedia* construido y navegado con este Modelo.

Un **Sistema Hipermedia** en el ámbito de SEM-HP es un conjunto de sistemas interrelacionados y en interacción, denominados Sistema de Conocimiento —formado por los *subsistemas de Memorización* y *Presentación*—, Sistema de Navegación y Sistema de Aprendizaje que permiten:

a) realizar una construcción y mantenimiento flexible de documentos hipermedia;

b) proporcionar una representación de la estructura conceptual del mismo;

c) ofrecer más de una presentación del sistema de información dentro del conjunto de las posibles, y

d) una navegación dinámica en donde es posible una navegación multidestino, multipropósito y con contextualización estructural.

Un Documento Hipermedia o **Hiperdocumento** construido con SEM-HP es un subconjunto de ítemes de información y de asociaciones posibles entre los mismos que determina el Sistema de Navegación, de acuerdo al conjunto de restricciones que se verifican en cada instante.

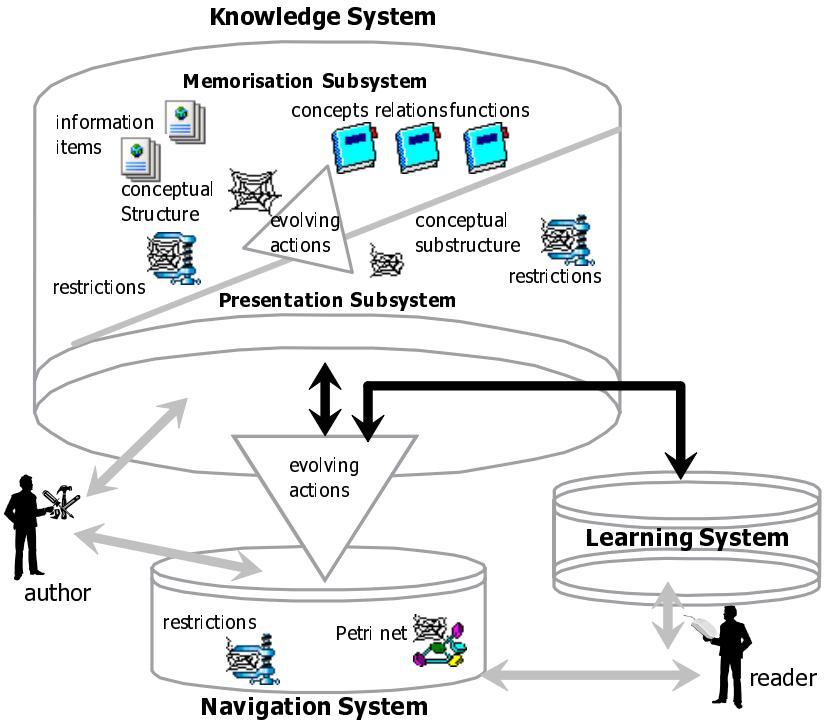


Figura 4.9 Arquitectura detallada del Modelo SEM-HP.

Resumiendo —figura 4.9—, el autor se dedica a representar sus dominios conceptuales mediante conceptos, asociaciones conceptuales y funcionales. Mediante este conocimiento caracteriza sus ítemes de información y construye sus estructuras conceptuales. Desde el Subsistema de Presentación el desarrollador selecciona una subestructura conceptual, y desde el Sistema de Navegación se ordena. Para ello cada Sistema proporciona acciones evolutivas que deben cumplir un subconjunto de las restricciones, que se comportan como precondiciones para esas operaciones.

7 El Proceso de Desarrollo Detallado

Se retoma en este apartado la descripción del Proceso de Desarrollo de Sistemas Hipermedia esbozado en la sección 2 de este mismo capítulo. El conocimiento de la terminología, y de la funcionalidad de los Sistemas y de sus componentes permite realizar una exposición mucho más detallada y precisa de este proceso de maduración evolutivo.

Justamente para gestionar este proceso evolutivo, el modelo SEM-HP sigue la arquitectura de dos niveles de abstracción descrita en el capítulo 3 y recordada en el apartado 2 de este mismo

capítulo. Cada uno de los Sistemas —figuras 4.10 y 4.11— cuenta, pues, con:

- una Meta-Sistema que se encarga de la construcción y cambio de la estructura de su Sistema Software —el Sistema Hipermedia en sus distintas fases—, y
- un Sistema Software que ofrece una determinada funcionalidad.

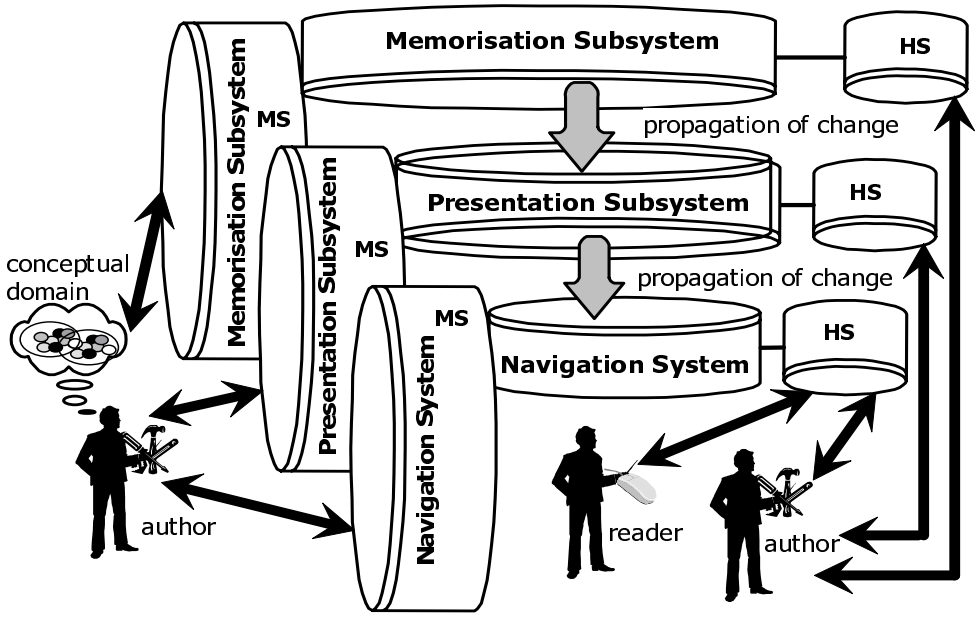


Figura 4.10 El modelo SEM-HP: Una arquitectura con dos niveles de abstracción.

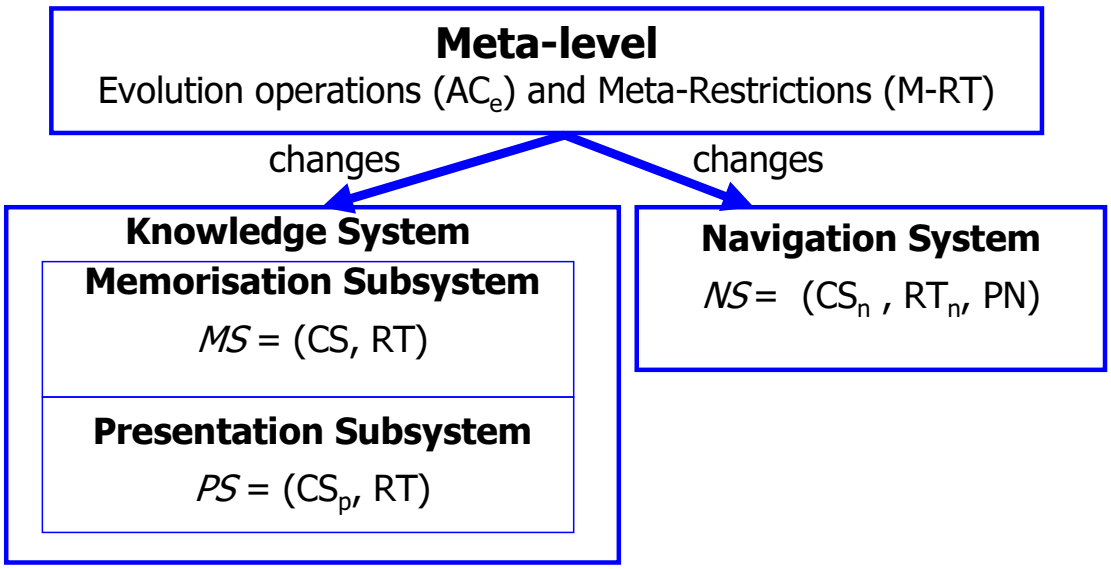


Figura 4.11 ESTRUCTURA y META-ESTRUCTURA de los Sistemas.

El autor interactúa con ambos niveles de abstracción y puede adoptar el papel de desarrollador o de usuario. Cuando interviene como desarrollador crea o modifica el Sistema Hipermedia mediante su comunicación con el Meta-Sistema. Por su parte, si hace uso del Sistema Hipermedia se dedicará a consultar o comprobar la ESTRUCTURA que está construyendo. El lector sólo interactúa con el Sistema de Navegación y únicamente realiza acciones funcionales —figuras 4.10 y 4.11—.

A continuación se explica pormenorizadamente el proceso de evolución de cada uno de los Sistemas del modelo SEM-HP haciendo hincapié en aquellos mecanismos que le dan soporte: Acciones Evolutivas, Restricciones, Propagación de los Cambios y Meta-Restricciones.

7.1 Evolución del Subsistema de Memorización

El autora, un experto en una determinada materia diseña la ESTRUCTURA de este Subsistema. En particular, representa su/s dominio/s complejos de conocimiento mediante los distintos componentes que define el *Subsistema de Memorización*. Su Meta-Sistema proporciona las Acciones Evolutivas necesarias para construir y modificar sus componentes.

Como resultado del anterior proceso de diseño, se construye una determinada Estructura Conceptual —EC_m—. El creador del hiperdocumento puede, además, especificar sus restricciones con objeto de guiar la construcción de la EC_m. Durante el desarrollo de estas actividades, el *Sistema de Memorización* debe garantizar, en todo momento, su consistencia.

Dos aspectos podrán cambiar en el Sistema: la EC_m —el grafo— y las Restricciones de Autor, RT_a. Se hará uso de la Teoría de Grafos para representar las acciones evolutivas del grafo y de sus Restricciones asociadas. Los cambios que se realicen en las Restricciones especificadas por el autor, RT_a, deben verificar un conjunto de Meta-Restricciones. Éstas se ocupan de asegurar que las modificaciones realizadas generan un conjunto de Restricciones válido.

Cuando el autor transforme la EC_m —añada, borre o modifique un concepto, ítem o asociación— el sistema deberá comprobar que:

- La EC_m verifica las restricciones definidas por el Sistema, y que las asociaciones satisfacen el conjunto de restricciones definidas por el autor. Las restricciones actúan como un conjunto de precondiciones para las acciones. La Acción Evolutiva será llevada a cabo sólo si cumple sus Restricciones asociadas —comprobación de integridad—.
- El subgrafo seleccionado por el *Subsistema de Presentación*, EC_p, es consistente con las modificaciones hechas en la EC_m. Si un concepto o asociación ha sido borrado en EC_m, el *Subsistema de Presentación* deberá borrar este concepto o asociación en la EC_p —propagación externa—.

Cuando el autor redefina —añada, borre o modifique— una restricción definida por el autor, RT_a, el sistema deberá asegurarse de que:

- El conjunto de restricciones definidas es consistente, esto es, está ausente de cualquier tipo de contradicción —comprobación de integridad—.
- La EC_m verifica el nuevo conjunto de restricciones mediante la teoría de grafos. El sistema debe detectar aquellas asociaciones que no satisfacen una o más restricciones y borrarlas —propagación interna—.

- La EC_p —el subgrafo seleccionado por el *Subsistema de Presentación*— cumple el nuevo conjunto de restricciones mediante teoría de grafos. El sistema debe detectar las asociaciones que no satisfacen estas restricciones y borrarlos —propagación externa—.

7.2 Evolución del Subsistema de Presentación

Continuando con el proceso de diseño, desde el Subsistema de Presentación, el autor elegirá un determinado subgrafo EC_p a partir de una de las Estructuras Conceptuales. Al igual que sucedía en el *Subsistema de Memorización*, la consistencia de tales operaciones evolutivas debe estar garantizada.

Cada vez que se cambie la EC_p —el diseñador seleccione otro conjunto de conceptos y asociaciones— el subsistema deberá comprobar que:

- La EC_p verifica las restricciones definidas por el sistema y que las asociaciones satisfacen el conjunto de restricciones definido por el autor. Todas estas restricciones son heredadas del *Subsistema de Memorización* —comprobación de integridad—.
- Se ha definido una nueva vista o presentación. En consecuencia, el creador/a del hiperdocumento debe definir de nuevo las restricciones de navegación —propagación externa—.

7.3 Evolución del Sistema de Navegación

Finalmente, el autor prosigue con su proceso de diseño y define sus restricciones de navegación, RT_n, desde el *Sistema de Navegación.* Por supuesto, en este Sistema también debe estar garantizada la consistencia de los cambios.

Cuando el modelador redefina —añada, borre o modifique— una restricción de navegación, el sistema deberá estar seguro de que:

- El conjunto de restricciones, que establece un orden de navegación, es consistente. Para especificar estas operaciones se hará uso un lenguaje basado en lógica temporal proposicional y para comprobar su validez se le asociarán Meta-Restricciones. Éstas estarán especificadas en un lenguaje que se apoya en la lógica temporal de predicados —comprobación de integridad—.
- La red de Petri que se fundamenta en estas restricciones de navegación debe evolucionar, generándose de nuevo —propagación interna—.

7.4 Evolución en el Modelo SEM-HP

Resumiendo, las Acciones de Evolución del Meta-Sistema nos permiten cambiar los distintos componentes de los Sistemas de SEM-HP. Cada una de estas modificaciones debe verificar un serie de Restricciones y debe ser propagada por el Meta-sistema interna y externamente.

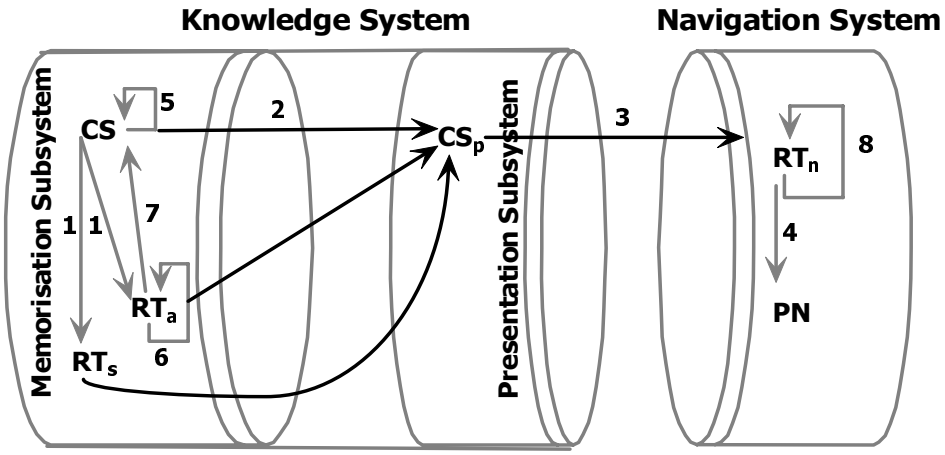


Figura 4.12 Evolución de SEM-HP: Propagación de los Cambios.

En particular —figura 4.12—, la Estructura Conceptual del *Subsistema de Memorización* debe acatar las Restricciones definidas por el sistema, RT_s, y por el autor, RT_a (1). La Estructura Conceptual, EC_m (5), las Restricciones de Autor, RT_a, (6) y las Restricciones de navegación, RT_n (8), pueden evolucionar. Cuando se cambian las RT_a, también se modifica la EC_m (7). La red de Petri debe ser reconstruida cada vez que se redefinen las RT_n (4). Finalmente, la evolución en el *Subsistema de Memorización* debe ser propagada al Subsistema de *Presentación* (2) y, más tarde, al Sistema de Navegación (3).

8 Referencias

[Bush, 1945] 75
[García-Cabrera, 2000] 72
[García-Cabrera, 2001b] 72
[Le Moigne, 1990] 69
[Parets-Torres, 1996] 69
[Peterson, 1981] 94
[Shipman, 1997] 72
[Sowa, 1991] 79
[Stotts, 1998] 86
[Uschold, 1996] 81

Capítulo 5

Formalización de SEM-HP

«La realidad es excesivamente compleja para poder ser directamente comprendida por nuestras limitadas entendederas. Lo único que podemos hacer es buscar en el universo matemático una estructura que se parezca en algún aspecto relevante a la porción de la realidad por la que nos interesamos, y usar esa estructura como modelo teórico simplificado de la realidad.»

Jesús Mosterín, Prólogo de Los Lógicos
Filósofo y escritor español (1941-).

Resumen

Este capítulo contiene la descripción formal de cada uno de los Sistemas del Modelo SEM-HP. Se comienza por introducir algunos algunos aspectos necesarios para el desarrollo de los siguientes apartados de este capítulo tales como:

- el esquema que se ha seguido en la descripción formal del Subsistema de Memorización, del Subsistema de Presentación y del Sistema de Navegación;
- los formalismos empleados en su especificación; y
- algunos conceptos básicos sobre Teoría de Grafos por ser un formalismo clave en el Modelo.

1 Introducción 105
1.1 Planteamiento General 105
1.2 Formalismos 107
1.3 Definiciones Previas: Teoría de Grafos 108

2 Especificación Formal del Subsistema de Memorización 111
3 Especificación Formal del Subsistema de Presentación 153
4 Especificación Formal del Sistema de Navegación 169
5 Consideraciones Generales 215

1 Introducción

Hasta ahora se ha presentado desde una perspectiva puramente conceptual el modelo SEM-HP. En concreto, se han descrito las distintas piezas de información que el modelo usa en la construcción de un documento, el contenido y las propiedades que nos permiten caracterizarlas. También se ha explicado de manera informal cómo la Estructura Conceptual, las Acciones de Evolución, sus precondiciones, las Restricciones, las Meta-Restricciones y la Propagación del Cambio consiguen controlar y hacer evolucionar los procesos de estructuración, presentación y navegación del Sistema Hipermedia.

Queda, por tanto, transformar ese modelo conceptual en un modelo teórico, en una representación formal que haga posible el desarrollo de una herramienta con la que se puedan crear dominios conceptuales y de información, seleccionar presentaciones o vistas de los mismos, y especificar navegaciones de manera dinámica y evolutiva.

Se dedica este capítulo, pues, a la descripción detallada de la especificación formal de cada uno de los Sistemas del Modelo SEM-HP. Dada la extensión de esta exposición se ha considerado preciso presentarlo separadamente estructurándolo en cinco subcapítulos: Introducción, Especificación Formal del Subsistema de Memorización, Especificación Formal del Subsistema de Presentación, Especificación Formal del Sistema de Navegación y Consideraciones Generales.

En el primero de ellos, en el que se encuentra, se introducen algunos aspectos fundamentales tales como:

- el planteamiento general adoptado a la hora de describir formalmente cada uno de los Sistemas o Subsistemas que forman parte del Modelo SEM-HP;
- la descripción de los formalismos empleados; y
- finalmente, algunas consideraciones previas sobre la Teoría de Grafos, por ser el formalismo en el que se fundamentan en gran medida todos los Sistemas de SEM-HP.

1.1 Planteamiento General

La especificación formal del Modelo que se propone no sólo está interesada en describir qué deben hacer cada uno de los Sistemas del hipermedia; además incorpora:

- operaciones que especifican cómo se pueden transformar y manipular el hipermedia, y
- reglas que guían este proceso de construcción y que garantizan la integridad del mismo.

Esto es, tal y como se describió anteriormente de manera abstracta, cada uno de los Sistemas será especificado mediante:

1. Uno o más entes o artefactos, $\{E_1, E_2, ..., E_n\}$, cuya misión será la de representar su visión particular del dominio conceptual y de información que se está diseñando.

2. Un conjunto de restricciones, que se notará mediante RT, con el que se controlará la construcción, y se garantizará la consistencia, de estos artefactos.
3. Un conjunto de Acciones de Evolución, que se notará con AC_e, con las que se podrán hacer y propagar cambios en los distintos Sistemas del hipermedia.

Esto quiere decir que la definición de los mismos se podrá hacer por medio de una n-upla:

$$SISTEMA = (E_1, E_2, ..., E_n, RT, AC_e).$$

Desde el punto de vista del método MEDES, los dos primeros tipos de elementos representan la ESTRUCTURA de los diferentes Sistemas. Esta ESTRUCTURA determina su FUNCIONALIDAD y puede ser modificada mediante un conjunto de operaciones que se denominan Acciones de Evolución —el tercer elemento de la n-upla—.

La especificación de estas operaciones introduce un segundo nivel de abstracción en el sistema hipermedia al que se denomina METASISTEMA. El modelador interactúa con este nivel para modificar la ESTRUCTURA de los distintos Sistemas. Estas operaciones deben verificar un conjunto de invariantes que se denominan Restricciones —RT— que, por supuesto, también pueden cambiar, aunque únicamente aquellas que hayan sido definidas por el autor.

Estas Restricciones impiden la violación de la integridad y la consistencia de los Sistemas. Para ello, siempre que se lleve a cabo una determinada Acción de Evolución perteneciente a uno de los Sistemas, éste desencadenará tres tareas:

1. En primer lugar, se deberá comprobar que el componente —que forma parte de la ESTRUCTURA del Sistema— al que se le ha realizado el cambio, conserva su integridad. Obviamente, aquellos aspectos que son básicos o fundamentales, definidos por el propio Sistema —los denominados invariantes— no admiten ningún tipo de modificación.

   Si las modificaciones se realizan a las Restricciones, la verificación de su consistencia se realizará en un nivel superior mediante las denominadas Meta-Restricciones. Éstas forman parte, junto a las Acciones de Evolución, de la META-ESTRUCTURA del METASISTEMA que, indudablemente, no está sujeta a ninguna alteración.
2. A continuación, el Sistema deberá propagar el cambio de este componente de la ESTRUCTURA al resto de los elementos del Sistema. Esto quiere decir, que se está haciendo una propagación del cambio dentro del propio Sistema con el fin de garantizar su consistencia.
3. Por último, el Sistema debe hacer partícipes del cambio realizado dentro del Sistema al resto de los Sistemas de SEM-HP. Esta propagación externa asegura la integridad de todos los Sistemas y, por tanto, del Sistema Hipermedia.

Indudablemente, cada acción de evolución y sus propagaciones podrá causar auténticos estragos en el Sistema Hipermedia. Un cambio puede conseguir la eliminación completa o casi completa de un dominio conceptual y su correspondiente dominio de información. Por esta razón, en el Modelo se incluyen las denominadas Políticas de Cambio que establecen tres formas distintas de actuación:

1. **Irreversible**. El autor decide asumir un cambio con todas las consecuencias que traiga consigo.
2. **Reversible**. El modelador sólo quiere simular un cambio para observar cuál sería el resultado final.
3. **Impacto**. El diseñador quiere informarse sobre las partes del componente, de los otros componentes del sistema y del resto de los sistemas que se verían afectados por el cambio.

El planteamiento general descrito con respecto a la especificación formal, comprobación de integridad y políticas de cambio, se repetirá en todas y cada una de las descripciones formales de cada uno de los Sistemas o Subsistemas que forman parte del Modelo SEM-HP.

La novedad de esta propuesta, desde el punto de vista de la evolución, es la incorporación de un Meta-nivel que posibilita el razonamiento reflexivo sobre la funcionalidad y la estructura de un sistema hipermedia que evoluciona.

1.2 Formalismos

Se han planteado en estas últimas décadas diversos mecanismos formales para capturar y modificar hipertextos. Algunas de esas propuestas se basan en la teoría de conjuntos [Parunak 1991], otras en la teoría de grafos [Afrati, 1990] [Bench-Capon, 1989] [Garg, 1988] [Koo, 1989] [Wang, 1995, 1998] y, por último, en teoría de autómatas, lógica temporal y redes de Petri [Sttots, 1989, 1998]. En las formalizaciones basadas en teoría de conjuntos se ofrece un razonamiento taxonómico con el objetivo de describir las funciones del hipermedia. En las propuestas basadas en grafos, los nodos y los enlaces son etiquetados para describir las funciones del hipermedia. La especificación basada en teoría de autómatas enfatiza la estructura inherente de los documentos a través de la denominada semántica de estructuración. Algunos lenguajes de especificación formal, tales como Z [Spivey, 1989] y VDM [Ince, 1990], han sido empleados para especificar los anteriores modelos [Chua, 1991] [Halasz, 1990] [Tochtermann, 1995]. Sin embargo, con frecuencia estos lenguajes formales describen qué puede hacer un sistema y no cómo debe hacerse.

En este trabajo no se opta por un sólo formalismo, sino por un conjunto de ellos. En la representación de los distintos Sistemas se hará uso de un subconjunto de los siguientes formalismos: teoría de grafos o redes semánticas, teoría de conjuntos, lógica temporal proposicional y de predicados, y redes de Petri. El uso de tantos y tan variados mecanismos formales no es gratuito. Parafraseando a Jesús Mosterín [Mosterín, 2000, página 11], únicamente se ha intentado buscar en el universo matemático estructuras que nos brinden una funcionalidad parecida a la que debe ofrecer cada uno de los componentes de los Sistemas que forman parte del Modelo, SEM-HP.

Los distintos formalismos permiten modelar el sistema de información y su estructura conceptual, seleccionar una determinada presentación y, por último, construir la navegación que se quiere realizar sobre la presentación elegida. En particular se empleará:

- la teoría de grafos —y la teoría de conjuntos en la que se basa— para formalizar la Estructura Conceptual. Un grafo es el mecanismo más adecuado para representarlo porque ofrece una gran versatilidad debido a sus posibilidades de navegabilidad e interacción visual.
- la lógica temporal proposicional para describir las Restricciones, debido a que con este formalismo es muy fácil expresar relaciones de orden.
- lógica temporal de predicados para especificar las Meta-Restricciones, porque necesitamos subir un nivel de abstracción.
- la teoría sobre redes de Petri, para convertir las Restricciones de navegación en un formalismo operacional con el que se pueda ejecutar una navegación y analizar sus propiedades.

Al final del capítulo, en el apartado 5, se incluye una justificación más amplia y detallada de los formalismos utilizados en cada uno de los componentes de los Sistemas del modelo SEM-HP.

1.3 Definiciones Previas: Teoría de Grafos

Dado que la especificación formal de los Sistemas se basa fundamentalmente en un grafo, incluimos algunas definiciones básicas sobre Teoría de Grafos [Christofides, 1975].

Un ***grafo*** es una colección de nodos o vértices n_1,n_2, ..., n_n —se denota con el conjunto N—, y una colección de enlaces o aristas l_1, l_2, ..., l_n —se denota con el conjunto L— que unen esos nodos. El grafo G se puede describir completamente mediante estos dos conjuntos y se denota como G = (N, L).

Si las aristas tienen una dirección —en este caso, se representan con flechas— se llaman ***arcos*** y el grafo que resulta se llama ***grafo dirigido*** (N, A). El conjunto de arcos es el producto Cartesiano N X N en donde un arco que vaya desde el nodo n_i al nodo n_j se denota como $<n_i, n_j>$. Al nodo n_i lo llamamos, ***nodo origen*** y al nodo n_j, ***nodo destino***. Un ***camino*** es una secuencia finita de enlaces $m = < n_1, n_2 >, < n_2, n_3 >, ..., < n_{n-1}, n_n >$, que a veces se denota como la secuencia de nodos destino de los arcos encabezada con el origen del primer destino: $\mu = n_1, n_2, n_3, ..., n_{n-1}, n_n$.

Un camino $a_1, a_2, ..., a_n$ en el que el nodo origen del arco a_1 coincide con el nodo destino del arco a_n es un ***circuito*** o *ciclo*.

Los arcos $a= <n_i, n_j>, n_i \neq n_j$ que tienen en común el nodo destino se llaman ***adyacentes***. Por tanto, dos nodos n_i y $n_j$son adyacentes si el arco $<n_i, n_j>$ o el arco $<n_j, n_i>$ o ambos existen en el grafo.

En un grafo G = (N, A), a un nodo n_j se le llama el ***sucesor*** de n_i si $<n_i, n_j> \in$ A; el *conjunto de sucesores* de n_i se denota como $\Gamma^+(n_i)$. De igual modo, a un nodo n_j se le llama ***predecesor*** de n_i si $<n_j, n_i> \in$ A y, el *conjunto de predecesores* de n_i se denota como $\Gamma^-(n_i)$.

Dado n_i, n_j, si existe un camino desde n_i a n_j, entonces n_j es un ***descendiente*** de n_i; mientras que si existe un camino desde n_j a n_i, entonces n_j es un ***ascendente*** de n_i. Se dice de un nodo n_j que es ***accesible-desde*** n_i, si n_j es un descendiente de n_i ó $n_j = n_i$; de igual modo, se dice que n_j ***alcanza-a*** n_i si es un ascendente de n_i ó $n_j = n_i$. El conjunto de nodos que son *accesibles-desde* y que *alcanzan-a* n_i se denotan como, $\Gamma^{*+}(n_i)$ y $\Gamma^{*-}(n_i)$ respectivamente.

Una manera alternativa y preferible es describir un grafo G mediante un conjunto de nodos N y una correspondencia Γ (Γ: N $\rightarrow$ N) que muestre cómo los nodos se relacionan unos con otros. Por tanto un grafo también se denota como G = (N, Γ). El *número de arcos* que parten del nodo n_i —que se denota como $|\Gamma^+(n_i)|$— o que tienen como nodo origen al nodo n_i se llama ***rango-de-salida*** —*outdegree*— de n_i. De modo similar, el número de arcos que llegan a n_i —esto es, $|\Gamma^-(n_i)|$— o que tienen como nodo destino n_i se llama ***rango-de-entrada*** —*indegree*— de nodo n_i. Es bastante obvio que la suma de los rangos de entrada o de salida de todos los nodos de un grafo es igual al número de arcos de G, esto es,

$$\sum_{i=1}^{n} | \Gamma^+(n_i) | = \sum_{i=1}^{n} | \Gamma^-(n_i) | = m$$

donde *n* es el número total de nodos y *m* el número total de arcos de G.

Un ***grafo acíclico*** es un grafo que no contiene ningún ciclo. Cualquier grafo acíclico contiene al menos un nodo que no tiene sucesores, y al menos un nodo que no tiene predecesores. Una condición necesaria y suficiente para que un grafo G = (N, A) sea acíclico es que sus nodos n_1, n_2, ... n_n puedan ser numerados (asignándoles índices), de modo que si $< n_i, n_j> \in$ A entonces i < j [Carre, 1979].

Se dice de un grafo que es ***débilmente conectado*** si existe al menos un camino que una cada par de nodos distintos. Dicho de otro modo, todo nodo de un grafo débil debe estar conectado al grafo.

G' = (N', A') es un ***subgrafo*** de un grafo dirigido G =(N, A) si G' es también un grafo y N' $\subseteq$ N y A'$\subseteq$ A. Todos los conceptos definidos para los grafos, tales como camino y ciclo son aplicables a sus subgrafos. Sin embargo, un subgrafo ofrece un contexto más especifico para estas nociones. Por ejemplo, un camino en un subgrafo de G' es también un camino en G pero un camino en G puede no ser un camino en alguno de sus subgrafos.

Una ***red semántica*** es un tipo de grafo dirigido, en el que nodos y arcos están etiquetados con alguna semántica. Un arco en una red semántica se representa con una tripleta $<n_o, r, n_d>$, donde n_o es el nodo origen, n_d el nodo destino, y r la relación que los une. Por tanto, una red semántica se puede denotar como G = (N, R, A), donde N son los nodos etiquetados, R las relaciones etiquetadas, y A: N X R $\rightarrow$ N, es decir, A $\subseteq$ N X R X N son las asociaciones entre nodos consistentes en tripletas.

Subcapítulo 5.2

Especificación Formal del Subsistema de Memorización

«La memoria parece grande por lo que muestra en recuerdos; lo es mucho más por lo que ciertamente esconde.»

Niceto Alcalá Zamora.
Abogado y político español (1877-1949).

«Sobrecargar la memoria ocasiona los mismos graves daños que el no ejercitarla.»

Johann M. Sailer.
Profesor de teología y sacerdote católico (1751-1832).

«La memoria es la centinela del cerebro.»

William Shakespeare.
Actor, escritor y empresario teatral inglés (1564-1616).

Resumen

Este apartado se dedica a describir formalmente el Subsistema de Memorización del Sistema de Conocimiento. En primer lugar, se especifican sus componentes: Estructura Conceptual, Diccionarios y Restricciones. A continuación se detallan las acciones evolutivas con las que crear y modificar estos artefactos. Posteriormente, se explica cómo se propagan las transformaciones que pueden suceder en estos elementos. Finalmente, se describe una de sus posibles utilidades: la composición de ítemes de información.

2 Especificación Formal del Subsistema de Memorización 113
2.1 La Estructura Conceptual del Subsistema de Memorización, ECm 113
2.2 El Diccionario del Subsistema de Memorización D 120
2.3 Las Restricciones del Subsistema de Memorización RTm 120
2.3.1 Restricciones de los Conceptos RTmc 121
2.3.2 Restricciones de las Relaciones Conceptuales RTmr 121
2.3.3 Restricciones Relacionadas con los Ítemes de Información RTmii 126
2.3.4 Restricciones Relacionadas con los Diccionarios RTmd 127
2.4 Acciones Evolutivas del Subsistema de Memorización ACem 128
2.4.1 Cambiando la Estructura Conceptual ECm 129
2.4.2 Cambiando las Restricciones de Autor 143
2.4.3 Cambiando el Diccionario D 145
2.5 Propagación Interna del Cambio 147
2.5.1 Propagación al Cambiar la Estructura Conceptual ECm 147
2.5.2 Propagación al Cambiar las Restricciones de Autor 147
2.6 Composición de Ítemes de Información 149

2 Especificación Formal del Subsistema de Memorización

El *Subsistema de Memorización* forma parte, junto al *Subsistema de Presentación*, del *Sistema de Conocimiento*. Este Subsistema es el encargado de almacenar, estructurar y mantener el sistema de información. Dicho de otro modo, será el responsable de:

1. Estructurar semánticamente el conocimiendo, creando y mateniendo dominios conceptuales.
2. Organizar el conjunto de ítemes de información y catalogarlos mediante estos dominios conceptuales, convirtiéndolos en dominios de información.

El *Subsistema de Memorización* contará con un conjunto de operaciones o acciones evolutivas AC_{em} mediante las cuales el autor podrá diseñar estos dominios conceptuales y de información. Para conducir y controlar este proceso dinámico de construcción, el Subsistema y el propio creador, proporcionarán un conjunto de restricciones RT_m que garantizarán la integridad del mismo. Estas restricciones establecen de manera explícita qué acciones son correctas semánticamente y cuáles no en cada momento, de manera que sólo se realizarán aquellas operaciones que después de ser aplicadas conservan el Subsistema en un estado consistente.

Def[sm1] El *Subsistema de Memorización*[1] SM se define como: SM = (EC_m, D, RT_m, AC_{em}), donde:

- EC_m es un *grafo dirigido débilmente conectado con raíz* que representa la estructura conceptual de un determinado dominio conceptual y de información de un sistema hipermedia;
- D es el diccionario de los distintos elementos que podrán ser usados en la EC_m;
- RT_m es el conjunto de restricciones que debe verificar los distintos componentes del Subsistema —la EC_m y el D—; y
- AC_{em} es el conjunto de acciones u operaciones evolutivas que nos permiten construir y mantener la EC_m, el D y las RT_m.

2.1 La Estructura Conceptual del Subsistema de Memorización, EC_M

El *Subsistema de Memorización* proporciona los mecanismos necesarios para representar un

1 Las definiciones formales del capítulo 4 estabán destinadas a facilitar las comprensión conceptual del modelo y fueron simplificadas. En estos subcapítulos dedicados a la especificación formal del modelo se proporcionan las descripciones completas de los Sistemas y de sus componentes.

dominio conceptual y de información mediante un grafo dirigido, en el que nodos y arcos están etiquetados semánticamente. Al domininio conceptual —formado por asociaciones entre conceptos—, junto con el dominio de información que identifica —asociaciones entre conceptos e ítemes—, se denomina Estructura Conceptual EC_m. Los distintos ítemes de información se pueden asociar —etiquetar— con uno o más conceptos de esta EC_m.

Def[ecm1] La *estructura conceptual* EC_m es un grafo etiquetado o *red semántica* que representa un determinado dominio conceptual de un sistema hipermedia.

Def[ecm2] A las etiquetas de los arcos de la estructura conceptual los llamaremos *relaciones* R. Esto es, cada nombre es una relación.

$$R = \{r / r \text{ es una relación}\}, R \neq \emptyset.$$

Def[ecm3] Se consideran dos tipos de nodos etiquetados que llamaremos *conceptos*, C, e *ítemes de información*, II.

$$C = \{c / c \text{ es un concepto}\}, C \neq \emptyset,$$
$$II = \{ii / ii \text{ es un ítem de información}\}.$$

Def[ecm4] Las relaciones R, pueden ser de dos tipos: *Conceptuales*, R_c, y *Funcionales*, R_f. Las Relaciones Conceptuales R_c únicamente pueden unir conceptos. Las relaciones funcionales R_f únicamente pueden unir un concepto y un ítem de información. En general, las etiquetas de las Relaciones serán definidas por el autor, de forma que

$$\forall\ r_1, r_2 \in R, r_1 \neq r_2$$
$$R_c, R_f \subseteq R, R_c \cup R_f = R, R_c \cap R_f = \emptyset.$$

Def[ecm5] Las Relaciones Conceptuales R_c pueden ser de dos tipos: *de Dependencia* R_d y *de Referencia* R_r. Las Relaciones de Dependencia R_d están definidas por el sistema y sólo pueden ser alguna de estas tres {*EsUn*, *ParteDe*, *TipoDe*}. Las Relaciones de Referencia serán definidas por el autor.

$$R_d, R_r \subseteq R_c, R_d \cup R_r = R_c, R_d \cap R_r = \emptyset.$$
$$R_d = \{EsUn, ParteDe, TipoDe\}.$$

Def[ecm6] A la correspondencia entre nodos etiquetados —conceptos, ítemes de información— mediante arcos etiquetados —relaciones— se la denomina Asociación A. Solamente son posibles dos tipos de asociaciones: *Conceptual* A_c y *Funcional* A_f.

$$A_c, A_f \subseteq A, A_c \cup A_f = A, A_c \cap A_f = \emptyset.$$

Def[ecm7] Una *Asociación Conceptual* A_c hace corresponder a cada pareja —concepto, relación conceptual—, otro concepto. Se trata de una correspondencia del tipo: $C \times R_c \rightarrow C$, donde C es el conjunto de conceptos y R_c el conjunto de relaciones conceptuales. Cada elemento de esta correspondencia se puede notar con la tripleta $<c_o, r_c, c_d>$, donde c_o y c_d son conceptos y r_c una *relación conceptual*. A c_o lo llamamos *concepto origen* de la relación, y a c_d *concepto destino* de la relación. No obstante, la asociación se leerá en sentido inverso, esto es, c_d mantiene una relación conceptual r_c con el concepto c_o.

A_c: C x R_c → C, donde
C = {c / c es un concepto}, C ≠ ∅,
R_c = {r_c / r_c es una relación conceptual}, R_c ≠ ∅.
A_d, A_r ⊆ A_c, A_d ∪ A_r = A_c, A_d ∩ A_r = ∅.

Def[ecm8] Una *Asociación Funcional* A_f hace corresponder a cada pareja —concepto, relación funcional—, un ítem de información. Se trata de una correspondencia del tipo: C x R_f → II, donde C es el conjunto de conceptos, R_f el conjunto de relaciones funcionales e II el conjunto de ítemes de información. Cada elemento de esta correspondencia se puede notar con la tripleta <c, r_f, ii>, donde c es un concepto, r_f una *relación funcional* e ii un ítem de información. Por tanto, los ítemes de información siempre serán nodos hoja en el grafo ya que no se puede dar una asociación cuyo origen sea un ítem de información.

A_c: C x R_f → II, donde
C = {c / c es un concepto}, C ≠ ∅,
R_f = {r_f / r_f es una relación funcional}, R_f ≠ ∅.
II = {ii / ii es un ítem de información}.

Obviamente, al ser A_f una correspondencia, un ítem de información puede estar asociado:

a) mediante relaciones funcionales a más de un concepto:

∀ ii ∈ II, ∃ C_k ⊆ C, tal que ∀ c ∈ C_k, ∃ <c, r_f, ii>, con r_f ∈ R_f; y

b) a un mismo concepto mediante más de una relación funcional:

∀ ii ∈ II, ∃ R_{fk} ⊆ R_f, tal que ∀ r_f ∈ R_{fk}, ∃ <c, r_f, ii>, con c ∈ C.

Def[ecm9] La estructura conceptual EC_m se define formalmente mediante: EC_m = (C, II, R_c, R_f, A_c, A_f), en la que C es un conjunto de *conceptos*, II es el conjunto de *ítemes de información*, R_c el conjunto de *relaciones conceptuales*, R_f conjunto *de relaciones funcionales*, A_c es el conjunto de *asociaciones conceptuales*, A_f es el conjunto de *asociaciones funcionales*.

EC_m = (C, II, R_c, R_f, A_c, A_f), donde
C = {c / c es un concepto}, C ≠ ∅,
II = {ii / ii es un ítem de información},
R_c = {r_c / r_c es una relación conceptual}, R_c ≠ ∅.
R_f = {r_f / r_f es una relación funcional}, R_f ≠ ∅.
A_c: C x R_c → C, A_c= {<c_o, r_c, c_d> / <c_o, r_c, c_d> es una asociación conceptual}, A_c ≠ ∅,
A_f: C x R_f → II, A_f = { <c, r_f, ii> / <c, r_f, ii> es una asociación funcional}, A_f ≠ ∅.

La estructura conceptual EC_m tiene los siguientes *invariantes*:

■ La estructura conceptual EC_m es un *grafo dirigido débilmente conectado*. Un grafo se dice débilmente conectado si todo nodo debe tener al menos un arco con otro nodo. Esto quiere decir que todo concepto de la estructura conceptual debe estar conectado al grafo. Dicho de otro modo, no se permite que existan nodos aislados.

EC_m = (C, II, R_c, R_f, A_c, A_f), donde
∀ c_i, c_j ∈ C, ∃ μ = c_i, c_k, ..., c_j ∈ EC_m, siendo μ un camino del grafo —subcapítulo 5.1, apartado 3.1—.

■ La estructura conceptual EC_m es un grafo que sólo admite arcos que conecten dos nodos. Por tanto, todo arco tiene que tener un nodo origen y un nodo destino.

$$EC_m = (C, II, R_c, R_f, A_c, A_f), \text{ donde}$$
$$\forall\, r \in R, \exists\, <c_o, r, o_d> \in A, \text{ donde } c_o \in C,\ o_d \in C \text{ ó } o_d \in II.$$

■ EC_m es un grafo *con raíz*. Esto es, uno sólo de sus conceptos no tiene predecesor. A este concepto se le llama *concepto-sistema*, $\mathfrak{s}$, porque es invisible para los usuarios. Existe otro concepto especial llamado *concepto-raíz,* $\mathfrak{r}$, que es el único sucesor del concepto de sistema. El concepto de sistema mantiene una relación conceptual especial llamada de *inicio* $\mathfrak{i}$ con el concepto-raíz.

$$EC_m = (C, II, R_c, R_f, A_c, A_f), \text{ donde}$$
$$\mathfrak{s} \in C, \forall\, c \in C, \forall\, r \in R, <c, r, \mathfrak{s}> \notin A_c$$
$$\mathfrak{r} \in C, \mathfrak{i} \in R_c, \text{ tal que } <\mathfrak{s}, \mathfrak{i}, \mathfrak{r}> \in A_c$$
$$\forall\, c \in C, \forall\, r \in R \text{ y } c \neq \mathfrak{s}, r \neq \mathfrak{i}, <c, r, \mathfrak{r}> \notin A_c.$$

Gracias a estos conceptos especiales —*sistema* y *raíz*—, y a la relación *inicio*, el Subsistema de Memorización puede mantener más de una Estructura Conceptual. Es decir, se pueden simular varias Estructuras Conceptuales bajo un sólo grafo.

■ Todos los ítemes de información creados en el sistema, por defecto, están asociados al concepto-sistema $\mathfrak{s}$ con el que mantienen una relación funcional especial llamada *genérica* g.

$$EC_m = (C, II, R_c, R_f, A_c, A_f), \text{ donde}$$
$$\forall\, ii \in II, \exists\, <\mathfrak{s}, g, ii> \in A_f, \text{ con } \mathfrak{s} \in C,\ g \in R_f.$$

Def[ecm10] Los conceptos, relaciones e ítemes de información son entidades u objetos de datos que pueden tener atributos o propiedades; cada propiedad se denota como *objetodato.propiedad*. La propiedad más inmediata es la etiqueta o nombre que recibe ese objeto. Por ejemplo, para hacer referencia al nombre de un concepto, c, lo haremos como c.nombre.

En la figura 5.1 se muestra un ejemplo genérico. En este momento, en el Subsistema de Memorización se han definido dos Estructuras Conceptuales cuyos grafos están formados por los siguientes conjuntos de nodos y asociaciones:

Estructura Conceptual de la izquierda,

C= {CA, CB, CC, CD, CE, CF, CG, CH, CI, CJ}, II= {IA, IB, IC, ID}

A_c= {<CB, RA, CA>, <CB, RE, CE>, <CB, RD, CC>, <CB, RF, CF>, <CC, RA, CA>, <CC, RF, CF>, <CC, RF, CG>, <CD, RA, CA>, <CD, RF, CH>, <CH, RG, CF>, <CI, RC, CH>, <CJ, RC, CH>}

A_f= {<CA, RX, IA>, <CF, RU, IB>, <CI, RV, IC>, <CI, RU, ID>, <CH, RX, IB>}

Concepto raíz: CJ

Estructura Conceptual de la derecha,

C= {CK, CL, CM, CN, CO, CP, CQ, CR, CS, CT, CU, CV}, II= {IE, IF, IG, IH, IJ}

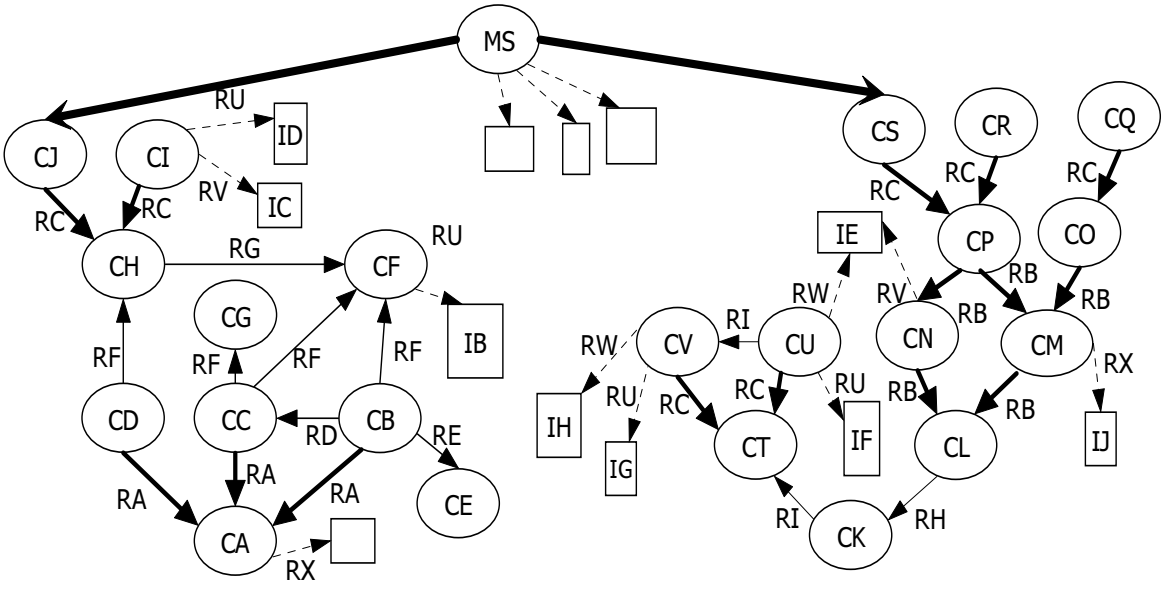


Figura 5.1 Un ejemplo genérico.

A_c= {<CL, RH, CK>, <CM, RB, CL>, <CN, RB, CL>, <CO, RB, CM>, <CP, RB, CM>, <CP, RB, CN>, <CQ, RC, CO>, <CR, RC, CP>, <CS, RC, CP>, <CK, RI, CT>, <CU, RC, CT>, <CV, RC, CT>, <CV, RI, CU>}

A_f= {<CU, RU, IE>, <CN, RV, IE>, <CU, RW, IF>, <CV, RU, IG>, <CV, RW, IH>, <CM, RX, IJ>}

Concepto raíz: CS

En el Subsistema de Memorización existen tres ítemes que no están asociados a ninguna Estructura Conceptual, y que son asociados al concepto raíz mediante g.

A las funciones de correspondencia Γ^+, Γ^-, Γ^{*+} y Γ^{*-} definidas en la teoría de grafos —apartado1.3 del subcapítulo 5.1— se añade una nueva función de correspondencia Λ: C x C → R_c ó Λ: C x II → R_f. Esta función permite calcular todas las relaciones que mantienen dos nodos del grafo. Estas relaciones serán conceptuales, si se pregunta por dos conceptos, o funcionales, si se pregunta por un concepto y un ítem de información. Esta correspondencia se leerá como *Relacionados-mediante*. Además adaptaremos las funciones *accesibles-desde* —Γ^+, Γ^{*+}— y *alcanzan-a* —Γ^-, Γ^{*-}— para que se pueda calcular el conjunto de conceptos que son accesibles mediante una relación conceptual en particular —con una determinada etiqueta—.

Def[ecm11] En nuestro sistema se definen tres funciones de correspondencia que llamaremos de alcanzabilidad: *Conceptos_accesibles_desde*, *Conceptos_alcanzan_a* y *Relacionados_mediante*.

■ *Conceptos_accesibles_desde*, $\Gamma^{+n}(c_o, r_c)$ admite como parámetros un número entero positivo n, un concepto origen c_o y una relación conceptual r_c. Se admiten las siguientes variantes:

Conceptos_accesibles_desde	Definición
$\Gamma^{+n}(c_o, r_c)$	Conjunto de conceptos que son accesibles desde c_o, mediante la relación conceptual r_c, recorriendo n tripletas.
$\Gamma^{+}(c_o, r_c) = \bigcup_{n=0}^{\infty} \Gamma^{+n}(c_o, r_c)$	Conjunto de conceptos que son accesibles desde c_o, mediante relación conceptual r_c, recorriendo las tripletas que sean necesarias.
$\Gamma^{+n}(c_o) = \bigcup_{rc} \Gamma^{+n}(c_o, r_c)$	Conjunto de conceptos que son accesibles desde c_o, mediante cualquier relación, recorriendo n tripletas.
$\Gamma^{+}(c_o) = \bigcup_{n=0}^{\infty} \Gamma^{+n}(c_o)$	Conjunto de conceptos que son accesibles desde c_o, mediante cualquier relación, recorriendo las tripletas que sean necesarias.

Tabla 5.1 Funciones de correspondencia: Conceptos_accesibles_desde.

■ *Conceptos_alcanzan_a*, $\Gamma^{-n}(r_c, c_d)$ admite como parámetros un número entero positivo n, un concepto destino c_d y una relación conceptual r_c. Se admiten las siguientes variantes:

Conceptos_alcanzan_a	Definición
$\Gamma^{-n}(r_c, c_d)$	Conjunto de conceptos que alcanzan a c_d, mediante la relación conceptual r_c, recorriendo n tripletas.
$\Gamma^{-}(c_d, r_c) = \bigcup_{n=0}^{\infty} \Gamma^{-n}(c_d, r_c)$	Conjunto de conceptos que alcanzan c_d, mediante relación conceptual r_c, recorriendo las tripletas que sean necesarias.
$\Gamma^{-n}(c_d) = \bigcup_{rc} \Gamma^{-n}(c_d, r_c)$	Conjunto de conceptos que alcanzan a c_d, mediante cualquier relación, recorriendo n tripletas.
$\Gamma^{-}(c_d) = \bigcup_{n=0}^{\infty} \Gamma^{-n}(c_d)$	Conjunto de conceptos que alcanzan a c_d, mediante cualquier relación, recorriendo las tripletas que sean necesarias.

Tabla 5.2 Funciones de correspondencia: Conceptos_alcanzan_a.

Obviamente, el conjunto formado por los *conceptos_accesibles_desde* y los *conceptos_alcanzan_a* estará compuesto por los conceptos que están conectados a uno dado. Se usará el símbolo Υ para hacer referencia a la unión de las dos anteriores funciones —$\Gamma^{+} \cup \Gamma^{-}$ ó $\Gamma^{*+} \cup \Gamma^{*-}$— en cualquiera de sus variantes.

■ *Relacionados_mediante*, $\Lambda^{n}(c_o, r, o_d)$ admite como parámetros un número entero positivo n, un concepto origen c_o, una relación r y un concepto destino o un ítem de información que se notará como o_d. Si el objeto destino es un concepto, estaremos preguntando por relaciones conceptuales; si es un ítem de información por relaciones funcionales. Se admiten las siguientes variantes:

Relacionados-mediante	Definición
$\Lambda^n(c_o, o_d)$	Conjunto de relaciones conceptuales o funcionales que conectan c_o con o_d, mediante n tripletas.
$\Lambda(c_o, o_d) = \bigcup_{n=0}^{\infty} \Lambda^n(c_o, o_d)$	Conjunto de relaciones conceptuales o funcionales que conectan c_o con o_d, mediante las tripletas que sean necesarias.
$\Lambda^n(c_o, r, o_d)$	¿El concepto c_o está relacionado con el objeto o_d, mediante la relación r, recorriendo n tripletas?
$\Lambda(c_o, r, o_d)$	¿El concepto c_o está relacionado con el objeto o_d, mediante la relación r, recorriendo las tripletas que sean necesarias?

Tabla 5.3 Funciones de correspondencia: Relacionados_mediante.

Def[ecm12] Para cada una de las tres anteriores funciones de correspondencia podremos preguntar por:

- el número de conceptos que son sucesores $|\Gamma^{+n}(c_d, r_c)|$,
- el número de conceptos que son predecesores de un concepto $|\Gamma^{-n}(c_d, r_c)|$ o
- el número de relaciones conceptuales que conectan a dos conceptos, o de relaciones funcionales que asocian a un concepto y a un ítem información $|\Lambda^n(c_o, r, o_d)|$.

Basándonos en el ejemplo de la figura 5.1 se muestran algunos casos de ejemplo de uso de las anteriores funciones:

Γ^{+}(CB) = {CA, CC, CE, CF}; $|\Gamma^{+}$(CB)| = 4

Γ^{+}(CA) = {}; $|\Gamma^{+}$(CA)| = 0

Γ^{+}(CQ, RB) = {CO, CM, CL}; $|\Gamma^{+}$(CQ, RB)| = 3

Γ^{-}(CT) = {CU, CV}; $|\Gamma^{-}$(CT)| = 2

Γ^{-2}(CL, RB) = {CM, CN, CO, CP}; $|\Gamma^{-2}$(CL, RB)| = 4

Υ (CV) = {CU, CT}; $|\Upsilon$ (CV)| = 2

Λ^1(CV, CT) = {RC}; $|\Lambda^1$(CV, CT)| = 1

Λ(CV, CT) = {RC, RI}; $|\Lambda$(CV, CT)| = 2

Λ(CV, RW, IH) = true

2.2 El Diccionario del Subsistema de Memorización D

Con el fin de que el diseñador de la estructura conceptual pueda ir definiendo conceptos, relaciones conceptuales y funcionales de manera flexible, sin que tengan que etiquetar en ese momento algún nodo o arco de la Estructura Conceptual, el Subsistema contará con los llamados diccionarios.

Def[ecm13] Los diccionarios del Subsistema de Memorización, D, se definen formalmente mediante la tripleta: D = (DC, DR_c, DR_f), en la que DC es el diccionario o conjunto de *conceptos*, DR_c es el diccionario o conjunto de relaciones conceptuales y DR_f es el diccionario o conjunto de relaciones funcionales. Los nombres incluidos en estos diccionarios DC, DR_c y DR_f serán aplicados a Conceptos C, Relaciones Conceptuales R_c y Relaciones Funcionales R_f, respectivamente.

$$D = (DC, DR_c, DR_f), \text{ donde}$$
$$DC = \{dc / dc \text{ es un nombre de un concepto}\},\ DC \neq \emptyset \text{ y } C \subseteq DC,$$
$$DR_c = \{dr_c / dr_c \text{ es un nombre de una relación conceptual}\},\ DR_c \neq \emptyset \text{ y } R_c \subseteq DR_c,$$
$$DR_f = \{dr_f / dr_f \text{ es un nombre de una relación funcional}\},\ DR_f \neq \emptyset \text{ y } R_f \subseteq DR_f.$$

Distinguir entre los conceptos y relaciones que aparecen en el grafo —C, R_c y R_f— y las posibles etiquetas de estos conceptos y relaciones —DC, DR_c y DR_f— no es necesario desde un punto de vista estrictamente formal, pero se ha optado por hacer esta separación porque:

1. Facilita la implementación y la manipulación evolutiva del grafo.
2. Permite asociar un conjunto de Restricciones distinto —ver apartado 2.4.2— a relaciones conceptuales con igual nombre, pero que pertenecen a Estructuras Conceptuales diferentes. Esto es, el autor podrá definir Restricciones para cada relación conceptual de cada Estructura Conceptual.

Por defecto, el sistema ofrece un nombre de concepto {*sistema*}, cuatro nombres de relaciones conceptuales {*inicio*, *EsUn*, *ParteDe*, *TipoDe*} y un repertorio de nombres de relaciones funcionales tales como: {*genérica*, *introducción*, *definición*, *ejemplo*, *aclaración*, *referencias*, *algoritmo*, *advertencias*, *explicación*, *opinión*, *comparación*, *análisis*, *justificación*, *valoración*, *antecedentes*, *enfoques*, *profundización*, *ampliación*} —ver tabla 4.1—

2.3 Las Restricciones del Subsistema de Memorización RT_M

Def[ecm14] Para cada uno de los elementos de la estructura conceptual –conceptos C, relaciones R, asociaciones A, ítemes de información II— y del diccionario –diccionario de conceptos DC, de relaciones conceptuales DR_c y de relaciones funcionales DR_f— del *Subsistema de Memorización* se definen un conjunto de restricciones RT_m. Un subconjunto de las restricciones definidas se convertirá en precondiciones y postcondiciones para las acciones AC_{em} que hacen evolucionar a la EC_m y al D.

$RT_{mc}, RT_{mr}, RT_{ma}, RT_{mii}, RT_{md} \subseteq RT_{m}$, siendo
RT_{mc}, las restricciones de los conceptos,
RT_{mr}, las restricciones de las relaciones conceptuales,
RT_{ma}, las restricciones de las asociaciones conceptuales,
RT_{mii}, las restricciones de los ítemes de información y
RT_{md}, las restricciones de los diccionarios.

A continuación se describen cada una de estas restricciones. No obstante, hay que aclarar que en la formalización del *Subsistema de Memorización* que aquí se presenta, el conjunto de restricciones relacionadas con las asociaciones RT_{ma} será vacío. En el apartado dedicado a los Trabajos Futuros se describe una primera aproximación del lenguaje que le permitiría al autor definir cualquier tipo de restricción asociativa.

Def[ecm15] Los conceptos, relaciones, asociaciones e ítemes de información deberán cumplir en todo momento el conjunto de restricciones RT_{m} definido en el Subsistema de Memorización. Estas restricciones pueden ser proporcionadas por el sistema o por el autor/a. Las primeras, serán fijas, no cambiarán. El segundo tipo será variable y puede evolucionar con el tiempo.

2.3.1 Restricciones de los Conceptos RT_{MC}

Todas las restricciones relacionadas con los conceptos son fijas y están predefinidas por el sistema. Todas ellas son bastantes evidentes pues se deducen de la propia definición que se ha realizado de la Estructura Conceptual EC_{m} —invariantes—. Se definen las siguientes:

RT[c1] Todos los conceptos deben estar etiquetados.

$EC_{m} = (C, II, R_{c}, R_{f}, A_{c}, A_{f})$, donde
$\forall c \in C$, c.nombre <> "".

RT[c2] Todos los conceptos que aparezcan en la Estructura Conceptual EC_{m} —que están asociados— pertenecerán al conjunto de conceptos C.

$EC_{m} = (C, II, R_{c}, R_{f}, A_{c}, A_{f})$, donde
$\forall <c_{o}, r, c_{d}> \in A_{c}, c_{o}, c_{d} \in C$.

RT[c3] En una misma Estructura Conceptual no pueden existir dos nodos etiquetados con el mismo nombre de concepto, esto es, el conjunto no admite elementos repetidos.

$EC_{m} = (C, II, R_{c}, R_{f}, A_{c}, A_{f})$, donde
$\forall c_{i}, c_{j} \in C, c_{i}.nombre \neq c_{j}.nombre$.

2.3.2 Restricciones de las Relaciones Conceptuales RT_{MR}

Entre las restricciones relacionadas con las relaciones conceptuales se pueden distinguir entre las definidas por el sistema y las que puede establecer el/la autor/a.

2.3.2.1 Restricciones de Sistema

Nuevamente las restriciones del sistema se deducen trivialmente de las anteriores definiciones y son las siguientes:

RT[r1] Todos las relaciones deben estar etiquetadas.

$$EC_m = (C, II, R_c, R_f, A_c, A_f), \text{ donde}$$
$$\forall\ r \in R_c,\ r.\text{nombre} <> \text{“”},$$
$$\forall\ r \in R_f,\ r.\text{nombre} <> \text{“”}.$$

RT[r2] Todas las relaciones que aparezcan en la Estructura Conceptual EC_m pertenecerán al conjunto de Relaciones Conceptuales R_c o al conjunto de Relaciones Funcionales R_f.

$$EC_m = (C, II, R_c, R_f, A_c, A_f), \text{ donde}$$
$$\forall\ <c_o, r_c, c_d> \in A_c,\ r_c \in R_c.$$
$$\forall\ <c_o, r_f, ii> \in A_f,\ r_f \in R_f.$$

RT[r3] El conjunto de relaciones de dependencia está predefinido por el sistema y está formado por:{*EsUn*, *ParteDe*, *TipoDe*}.

$$EC_m = (C, II, R_c, R_f, A_c, A_f), \text{ donde}$$
$$\forall\ r_d \in R_d,\ r_d.\text{nombre} \in \{EsUn, ParteDe, TipoDe\}, \text{ con } R_d \subseteq R_c.$$

2.3.2.2 Restricciones de Autor

Entre las Restricciones variables se distingen un total de 16 restricciones que podrán ser asociadas a una determinada relación conceptual y que, por tanto, deberán verificar las Relaciones Conceptuales R_c que se etiqueten con este nombre de relación. Estas restricciones se pueden catalogar en seis grupos que denominamos: Acíclicas —Ac—, Antitransitiva —At—, Salida —S—, Llegada —Ll—, Transmisión —Tr— y Asociación —As—. Estas restricciones no se deducen de las definiciones anteriores y el propio creador/a del sistema hipermedia deberá caracterizar sus relaciones conceptuales con un subconjunto de las mismas —tabla 5.4—.

Restricciones Acíclicas

Ac1 Antirreflexiva. El concepto origen y destino de una relación no puede coincidir.

$$\forall\ < c_1, r_c, c_2 > \in A_c \Rightarrow c_1 \neq c_2 \text{ ó } \Gamma^{+1}(c_1, r_c) \neq c_1.$$

Ac2 Antisimétrica. Si un concepto es el origen de una relación, el concepto destino de esa relación no puede ser el concepto origen del anterior.

$$\forall\ < c_1, r_c, c_2 > \in A_c \Rightarrow < c_2, r_c, c_1 > \notin A_c \text{ ó } \Gamma^{+1}(c_2, r_c) \neq c_1.$$

Ac3 Aciclicidad. No puede existir una cadena cerrada de relaciones entre los conceptos de la EC_m. Esto es, para todo camino formado por relaciones con el mismo nombre el concepto origen de la primera relación del camino no puede coincidir con el concepto destino de la última relación.

$$\forall \mu = < c_1, r_{ci}, c_2 >, < c_2, r_{ci}, c_3 >, ..., < c_{n-1}, r_{ci}, c_n > \in EC_m, \text{ tal que } r_{ci} \in R_c \Rightarrow c_1 \neq c_n$$
$$\text{ó } c_1 \not\subset \Gamma^{+}(c_1, r_{ci}).$$

Restricción Antitransitiva

At Antitransitiva. Si un concepto c_1 es el origen de una relación r_{ci}, y el destino de ésta c_2 es el concepto origen de la misma relación para un tercer concepto c_3 entonces el primer concepto c_1 no puede ser el concepto origen de este tercer concepto c_3 con la misma relación. En general, para cualquier número de conceptos se puede enunciar como: para todo camino formado por una determinada relación r_{ci}, entre el concepto origen de la primera relación del camino y el concepto destino de la última relación no puede existir la misma relación.

$$\forall \mu = < c_1, r_{ci}, c_2 >, < c_2, r_{ci}, c_3 >, ..., < c_{n-1}, r_{ci}, c_n > \in EC_m \Rightarrow < c_1, r_{ci}, c_n > \notin A_c$$
$$\text{ó } \Gamma^{+1}(c_1, r_{ci}) \neq c_1.$$

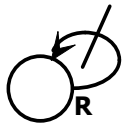
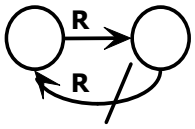
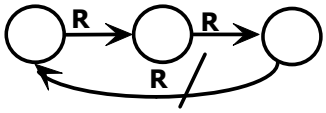
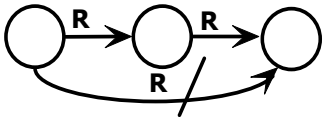
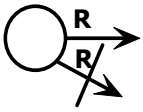
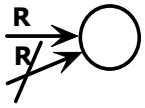
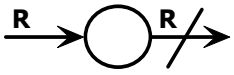
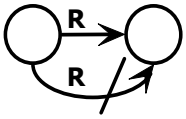
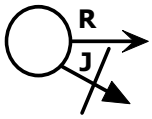
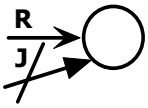
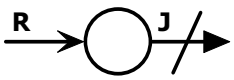
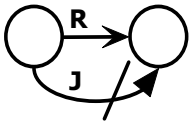
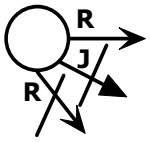
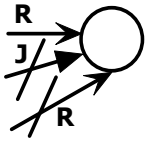
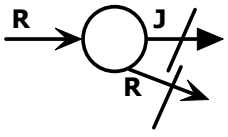
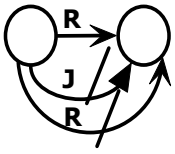
Acíclicas			Antitransitiva
Ac1	Ac2	Ac3	At
			
Salida	Llegada	Transmisión	Asociación
S1	LI1	Tr1	As1
			
S2	LI2	Tr2	As2
			
S3	LI3	Tr3	As3
			

Tabla 5.4 Restricciones de las Relacionas Conceptuales.

Restricciones de Salida

S1 Un concepto no puede ser origen de más de una asociación conceptual en las que participe la misma relación conceptual. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de salida***.**

$$\forall c_i \in C \text{ y } r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1.$$

S2 Si un concepto es origen en una asociación conceptual en la que participe la relación conceptual r_c, ese concepto no puede ser origen en asociaciones en las que intervengan el resto de las relaciones. Por tanto, *la relación conceptual presenta una Incompatibilidad con las otras relaciones.*

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^+(c_i, r_q)| = 0.$$

S3 Si un concepto es origen en una asociación conceptual en la que participe la relación conceptual r_c no puede ser origen en ninguna otra asociación. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^+(c_i, r)| = 0.$$

Restricciones de Llegada

LI1 Un concepto no puede ser destino de más de una asociación conceptual en las que participe la misma relación conceptual. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de llegada*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 1.$$

LI2 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, ese concepto no puede ser destino en asociaciones en las que intervengan el resto de las relaciones. Por tanto, la relación conceptual presenta una *Incompatibilidad con las otras relaciones*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^-(c_i, r_q)| = 0.$$

LI3 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c no puede ser destino en ninguna otra asociación. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^-(c_i, r)| = 0.$$

Restricciones de Transmisión

Tr1 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c entonces no puede ser origen en otra asociación conceptual en la que intervenga la misma relación, aunque puede ser origen de asociaciones en cuyas tripletas aparezcan otras relaciones conceptuales. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de transmisión*.

$$\forall\ c_i \in C,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_c)| = 0.$$

Tr2 Si un concepto es el destino en una asociación conceptual en la que participe la relación conceptual r_c entonces no puede ser origen de asociaciones en cuyas tripletas aparezcan las otras relaciones conceptuales. Por tanto, la relación conceptual presenta *Incompatibilidad con las otras relaciones*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^+(c_i, r_q)| = 0.$$

Tr3 Si un concepto es el destino en una asociación conceptual en la que participe la relación conceptual r_c entonces no puede ser origen en ninguna otra asociación conceptual. Se trata de un concepto hoja o *dead-end*. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^+(c_i, r)| = 0.$$

Restricciones de Asociación

As1 Si un concepto origen y un concepto destino están asociados mediante la relación conceptual r_c, no podrán existir otra asociación conceptual con los mismos elementos. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de asociación*. En realidad, esta restricción tiene que verificarla toda asociación porque no tendría sentido que existan elementos repetidos en un conjunto. A esta restricción la llamamos *universal* para todas las relaciones y será la cuarta invariante que tiene que verificar toda relación conceptual.

$$\forall\ <c_o, r_c, c_d> \in A_c,\ \nexists\ <c_o, s_c, c_d> \in A_c, \text{ tal que } r_c.\text{nombre} = s_c.\text{nombre}.$$

As2 Si entre un concepto origen y un concepto destino existe una asociación en la que participe la relación conceptual r_c, no podrá existir ninguna asociación entre ambos conceptos en la que intervengan las otras relaciones conceptuales. Por tanto, la relación conceptual presenta una *Incompatibilidad con las otras relaciones*.

$$\forall\ < c_o, r_c, c_d> \in A_c \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ <c_o, r_q, c_d> \notin A_c.$$

As3 Si entre un concepto origen y un concepto destino existe una asociación conceptual en la que participe la relación conceptual r_c no podrá existir ninguna otra asociación entre ambos. *Incompatibilidad total*.

$$\forall < c_o, r_c, c_d> \in A_c \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c\ , <c_o, r, c_d> \notin A_c.$$

2.3.2.3 Caracterización de las Relaciones de Dependencia R_D

Tal y como se definió **Def[ecm5]** existen tres tipos de relaciones de dependencia R_d: *EsUn*, *ParteDe* y *TipoDe*. Estas relaciones están proporcionadas y son caracterizadas por el sistema con un subcojunto de las restricciones relacionales de antemano. Por tanto, el conjunto de restricciones relacionales asociado a cada una de estas relaciones de dependencia no puede cambiar y debe ser cumplido en todo momento.

La relación de dependencia *EsUn* debe verificar el siguiente subconjunto de restricciones {Ac3, At, Tr3, As3}. La relación de dependencia *ParteDe* y *TipoDe* deben cumplir el siguiente subconjunto de restricciones {Ac3, As3}. Por tanto podemos concluir que:

- Todo tipo de relación de dependencia es acíclica Ac3.
- Todo tipo de relación de dependencia es transitiva con la excepción de *EsUn* que verifica At.
- Todo tipo de relación de dependencia es incompatible con los restantes tipos de relaciones de dependencia Tr3.
- El grafo formado por todos aquellos conceptos que están asociados con una determinada relación de dependencia debe ser acíclico.

2.3.3 Restricciones Relacionadas con los Ítemes de Información RT_{MII}

Los ítemes de información junto a los conceptos son nodos de la estructura conceptual —**Def[ecm9]**— y mantienen con ellos relaciones funcionales. Por tanto, a cualquier concepto de la estructura conceptual se le pueden asociar ítemes de información mediante relaciones funcionales. Estas asociaciones se expresan mediante las tripletas <c, r_f, ii>, con *c* un concepto, r_f una relación funcional y *ii* un ítem de información. Como ya se dijo anteriormente todos los ítemes de información mantienen una relación funcional especial denominada *genérica* con el *concepto de sistema*. Además un ítem de información puede estar asociado a más de un concepto y un ítem de información puede estar asociado a un concepto mediante más de una relación funcional. Para la entidad u objeto ítem de información se definen las siguientes propiedades:

- Identificador que debe ser único. Esta propiedad se denota como *Id*. El identificador está formado por: un número de ítem y un número de variante separados por un punto, *Id = numii.numvar*. Por defecto, el número de variante para el primero de los ítemes será 0, esto es, *Id = numii.0*
- Lenguaje de expresión (texto, sonido, gráfico, imagen, animación, ejecución, hipermedia). Esta propiedad se denota como *lenguaje*.
- Aplicación a la que está asociado el ítem de información con la que se edita el ítem. Esta propiedad se denota como *aplic*.

- Formato o tipo de fichero que contiene el ítem de información. Esta propiedad se denota como *formato*.
- Complejidad o nivel de especialización. Esta propiedad se denota como *nivel*.
- Propiedades de edición: idioma de expresión, autor/es, organismo, fecha de creación, última fecha de actualización, calidad, número de visitantes, tipo de visitantes, organismos que lo recomiendan, personas que lo recomiendan. Este conjunto de propiedades se denota como *edición*.

Las restricciones predefinidas por el sistema para los ítemes de información están relacionadas con algunas de estas propiedades. Se definen las siguientes:

RT[ii1] Cada ítem de información tendrá un único *número de identificación*. Este número será el mismo durante toda su vida, desapareciendo cuando sea borrado.

$$\forall\ ii \in II \not\exists\ ii' \text{ tal que } ii.id = ii'.id$$

RT[ii2] Este número estará formado por otros dos: *número de ítem* y *número de variante*. Por defecto, el número de variante para el primero de los ítemes será 0.

$$\forall\ ii \in II,\ ii.id = ii.numii.numvar, \text{ con } numii \in \mathbb{N},\ numvar \in \mathbb{N}^*$$
$$\forall\ \{ii_1, ii_2, ..., ii_n\} \subseteq II, \text{ tal que } ii_1.numii = ii_2.numii = ... = ii_n.numii,\ ii_1.numvar = 0$$

RT[ii3] Todas las *variantes* de un ítem de información deben estar asociadas a los mismos conceptos y mediante las mismas funciones.

$$\forall\ \{ii_1, ii_2, ..., ii_n\} \subseteq II, \text{ tal que } ii_1.numii = ii_2.numii = ... = ii_n.numii,$$
$$\Gamma^-(ii_1) = \Gamma^-(ii_2) = ... = \Gamma^-(ii_n) = C_k \text{ y}$$
$$\forall\ c \in C_k,\ \Lambda(c, ii_1) = \Lambda(c, ii_2) = ... = \Lambda(c, ii_n).$$

Sin embargo, aquellos ítemes de información que coincidan en sus asociaciones funcionales no tienen por qué ser variantes.

> Por ejemplo, imagine que tiene de un pintor toda su obra, de tal forma, que cada cuadro está almacenado en un ítem de información distinto. Todos estos ítemes estarían asociados al mismo concepto, *el nombre del pintor*, con la misma relación funcional, *obra*.

Las restricciones variables que se definen para los ítemes de información están relacionadas con la posibilidad de componer ítemes de información. La explicación de esta operación y de sus restricciones asociadas se pospone, debido a que la composición de ítemes depende de si los conceptos a los que están asociados reciben una relación conceptual que permita agruparlos. Se retomará esta cuestión en la última sección —2.6— dedicada a la composición.

2.3.4 Restricciones Relacionadas con los Diccionarios RT_{MD}

Todas las restricciones relacionadas con los diccionarios son fijas y, por tanto, están predefinidas por el sistema. Casi todas estas restricciones son obvias pues se deducen de la propia definición de conjunto o son consecuencia de las anteriores definiciones.

RT[d1] No puede haber dos etiquetas de conceptos iguales en el diccionario de conceptos.

$$\forall\ dc \in DC, \nexists\ dc' \text{ tal que } dc = dc'.$$

RT[d2] No se puede borrar una etiqueta de concepto si se está utilizando en alguno de los conceptos de alguna estructura conceptual del sistema o en algún ítem de información del sistema.

$$\forall\ c \in C, c.nombre \in DC.$$

RT[d3] No puede haber dos etiquetas de relaciones iguales en el diccionario de relaciones conceptuales.

$$\forall\ dr_c \in DR_c, \nexists\ dr_c' \text{ tal que } dr_c = dr_c'.$$

RT[d4] No se puede borrar una etiqueta de relación conceptual si se está utilizando en alguna de las relaciones entre conceptos de alguna estructura conceptual del sistema.

$$\forall\ r_c \in R_c, r_c.nombre \in DR_c.$$

RT[d5] No puede haber dos etiquetas de relaciones funcionales iguales en el diccionario de funciones.

$$\forall\ dr_f \in DR_f, \nexists\ dr_f' \text{ tal que } dr_f = dr_f'.$$

RT[d6] No se puede borrar una etiqueta de función si se está utilizando en algún ítem de información del sistema.

$$\forall\ r_f \in R_f, r_f.nombre \in DR_f.$$

RT[d7] Los nombres o etiquetas definidos por el sistema no se pueden borrar de los diccionarios. Los nombres {*sistema*}, {*inicio*} y {*generica*} únicamente pueden ser asignados por el sistema.

$$\{sistema\} \in DC,$$

$$\{inicio, EsUn, ParteDe, TipoDe\} \in DR_c,$$

$$\{genérica, introducción, definición, ejemplo, aclaración, referencias, algoritmo, advertencias, explicación, opinión, comparación, análisis, justificación, valoración, antecedentes, enfoques, profundización, ampliación\} \in DR_f.$$

2.4 Acciones Evolutivas del Subsistema de Memorización AC_{EM}

El Subsistema de Memorización es el gran almacén de información del sistema hipermedia. Este Subsistema es el taller en el que el autor/a de los hiperdocumentos puede crear sus dominios conceptuales y de información. Dicho de otro modo, este Subsistema debe ofrecer un repertorio de operaciones mediante las cuales sea posible la especificación, modificación y evolución de la estructura conceptual EC_m, del diccionario D y de las restricciones de autor RT_a que caracterizan a cada una de las relaciones conceptuales de referencia R_{cf}. En consecuencia, es necesario definir acciones evolutivas

que deberán respetar un subconjunto de las restricciones antes descritas para garantizar la consistencia del cambio. Además, el susbsistema deberá propagar dicho cambio con objeto de hacer evolucionar al resto de los componentes del Subsistema y al resto de los sistemas del Modelo SEM-HP —estas propagaciones externas se expondrán al final, una vez que se haya descrito la totalidad de los sistemas—.

Def[ecm16] Para cada uno de los elementos del *Subsistema de Memorización*,— Estructura Conceptual EC_m, Diccionario D y Restricciones de autor RT_a— se definen un conjunto de acciones evolutivas AC_{em}. Estas operaciones deben respetar en todo momento las restricciones definidas que se convierten en precondiciones y postcondiciones para las acciones AC_{em} que hacen evolucionar a la EC_m, al diccionario D y a las Restricciones de autor RT_a.

$$AC_{mec}, AC_{md}, AC_{mrt} \subseteq AC_{em},$$
$$AC_{mec} \cup AC_{md} \cup AC_{mrt} = AC_{em},$$
$$AC_{mec} \cap AC_{md} \cap AC_{mrt} = \varnothing.$$

2.4.1 Cambiando la Estructura Conceptual EC_m

La Estructura Conceptual es un grafo débilmente conectado, esto es, no se permiten conceptos aislados ni relaciones que no asocien bien conceptos o bien conceptos e ítemes de información. Por esta razón, las acciones que cambian la EC se dedicarán a crear, borrar o modificar asociaciones y no a modificar únicamente conceptos. A este grupo de operaciones las llamaremos acciones evolutivas asociativas AC_{ma}. Por otro lado, es necesario crear, editar, borrar y modificar ítemes de información. Al ser hojas de la EC, su manipulación no la invalida. Para la modificación de estos objetos se contará con otro grupo de acciones evolutivas AC_{mii}.

$$AC_{ma}, AC_{mii} \subseteq AC_{mec},$$
$$AC_{ma} \cup AC_{mii} = AC_{mec},$$
$$AC_{ma} \cap AC_{mii} = \varnothing.$$

2.4.1.1 Operaciones Básicas Relacionadas con la Estructura Conceptual AC_b

Las acciones evolutivas relacionadas con las asociaciones AC_{ma} se apoyan en un grupo de operaciones básicas AC_b o de sistema. Es decir, cualquier operación asociativa es una composición de las operaciones básicas. En realidad, las acciones asociativas serán más bien transacciones. El usuario no podrá hacer un uso directo o individual de estas operaciones básicas que forman parte de una transacción. En consecuencia, las operaciones básicas no tienen por qué cumplir todas las restricciones definidas para la EC_m que, sin embargo, si deberán cumplirlas las acciones asociativas AC_{ma}.

AC_b = {crea-nodo, borra-nodo, crea-arco, borra-arco, pon-etiqueta-nodo, pon-etiqueta-arco, quita_etiqueta-nodo, quita-etiqueta-arco, nodos-accesibles-desde, nodos-alcanzan-a, nodos-conectados-a, relacionados-mediante, marca-nodo-raíz, quita-nodo-raíz}

Dada la estructura conceptual que será manipulada, $EC_m = (C, II, R_c, R_f, A_c, A_f)$, llamaremos, $EC_m' = (C', II', R_c', R_f', A_c', A_f')$, a la estructura conceptual una vez que haya sido manipulada.

Las operaciones básicas se van a dividir en dos grupos: el de modificación y el de consulta.

AC[b1]	**Crea-nodo**(EC_m): c′
Definición	Se crea un nuevo nodo.
Precondición	
Efecto	→ EC_m′, C′ = {c′} ∪ C.

Tabla 5.5 Operación Básica: Crea-nodo.

AC[b2]	**Borra-nodo**(c, EC_m): boolean
Definición	Se borra un nodo.
Precondición	c ∈ C y c.nombre = "".
Efecto	→ EC_m′, C′ = C – {c}.

Tabla 5.6 Operación Básica: Borra-nodo.

AC[b3]	**Crea-arco**(c_o, o_d, EC_m): <c_o, r′, o_d>
Definición	Se crea un arco entre dos nodos existentes
Precondición	c_o ∈ C, o_d ∈ C ó o_d ∈ II, r′ ∉ R y r′.nombre= "".
Efecto	→ EC_m′, R′ = {r′} ∪ R.

Tabla 5.7 Operación Básica: Crea-arco.

AC[b4]	**Borra-arco**(<c_o, r, c_d>, EC_m)
Definición	Se borra un arco
Precondición	c_o ∈ C, c_d ∈ C, r ∈ R y r.nombre= ""
Efecto	→ EC_m′, R′ = R – {r}.

Tabla 5.8 Operación Básica: Borra-arco.

AC[b5]	**Pon-etiqueta-nodo**(c, et, EC_m): boolean
Definición	Se etiqueta un nodo con una de las etiquetas existentes o con una nueva etiqueta.
Precondición	$c \in C$ y $\forall\ c_i, c_j \in C$, c_i.nombre $\neq$ c_j.nombre RT[c3]
Efecto	$\rightarrow EC_m'$, c.nombre = et.
Propagación interna	Si et $\notin$ DC entonces DC' = DC $\cup$ {et}.

Tabla 5.9 Operación Básica: Pon-etiqueta-nodo.

AC[b6]	**Pon-etiqueta-arco**(r, et, EC_m): boolean
Definición	Se etiqueta un arco con una de las etiquetas existentes o con una nueva etiqueta.
Precondición	$r \in R$.
Efecto	$\rightarrow EC_m'$, Si $r \in R_c$, entonces r.nombre = et, Si $r \in R_f$, entonces r.nombre = et.
Propagación interna	Si et $\notin$ DR entonces DR' = DR $\cup$ {et}. Si et $\notin$ DF entonces DF' = DF $\cup$ {et}.

Tabla 5.10 Operación Básica: Pon-etiqueta-arco.

AC[b7]	**Quita-etiqueta-nodo**(c, EC_m): boolean
Definición	Se quita la etiqueta de un nodo.
Precondición	$c \in C$.
Efecto	$\rightarrow EC_m'$, c.nombre = "".

Tabla 5.11 Operación Básica: Quita-etiqueta-nodo.

AC[b8]	**Quita-etiqueta-arco**(r, EC_m): boolean
Definición	Se quita la etiqueta de un arco.
Precondición	$r \in R$.
Efecto	$\rightarrow EC_m'$, r.nombre = "".

Tabla 5.12 Operación Básica: Quita-etiqueta-arco.

AC[b9]	**Marca-nodo-raíz**(c, EC_m): boolean
Definición	Se hace conectar este nodo con el nodo-sistema mediante un arco de inicio y se convierte este nodo, en el nodo raíz de la estructura conceptual.
Precondición	$c \in C$, $c \neq r$.
Efecto	$\rightarrow EC_m'$, $A_c' = <\mathfrak{s}, i, c> \cup A_c$, $c = r$.

Tabla 5.13 Operación Básica: Marca-nodo-raíz.

AC[b10]	**Quita-nodo-raíz**(c, EC_m): boolean
Definición	Se hace desconectar el nodo-raíz del nodo-sistema.
Precondición	$c \in C$, $c = r$.
Efecto	$\rightarrow EC_m'$, $A_c' = A_c - <\mathfrak{s}, i, c>$, $c \neq r$.

Tabla 5.14 Operación Básica: Quita-nodo-raíz.

Operaciones Básicas de Consulta

AC[b11]	**Nodos-accesibles-desde**(n, c, EC_m): C_k
Definición	Calcula el conjunto de nodos que son accesibles desde un nodo dado c mediante un camino máximo de n arcos. Lo que en grafos se denomina la función de correspondencia Γ^{+n}, siendo n el número de arcos.
Precondición	$n \in \mathbb{N}$, $c \in C$.
Efecto	$\rightarrow (EC_m, C_k)$, $\forall c_k \in C_k$, $\exists \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n >$, tal que $c_1 = c$ y $c_n = c_k$ y $\|\mu\| = n$.

Tabla 5.15 Operación Básica: Nodos-accesibles-desde.

AC[b12]	**Nodos-alcanzan-a**(n, c, EC_m): C_k
Definición	Se calcula el conjunto de nodos que alcanzan a un determinado nodo mediante un camino máximo de n arcos. La función de correspondencia Γ^{-n}, siendo n el número de arcos.
Precondición	$n \in \mathbb{N}$, $c \in C$.
Efecto	$\rightarrow (EC_m, C_k)$, $\forall c_k \in C_k$, $\exists \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n >$, tal que $c_1 = c_k$ y $c_n = c$ y $\|\mu\| = n$.

Tabla 5.16 Operación Básica: Nodos-alcanzan-a.

AC[b13]	**Nodos-conectados-a**(n, c, EC_m): C_k
Definición	Se calcula el conjunto de nodos que están conectados a un determinado nodo mediante una secuencia de asociaciones en cualquier dirección. La función de correspondencia Υ.
Precondición	$n \in \mathbb{N}$, $c \in C$.
Efecto	$\rightarrow (EC_m, C_k)$, $\Upsilon^1(c) = \Gamma^+(c) \cup \Gamma^-(c)$ $\Upsilon^n(c) = \Upsilon^{n-1}(\Gamma^+(c) \cup \Gamma^-(c)) = \Upsilon^{n-2}(\Gamma^+(\Gamma^+(c) \cup \Gamma^-(c))$ $\cup \Gamma^-(\Gamma^+(c) \cup \Gamma^-(c)))$

Tabla 5.17 Operación Básica: Nodos-conectados-a.

AC[b14]	**Relacionados-mediante**(c_o, o_d, EC_m): R_k
Definición	Se calcula el conjunto de arcos que parten de un nodo origen y llegan a un nodo destino. La función de correspondencia Λ.
Precondición	$c_o \in C$, $o_d \in C$ ó $o_d \in II$.
Efecto	$\rightarrow (EC_m, R_k)$, $\forall r_k \in R_k$, $\exists <c_o, r_k, o_d>$.

Tabla 5.18 Operación Básica: Relacionados-mediante.

2.4.1.2 Acciones Evolutivas Asociativas AC_{MA}

Tal y como se indicó al principio del apartado 2.4.1, las acciones de evolución son transacciones que están formadas por subconjuntos de las operaciones básicas anteriormente descritas.

AC_{ma} = {Crea-concepto, Crea-asociación-conceptual, Crea-asociación-funcional, Borra-asociación-conceptual, Borra-asociación-funcional, Crea-A_c-y-concepto, Modifica-asociación-conceptual, Modifica-asociación-funcional, Marca-concepto-raíz, Cambia-etiqueta}

Dada la estructura conceptual que será manipulada, $EC_m = (C, II, R_c, R_f, A_c, A_f)$, llamaremos, $EC_m' = (C', II', R_c', R_f', A_c', A_f')$, a la estructura conceptual una vez que haya sido manipulada.

AC[a1]	**Crea-concepto**(et, EC_m): c′
Definición	Se crea un nodo y se etiqueta con una de las etiquetas del diccionario de conceptos o con una nueva etiqueta. Este nodo se hará colgar del concepto-sistema y será marcado como concepto-raíz de una nueva estructura conceptual.
Transacción	c′ = Crea-nodo(EC_m) Pon-etiqueta-nodo(c′, et, EC_m) Marca-nodo-raíz(c′, EC_m).
Precondición	$\forall$ c $\in$ C, c.nombre $\neq$ c′.nombre RT[c3].
Efecto	$\rightarrow$ EC_m′, c′.nombre = et, C′ = {c′} $\cup$ C, A_c′ = <$\mathfrak{s}$, i, c> $\cup$ A_c, c′ = r.
Propagación Interna	et $\notin$ DC $\Rightarrow$ Crea-etiqueta-concepto.

Tabla 5.19 Acciones Evolutivas Asociativas: Crea-concepto.

AC[a2]	**Crea-asociación-conceptual**(c_o, c_d, et, EC_m): <c_o, r_c′, c_d >
Definición	Se crea un arco entre dos conceptos existentes, y se etiqueta con una de las etiquetas del diccionario de tipos de relaciones conceptuales o con un nuevo nombre de relación conceptual.
Transacción	<c_o, r_c′, c_d> = Crea-arco(c_o, c_d, EC_m) Pon-etiqueta-arco(r_c′, et, EC_m)
Precondición	c_o, c_d $\in$ C, <c_o, r_c′, c_d > $\notin$ A_c Se deben verificar las restricciones variables relacionales que el autor ha definido para esa relación conceptual.
Efecto	$\rightarrow$ EC_m′, r_c′.nombre = et, A_c′ = <c_o, r_c′, c_d> $\cup$ A_c.
Propagación Interna	Si et $\notin$ DR_c $\Rightarrow$ Crea-etiqueta-Rconceptual

Tabla 5.20 Acciones Evolutivas Asociativas: Crea-asociación-conceptual.

AC[a3]	**Crea-asociación-funcional**(c_o, ii, et, EC_m): <c_o, r_f′, ii >
Definición	Se crea un arco entre un concepto y un ítem de información, y se etiqueta con una de las etiquetas del diccionario de relaciones funcionales o con un nuevo nombre de relación funcional.
Transacción	<c_o, r_f′, ii > = Crea-arco(c_o, ii, EC_m) Pon-etiqueta-arco(r_f′, et, EC_m)
Precondición	c_o $\in$ C, ii $\in$ II, <c_o, r_f′, ii > $\notin$ A_f.
Efecto	$\rightarrow$ EC_m′, r_f′.nombre = et, A_f′ = <c_o, r_f′, ii> $\cup$ A_f,
Propagación Interna	Si et $\notin$ DR_c $\Rightarrow$ Crea-etiqueta-Rfuncional

Tabla 5.21 Acciones Evolutivas Asociativas: Crea-asociación-funcional.

AC[a4]	**Borra-asociación-conceptual**(<c_o, r_c, c_d>, EC_m): boolean
Definición	Se borra una relación conceptual y cualquier concepto —junto con las relaciones conceptuales y funcionales unidas al concepto— que se convierta en inaccesible después de borrar esa relación Def[ecm9]. Esto es, que quede desconectado de la EC_m. En el caso de que esta relación conceptual sea la de comienzo, se borraría toda una estructura conceptual.

Transacción	Quita-etiqueta-arco(r_c, EC_m)
	Borra-arco(<c_o, r_c, c_d>, EC_m)
	C_k = C - (Nodos-conectados-a(1, r, EC_m))
	$\forall$ $c_k \in C_k$
	O_h = Nodos-accesibles-desde(1, c_k, EC_m) //conceptos e ítemes
	C_j = Nodos-alcanzan-a(1, c_k, EC_m) //conceptos
	$\forall$ $o_h \in O_h$ //se borran todos los arcos que parten de c_k
	R_h = Relacionados-mediante(c_k, o_h, EC_m)
	$\forall$ $r_h \in R_h$
	Quita-etiqueta-arco(r_h, EC_m),
	Borra-arco(<c_k, r_h, o_h>, EC_m)
	$\forall$ $c_j \in C_j$ //se borran todos los arcos que llegan a c_k
	R_{cj} = Relacionados-mediante(c_j, c_k, EC_m)
	$\forall$ $r_{ci} \in R_{cj}$
	Quita-etiqueta-arco(r_{cj}, EC_m),
	Borra-arco(<c_j, r_{cj}, c_k>, EC_m)
	Quita-etiqueta-nodo(c_k, EC_m) //se borra ese nodo y su etiqueta
	Borra-nodo(c_k, EC_m)

Precondición	c_o, $c_d \in C$, <c_o, r_c, c_d> $\in A_c$, Antes hay que verificar que al quitar esas relaciones y conceptos se siguen verificando las restricciones de cada relación conceptual. Si alguna restricción no se cumple, entonces no se puede borrar la asociación conceptual.
Efecto	$\rightarrow$ EC_m', $A_c' = A_c$ - <c_o, r_c, c_d>, $\Upsilon^*(r) = C_k$, $\forall$ $c \notin C_k$, $\Gamma^{+1}(c) = C_h \cup II_h = O_h$, $\Gamma^{-1}(c) = C_j$, con C_h, C_j, $C_k \subseteq C$ y $II_h \subseteq II$ $\forall$ $c_h \in C_h$, $\Lambda(c, c_h) = R_h \subseteq R$, $\forall$ $r_h \in R_h$, //conceptual y funcional $A' = A$ - <c, r_h, o_h>, $II_p' = II_p$ - o_h ó $C_p' = C_p - o_h$ $\forall$ $c_j \in C_j$, $\Lambda(c_j, c) = R_{cj} \subseteq R_c$, $\forall$ $r_{cj} \in R_{cj}$, $A_c' = A_c$ - <c_j, r_{cj}, c>.

Tabla 5.22 Acciones Evolutivas Asociativas: Borra-asociación-conceptual.

Por ejemplo —figura 5.2—, si queremos borrar la relación que va desde el concepto c al concepto f, se borra ese arco y su etiqueta y se calcula Nodos-conectados (a) = {g, b, c, d, h}, luego el complementario es {e,f}, se borran esos conceptos y las relaciones que llegan o parten a estos nodos. Los ítemes siempre permanecen asociados mediante la relación funcional especial genérica al concepto-sistema. Por tanto, no hay que preocuparse si no están asociados a ningún concepto.

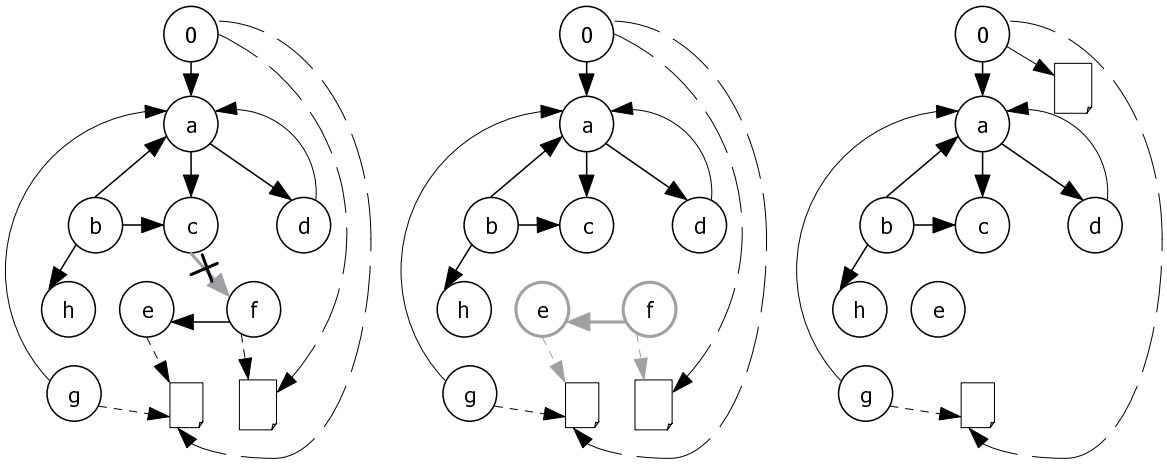


Figura 5.2 Ejemplo de la acción evolutiva asociativa: Borra-asociación-conceptual.

AC[a5]	**Borra-asociación-funcional**(<c_o, r_f, ii>, EC_m): boolean
Definición	Se borra una relación funcional entre un concepto y un ítem de información. No hay que preocuparse si para el ítem es la única relación funcional que mantiene porque, al menos, colgará del concepto-sistema mediante la relación funcional especial *genérica* **Def[ecm9]**.
Transacción	Quita-etiqueta-arco(r_f', EC_m) Borra-arco(<c_o, r_f', ii>, EC_m)
Precondición	$c_o \in C$, <c_o, r_f, ii> $\in A_f$, ii $\in$ II.
Efecto	→ EC_m', $A_f' = A_f$ - <c, r_f, ii>.

Tabla 5.23 Acciones Evolutivas Asociativas: Borra-asociación-funcional.

AC[a6]	**Crea-A_c-y-concepto**(c_o, et_c, et_r, EC_m): <c_o, r_c', c_d'>
Definición	Se crea una relación conceptual y un concepto destino. Debe existir el concepto origen.
Transacción	c_d' = Crea-nodo(EC_m) Pon-etiqueta-nodo(c_d', et_c, EC_m) <c_o, r_c', c_d'> = Crea-arco(c_o, c_d', EC_m) Pon-etiqueta-arco(r_c', et_r, EC_m)
Precondición	$c_o \in C$, $c_d' \notin C$ Se deben verificar las restricciones variables relacionales que el autor ha definido para esa relación conceptual
Efecto	→ EC_m', c_d'.nombre = et_c, r_c'.nombre = et_r $A_c' =$ <c_o, r_c', c_d'> $\cup$ A_c.
Propagación Interna	Si $et_c \notin DC \Rightarrow$ Crea-etiqueta-concepto Si $et_r \notin DR_c \Rightarrow$ Crea-etiqueta-Rconceptual

Tabla 5.24 Acciones Evolutivas Asociativas: Crea-A_c-y-concepto.

AC[a7]	**Modifica-asociación-conceptual**(c_o, c_n, EC_m): <c_o, r_c, c_d>'
Definición	Se cambia el concepto origen o destino de una relación conceptual por otro concepto de la E-C. Debe existir el nuevo concepto origen o destino en la estructura conceptual. Al modificar una relación hay que calcular los conceptos que dejan de estar conectados desde el concepto-raíz. Todos esos conceptos deben borrarse y todas las relaciones que llegan y parten de él. Para aquellos conceptos que mantengan relaciones funcionales con ítemes se borra esta relación funcional.

Transacción	Quita-etiqueta-arco(r_c, EC_m)
	Borra-arco(<c_o, r_c, c_d>, EC_m)
	<c_o, r_c, c_n> = Crea-arco(c_o, c_n, EC_m) ó
	<c_n, r_c, c_d> = Crea-arco(c_n, c_d, EC_m)
	Pon-etiqueta-arco(r_c, et, EC_m)
	
	C_k = C - (Nodos-conectados-a(1, r, EC_m))
	$\forall$ $c_k \in C_k$
	O_h = Nodos-accesibles-desde(1, c_k, EC_m) //conceptos e ítemes
	C_j = Nodos-alcanzan-a(1, c_k, EC_m) //conceptos
	
	$\forall$ $o_h \in O_h$ //se borran todos los arcos que parten de c_k
	R_h = Relacionados-mediante(c_k, o_h, EC_m)
	$\forall$ $r_h \in R_h$
	Quita-etiqueta-arco(r_h, EC_m),
	Borra-arco(<c_k, r_h, o_h>, EC_m)
	
	$\forall$ $c_j \in C_j$ //se borran todos los arcos que llegan a c_k
	R_{cj} = Relacionados-mediante(c_j, c_k, EC_m)
	$\forall$ $r_{ci} \in R_{cj}$
	Quita-etiqueta-arco(r_{cj}, EC_m),
	Borra-arco(<c_j, r_{cj}, c_k>, EC_m)
	
	Quita-etiqueta-nodo(c_k, EC_m) //se borra ese nodo y su etiqueta
	Borra-nodo(c_k, EC_m)

Precondición	c_o, c_d, $c_k \in C$, <c_o, r_c, c_d> $\in A_c$. Se deben verificar las restricciones variables relacionales que el autor ha definido para esa relación conceptual
Efecto	$\rightarrow EC_m'$, Si c_o cambia, entonces $A_c' = A_c$ - <c_o, r_c, c_d>, $A_c' =$ <c_n, r_c, c_d> $\cup A_c$. Si c_d cambia, entonces $A_c' = A_c$ - <c_o, r_c, c_d>, $A_c' =$ <c_o, r_c, c_n> $\cup A_c$ Υ^* (r) = C_k, $\forall$ c $\notin C_k$, Γ^{+1}(c) = C_h, Γ^{-1}(c) = C_j, con C_h, C_j, $C_k \subseteq C$ $\forall$ $c_h \in C_h$, Λ(c, c_h) = $R_h \subseteq R$, $\forall$ $r_h \in R_h$, A' = A - <c, r_h, c_h> //conceptual y funcional $\forall$ $c_j \in C_j$, Λ(c_j, c) = $R_{cj} \subseteq R_c$, $\forall$ $r_{cj} \in R_{cj}$, $A_c' = A_c$ - <c_j, r_{cj}, c>. $\forall$ c $\notin C_k$, $\forall$ $r_f \in R_{fh}$, Γ^{+}(c, r_{fh}) = $II_h \subseteq II$, $\forall$ $ii_h \in II_h$, $A_f' = A_f$ - <c, r_{fh}, ii_h>.

Tabla 5.25 Acciones Evolutivas Asociativas: Modifica-asociación-conceptual.

AC[a8]	**Modifica-asociación-funcional**(<c_o, r_f, ii >, o_n, EC_m): <c_o, r_f, ii >'
Definición	Se cambia el concepto origen o el ítem destino de una relación funcional. Tiene que existir el concepto al que se conecta el ítem o el ítem al que se quiere conectar el concepto.
Transacción	Quita-etiqueta-arco(r_f, EC_m) Borra-arco(<c_o, r_c, ii>, EC_m) <c_o, r_f, o_n> = Crea-arco(c_o, o_n, EC_m) ó <o_n, r_f, ii> = Crea-arco(o_n, ii, EC_m) Pon-etiqueta-arco(r_f, et, EC_m)
Precondición	$c_o \in C$, ii $\in$ II, $o_k \in C$ ó $o_k \in$ II, <c_o, r_f, ii > $\in A_f$.
Efecto	→ EC_m' Si $o_n \in C$, entoces $A_f' = A_f$ - <c_o, r_f, ii>, A_f' = <o_n, r_f, ii> $\cup A_f$. Si $o_n \in$ II, entoces $A_f' = A_f$ - <c_o, r_f, ii>, A_f' = <c_o, r_f, o_n> $\cup A_f$.

Tabla 5.26 Acciones Evolutivas Asociativas: Modifica-asociación-funcional.

AC[a9]	**Marca-concepto-raíz**(c_i, EC_m): boolean
Definición	Se hace conectar este concepto con el concepto-sistema mediante una relación de comienzo y se convierte este concepto, en el concepto raíz de la estructura conceptual. Para ello, antes se desconecta el anterior concepto raíz del concepto-sistema.
Transacción	Quita-nodo-raíz(c, EC_m) // el original, Marca-nodo-raíz(c_i, EC_m) // el nuevo nodo
Precondición	$c_i \in C$
Efecto	→ EC_m', $A_c' = A_c$ - <ɕ, i, r>, c_i = r, A_c'' = <ɕ, i, c_i> $\cup A_c'$.

Tabla 5.27 Acciones Evolutivas Asociativas: Marca-concepto-raíz.

En la figura 5.3 se marca un nuevo concepto como raíz. Antes el concepto raíz era el a, despues de la acción evolutiva, el concepto c se convierte en el raíz.

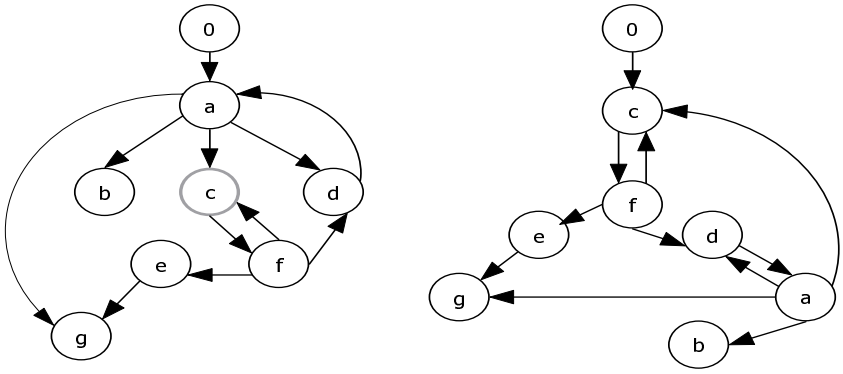


Figura 5.3 Ejemplo de la acción evolutiva asociativa: Marca-concepto-raíz.

AC[a10]	**Cambia-etiqueta**(o_k, et, EC_m): boolean
Definición	Se cambia la etiqueta de un concepto o de una relación por otra etiqueta tomada del diccionario de conceptos o relaciones, o por una nueva.
Transacción	Si $o_n \in C$ Quita-etiqueta-nodo(o_n, EC_m) Pon-etiqueta-nodo(o_n, et, EC_m) Si $o_n \in R$ Quita-etiqueta-arco(o_n, EC_m) Pon-etiqueta-arco(o_n, et, EC_m)
Precondición	$o_k \in C \; \forall \; c \in C$, c.nombre $\neq$ c'.nombre **RT[c3]** ó $o_k \in R_c$ y se deben verificar las restricciones variables relacionales que el autor ha definido para esa relación conceptual ó $o_k \in R_f$
Efecto	$\rightarrow EC_m'$, o_n.nombre = et.
Propagación interna	Si $o_n \in C \Rightarrow$ Si et $\notin$ C $\Rightarrow$ Crea-etiqueta-concepto Si $o_n \in R_c \Rightarrow$ Si et $\notin DR_c \Rightarrow$ Crea-etiqueta-Rconceptual Si $o_n \in R_f \Rightarrow$ Si et $\notin DR_f \Rightarrow$ Crea-etiqueta-Rfuncional

Tabla 5.28 Acciones Evolutivas Asociativas: Cambia-etiqueta.

2.4.1.3 Acciones Evolutivas Relacionadas con los Ítemes de Información AC_{mii}

Los ítemes son piezas de información que pueden ser almacenadas con distintos formatos y editadas con sus respectivas aplicaciones. Asimismo, estas unidades también tienen una serie de propiedades relacionadas con su identificación, modo de expresión y edición. Por tanto, la acciones de evolución relacionadas con los ítemes se dedicarán a crear, modificar y borrar tanto su contenido como sus propiedades —ver apartado 2.3.3 donde se describen sus propiedades y restricciones—.

AC_{mii} = {Crea-ítem, Edita-contenido-ítem, Edita-propiedades-ítem, Borra-ítem, Crea-ítem-compuesto, Crea-variante, Convierte-en-variante, Crea-ítem-copia, Crea-ítem-copia-contenido Crea-ítem-copia-propiedades }

Dada la estructura conceptual que será manipulada, $EC_m = (C, II, R_c, R_f, A_c, A_f)$, llamaremos, $EC_m' = (C', II', R_c', R_f', A_c', A_f')$, a la estructura conceptual una vez que haya sido manipulada.

AC[ii1]	**Crea-ítem**(EC_m): ii'
Definición	Se crea un ítem de información y el sistema le asigna un número de ítem y el número de variante 0, y la fecha de creación. Este número no puede coincidir con ninguno de los existentes **RT[ii1]**. Su contenido será vacío. Por defecto, todo ítem se asocia al concepto-sistema mediante la relación funcional *genérica* **Def[ecm9]**.
Precondición	$\forall$ ii $\in$ II $\nexists$ ii' tal que ii.id = ii'.id. $\forall$ ii $\in$ II, $\exists$ <$\mathfrak{s}$, g, ii> $\in A_f$, con $\mathfrak{s} \in C$, $g \in R_f$.
Efecto	$\rightarrow EC_m'$, ii'.id = ii'.numii'.0, ii'.fecha-creación=fecha-actual, ii'.contenido="" II' = {ii'} $\cup$ II, A_f' = <$\mathfrak{s}$, g, ii'> $\cup$ A_f.

Tabla 5.29 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-ítem.

AC[ii2]	**Edita-contenido-ítem**(ii, EC_m): boolean
Definición	Se ejecuta la aplicación con la que se edita el contenido del ítem
Precondición	ii $\in$ II.
Efecto	$\rightarrow EC_m'$, ii.contenido = ii.aplic(ii.contenido).

Tabla 5.30 Acciones Evolutivas para cambiar los Ítemes de Información: Edita-contenido-ítem.

La acción evolutiva dedicada a editar las propiedades de un ítem mecere algunos comentarios relacionados con la naturaleza de las mismas. A continuación, se enumeran estas propiedades describiendo, en su caso, sus particularidades.

- No podrá borrarse ni modificarse su identificador, ni tampoco su fecha de creación y de actualización. Estas propiedades son mantenidas automáticamente por el sistema.
- El lenguaje de expresión podrá ser: texto, sonido, gráfico, imagen, animación, ejecución, e hipermedia.
- La aplicación a la que está asociado, esto es, con la que se edita el ítem de información, debe estar disponible.
- Para cambiar de formato y de aplicación deberá existir un conversor entre los dos tipos de formato.
- La propiedad idioma nos permitirá etiquetarlo con la lengua en la que se ha escrito el ítem.
- Se pueden indicar cuáles son el/los autor/es, organismo/s que proporcionan estos ítemes.
- La calidad, número de visitantes, tipo de visitantes, organismos que lo recomiendan, personas que lo recomiendan: propiedades mantenidas desde el sistema de navegación.

AC[ii3]	**Edita-propiedades-ítem**(ii, prop, EC_m): boolean
Definición	Se asignan, modifican o borran sus propiedades.
Precondición	ii $\in$ II, con prop <> {id, fecha-creación} si prop = aplic $\Rightarrow \exists$ aplic
Efecto	$\rightarrow EC_m'$, ii.*propiedad* = prop.

Tabla 5.31 Acciones Evolutivas para cambiar los Ítemes de Información: Edita-propiedades-ítem.

AC[ii4]	**Borra-ítem**(ii, EC_m): boolean
Definición	Se borra su contenido y sus propiedades y su identificador. Si existen varias variantes del mismo número de ítem, se reajustan los números de variante de los ítemes con el mismo número de ítem.
Precondición	ii $\in$ II
Efecto	$\rightarrow EC_m'$, $\forall$ <c, r_f, ii> $\in A_f$, $A_f' = A_f - \{<c, r_f, ii>\}$, con c $\in$ C y $r_f \in R_f$, II' = II – {ii}.

Tabla 5.32 Acciones Evolutivas para cambiar los Ítemes de Información: Borra-ítem.

AC[ii5]	**Crea-variante**(ii, EC_m): boolean
Definición	Se crea una nueva variante de un ítem dado. Para ello, se copia el contenido y las propiedades de un ítem. Se le asigna como identificador, el mismo número de ítem y un nuevo número de variante, se le asigna una nueva fecha de creación.
Precondición	ii $\in$ II, ii' $\notin$ II **RT[ii1]** $\forall$ ii $\in$ II $\not\exists$ ii' tal que ii.id = ii'.id **RT[ii3]** $\forall$ {ii_1, ii_2, ..., ii_n} $\in$ II, tal que ii_1.numii = ii_2.numii = ... = ii_n.numii, $\Gamma^-(ii_1) = \Gamma^-(ii_2) = ... = \Gamma^-(ii_n) = C_k$ y $\forall$ c $\in C_k$, $\Lambda(c, ii_1) = \Lambda(c, ii_2) = ... = \Lambda(c, ii_n)$.
Efecto	$\rightarrow EC_m'$, ii'.id = ii.numii.numvar+1, ii'.prop = ii.prop, ii.contenido=ii'.contenido, ii'.fecha-creación=fecha-actual.

Tabla 5.33 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-variante.

AC[ii6]	**Convierte-en-variante**(ii_1, ii_2, EC_m): boolean
Definición	Un ítem de información se convierte en una variante de otro ítem de información. Se cambia su identificador, dándole el mismo número de ítem del que va a ser variante y el número de variante que le corresponda. Los conceptos y funciones deben coincidir.
Precondición	$ii_1, ii_2 \in II$, $\Gamma^-(ii_1) = \Gamma^-(ii_2) = C_k$, $\forall c \in C_k$, $\Lambda(c, ii_1) = \Lambda(c, ii_2)$ **RT[ii1]** $\forall ii \in II \not\exists ii'$ tal que ii.id = ii'.id **RT[ii3]** $\forall \{ii_1, ii_2, ..., ii_n\} \in II$, tal que ii_1.numii = ii_2.numii = ... = ii_n.numii, $\Gamma^-(ii_1) = \Gamma^-(ii_2) = ... = \Gamma^-(ii_n) = C_k$ y $\forall c \in C_k$, $\Lambda(c, ii_1) = \Lambda(c, ii_2) = ... = \Lambda(c, ii_n)$.
Efecto	$\rightarrow EC_m'$, ii_1.id = ii_2.numii.numvar +1, ii_1.fecha-creación=fecha-actual

Tabla 5.34 Acciones Evolutivas para cambiar los Ítemes de Información: Convierte-en-variante.

AC[ii7]	**Crea-ítem-copia**(ii, EC_m): ii'
Definición	Se crea un nuevo ítem a partir de otro. Se copia el contenido y las propiedades del ítem. Se le asigna un nuevo número de ítem, el número de variante 0 y la fecha de creación.
Precondición	$ii \in II$, **RT[ii1]** $\forall ii \in II \not\exists ii'$ tal que ii.id = ii'.id
Efecto	$\rightarrow EC_m'$, ii'.id = ii.numii'.0, ii'.prop = ii.prop, ii.contenido=ii'.contenido, ii'.fecha-creación=fecha-actual. $II' = \{ii'\} \cup II$, $\forall c \in C$ y $\forall r_f \in R_f$ tal que <c, r_f, ii> $\in A_f \Rightarrow A_f' =$ <c, r_f, ii'> $\cup A_f$.

Tabla 5.35 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-ítem-copia.

AC[ii8]	**Crea-ítem-copia-contenido**(ii, EC_m): ii'
Definición	Se crea un nuevo ítem de información a partir de otro con igual contenido y con un nuevo número de ítem y número de variante 0, fecha de creación y el resto de las propiedades vacías. Se cuelga del concepto del sistema.
Precondición	$ii \in II$, **RT[ii1]** $\forall ii \in II \not\exists ii'$ tal que ii.id = ii'.id.
Efecto	$\rightarrow EC_m'$, ii'.id = ii.numii'.0, ii'.prop = "", ii'.contenido=ii.contenido, ii'.fecha-creación=fecha-actual. $II' = \{ii'\} \cup II$, y $A_f' =$ <$\mathfrak{s}$, g, ii'> $\cup A_f$

Tabla 5.36 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-ítem-copia-contenido.

AC[ii9]	**Crea-ítem-copia-propiedades**(ii, EC_m): ii'
Definición	Se crea un nuevo ítem de información con las mismas propiedades que el anterior menos el identificador (un nuevo número de ítem y variante 0) y fecha de creación. El contenido será vacío.
Precondición	ii ∈ II, **RT[ii1]** ∀ ii ∈ II ∄ ii' tal que ii.id = ii'.id
Efecto	→ EC_m', ii'.id = ii.numii'.0, ii'.prop = ii.prop, ii.contenido="", ii'.fecha-creación=fecha-actual. II' = {ii'} ∪ II, ∀ c ∈ C y ∀ r_f ∈ R_f tal que <c, r_f, ii> ∈ A_f entonces A_f'= <c, r_f, ii'> ∪ A_f.

Tabla 5.37 Acciones Evolutivas para cambiar los Ítemes de Información: Crea-ítem-copia-propiedades.

2.4.2 Cambiando las Restricciones de Autor

Las únicas restricciones que se pueden redefinir son las variables o de autor, RT_a. Las únicas restricciones de este tipo son las relacionadas con las relaciones conceptuales RT_{mr}. Tal y como se describió en el apartado 2.3.2, las relaciones conceptuales se pueden caracterizar asociándoles un subconjunto de las dieciséis restricciones —en realidad quince porque la restricción As1, es la restricción *universal*, que debe ser cumplida por cualquier relación conceptual y también funcional—.

$$\text{Sea } \{Ac1, Ac2, Ac3, At, S1, S2, S3, Ll1, Ll2, Ll3, Tr1, Tr2, Tr3, As1, As2, As3\} = RT_a \subset RT_{mr},$$
$$\forall r_c \in R_c, \exists RT_{ai} \subseteq RT_a, \text{ tal que}$$
$$\forall <c_o, r_c, c_d> \in A_c, r_c \text{ verifica } RT_{ai,} \text{ con } c_o, c_d \in C \text{ y } r_c \in R_c.$$

En cualquier momento, se puede redefinir una relación conceptual, caracterizándola con un nuevo subconjunto de RT_a. El Sistema de Memorización cuenta con dos acciones de evolución para ello: Añade-RT_a y Borra-RT_a.

$\{\text{Añade-}RT_a, \text{Borra-}RT_a\} \subseteq AC_{mrt}$

Dado r_{ci} uno de los elementos de R_c y RT_{ai}, el conjunto restricciones que lo caracteriza, RT_{ai}' será el conjunto de restricciones que lo caracteriza después de aplicar alguna de las acciones de evolución relacionadas con las Restricciones.

AC[rt1]	**Añade-RT_{mr}**(r_{ci}, rt_{mr}, RT_{mri}): RT_{mri}'
Definición	Se añade una nueva restricción relacional a un determinado nombre de relación conceptual.
Precondición	$r_{ci} \in R_c$ y $rt_{mr} \notin RT_{mri}$ RT_{mri}' debe ser una de las combinaciones válidas
Efecto	→ RT_{mri}', RT_{mri}' = {rt_{mr}} ∪ RT_{mri}
Propagación interna	∀ $r_c \in R_c \Rightarrow$ verificar que cumple RT_{mri}

Tabla 5.38 Acciones Evolutivas para cambiar las Restricciones: Añade-RT_a.

AC[rt2]	**Borra-RT$_{mr}$**(r_{ci}, rt_{mr}, RT_{mri}): RT_{mri}'
Definición	Se elimina una nueva restricción relacional a un determinado nombre de relación conceptual.
Precondición	$r_{ci} \in R_c$ y $rt_{mr} \notin RT_{mri}$ RT_{mri}' debe ser una de las combinaciones válidas
Efecto	→ RT_{mri}', RT_{mri}' = RT_{mri} - {rt_{mr}}

Tabla 5.39 Acciones Evolutivas para cambiar las Restricciones: Borra-RT_a.

2.4.2.1 Consistencia de las Restricciones de Autor

En todo momento, el conjunto de restricciones asociado a una relación conceptual debe ser consistente. Tal y como se demuestra en los apéndices B y C, cualquier combinación de restricciones no tiene por qué ser válida. Por tanto, hay que determinar el conjunto de combinaciones válidas y, en consecuencia, consistentes. Antes de añadir o borrar una de las restricciones que caracterizan a una relación conceptual se debe comprobar nuevamente, que el conjunto de restricciones es consistente. Para ello únicamente deberemos comprobar que es una de las combinaciones válidas.

Resumimos en este apartado el procedimiento que se ha seguido para determinar el conjunto de combinaciones válidas. Se ha optado por una demostración matemática y computacional. Este tipo de demostraciones se viene considerando igualmente válido desde 1976 cuando por primera vez Kenneth Appel y Wolfgang Haken utilizaron el ordenador para convertir 'la conjetura de los cuatro colores' —enunciada por Francis Guthrie en 1852— en 'el Teorema de los cuatro colores' [Barrow, 1993, páginas 227-234]. Esquemáticamente esta demostración ha consistido en los siguientes pasos:

1. Generar con un algoritmo todas las combinaciones posibles que se pueden generar con las 16 restricciones relacionales —se pueden hacer hasta 2^{16} caracterizaciones posibles a una relación conceptual—.
2. Rechazar aquellas combinaciones que son inconsistentes —se ha conseguido reducir el anterior número a un total de 290 combinaciones a priori válidas—.
3. Demostrar que las combinaciones que han quedado después de eliminar las no válidas son consistentes.

Para alcanzar el objetivo del paso 2:

1. Se han establecido reglas de inclusión e incompatibilidad entre algunas de las restricciones.
2. Se ha demostrado la verdad de estas reglas.
3. Se ha escrito un algoritmo que nos eliminaba aquellas combinaciones no válidas, esto es, que contenían estas restricciones incluyentes o incompatibles.

Finalmente, para demostrar que las combinaciones que han resultado del anterior proceso son válidas se ha actuado de la siguiente manera:

1. Dado el gran número de combinaciones —un total de 290 combinaciones— que hace inviable en la práctica la demostración una a una, se ha optado por agruparlas en representantes.
2. Se han demostrado las reglas representantes que permiten resumir las combinaciones válidas.

3. Con los representantes se han establecido reglas o axiomas de consistencia.
4. Se han demostrado estas reglas de consistencia.
5. Se ha escrito una algoritmo que comprueba que todas las combinaciones válidas no eliminadas contienen un subconjunto de estas reglas válidas.
6. Por último, se ha comprobado que este conjunto de reglas de consistencia es completo. Para ello, se ha elaborado un algoritmo en Prolog que a partir de estos axiomas genera las mismas 290 combinaciones válidas del paso 2.

2.4.3 Cambiando el Diccionario D

El diccionario mantiene los nombres o etiquetas que pueden asignarse a conceptos DC, a las relaciones conceptuales DR_c y a las relaciones funcionales DR_f. El autor puede ir construyendo sus terminologías con independecia de si se están o no aplicando a una estructura conceptual concreta. Para el mantemiento de estos diccionarios cuenta con operaciones que le permiten añadir y borrar elementos a cada uno de estos conjuntos DC, DR_c y DR_f.

AC_{md} = {Crea-etiqueta-concepto, Borra-etiqueta-concepto, Crea-etiqueta-Rconceptual, Borra-etiqueta-Rconceptual, Crea-etiqueta-Rfuncional, Borra-etiqueta-Rfuncional}

Dado D = (DC, DR_c, DR_f) el diccionario que será manipulado, D = (DC, DR_c, DR_f), será el diccionario una vez que haya sido manipulado.

AC[d1]	**Crea-etiqueta-concepto**(D): dc'
Definición	Se incluye una nueva etiqueta en el diccionario de conceptos.
Precondición	**RT[d1]** $\forall$ dc $\in$ DC', $\nexists$ dc' tal que dc = dc'
Efecto	$\rightarrow$ D', DC' = {dc'} $\cup$ DC.

Tabla 5.40 Acciones Evolutivas para cambiar el Diccionario: Crea-etiqueta-concepto.

AC[d2]	**Borra-etiqueta-concepto**(dc, D): boolean
Definición	Se borra la etiqueta de concepto del diccionario de conceptos
Precondición	dc $\in$ DC y $\nexists$ c $\in$ C, c.nombre = dc **RT[d2]** $\forall$ c $\in$ C, c.nombre $\in$ DC
Efecto	$\rightarrow$ D', DC' = DC – {dc}.

Tabla 5.41 Acciones Evolutivas para cambiar el Diccionario: Borra-etiqueta-concepto.

AC[d3]	**Crea-etiqueta-Rconceptual**(D): dr_c'
Definición	Se incluye una nueva etiqueta en el diccionario de relaciones conceptuales.
Precondición	**RT[d3]** $\forall dr_c \in DR_c'$, $\nexists dr_c'$ tal que $dr_c = dr_c'$
Efecto	$\rightarrow D', DR_c' = \{dr_c'\} \cup DR_c$.

Tabla 5.42 Acciones Evolutivas para cambiar el Diccionario: Crea-etiqueta-Rconceptual.

AC[d4]	**Borra-etiqueta-Rconceptual**(dr_c, D): boolean
Definición	Se borra la etiqueta del diccionario de relaciones conceptuales.
Precondición	$dr_c \in DR_c$ $\nexists r_c \in R_c$, r_c.nombre = dr_c y $dr_c \notin$ {EsUn, ParteDe, TipoDe} **RT[d4]** $\forall dr_c \in DR_c$, $\nexists dr_c'$ tal que $dr_c = dr_c'$ **RT[d7]** $DR_c \neq \emptyset$ y {EsUn, ParteDe, TipoDe} $\in DR_c$
Efecto	$\rightarrow D', DR_c' = DR_c - \{dr_c\}$.

Tabla 5.43 Acciones Evolutivas para cambiar el Diccionario: Borra-etiqueta-Rconceptual.

AC[d5]	**Crea-etiqueta-Rfuncional**(D): dr_f'
Definición	Se incluye una nueva etiqueta en el diccionario de relaciones funcionales.
Precondición	**RT[d5]** $\forall dr_f \in DR_f'$, $\nexists dr_f'$ tal que $dr_f = dr_f'$
Efecto	$\rightarrow D', DR_f' = \{dr_f'\} \cup DR_f$.

Tabla 5.44 Acciones Evolutivas para cambiar el Diccionario: Crea-etiqueta-Rfuncional.

AC[d6]	**Borra-etiqueta-Rfuncional**(dr_f, D): boolean
Definición	Se borra la etiqueta del diccionario de relaciones funcionales.
Precondición	$dr_f \in DR_f$ $\nexists r_f \in R_f$, r_f.nombre = dr_f **RT[d6]** $\forall dr_f \in DR_f$, $\nexists dr_f'$ tal que $dr_f = dr_f'$
Efecto	$\rightarrow D', DR_f' = DR_f - \{dr_f\}$.

Tabla 5.45 Acciones Evolutivas para cambiar el Diccionario: Borra-etiqueta-Rfuncional.

2.5 Propagación Interna del Cambio

A lo largo de la descripción detallada de las acciones de evolución, se ha visto cómo las restricciones definidas por el sistema y por el autor garantizan la consistencia de cada uno de los elementos. Pero, además, la ejecución de algunas de estas operaciones obligará al *Subsistema de Memorización* a propagar el cambio al resto de los elementos con objeto de que todo el Subsistema conserve su integridad. En concreto:

1. Algunas de las acciones asociativas llevadas a cabo con la EC_m tendrá algún tipo de consecuencia sobre el diccionario D.
2. El cambio de las Restricciones de Autor supone realizar actualizaciones en la EC_m
3. La modificación del Diccionario no acarrea ningún tipo de propagación interna porque no se pueden hacer cambios si los términos están en uso en la Estructura Conceptual.

2.5.1 Propagación al Cambiar la Estructura Conceptual EC_M

Mediante las acciones de evolución asociativas AC_{ma} y las relacionadas con los ítemes de información AC_{mii}, el autor va creando su Estructura Conceptual. Algunas de estas operaciones tendrán algún tipo de impacto sobre alguno/s elemento/s del Sistema de Memorización.

Únicamente se ha observado una posible propagación al Diccionario. Siempre que con alguna acción de evolución asociativa se añada un concepto, relación conceptual o relación funcional y, la etiqueta que se asigna a ese objeto no pertenezca a su correspondiente diccionario, el *Subsistema de Memorización* deberá agregar estos nombres. En las secciones anteriores se especifica para cuáles de estas acciones es necesaria esta propagación.

2.5.2 Propagación al Cambiar las Restricciones de Autor

Al cambiar las Restricciones se vuelve a plantear un problema de evolución, que en nuestro caso estará relacionado con las restricciones que debe verificar la estructura conceptual en cada momento. El Subsistema no se tiene que preocupar en absoluto de las restricciones fijas o de sistema porque todas las operaciones que gestionan la estructura conceptual verifican estas restricciones antes y después de su ejecución. Pero las restricciones variables asociadas a una determinada relación conceptual van cambiando a lo largo del tiempo. Esto es, en un instante del pasado, la EC_m era consistente con las Restricciones de Autor definidas en ese momento pero en un tiempo posterior se han redefinido esas restricciones. Por tanto, puede que las relaciones conceptuales que se han ido estableciendo en la EC_m y las composiciones de ítemes que se han realizado sean inconsistentes con el nuevo conjunto de restricciones definido.

Cada vez que se añada o borre una restricción relacional a una determinada etiqueta de relación conceptual se debe aplicar el siguiente algoritmo—tabla 5.46—:

Algoritmo 1: Redefinición de las Restricciones relacionales en el SM
R_{ci} = Conjunto de relaciones conceptuales que no cumplen las Restricciones de Autor $\forall$ <c_o, r_{ci}, c_d> tal que $r_{ci} \in R_{ci}$ Borra-asociación-conceptual(<c_o, r_{ci}, c_d>, EC_m) R_{ci} = Conjunto de relaciones conceptuales que no cumplen las Restricciones de Autor

Tabla 5.46 Algoritmo 1 que redefine las Restricciones Relacionales Conceptuales.

La aplicación de este algoritmo podría suponer borrar por completo o casi por completo la EC_m. El autor deberá decidir si prefiere redefinir la restricción relacional de autor y perder parte de la EC_m o no. Una solución intermedia sería asesorar al autor dándole una relación de las asociaciones que no cumplen con las Restricciones, indicándole por qué no las cumple y dándole la oportunidad de que arregle la EC_m para que sea consistente antes de modificar el conjunto de Restricciones de Autor.

Resumiendo —figura 5.4—, el cambio de la Estructura Conceptual EC_m (1) debe ser consistente con las Restricciones de sistema RT_s (2) y de autor RT_a (2), que se hayan especificado hasta el momento, al tiempo que, podría suponer la actualización de algunos de los diccionarios D (3). Por otro lado, la redefinición de las Restricciones de Autor (4) debe conservar la integridad y podría provocar la reconstrucción de la Estructura Conceptual EC_m (5) con objeto de que ésta verifique el conjunto de restricciones de autor definido. La modificación del Diccionario ha de ser consistente —no se puede eliminar ningún elemento que se esté utilizando en la EC_m— pero no tiene mayores consecuencias.

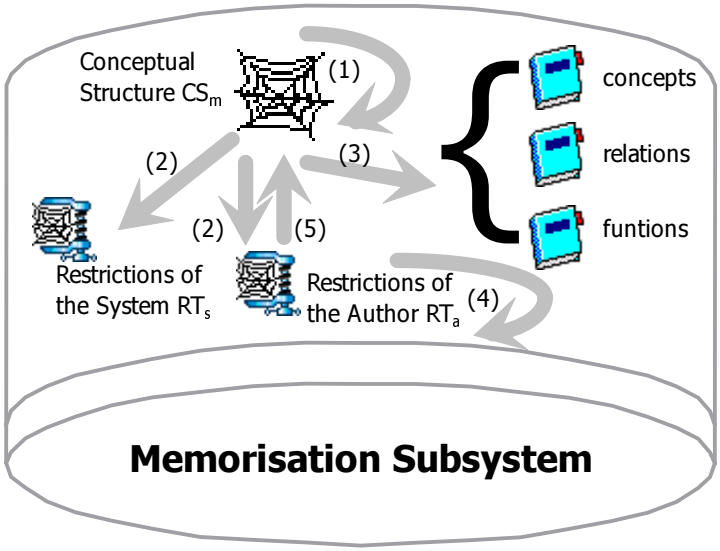


Figura 5.4 Propagación Interna en el Subsistema de Memorización.

2.6 Composición de Ítemes de Información

Como ya se comentó —apartado 5.2.2 del capítulo 4— las Asociaciones Conceptuales hacen posible la composición de ítemes de información de una forma lógica. La composición es una de las acciones evolutivas que se pueden realizar con los ítemes de información. Esto es, a partir de un conjunto de ítemes de información se puede crear uno nuevo agrupándolos.

Para que estos ítemes se puedan componer deben cumplir las siguientes restricciones de sistema:

rt[ii1] Los ítemes que se compongan deben estar etiquetados con algún concepto que sea destinatario de la relación conceptual que admite composición.

- Sea II_i el conjunto de ítemes que queremos componer y r_c, la relación conceptual que admite composición, entonces

$$\forall\ ii \in II_i \subseteq II, \exists\ <c, r_f, ii> \in A_f, \text{ con } c \in C \text{ y } r_f \in R_f, \text{ tal que } |\Gamma^{-}(c, r_c)| \neq 0\ , \text{con } r_c \in R_c.$$

rt[ii2] Los ítemes que se compongan mediante la relación conceptual r_c deben estar etiquetados con conceptos hermanos.

- Sea II_i el conjunto de ítemes que queremos componer y r_c, la relación que admite composición, entonces

$$\forall\ ii \in II_i \subseteq II, \exists\ <c, r_f, ii> \in A_f, \text{ con } c \in C \text{ y } r_f \in R_f, \text{ tal que } \Gamma^{-}(c, r_c) = c_j\ , \text{con } r_c \in R_c$$

Esto es, los ítemes que mantienen una relación funcional con conceptos que son alcanzados de forma directa desde otro concepto con una relación conceptual se pueden componer con otros ítemes que mantengan una relación funcional con conceptos hermanos asociados con la misma relación conceptual.

Por ejemplo, en la estructura conceptual de la derecha de la figura 5.1 se pueden componer los ítemes IJ e IE. El primer ítem está asociado al concepto CM y el segundo al concepto CN. Estos conceptos son hermanos, están asociados al mismo padre CP y mediante las misma relación RB. Por tanto, los ítemes IJ e IE se pueden componer.

El nuevo ítem compuesto se asociará al concepto padre común mediante nuevas relaciones funcionales en el caso de que el autor lo indique. En asusencia de estas nuevas relaciones funcionales, el concepto se asocia al concepto padre con tantas relaciones funcionales como las que mantenían originalmente los ítemes con los conceptos hijos.

Siguiendo con el ejemplo, el ítem compuesto por los ítemes IJ e IE se asociará al concepto CP. Dado que estos ítemes estaban asociados a los conceptos hijos con las funciones RX y RV, el nuevo concepto compuesto se asocia a CP con RX y RV —figura 5.5—. Observe, que la operación de composición no borra los ítemes que se han compuesto únicamente crea un nuevo ítem.

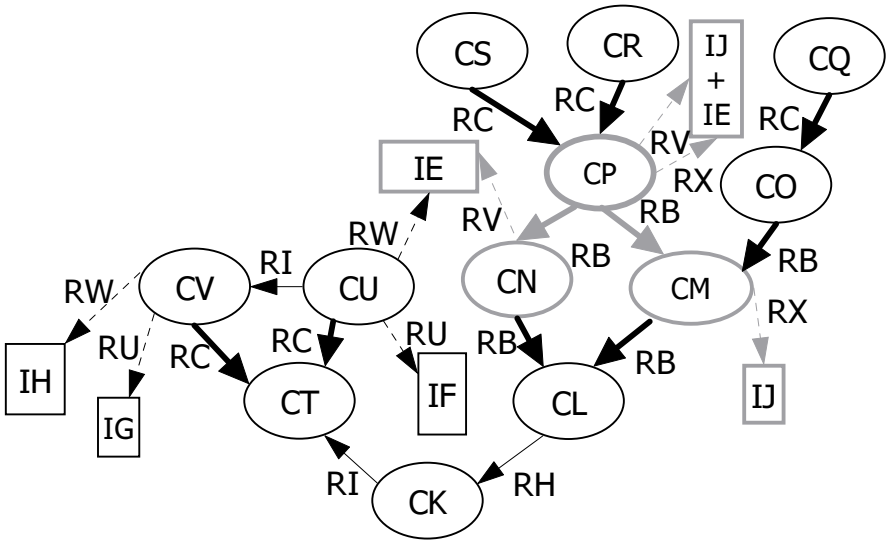


Figura 5.5 Ejemplo de composición de ítemes de información.

La Acción de Evolución que permite hacer estas composiciones se llama Crea-ítem-compuesto y es la décima operación que admiten los ítemes de información.

AC[ii10]	**Crea-ítem-compuesto** (c_p, r_c, DR_{fi}, EC_m): ii'
Definición	Se crea un nuevo ítem agrupando un conjunto de ítemes de información que están asociados a conceptos hermanos, hijos todos de un concepto común mediante un mismo tipo de relación conceptual. Se proporciona el concepto padre común y la relación conceptual que deberá admitir composición. El nuevo ítem se etiqueta con ese concepto padre común. El ítem compuesto será etiquetado con el conjunto de relaciones funcionales que se proporciona. Si este conjunto es vacio, se etiqueta con las relaciones funcionales de los ítemes asociados a los conceptos hijos.
Precondición	ii' $\notin$ II, $c_p \in C$, $r_c \in R_c$, $DR_{fi} \subseteq DR_f$, rt[ii1] $\forall$ ii $\in II_i \subseteq II$, $\exists$ <c, r_f, ii> $\in A_f$, con c $\in$ C y $r_f \in R_f$, tal que $\lvert\Gamma^-(c, r_c)\rvert \neq 0$, con $r_c \in R_c$ rt[ii2] $\forall$ ii $\in II_i \subseteq II$, $\exists$ <c, r_f, ii> $\in A_f$, con c $\in$ C y $r_f \in R_f$, tal que $\Gamma^-(c, r_c) = c_j$, con $r_c \in R_c$
Efecto	$\rightarrow EC_m'$ $\Gamma^{+1}(c_p, r_c) = C_p \subseteq C$, $\forall$ c $\in C_p$, $\Gamma^+(c, r_f) = II_p \subseteq II$, con r_f cualquier relación funcional $\forall$ ii $\in II_p$, ii'.contenido = $\cup$ {ii.contenido}, ii'.conceptos = c_p, II' = {ii'} $\cup$ II Si $DR_{fi} \neq \emptyset$, entonces $\forall$ $r_f \in R_{fi}$, $A_f' = <c_p, r_f, ii'> \cup A_f$ Si $DR_{fi} = \emptyset$, entonces , $\forall$ $c_p \in C_p$, $\forall$ $ii_p \in II_p$, $\Lambda(c_p, ii_p) = R_{fp}$, $\forall$ $r_{fp} \in R_{fp}$, $A_f' = <c_p, r_{fp}, ii'> \cup A_f$.

Tabla 5.47 Acciones Evolutivas Asociativas: Crea-ítem-compuesto.

Por defecto, todas las relaciones conceptuales admiten composición. No obstante, el autor podrá impedir esta posibilidad negando la composición a determinadas relaciones conceptuales. El *Subsistema de Memorización* cuenta con dos acciones de evolución para ello: Componer-R_c y No-Componer-R_c. Si una relación deja de admitir composición entonces todos los ítemes de información que sean composición deberán borrarse. Pero estas acciones pertenecen a la propagación de la acción evolutiva interna tal y como se verá más adelante.

$\{\text{Componer-}R_c, \text{No-Componer-}R_c\} \subseteq AC_{mrt}$

Dado r_{ci} uno de los elementos de R_c y cmp, una variable booleana, la operación Componer-R_c pondrá a *true* esa variable y No-Componer-R_c, le asignará el valor *false*.

AC[rt3]	**Componer-R_c**(r_c, cmp): cmp'
Definición	Las relaciones conceptuales etiquetadas con ese nombre de relación conceptual admiten composición.
Precondición	$r_c \in R_c$
Efecto	→ cmp' cmp = true

Tabla 5.48 Acciones Evolutivas para cambiar las Restricciones: Componer-R_c.

AC[rt4]	**No-Componer-R_c**(r_c, cmp): cmp'
Definición	Las relaciones conceptuales etiquetadas con ese nombre de relación conceptual admiten composición.
Precondición	$r_c \in R_c$
Efecto	→ cmp' cmp = false
Propagación interna	$\forall$ ii $\in$ II compuesto mediante $r_c \Rightarrow$ Borrar-ítem

Tabla 5.49 Acciones Evolutivas para cambiar las Restricciones: No-Componer-R_c

Cada vez que un tipo de relación conceptual pase de admitir composición a no admitirla, se debe propagar internamente el cambio aplicando el siguiente algoritmo:

Algoritmo 2: Redefinición de las Restricciones relacionales en el SM
$\forall$ ii compuesto mediante r_c Borrar-ítem(ii)

Tabla 5.50 Algoritmo 2 que redefine las Restricciones Relacionales.

También en este caso, como con el resto de las acciones evolutivas, el Subsistema puede dar la opción de que el autor se pueda arrepentir de este cambio.

Subcapítulo 5.3

Especificación Formal del Subsistema de Presentación

«La diferencia entre paisaje y paisaje es poca, pero hay una gran diferencia entre los que lo miran.»

Ralph W. Emerson.
Poeta y filósofo estadounidense (1803-1882).

«La abeja y la avispa liban la mismas flores, pero no logran la misma miel.»

Joseph Joubert.
Moralista y crítico francés (1754-1824).

«La unidad es la variedad, y la variedad en la unidad es la ley suprema del universo.»

Isaac Newton.
Matemático, físico, astrónomo y filósofo inglés (1642-1727).

Resumen

En este apartado se realiza la descripción formal del Subsistema de Presentación, uno de los dos Subsistemas del Sistema de Conocimiento. En él se especifican sus componentes básicos, la Estructura Conceptual de Presentación y las Estructuras Virtuales. Para cada uno de estos artefactos se describen sus Restricciones y Acciones de Evolución. Para calcular las Estructuras Virtuales se hará uso de la Teoría de Grafos, en concreto, de los llamados grafos condensados. Finalmente, se explica en qué consiste en este Subsistema la propagación interna de los cambios.

3 Especificación Formal del Subsistema de Presentación 155
3.1 La Estructura Conceptual del Subsistema de Presentación ECp 155
3.2 Las Restricciones del Subsistema de Presentación RTp 156
RTpec Ì RTm 156
3.3 Estructuras Conceptuales Virtuales ECvi 157
3.3.1 Cálculo Automático de Componentes Fuertes y del Grafo Condensado 157
3.3.2 Cálculo Manual de los Componentes Fuertes y del Grafo Condensado 160
3.3.3 Cálculo Semiautomático 160
3.4 Las Acciones Evolutivas del Subsistema de Presentación ACep 162
3.4.1 Cambiando la Estructura Conceptual ECp 162
3.4.2 Selección del Subsistema de Información 165
3.4.3 Cambiando las Estructuras Conceptuales Virtuales ECpv 166
3.5 Propagación Interna del Cambio 167
3.5.1 Propagación al Cambiar la Estructura Conceptual ECp 167
3.5.2 Propagación al Cambiar las Estructuras Virtuales ECvi 168

3 Especificación Formal del Subsistema de Presentación

El *Subsistema de Presentación* realiza un filtrado de la estructura conceptual proporcionada por el Subsistema de Memorización EC_m. Por tanto, el Subsistema de Presentación permitirá reducir los conceptos, las relaciones conceptuales —esto es, el dominio conceptual— y los ítemes de información que pertenezcan al dominio conceptual definido por la estructura conceptual. Dicho de otro modo, el dominio conceptual y el dominio de información que podrá ver el lector —en última instancia, a través del Sistema de Navegación— es un subconjunto del dominio conceptual y del dominio de información que proporciona el Subsistema de Memorización.

Def[sp1] El *Subsistema de Presentación* evolutivo SP se define como: SP = (EC_p, EC^v_i, RT_p, AC_{ep}), donde

- EC_p es una subestructura conceptual de la estructura conceptual proporcionada por el Subsistema de Memorización EC_m **Def[ecm9]**,
- EC^v_i es un conjunto de estructuras conceptuales virtuales que simplifican la presentación de EC_p,
- RT_p es el conjunto de restricciones que debe verificar los componentes de este Subsistema —EC_p y EC^v_i—, y
- AC_{ep} es el conjunto de acciones evolutivas que nos permiten construir y hacer evolucionar la estructura del Subsistema de Presentación que viene determinada por la EC_p, las EC^v_i y RT_p.

3.1 La Estructura Conceptual del Subsistema de Presentación EC_p

Def[ecp1] La estructura conceptual EC_p, es un grafo etiquetado o *red semántica* que representa la estructura conceptual de un subconjunto de un dominio conceptual de un Sistema Hipermedia, en concreto, del dominio conceptual y de información proporcionado por el Subsistema de Memorización. En el caso de que el/la autor/a no aplique ninguna acción de reducción o el conjunto de acciones aplicadas desde el SP sea equivalente a no aplicar ninguna selección, la EC_p elegida será la misma EC_m.

Def[ecp2] EC_p se define formalmente mediante la sextupla: $EC_p = (C_p, II_p, R_{cp}, R_{fp}, A_{cp}, A_{fp})$, en la que C_p es un subconjunto del conjunto C de *conceptos*, II_p es un subconjunto del conjunto II de *ítemes de información*, R_{cp} es un subconjunto del conjunto R_c de *relaciones conceptuales*, R_{fp} es un subconjunto del conjunto R_f de *relaciones funcionales*, A_{cp} es un subconjunto del conjunto de *asociaciones conceptuales* A_c, A_{fp} es un subconjunto del conjunto de *asociaciones funcionales* A_f.

$$EC_p = (C_p, II_p, R_{cp}, R_{fp}, A_{cp}, A_{fp}), \text{ donde}$$
$$C_p \subseteq C,\ C_p \neq \emptyset,\ II_p \subseteq II,$$
$$R_{cp} \subseteq R_c,\ R_{cp} \neq \emptyset,\ R_{fp} \subseteq R_f,$$
$$A_{cp} \subseteq A_c,\ A_{cp} \neq \emptyset,\ A_{fp} \subseteq A_f.$$

3.2 Las Restricciones del Subsistema de Presentación RT_P

Def[ecp3] Para la subestructura conceptual EC_p y las estructuras virtuales EC^v_i del *Subsistma de Presentación*, se definen un conjunto de restricciones RT_p. Un subconjunto de estas restricciones definidas se convertirá en precondiciones y postcondiciones para las acciones AC_{ep} que hacen evolucionar a la EC_p y a las EC^v_i.

Def[ecp4] Las restricciones que se pueden realizar desde el Subsistema de Presentación RT_p pueden pertenecer a dos tipos: las que afectan a la Estructura Conceptual de Presentación EC_p, RT_{pec} y las que están relacionadas con las *Estructuras Conceptuales Virtuales* EC^v_i, RT_{pev}

$$RT_{pec} \cup RT_{pev} = RT_p.$$

El Subsistema de Presentación proporcionará al autor/a un conjunto de operaciones AC_{ep} con las que podrá seleccionar un subdominio conceptual y de información. Al igual que en el Subsistema de Memorización, el subconjunto de conceptos y relaciones conceptuales elegido debería formar una subestructura conceptual, esto es, todos los conceptos elegidos deben estar conectados **Def[ecm9]**.

Asimismo, la subestructura conceptual deberá cumplir el conjunto de restricciones del sistema y del autor —las asociadas a las relaciones conceptuales— relacionadas con los conceptos y relaciones conceptuales seleccionadas. En caso de no cumplir todas estas restricciones, el sistema rechazaría tal reducción por no cumplir las reglas de integridad impuestas desde el Subsistema de Memorización. Esto quiere decir que las restricciones relacionadas con la EC_p son al menos las establecidas en el SM. El *Subsistema de Presentación* adquiere estas restricciones y las convierte en precondición de las acciones encaminadas a reducir de dominio conceptual y de información.

Def[ecp5] Las restricciones afectan a la Estructura Conceptual de Presentación EC_p, RT_{pec} son un subconjunto de las restricciones establecidas en el Subsistema de Memorización.

$$RT_{pec} \subset RT_m$$

3.3 Estructuras Conceptuales Virtuales EC^v_I

En algunas ocasiones la subestructura conceptual seleccionada por el autor contendrá gran número de conceptos y relaciones conceptuales. Quizás algunos de estos conceptos estén íntimamente relacionados y, por esa razón, deberían ser presentados de tal modo que reflejen esa interrelación. En consecuencia, se hace necesario ofrecer una superestructuración del dominio conceptual elegido mostrando un nivel de abstracción mayor en las relaciones. Para ello, el Subsistema de Presentación ofrece mecanismos que nos permiten simplificar la presentación de EC_p definiendo subestructuras conceptuales que se mapean en un único concepto pudiendo definir cuantos niveles admita la estructura conceptual.

En suma, se está incorporando una ayuda de navegación que asiste a los usuarios en la tarea de encontrar la información que están buscando, procurando evitar su desorientación dentro del hiperdocumento. Se asemeja este facilidad a las denominadas *vistas* descritas en el apartado 5.2.2 del capítulo 1.

Esta estructuración puede ser llevada a cabo automáticamente por el sistema, manualmente por el autor o bien de manera semiautomática en el caso de que el sistema asesore de algún modo y finalmente el autor decida la estructuración final que desea. En cualquier caso, el proceso de estructuración tiene como objetivos:

a) determinar subgrafos fuertemente interrelacionados,

b) mapear estas subestructuras en un superconcepto y,

c) construir con estos superconceptos, una superestructura conceptual.

En teoría de grafos se denomina a estas subestructuras *componentes fuertemente conectadas* y, a la superestructura, *grafo condensado* [Christofides, 1975, página 12]. Por tanto, nuestro objetivo será calcular estos componentes fuertemente conectadas y construir el grafo condensado.

3.3.1 Cálculo Automático de Componentes Fuertes y del Grafo Condensado

Un grafo se dice ***fuertemente conectado*** si para cada dos nodos —en nuestro caso, conceptos— existe al menos un camino que va desde uno de ellos al otro.

Un ***componente fuerte*** de un Grafo G es el subgrafo máximo fuertemente conectado de G. Ya que, en un grafo fuertemente conectado, el concepto c_j es alcanzable desde c_i y vicevesa para cualquier c_i y c_j, el componente fuerte que contenga a c_i es única y c_i aparecerá en el conjunto de conceptos de una y sólo un componente fuerte.

Esta afirmación es bastante obvia ya que si c_i aparece en dos o más componentes fuertes, entonces debe existir un camino desde cualquier concepto de un componente fuerte a cualquier concepto del otro componente fuerte. Y por tanto, la unión de ambos componentes sería fuertemente conectado, no siendo máximas las anteriores y contradiciendo la definición. Luego, cada uno de los conceptos sólo puede pertenecer a uno de los componentes fuertes [Christofides, 1975, página 21].

Rt[e1] Cada uno de los conceptos sólo puede pertenecer a uno de los componentes fuertes de la superestructura.

Dicho esto, veamos cómo se pueden ir calculando estos componentes fuertes. Para un concepto c_i, se calcula el llamado ***conjunto de conceptos esenciales***, esto es, aquellos que están, tanto en la matriz de alcanzabilidad $R(c_i)$ —los conceptos que son alcanzables desde c_i— como los que están en la matriz de alcanzabilidad traspuesta $Q(c_i) = R^t(c_i)$ —los conceptos que alcanzan o llegan a c_i—.

$$R(c_i) = \{ c_i \} \cup \Gamma(c_i) \cup \Gamma^2(c_i) \cup ... \cup \Gamma^p(c_i)$$
$$Q(c_i) = \{ c_i \} \cup \Gamma^{-1}(c_i) \cup \Gamma^{-2}(c_i) \cup ... \cup \Gamma^{-p}(c_i)$$
$$R(c_i) \cap Q(c_i).$$

El conjunto $R(c_i) \cap Q(c_i)$ será el primer componente fuerte. Si se quitan al grafo estos conceptos C- $\{R(c_i) \cap Q(c_i)\}$ podemos continuar la búsqueda de componentes fuertes del mismo modo a partir de uno de los conceptos que quedan. Este proceso se repetirá hasta que todos los conceptos de la estructura conceptual EC_p sean asignados a uno de los componentes fuertes. Esto es, el grafo ha sido particionado en componentes fuertes.

A partir estos componentes fuertes podemos construir el grafo condensado. Un grafo condensado o virtual $G^v = (N^v, G^v)$ es un grafo cuyos nodos —en nuestro caso conceptos— representan el conjunto de conceptos de un Componente Fuerte de G, y una relación conceptual entre c_i^v y c_j^v existe si y sólo si existe una relación entre (c_i, c_j) en G para algún $c_i \in c_i^v$ y algún $c_j \in c_j^v$. En un grafo condensado obviamente no encontraremos circuitos —definición en el apartado 1.3—. Estos circuitos están embebidos en esos componentes fuertes.

Para dejar más claro el procedimiento descrito veamos un ejemplo. Dado el grafo de la figura 5.6, vamos a encontrar los componentes fuertes y calcular su grafo condensado.

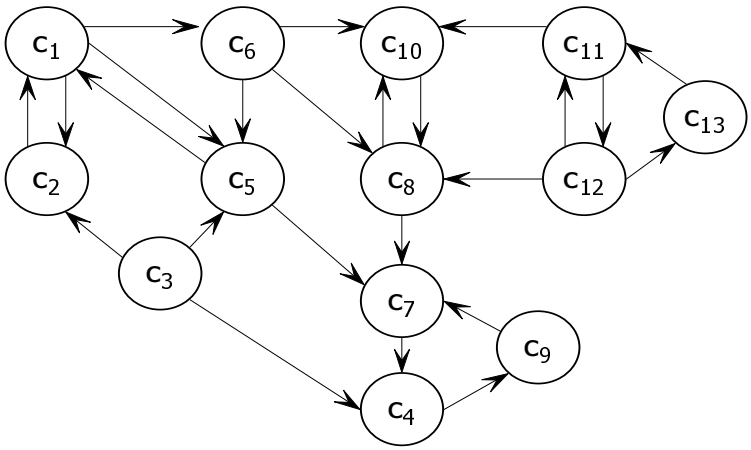


Figura 5.6 Un ejemplo de grafo al que queremos calcular su grafo condensado.

Supongamos que empezamos a calcular el componente fuerte que contiene a c_1, c_1^v, entonces

$$R(c_1) = \{c_1, c_2, c_4, c_5, c_6, c_7, c_8, c_9, c_{10}\}$$
$$Q(c_1) = \{c_1, c_2, c_3, c_5, c_6\}$$
$$R(c_i) \cap Q(c_i) = \{c_1, c_2, c_5, c_6\} = c_1^v.$$

De modo similar, seguimos calculando y llegaremos al grafo condensado de la figura 5.7.

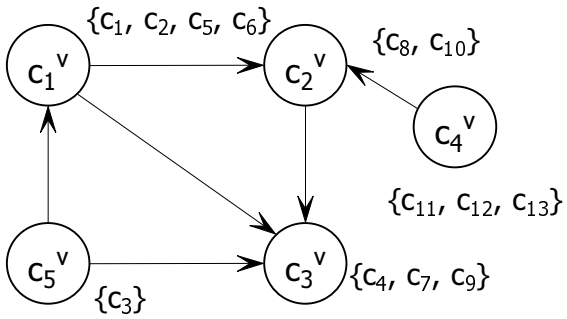


Figura 5.7 Grafo condensado del grafo de la figura 5.6.

Las operaciones anteriormente descritas se pueden realizar de forma más directa mediante las matrices R y Q [Christofides, 1975, página 23]. Si calculamos R ⊗ Q, esto es, la multiplicación de ambas matrices elemento a elemento, entonces para aquellas casillas (c_i, c_j) cuyo valor sea 1, querrá decir que son conceptos mutuamente alcanzables. Y dos conceptos están en el mismo componente fuerte si y sólo si sus correspondientes filas —o columnas— son idénticas. Por tanto, la matriz R ⊗ Q puede transformarse por transposición de filas y columnas en bloques puestos a 1, cada uno de ellos será una componentre fuerte o subestructura del grafo —tabla 5.51—.

	c_1	**c_2**	**c_5**	**c_6**	**c_8**	**c_{10}**	**c_4**	**c_7**	**c_9**	**c_{11}**	**c_{12}**	**c_{13}**	**c_3**
c_1	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0
c_2	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0
c_5	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0
c_6	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0
c_8	0	0	0	0	**1**	**1**	0	0	0	0	0	0	0
c_{10}	0	0	0	0	**1**	**1**	0	0	0	0	0	0	0
c_4	0	0	0	0	0	0	**1**	**1**	**1**	0	0	0	0
c_7	0	0	0	0	0	0	**1**	**1**	**1**	0	0	0	0
c_9	0	0	0	0	0	0	**1**	**1**	**1**	0	0	0	0
c_{11}	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	0
c_{12}	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	0
c_{13}	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	0
c_3	0	0	0	0	0	0	0	0	0	0	0	0	**1**

Tabla 5.51 Cálculo de las componentes fuertes mediante R ⊗ Q.

El cálculo del grafo condensado nos permite calcular el conjunto de conceptos desde los cuales se puede alcanzar cualquier concepto de la estructura conceptual. A este conjunto se le llama en teoría de grafos ***base B***.

En un grafo condensado claramente las bases serán aquellos conceptos compuestos con rango-de-entrada —*indegree*— 0. En el ejemplo anterior son base el conjunto $\{c_5^v, c_4^v\} = \{ c_3, c_{11}, c_{12}, c_{13}\}$. De igual modo, es fácil calcular los conceptos que son alcanzables por todos los demás conceptos o ***contra-base***. En un grafo condensado estas contrabases están formadas por aquellos conceptos compuestos con rango-de-salida —*outdegree*— 0. Volviendo al ejemplo —figura 5.7—, una contra-base es $\{ c_3^v\} = \{ c_4, c_7, c_9\}$.

El cálculo del grafo condensado únicamente nos permite definir un único nivel debido a que no hay límites de alcanzabilidad. En el caso en el que queramos restringir este límite a un solo arco las bases que nos proporcione el grafo condensado se llamarán ***conjuntos dominantes minimales***. De manera general, se puede limitar la alcanzabilidad a un determinado número de arcos, q. En este caso, el grafo condensado $G^v =(N^v, \Gamma^v)$ es un grafo cuyos nodos —en nuestro caso conceptos— representan el conjunto de conceptos de un Componente Fuerte de G. Además, una relación conceptual entre c_i^v y c_j^v existe si y sólo si existe un camino de cardinalidad menor o igual a q, entre (c_i, c_j) en G para algún $c_i \in c_i^v$ y algún $c_j \in c_j^v$. Este grafo condensado, se podría nuevamente condensar generando un nivel de abstracción mayor. Obviamente, llegará un momento en que el grafo es tan simple que no contiene ningún circuito y es imposible reducirlo más.

El cálculo de las Estructuras Virtuales mediante componentes fuertes —o conjuntos dominantes minimales— permiten agrupar aquellos conceptos de grafo que están fuertemente conectados. En este caso, se está suponiendo que las sintaxis o conexiones entre ellos es fiel reflejo de una gran interdependencia semántica entre estos conceptos. Sin duda, esto no tiene por qué ser siempre así. Por esta razón, el desarrollador podrá seleccionar libremente las componenetes fuertes y hacer su propia condensación del grafo.

3.3.2 Cálculo Manual de los Componentes Fuertes y del Grafo Condensado

En el caso de que el autor decida calcular las subestructuras manualmente, deberá seleccionar los conceptos que formarán parte de un componente fuerte teniendo que respetar en todo momento la restricción **RT[e1]** —cada uno de los conceptos sólo puede pertenecer a uno de los componentes fuertes de la superestructura—.

En ambos casos, tanto si decide estructurar automáticamente o manualmente, el autor deberá indicar el nombre del concepto compuesto —del componente fuerte— y de las relaciones conceptuales entre los conceptos compuestos. Estas etiquetas serán aportadas desde el Subsistema de Presentación. Para un mismo subdominio conceptual son posibles distintas estructuraciones y, en consecuencia, distintas presentaciones.

3.3.3 Cálculo Semiautomático

En muchas ocasiones, el autor dudará y no sabrá cómo agrupar y estructurar los conceptos. En estas circunstancias el sistema puede ayudarle a localizar aquellos conceptos que están más próximos al resto. Para ello nuevamente recurrimos a la teoría de grafos que nos permite calcular estos centros [Christofides, 1975, página 79].

Nuevamente, se está suponiendo que aquellos conceptos que están más conectados deben ser los líderes, es decir, los conceptos a partir de los cuales se deben calcular los componentes fuertes.

Localización de Centros

Para cada nodo —concepto— c_i del grafo G =(N, Γ), se calcula el conjunto de conceptos c_j de G que son alcazables desde c_i y la longitud de sus caminos, y el conjunto de nodos c_k que alcanza a c_i y la longitud de sus caminos. Con estos números se construye una matriz bidimensional, una ***matriz de distancias*** D(G). El número máximo para cada fila es la separación de salida máxima $s_s(c_i)$, y el máximo número por columnas es la separación de entrada máxima para cada nodo $s_e(c_i)$. Los posibles centros serán aquellos nodos que están más cerca del resto. Y, por tanto, serían los candidatos a estar situados en diferentes componentes fuertes.

Por ejemplo, dado el grafo de la figura 5.8, su matriz de distancias será el que muestra la tabla 5.52. Los centros o conceptos que están a menos distancia del resto serán los posibles candidatos: c_2, c_3, c_5 y c_6. Si bien, con distinto grado de importancia, en primer lugar estaría c_5, luego c_2 y en igualdad de condiciones c_3 y c_6.

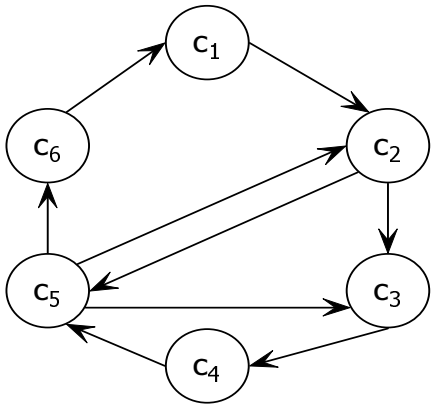


Figura 5.8 Ejemplo de grafo al que se le calculan los centros para ayudar al autor en la creación de estructuras virtuales.

		(2)	(3)		(1)	(3)	
	c_1	**c_2**	**c_3**	**c_4**	**c_5**	**c_6**	**$s_s(ci)$**
c_1	0	1	2	3	2	3	3
c_2	3	0	1	2	1	2	3
c_3	4	3	0	1	2	3	4
c_4	3	2	2	0	1	2	3
c_5	2	1	1	2	0	1	2^v
c_6	1	2	3	4	3	0	4
$s_e(ci)$	4	3^v	3^v	4	3^v	3^v	4

Tabla 5.52 Matriz de distancias D(G) para el grafo de la figura 5.8.

3.4 Las Acciones Evolutivas del Subsistema de Presentación AC_{EP}

El *Subsistema de Presentación*, basándose en uno de los dominios conceptuales y de información que le proporciona el *Subsistema de Memorización*, selecciona una perspectiva concreta del mismo. Por tanto, el Subsistema de Presentación debe ofrecer un repertorio de acciones mediante las cuales el autor/a pueda especificar la Estructura Conceptual de presentación EC_p y las Estructuras Conceptuales Virtuales EC^v_i.

Estas acciones evolutivas deberán respetar las restricciones antes descritas —las definidas en el Subsistema de Memorización para el grafo y las relaciones conceptuales— para garantizar la consistencia del cambio. Además, el Susbsistema deberá propagar dicho cambio con objeto de hacer evolucionar al resto de los componentes del Subsistema y al resto de los sistemas del Modelo SEM-HP —estas propagaciones externas se expondrán al final, una vez que se haya descrito la totalidad de los Sistemas—.

Def[ecp6] Para cada uno de los elementos del *Subsistema de Presentación*, Estructura Conceptual EC_p y Estructuras Conceptuales Virtuales EC^v_i se definen un conjunto de acciones evolutivas AC_{ep}. Estas operaciones deben respetar en todo momento las restricciones definidas que se convierten en precondiciones y postcondiciones para las acciones AC_{ep} que hacen evolucionar a la EC_p, a las EC^v_i.

$$AC_{pec}, AC_{pev} \subseteq AC_{em},$$
$$AC_{pec} \cup AC_{pev} = AC_{em},$$
$$AC_{pec} \cap AC_{pev} = \varnothing.$$

3.4.1 Cambiando la Estructura Conceptual EC_P

Tal y como se ha descrito anteriormente —**Def[ecp1]**, **Def[ecp2]**— desde *el Subsistema de Presentación* se puede seleccionar un subdominio conceptual y de información mediante el uso de acciones de evolución. El usuario/a contará con un conjunto de operaciones para ello, que se pueden catalogar como: las que se dedican a seleccionar un subdominio conceptual y las que se ocupan de elegir un subdominio de información. No obstante, al restringir el dominio conceptual también, de manera indirecta, se estará reduciendo el dominio de información. Éste quedará limitado al conjunto de ítemes que están etiquetados con alguno de los conceptos seleccionados y al conjunto de ítemes asociados a conceptos que están conectados mediante las relaciones conceptuales seleccionadas.

El *Subsistema de Presentación* parte de la Estructura Conceptual proporcionada por el *Subsistema de Memorización*, esto es, en un primer momento la EC_p es exactamente igual a la EC_m. A partir de aquí, el diseñador/a de la presentación se dedicará a realizar acciones de evolución con objeto de ir reduciendo u ocultando parte del dominio conceptual y de información original.

El conjunto de operaciones utilizado es muy parecido —un subconjunto adaptado— a las proporcionadas en el Subsistema de Memorización para la construcción del grafo. Es lógico que sea así porque el objetivo es muy parecido, la única diferencia es que en este caso estamos condicionados por la estructura conceptual suministrada. Esto quiere decir, que no se pueden añadir nuevos conceptos, ítemes y nuevas relaciones conceptuales y funcionales. Lo único que se puede hacer es ocultar parte de la estructura conceptual visible o posteriormente volver a mostrar una porción de la estructura

conceptual que se enmascaró. De nuevo, el conjunto de acciones garantizará la consistencia:

a) impidiendo que se puedan mostrar conceptos desconectados o relaciones conceptuales que no unen dos conceptos o relaciones funcionales que no asocian conceptos con ítemes, y

b) obligando a que el subdominio conceptual cumpla las restricciones de autor relacionadas con las relaciones conceptuales.

AC_{pec} = {Oculta-AC, Oculta-AF, Muestra-AC, Muestra-AF}

Dada la estructura conceptual de presentación que serán manipulada, $EC_p = (C_p, II_p, R_{cp}, R_{fp}, A_{cp}, A_{fp})$, llamaremos, $EC_p' = (C_p', II_p', R_{cp}', R_{fp}', A_{cp}', A_{fp}')$, a la estructura conceptual de presentación una vez que haya sido manipulada. EC_p es la estructura conceptual de presentación que es visible mientras que $\mathbf{EC_p} = (\mathbf{C_p}, \mathbf{II_p}, \mathbf{R_{cp}}, \mathbf{R_{fp}}, \mathbf{A_{cp}}, \mathbf{A_{fp}})$, será la estructura conceptual de presentación que está oculta, de modo que,

$$EC_m = EC_p \cup \mathbf{EC_p}.\ EC_m - \mathbf{EC_p} = EC_p.$$
$$\text{Si } \mathbf{EC_p} = \emptyset \text{ entonces } EC_p = EC_m.$$

AC[pec1]	**Oculta-AC**($<c_o, r_c, c_d>$, $\mathbf{EC_p'}$, EC_p'): boolean
Definición	Se oculta una asociación conceptual y cualquier concepto —junto con las relaciones conceptuales y funcionales unidas al concepto— que se convierta en inaccesible después de ocultar esa asociación —esta operación es muy parecida a **AC[a4]**—. Esto es, que quede desconectado de la EC_p. En el caso de que esta relación conceptual sea la de comienzo, se ocultaría toda una estructura conceptual. Toda esta subestructura pasará a formar parte de $\mathbf{EC_p}$.
Precondición	$c_o, c_d \in C_p$, $<c_o, r_c, c_d> \in A_{pc}$, $<c_o, r_c, c_d> \notin \mathbf{A_{pc}}$ Antes hay que verificar que al quitar esas asociaciones y conceptos se siguen verificando las restricciones de cada relación conceptual. Si alguna restricción no se cumple, entonces no se puede ocultar la asociación conceptual.
Efecto	$\rightarrow EC_p', \mathbf{EC_p'}$ $A_{pc}' = A_{pc} - \{<c_o, r_c', c_d>\}$ $\mathbf{A_{pc}'} = \{<c_o, r_c', c_d>\} \cup \mathbf{A_{pc}}$ $\Upsilon^*(r) = C_k, \forall c \notin C_k, \Gamma^{+1}(c) = C_h \cup II_h = O_h, \ \Gamma^{-1}(c) = C_j,$ con $C_h, C_j, C_k \subseteq C_p$ y $II_h \subseteq II_p$ $\forall o_h \in O_h, \Lambda(c, c_h) = R_h \subseteq R_p, \forall r_h \in R_h$, //conceptual y funcional $A_p' = A_p - <c, r_h, o_h>$, $\mathbf{A_p'} = <c, r_h, o_h> \cup \mathbf{A_p'}$, $II_p' = II_p - o_h$ ó $C_p' = C_p - o_h$, $\mathbf{II_p'} = o_h \cup \mathbf{II_p'}$ ó $\mathbf{C_p'} = o_h \cup \mathbf{C_p'}$ $\forall c_j \in C_j, \Lambda(c_j, c) = R_{cj} \subseteq R_c, \forall r_{cj} \in R_{cj}$, $A_c' = A_c - <c_j, r_{cj}, c>$, $\mathbf{A_p'} = <c_j, r_{cj}, c> \cup \mathbf{A_p'}$ $C_p' = C_p - o_h$, $\mathbf{C_p'} = o_h \cup \mathbf{C_p'}$.
Propagación Interna	Calcula las estructuras virtuales

Tabla 5.53 Acciones Evolutivas para cambiar la Estructura Conceptual de Presentación: Oculta-AC.

Si queremos ocultar la relación que va desde el concepto c al concepto f, se oculta ese arco y su etiqueta y se calcula Nodos-conectados (a) = {g, b, c, d, h}, luego el complementario es {e,f}, se ocultan esos conceptos y las relaciones que llegan o parten de estos nodos. Los ítemes siempre permanecen asociados mediante la relación funcional especial genérica al concepto-sistema. Por tanto, no hay que preocuparse si no están asociados a ningún concepto. El procedimiento es el mismo que el empleado para borrar asociaciones conceptuales en la estructura conceptual del Subsistema de Memorización.

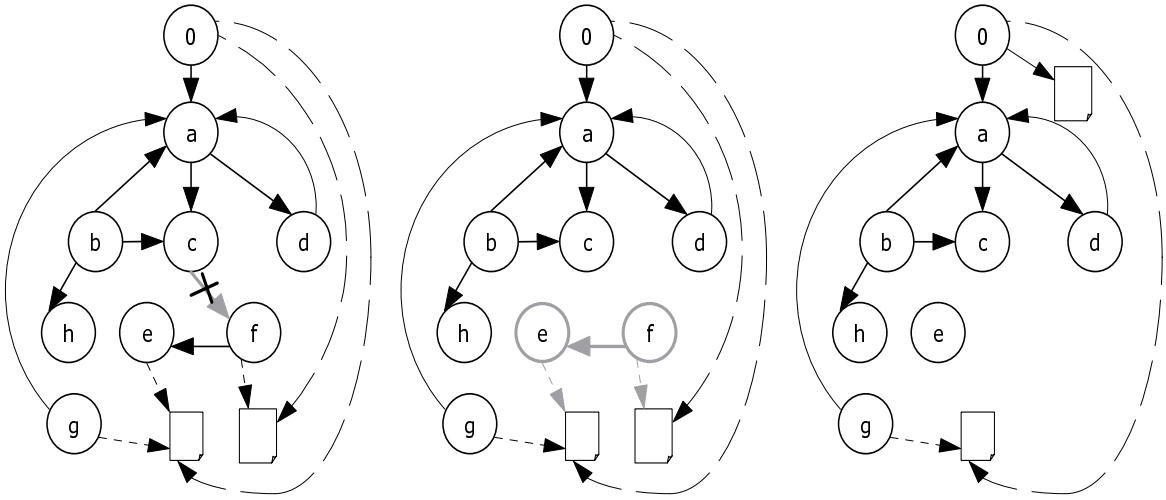


Figura 5.9 Ejemplo de la acción evolutiva: Oculta-AC.

AC[pec2]	**Oculta-AF**(<c_o, r_f, ii>, **EC_p'**, EC_p'): boolean
Definición	Se oculta una asociación funcional entre un concepto y un ítem de información. No hay que preocuparse si para el ítem es la única asociación funcional que mantiene porque, al menos, colgará del *concepto-sistema* mediante la relación funcional especial *genérica* —**Def[ecm9]**—.
Precondición	$c_o \in C_p$, <c_o, r_f, ii>$\in A_{pc}$, <c_o, r_f, ii> $\notin$ **A_{pc}** ii $\in II_p$, ii $\notin$ **II_p**.
Efecto	$\rightarrow EC_p'$, **EC_p'** $II' = II - \{ii\}$, $A_{pf}' = A_{pf} - \{<c_o, r_f', ii>\}$ **II_p'** = {ii} $\cup$ **II_p**, **A_{pc}'** = {<c_o, r_c', c_d>} $\cup$ **A_{pc}**

Tabla 5.54 Acciones Evolutivas para cambiar la Estructura Conceptual de Presentación: Oculta-AF.

AC[pec3]	**Muestra-AC**(<c_o, r_c, c_d>, **EC_p'**, EC_p'): boolean
Definición	Se muestra una relación conceptual y un concepto destino en EC_p' al tiempo que desaparece de **EC_p'**. Debe existir el concepto origen.
Precondición	$c_o \in C_p$, <c_o, r_c, c_d> $\notin A_{pc}$, $c_d \in$ **C_p**, <c_o, r_c, c_d> $\in$ **A_{pc}** Se deben verificar las restricciones variables relacionales que el autor ha definido para esa relación conceptual
Efecto	→ EC_p', **EC_p'** $C_p' = \{c_d'\} \cup C_p$, $A_{pc}' = \{<c_o, r_c', c_d>\} \cup A_{pc}$ **C_p'** = **C_p'**- $\{c_d'\}$, **A_{pc}'** = **A_{pc}** - $\{<c_o, r_c', c_d>\}$ $A_p' = A_p$ - <c, r_h, o_h>, **A_p'** = <c, r_h, o_h> $\cup$ **A_p'**, $II_p' = II_p - o_h$ ó $C_p' = C_p - o_h$, **II_p'** = $o_h \cup$ **II_p'** ó **C_p'** = $o_h \cup$ **C_p'** $A_c' = A_c$ - <c_j, r_{cj}, c>, **A_p'** = <c_j, r_{cj}, c> $\cup$ **A_p'** $C_p' = C_p - o_h$, **C_p'** = $o_h \cup$ **C_p'**.
Propagación Interna	Calcula las estructuras virtuales

Tabla 5.55 Acciones Evolutivas para cambiar la Estructura Conceptual de Presentación: Muestra-AC.

AC[pec4]	**Muestra-AF**(<c_o, r_f, ii >, **EC_p'**, EC_p'): boolean
Definición	Se muestra una de las relaciones funcionales de un concepto de **EC_p** y su ítem asociado. El concepto origen debe existir.
Precondición	$c_o \in C_p$, <c_o, r_f, ii > $\notin A_{pc}$, <c_o, r_f, ii > $\in$ **A_{pc}** ii $\notin$ II, $r_f' \in$ **II**.
Efecto	→ EC_p', **EC_p'** II′ = {ii} $\cup$ II, $A_{pf}' = \{<c_o, r_f, ii>\} \cup A_{pf}$ **II_p'** = **II_p** - {ii}, **A_{pc}'** = **A_{pc}** – $\{<c_o, r_c, c_d>\}$

Tabla 5.56 Acciones Evolutivas para cambiar la Estructura Conceptual de Presentación: Muestra-AF.

3.4.2 Selección del Subsistema de Información

El Subsistema de Presentación contará además con un lenguaje al que se ha denominado **I**, parecido a los empleados en los buscadores web, con el que se podrá seleccionar aquellos ítemes de información que tienen algunos valores para algunas de sus propiedades —identificación, variantes, fecha de creación, autor, aplicación, formato, nivel de especialización, funciones, etc—.

Este lenguaje tendrá tantos predicados como propiedades, a cada uno de ellos se le podrá asignar un valor —o rango de valores— o una variable en cuyo caso se seleccionarán todos. Estos predicados se podrán combinar mediante operadores lógicos.

Para indicar o enumerar más de un término para un predicado se usará el punto y coma. Para separar los rangos un guión, leyéndose entonces como: *seleccione los ítemes que contienen desde este valor a este valor*. Asimismo, se podrá preguntar por familias de valores mediante el uso de comodines que se simboliza con el asterisco. Por último, para indicar el orden de precedencia se utilizarán paréntesis.

Recapitulando, el lenguaje **I** con el que el se podrá seleccionar un subdominio de información estará compuesto por:

- **Predicados:** numii.numvar, lenguaje, aplic, formato, nivel, edicion —autor, idioma de expresión, autor/es, organismo, fecha de creación, última fecha de actualización, calidad, número de visitantes, tipo de visitantes, organismos que lo recomiendan, personas que lo recomiendan—.
- **Operadores lógicos:** and**,** or**,** not**.**
- **Símbolos especiales:** ; - * ( ).

A continuación se exponen algunos ejemplos de uso:

- "Selecciona los ítemes de información cuyo autor sea Parets que estén en español y también todos los ítemes que se han creado en el año 2000":

  Idioma((Autores(Parets)), español) and Fecha-creación(//2000)

- "Selecciona aquellas imágenes en formato *.gif creadas por autores cuyo apellido empieza por Gar":

  Autores((Formato(Lenguaje(imagen)), gif), Gar*)

- "Selecciona los ítemes de información creados entre el 1998 y 2001 en inglés y en español y rechaza los que sean de Soler "

  not Autores(Idioma(fecha-creación(//1998-//2001)), español; inglés), Soler)

- "Selecciona los ítemes de información cuyo autor sea Parets que estén en español y también todos los ítemes que se han creado en el año 2000"

  Idioma((Items, autores(Parets)), español) and fecha-creación(Items, //2000)

- "Selecciona los ítemes de información que han recibido entre 1000 y 2000 visitas y los que sean definiciones o introducciones"

  visitantes(1000-2000) and (función(definición) or función(introducción))

3.4.3 Cambiando las Estructuras Conceptuales Virtuales EC_P^V

Teniendo en cuenta las explicaciones anteriores —apartado 3.3—, podemos definir las estructuras conceptuales virtuales del siguiente modo:

Def[ecp7] EC^v_1 es una estructura conceptual virtual que simplifica la estructura conceptual de presentación EC_p mediante una acción evolutiva de estructuración AC_{pev} —que calcula el grafo condensado— perteneciente a AC_{ep} y que debe verificar la restricción de estructuración **RT[e1]**.

$$AC_{pev}(EC_p) = EC^v_1, \text{ tal que } EC^v_1 \text{ cumple } \mathbf{RT[e1]}$$
$$EC^v_1(C^v, R^v_c, A^v_c), \text{ donde}$$
$$R \otimes Q\ (C_p) = C^v,$$
$$\forall < c_o, r_c, c_d> \in A_c \text{ tal que } c_o \in c^v_o \text{ y } c_d \in c^v_{d,} \text{ entonces } <c^v_o, r^v_c, c^v_d> \in A_{cv}.$$

Def[ecp8] EC^v_i es un conjunto de estructuras conceptuales virtuales, de tal modo que cada EC^v_i simplifica la anterior estructura conceptual EC^v_{i-1} mediante una operación de estructuración AC_{pev}, siempre y cuando se respete la restricción de estructuración **RT[e1]**.

$$AC_{pev}\ (EC^v_{i-1}) = EC^v_i, \text{ tal que } EC^v_i \text{ cumple } \mathbf{RT[e1]}.$$

En consecuencia, el Subsistema de Presentación contará con acciones de evolución que nos permiten calcular el grafo condensado, seleccionar componentes fuertes, asignar nombres a los conceptos o relaciones conceptuales virtuales y localizar centros.

3.5 Propagación Interna del Cambio

El *Subsistema de Presentación* ofrece una serie de operaciones para la especificación de la Estructura Conceptual de presentación, EC_p, y de sus posibles Estructuras Virtuales, EC^v_i. No obstante, la realización de cambios en ambos componentes no supone ningún peligro de integridad, pues, las precondiciones de estas acciones garantizan la consistencia de los mismos. Sin embargo, esto no es suficiente, porque el Subsistema debe salvaguardar la integridad de todos y cada uno de sus partes. En concreto, siempre que se modifique el subdominio conceptual se deberán calcular, de nuevo, sus estructuras virtuales asociadas.

3.5.1 Propagación al Cambiar la Estructura Conceptual EC_P

Si el autor cambia el conjunto de conceptos y relaciones conceptuales que quiere presentar, entonces también deberá cambiar la estructuración para ese nuevo subdominio conceptual. En consecuencia, el sistema recalculará la/s superestructura/s nuevamente en el caso de que se hubiesen hecho automáticamente. Si el autor las definió manualmente, el autor deberá volver a construir estas superestructuras o generarlas automáticamente. Cada vez que se redefina el subdominio conceptual de EC_p se debe aplicar el siguiente algoritmo:

Algoritmo 1: Redefinición de la Subestructura Conceptual de EC_p

Si EC^v_i entonces
  Si EC^v_i automáticamente
  Recalcular EC^v_i
  En caso contrario
  Solicitar Recalcular EC^v_i

Tabla 5.57 Algoritmo 1: Redefinición de la Subestructura Conceptual de EC_p.

La aplicación de este algoritmo podría suponer borrar por completo EC^v_i. El autor deberá decidir si prefiere redefinir EC_p y perder las superestructuras o no.

3.5.2 Propagación al Cambiar las Estructuras Virtuales EC^v_i

El cambio de una de las posibles Estructuras Virtuales afectará a las siguientes que se han desarrollado a partir de ésta. En consecuencia, si se modifica la EC^v_i habrá que recalcular las siguientes, desde la EC^v_{i+1} hasta la EC^v_n.

Se aplicaría el mismo algoritmo que antes pero únicamente se generan aquellas Estructuras Virtuales que se ven afectadas por el cambio.

Resumiendo —figura 5.10—, los cambios que se realicen bien en la Estructura Conceptual (1), EC_p, bien en las superestructuras virtuales, EC^v_i, deben cumplir todas y cada una de las restricciones asociadas (2). La modificación de la EC_p implica una actualización de todas las estructuras virtuales (3). Sin embargo, el cambio de alguna de ellas, EC^v_i, sólo afecta a las siguientes (4).

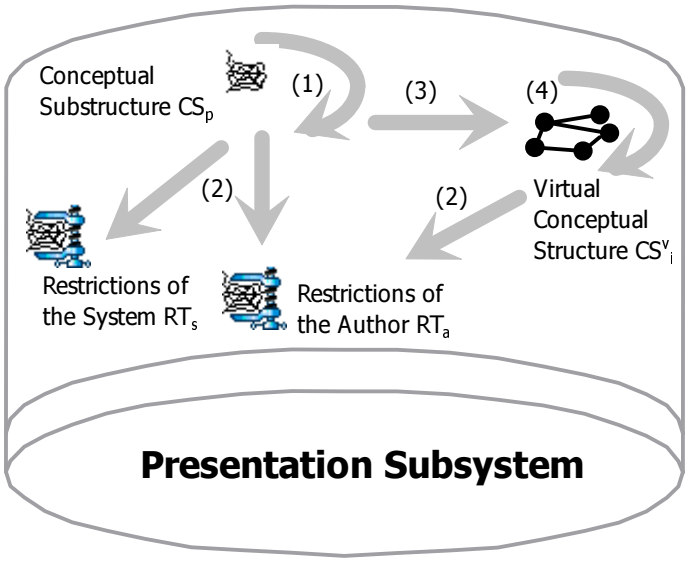


Figura 5.10 Propagación Interna del Cambio en el Subsistema de Presentación.

Subcapítulo 5.4

Especificación Formal del Sistema de Navegación

«No se viaja por viajar, sino por haber viajado.»

Alphonse Karr.
Escritor satírico francés (1808-1890).

«No basta dar pasos que un día puedan conducir a la meta, sino que cada paso ha de ser una meta, sin dejar de ser un paso..»

Johann P. Eckermann.
Filósofo y místico alemán (1642-1727).

«No abandones tu embarcación en el mar de la suerte; sigue remando, pero rema con desenvoltura, y reflexiona una vez más..»

August von Platen.
Poeta y dramaturgo alemán (1796-1835).

Resumen

Este apartado incluye la especificación formal del Sistema de Navegación. De igual modo que antes, se describen las componentes de su ESTRUCTURA: la Estructura Conceptual de navegación, las Restricciones de Navegación, la Red de Petri y los Árboles de Navegación. Para luego, explicar sus Acciones de Evolución, Meta-Restricciones y las propagación del cambio que desencadena la aplicación de cada una de ellas.

Se dedica especial atención al lenguaje **N** basado en lógica temporal con el que se describen las Restricciones de Orden, al proceso de transformación de éstas en una red de Petri y al cálculo del árbol de navegación a partir de esta última.

Con el fin de no perder continuidad en la exposición, se ha decido dejar la explicación de los mecanismos adicionales —red de Petri y árbol de navegabilidad— para el final del subcapítulo.

4 Especificación Formal del Sistema de Navegación 171
4.1 Estructura Conceptual del Sistema de Navegación ECn 172
4.2 Las Restricciones del Sistema de Navegación RTn 172
4.2.1 Lógica Temporal para las Restricciones de Orden RTo 173
4.3 Definición del Lenguaje N 178
4.3.1 Sintaxis del Lenguaje N 178
4.3.2 Metalógica para el Lenguaje N 181
4.3.3 Semántica del Lenguaje N 183
4.4 Las Acciones Evolutivas del Sistema de Navegación ACen 187
4.4.1 Cambiando las Restricciones de Navegabilidad RTnb 188
4.4.2 Cambiando las Restricciones de Orden RTo 189
4.4.3 Propagación de los Cambios en el Sistema de Navegación 195
4.5 La Red de Petri RP 197
4.5.1 Definición Formal de una Red de Petri, RP 199
4.5.2 Transformación de Cláusulas en lenguaje N a RP 201
4.6 Árboles de Alcanzabilidad o de Navegación 209
4.6.1 Definiciones Previas 209
4.6.2 Aplicaciones del Árbol de Alcanzabilidad o Navegación 211

4 Especificación Formal del Sistema de Navegación

El *Sistema de Navegación* nos permite ordenar de algún modo la estructura conceptual y, en consecuencia, el Subsistema de Información asociado a la misma —el dominio de información—, ambos proporcionados por el *Subsistema de Presentación*. Por tanto, la navegación se puede ver como la ejecución de una determinada presentación. Esta ejecución debe respetar, en primer lugar, las restricciones impuestas por el Sistema de Conocimiento y, en segundo lugar, aquellas otras restricciones que provengan desde el propio Sistema de Navegación. Todas estas restricciones determinarán el conjunto de recorridos o navegaciones posibles para una determinada presentación.

Con este Sistema interactúan tanto el lector como el autor del Sistema Hipermedia. Sin embargo, el primero de ellos puede comunicarse con sus dos niveles de abstracción —apartado 2 del capítulo 4— mientras que el segundo únicamente se dedica a ejecutar las navegaciones que para él diseña el desarrollador.

Antes de continuar con la especificación formal es necesario realizar una pequeña puntualización con respecto al tipo de navegación que admiten las asociaciones conceptuales. Con independencia del sentido del arco una asociación conceptual se puede navegar en ambos sentidos. Esto es, desde el concepto origen al destino o viceversa. Dicho de otro modo, la navegabilidad viene determinada por la conectividad y no por la alcanzabilidad. No obstante, el autor podrá si así lo desea restringir esta posibilidad y obligar al lector a navegar únicamente en el sentido que determina la asociación conceptual. Estos conceptos serán tratados con detenimiento en el apartado dedicado a las Restricciones de Navegación. Dicho esto,

Def[sn1] El sistema de navegación SN se define como: SN = (EC_n, RT_n, RP, AN, AC_{en}), donde

- EC_n es la estructura conceptual EC_p **Def[ecp1]** a la que se han aplicado las posibles restricciones de navegabilidad RT_{nb} pertenecientes al conjunto de restricciones que se establecen desde el sistema de navegación RT_n,
- RT_n son las restricciones de navegación que están formadas por unión de las restricciones de navegabilidad RT_{nb} y de orden RT_o,
- RP es la red de Petri que establece una semántica operacional a las Restricciones de Orden RT_o pertenecientes también a RT_n,
- AN es el árbol de alcanzabilidad —navegación— que se genera a partir de la red de Petri y
- AC_{en} es el conjunto de acciones evolutivas que se pueden realizar para cambiar este sistema.

4.1 Estructura Conceptual del Sistema de Navegación EC_N

Def[ecn1] La estructura conceptual del sistema de navegación EC_n, es un grafo etiquetado o *red semántica* que representa el dominio conceptual proporcionado por el sistema de presentación EC_p al que se le han aplicado las restricciones de navegabilidad RT_{nb} que determine el sistema de navegación. RT_{nb} especifica en qué sentido es navegable una relación conceptual y, en consecuencia, su sistema de información.

$$RT_{nb}(EC_p) \rightarrow EC_n, \text{ donde}$$
$$EC_p = (EC_m, RT_r)$$
$$RT_{nb} = \{rt_{nb} / rt_{nb} \text{ es una restricción de navegabilidad}\} \text{ donde } RT_{nb} \subset RT_n.$$

4.2 Las Restricciones del Sistema de Navegación RT_N

Def[ecn2] Las restricciones que el autor puede especificar desde el sistema de navegación son de dos tipos: de navegabilidad RT_{nb} y de orden RT_o.

$$RT_n = RT_{nb} \cup RT_o, \text{ donde}$$
$$RT_{nb} = \{rt_{nb} / rt_{nb} \text{ es una restricción de navegabilidad}\} \text{ y}$$
$$RT_o = \{rt_o / rt_o \text{ es una restricción de orden}\}.$$

Def[ecn3] Las *Restricciones de Navegabilidad*, RT_{nb}, determinan si la navegabilidad viene determinada por la alcanzabilidad o por la conectividad. En el primer caso, la asociación es navegable en el sentido del arco —desde el concepto A se puede ir al concepto B si existe una relación conceptual que parta del concepto A y llegue al concepto B—. En el segundo caso, la asociación es navegable en ambos sentidos —si existe una relación conceptual entre los conceptos A y B, se puede ir desde A a B o desde B a A—. Por defecto, el conjunto de restricciones de navegabilidad es vacío, en cuyo caso, la estructura conceptual que se muestra al lector es la proporcionada por el sistema de presentación.

$$\text{Si } RT_{nb} = \phi \Rightarrow EC_n = EC_p$$

Def[ecp4] Por defecto, la navegabilidad de todas las asociaciones conceptuales definidas por el autor viene determinada por la conectividad. Una asociación es navegable en ambos sentidos, desde el concepto origen al destino y viceversa. Toda asociación conceptual tiene inversa —en la mayoría de los casos, el nombre de la relación conceptual será la misma pero expresada en pasiva— y el sistema las notará, por defecto, con el mismo nombre y el sufijo *inv*.

Def[ecp5] La navegabilidad de todas las asociaciones de dependencia {*ParteDe*, *TipoDe*, *EsUn*} viene determinada por la conectividad, esto es, son navegables en ambos sentidos.

Def[ecp6] Las *Restricciones de Orden* RT_o establecen si desde un ítem se puede ir hacia otro ítem, haciendo depender esta navegación de las relaciones conceptuales que el lector haya seguido e ítemes que el lector haya visitado anteriormente. Por supuesto, estos recorridos están circunscritos a los caminos que nos ofrece la EC_n dependiendo estos de la alcanzabilidad o conectividad de cada una de las asociaciones conceptuales de la misma.

Def[ecp7] Las *Restricciones de Orden*, RT_o, serán expresadas mediante reglas en lógica temporal proposicional [Rodríguez-Fórtiz, 2000a]. Estas reglas temporales serán generadas en principio por el sistema –tal y como se describe en el apartado 4.2.1— a partir de la estructura conceptual de navegación EC_n y luego limitadas por el autor cuando indique sus propias Restricciones de Orden.

4.2.1 Lógica Temporal para las Restricciones de Orden RT_o

Tal y como se ha comentado anteriormente, las Restricciones de Orden serán especificadas mediante reglas en lógica temporal proposicional. Estas reglas deben respetar la estructura conceptual de navegación. Por tanto, en principio, será el propio Sistema de Navegación el que proporcione estas Restricciones de Orden RT_o.

Para facilitar la descripción del proceso de construcción de estas Restricciones de Orden a partir de la estructura conceptual de navegación EC_n así como la explicación de la sintaxis y, la interpretación o semántica de las mismas, nos apoyaremos en un ejemplo. Posteriormente, se hará una especificación más formal del lenguaje que usaremos para escribir dichas restricciones.

4.2.1.1 Generando Restricciones de Orden

Tenemos una EC_p que nos ofrece el sistema de presentación —la EC que muestra la figura 5.11 de la derecha—. Imaginemos que el autor aplica restricciones de navegabilidad RT_{nb} y decide que r2, r5, r6 y r7 sólo son navegables en el sentido de la flecha. Por tanto, aplicando estas restricciones de navegabilidad se obtiene el grafo —la EC_n— de la figura de la izquierda. Para aquellas relaciones para las que es posible la navegación en ambos sentidos, la otra relación se llamará igual a la primera pero con sufijo *inv* —$r1_{inv}$, $r3_{inv}$, $r4_{inv}$ y $r8_{inv}$, estas relaciones se dibujan con un tipo distinto de flecha—.

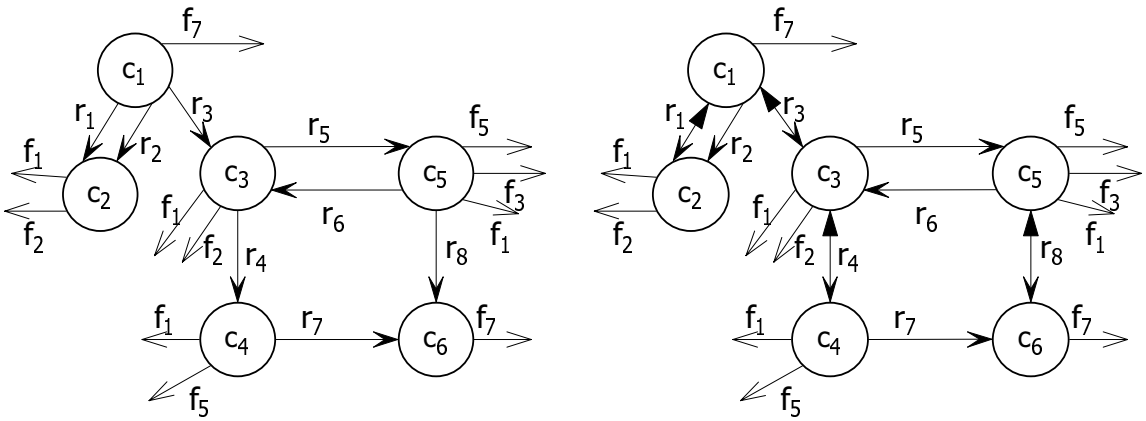


Figura 5.11 Un ejemplo de Estructura Conceptual sobre la que quieren imponer Restricciones de Navegabilidad.

Si el autor no impone ninguna Restricción de Orden adicional —más adelante, en el apartado 4.2.1.2, se describe cómo añadir más limitaciones—, la estructura conceptual de navegación nos está indicando que:

> *Si el lector se encuentra en un determinado ítem de un determinado concepto, podrá navegar hacia otro ítem, siempre y cuando, ese ítem esté asociado mediante una relación funcional con un concepto que, a su vez, está asociado mediante una relación conceptual, al anterior concepto –esto es, si es sucesor de éste—.*

Por ejemplo, si el lector se encuentra en el ítem que está asociado al concepto c3 mediante la relación funcional f2 —el ítem se nota como c3.f2, esto es, con el concepto al que está asociado c3 y la relación funcional que los une f2—, puede navegar:

- a través de r5 a tres posibles ítemes mediante f5, f3 ó f1, ó bien,
- mediante r4 a dos ítemes siguiendo f1 ó f5.

Dicho de otro modo, se puede ir a cualquier ítem de un concepto si previamente —inmediatamente antes— se ha estado en alguno de los ítemes del concepto que precede a ese concepto.

Restricción de orden implícita				**Restricción de orden explícita**		
Fija	**Modificable**	**Fija**		**Fija**	**Modificable**	**Fija**
c1.all ←	previo(c2.allor)	and previo(r1inv)		c1.f7 ←	previo(c2.f1) or previo(c2.f2)	and previo(r1inv)
c1.all ←	previo(c3.allor)	and previo(r3inv)		c1.f7 ←	previo(c3.f1) or previo(c3.f2)	and previo(r3inv)
c2.all ←	previo(c1.allor)	and previo(r1)		c2.f1 ← c2.f2 ←	previo(c1.f7) previo(c1.f7)	and previo(r1) and previo(r1)
c2.all ←	previo(c1.allor)	and previo(r2)		c2.f1 ← c2.f2 ←	previo(c1.f7) previo(c1.f7)	and previo(r2) and previo(r2)
c3.all ←	previo(c1.allor)	and previo(r3)		c3.f1 ← c3.f2 ←	previo(c1.f7) previo(c1.f7)	and previo(r3) and previo(r3)
c3.all ←	previo(c4.allor)	and previo(r4inv)		c3.f1 ← c3.f2 ←	previo(c4.f1) or previo(c4.f5) previo(c4.f1) or previo(c4.f5)	and previo(r4inv) and previo(r4inv)
c3.all ←	previo(c5.allor)	and previo(r6)		c3.f1 ← c3.f2 ←	previo(c5.f1) or previo(c5.f3) or previo(c5.f5) previo(c5.f1) or previo(c5.f3) or previo(c5.f5)	and previo(r6) and previo(r6)
c4.all ←	previo(c3.allor)	and previo(r4)		c4.f1 ← c4.f5 ←	previo(c3.f1) or previo(c3.f2) previo(c3.f1) or previo(c3.f2)	and previo(r4) and previo(r4)
c5.all ←	previo(c3.allor)	and previo(r5)		c5.f1 ← c5.f3 ← c5.f5 ←	previo(c3.f1) or previo(c3.f2) previo(c3.f1) or previo(c3.f2) previo(c3.f1) or previo(c3.f2)	and previo(r5) and previo(r5) and previo(r5)
c5.all ←	previo(c6.allor)	and previo(r8inv)		c5.f1 ← c5.f3 ← c5.f5 ←	previo(c6.f7) previo(c6.f7) previo(c6.f7)	and previo(r8inv) and previo(r8inv) and previo(r8inv)
c6.all ←	previo(c5.allor)	and previo(r8)		c6.f7 ←	previo(c5.f1) or previo(c5.f3) or previo(c5f.5)	and previo(r8)
c6.all ←	previo(c4.allor)	and previo(r7)		c6.f7 ←	previo(c4.f1) or previo(c4.f5)	and previo(r7)

Tabla 5.58 Restricciones de Orden que automáticamente genera el Sistema de Navegación a partir del grafo de la figura 5.11.

En la tabla 5.58 se detallan las Restricciones de Orden que se derivan de la EC_n de la figura 5.11. Las tres primeras columnas de la tabla —vistas de izquierda a derecha— las describen de forma resumida mediante las variables:

- *all* cuyo significado es que se puede seguir cualquier relación funcional que parte del concepto –todos los ítemes asociados al concepto—, y
- *allor* que quiere decir que se ha seguido alguna relación funcional del concepto y, por tanto, visitado alguno de sus ítemes.

En las tres últimas columnas la tabla se especifican todas y cada una de las reglas de Restricción de Orden.

A partir de la tabla con las reglas temporales o Restricciones de Orden que genera el ejemplo, se puede observar que el número de estas restricciones para cada concepto es igual al número de relaciones funcionales que parten del concepto multiplicado por el número de relaciones conceptuales que recibe. Se puede, por tanto, saber de antemano el número de Restricciones de Orden que genera una determinada estructura conceptual de navegación. En nuestro ejemplo, este número es igual a 22 tal y como se describe en la tabla 5.59.

Concepto	Nº R_f	Nº R_c	Nº RT_o
c1	1	2	2
c2	2	2	4
c3	2	3	6
c4	2	1	2
c5	3	2	6
c6	1	2	2
Nº de restricciones de orden			22

Tabla 5.59 Número de Restricciones de Orden que se generan a partir de la Estructura Conceptual de la figura 5.11.

4.2.1.2 Completando las Restricciones de Orden

Las anteriores Restricciones de Orden son las que generaría el Sistema de Navegación de forma directa basándose en la estructura conceptual de navegación EC_n. Pero el autor puede, si así lo desea, limitar aún más la navegación al lector. La estructura conceptual de navegación únicamente contiene información de qué ítemes de información pueden ser visitados una vez que el lector está posicionado en alguno de los ítemes que son predecesores de estos –de modo inmediato o en un solo paso—. Sin embargo, el autor puede llegar a condicionar de tres formas distintas la inspección de un ítem, veamos cuáles son:

- Supongamos que la estructura conceptual de navegación EC_n nos permite alcanzar un ítem de información siguiendo más de una relación conceptual. Este caso se presentará, siempre y cuando, dos conceptos estén unidos por más de una relación conceptual.

Por ejemplo, desde el ítem asociado a c1, c1.f7, se pueden alcanzar los dos ítemes del concepto c2, c2.f1 y c2.f2, a través de la relación conceptual r1 ó r2. Sin embargo, el autor decide mostrar sólo parte de los ítemes cuando se llega a un concepto por una determinada relación conceptual. Supongamos que, el autor determina que si se pulsa r1 únicamente se debe mostrar c2.f1 y, que si sigue r2, se mostrará sólo el ítem c2.f2. Por tanto —tabla 5.60 y figura 5.12—, esto supondría borrar dos Restricciones de Orden: c2.f2 ← **previo**(c1.f7) **and previo**(r1) y c2.f1 ← **previo**(c1.f7) **and previo**(r2).

Restricción1	c2.f1 ← **previo**(c1.f7) **and previo**(r1)
Restricción2	c2.f2 ← **previo**(c1.f7) **and previo**(r1)
Restricción3	c2.f1 ← **previo**(c1.f7) **and previo**(r2)
Restricción4	c2.f2 ← **previo**(c1.f7) **and previo**(r2)

Tabla 5.60 Imponiendo más Restricciones de Orden: Borrar reglas temporales.

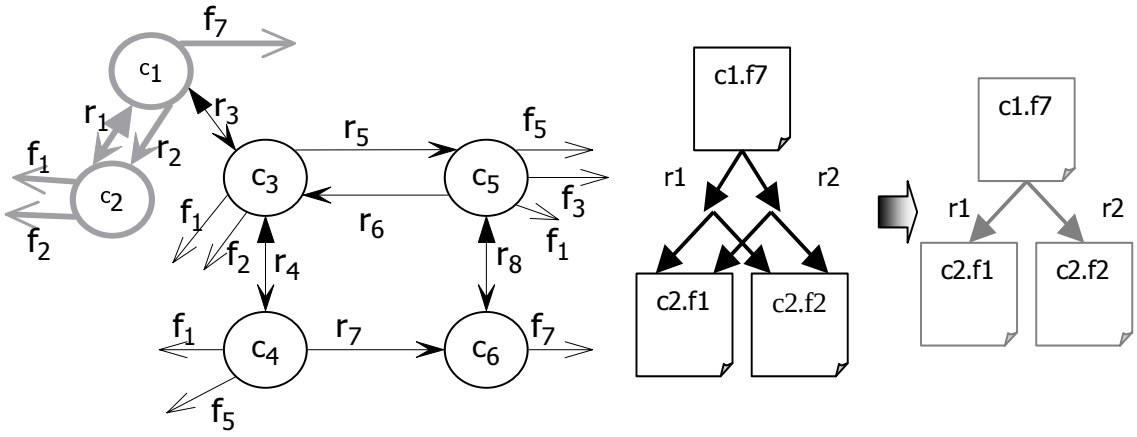


Figura 5.12 Recorridos posibles sobre la Estructura Conceptual antes y después de restringir la navegación como se indica en la tabla 5.60.

- El autor decide que el lector debe visitar más de un ítem del concepto inmediatamente anterior para ver tal ítem. Esto supone cambiar algunos de los operadores **or** del cuerpo de la Restricción de Orden por **and**.

Por ejemplo —tabla 5.61—, para poder mostrar el ítem c3.f1, el lector debe haber visitado previamente los ítemes c4.f1 y c4.f5.

Original	c3.f1 ← (**previo**(c4.f1) **or previo**(c4.f5)) **and previo**(r4inv)
Modificación	c3.f1 ← (**previo**(c4.f1) **and previo**(c4.f5)) **and previo**(r4inv)
Original	c3.f2 ← (**previo**(c5.f1) **or previo**(c5.f3) **or previo**(c5.f5)) **and previo**(r6)
Modificación	c3.f2 ← (**previo**(c5.f1) **and previo**(c5.f3) **or previo**(c5.f5)) **and previo**(r6)

Tabla 5.61 Imponiendo más Restricciones de Orden: Cambiar **or** por **and**.

- El autor impone más restricciones además de las que se deducen de la estructura conceptual de navegación. Es decir, el autor quiere que el lector visite antes algunos otros ítemes para mostrar ese ítem, ítemes que no están asociados al concepto inmediatamente anterior del concepto al que está asociado el ítem de la cabeza.

  Esto supondría añadir restricciones donde aparezcan los operadores temporales **antes** y **desde** —en el siguiente apartado se explicará la sintaxis y semántica de los mismos—. Estos operadores únicamente se pueden aplicar a ítemes que estén asociados a conceptos que son predecesores en más de un paso del concepto asociado al ítem que aparece en la cabeza.

  El sistema debe garantizar que los operadores **antes** y **desde** se aplican correctamente y que el conjunto de estas Restricciones de Orden no tiene ningún tipo de inconsistencia tal y como se describe en el apartado 4.4.2.

Volviendo al ejemplo —tabla 5.61—, el autor podría considerar que es imprescindible que el lector haya visitado anteriormente el ítem c5.f1 y el c4.f1 para poderle mostrar el ítem c2.f1, por tanto, a la Restricción de Orden anterior, c2.f1← **previo**(c1.f7) **and previo**(r1) añade (**antes**(c5.f1) **and antes**(c4.f1)).

Original	c2.f1 ← **previo**(c1.f7) **and previo**(r1)
Modificación	c2.f1 ←(**previo**(c1.f7)) **and** (**antes**(c5.f1) **and antes**(c4.f1)) **and previo**(r1)

Tabla 5.62 Imponiendo más Restricciones de Orden: Añadir operadores **antes** y **previo**.

Por tanto, tal y como se ha descrito, las Restricciones de Orden se pueden:

1. borrar —siempre y cuando se garantice la consistencia— o
2. redefinir pero únicamente en dos sentidos:

   a) cambiando los operadores lógicos que unen los operadores temporales **previo**, o bien,

   b) añadiendo más condiciones a la fórmula temporal mediante operadores temporales **antes** y **desde** aplicados a ítemes.

Estos cambios forman parte de las acciones de evolución de navegación y serán descritos con mayor detalle en el apartado dedicado a tal efecto.

El ejemplo anterior describe de forma intuitiva cómo se generan y completan restricciones a partir de la estructura conceptual de navegación EC_n y cuál es su significado. En el siguiente apartado pasamos a especificar más formalmente la sintaxis y la semántica de las mismas.

4.3 Definición del Lenguaje N

Todas las Restricciones de Orden RT_o serán expresadas mediante reglas muy simples, en realidad, un subconjunto reducido de las posibles cláusulas que se pueden llegar a expresar con lógica temporal proposicional. Para ello, definimos un lenguaje que denominamos de Navegación o simplemente **N**. El lenguaje **N** es una adaptación y simplificación del lenguaje L basado también en lógica temporal propuesto por [Rodríguez-Fórtiz, 2000a, página 99] para la formalización de software evolutivo.

A continuación se describe, en primer lugar, la sintaxis de lenguaje. Posteriormente, dadas las características específicas del Sistema de Navegación, se pormenoriza el sentido de esta lógica, es decir, la metalógica del lenguaje para luego hacer una definición detallada de su semántica.

4.3.1 Sintaxis del Lenguaje N

Como puede observarse, en el ejemplo anterior, todas estas Restricciones de Orden RT_o siguen el siguiente formato o patrón:

concepto.relación-funcional ← *fórmula-temporal* **and previo(relación-conceptual)**

De forma general, esta cláusula modela la relación entre dos acciones, de forma, que la acción —o el conjunto de las acciones del cuerpo— es una precondición de la acción de la cabeza que se modela con la regla:

$$p(x) \leftarrow \text{previo } q(y)$$

En nuestro caso, las acciones del cuerpo estarán relacionadas con haber visitado o no ciertos ítemes y haber seguido cierta relación conceptual. Estas acciones se convierten en una condición fundamental para poder llegar a visitar el ítem que aparece en la cabeza. Tal y como se ha visto, cada una de las Restricciones de Orden se interpreta como: "*se puede visitar el ítem de la cabeza si se cumple la condición impuesta en el cuerpo de la regla temporal y, se ha disparado o seguido la relación conceptual indicada en el cuerpo de la Restricción de Orden* ".

Por tanto, en este lenguaje únicamente se utilizarán dos predicados: *visitar*(x) y *seguir* (x). Estos predicados siempre estarán instanciados. El predicado *visitar* únicamente se puede aplicar a ítemes de información y el predicado *seguir* únicamente se podrá formular sobre las relaciones conceptuales. En el ejemplo anterior estos predicados están implícitos en la Restricción de Orden.

Por ejemplo, la regla anterior:

c6.f7 ← (**previo**(c4.f1) **or previo**(c4.f5)) **and previo**(r7)

puede reescribirse de manera explícita de la siguiente forma:

visitar(c6.f7) ← (**previo**(*visitar*(c4.f1)) **or previo**(*visitar*(c4.f5))) **and previo**(*seguir*(r7))

Por simplicidad, las cláusulas temporales siempre se expresarán de manera implícita. En nuestro lenguaje **N** todos los ítemes sobre los que se aplica un operador temporal se identifican a través del concepto al que está asociado y la relación funcional que los une:

Ítem = *concepto.relación-funcional*, por ejemplo, c5.f3.

En la cabeza de la regla temporal siempre aparecerá un único ítem que será el ítem que se pretende alcanzar o visitar. Llegar a ese ítem dependerá de las condiciones impuestas en el cuerpo que estará formado por dos partes:

1. La primera de ellas será una *fórmula temporal* en la que puede aparecer uno o más operadores temporales aplicados a ítemes de la estructura conceptual de navegación que deben ser predecesores del ítem de la cabeza.
2. El cuerpo de la regla siempre terminará con el operador lógico **and** y el operador temporal **previo** aplicado a una de las relaciones conceptuales que nos lleva al concepto de la cabeza.

A su vez, en la *fórmula-temporal* distinguimos dos fragmentos, unidos mediante un operador **and**:

1. El primero de ellos es la denominada *fórmula temporal del sistema* formada únicamente por operadores temporales **previo** aplicados a ítemes unidos mediante operadores **and** y/o **or**. Esta parte, tal y como se ha visto, es generada por el sistema de acuerdo a la estructura conceptual de navegación establecida EC_n.
2. El segundo fragmento lo denominados *fórmula temporal del autor* formada únicamente por operadores temporales **antes** y **desde** aplicados a ítemes unidos mediante operadores **and** y/o **or**. Esta porción de la restricción será construida por el autor en el caso de que desee limitar o conducir aún más la navegación del lector.

concepto.relación-funcional ← (*fórmula-temporal-sistema* **and** *fórmula-temporal-autor***)**
and previo(relación-conceptual)

Recapitulando —en la tabla 5.63 se resume esta sintaxis—, las Restricciones de Orden RT_o serán expresadas mediante el lenguaje **N** que estará compuesto por:

- **Constantes:** los ítemes II que se notan mediante *concepto.relación-funcional* —$c.r_f$, con $c \in C$ y $r_f \in R_f$— y las relaciones conceptuales —r_c, con $r_c \in R_c$—.
- **Predicados:** el predicado *visitar* que únicamente se puede aplicar a los ítemes —por ejemplo, *visitar*($c.r_f$)— y el predicado *seguir* que únicamente se puede aplicar a las relaciones conceptuales r_c. Estos predicados normalmente aparecerán de manera implícita en la Restricción de Orden.
- **Operadores lógicos: and**, **or**, **not**. El operador **not** únicamente podrá acompañar al operador temporal del pasado **antes**.
- **Operadores temporales: previo** —inmediatamente antes—, **antes** y **desde**. Únicamente se han incluido aquellos operadores del pasado que se han visto necesarios. El operador **previo** se puede aplicar a ítemes y relaciones conceptuales. Los operadores **antes** y **desde** solamente se pueden aplicar a ítemes.

- **Fórmulas:** una fórmula es una expresión del lenguaje que se obtiene recursivamente de la aplicación de operadores a los predicados aunque con ciertas limitaciones. Son posibles los siguientes tipos de fórmulas:
  - **Fórmulas atómicas**: **previo**(ítem); **antes**(ítem); **not antes**(ítem).
  - **Fórmulas simples**:
    previo(ítem1) **and previo** (ítem2);
    previo(ítem) **or previo** (ítem2);
    [**not**]**antes** (ítem1) **and** [**not**]**antes** (ítem2);
    [**not**]**antes** (ítem1) **or** [**not**]**antes** (ítem2);
    (ítem1) **desde** (ítem2).
    Los corchetes se interpretan como opcionales.
  - **Fórmulas compuestas:** *f* **or** *g*; *f* **and** *g*; donde f y g pueden ser una fórmula atómica, simple o compleja.

En una fórmula temporal tienen mayor precedencia los operadores temporales que los lógicos. Además, se establece un orden de precedencia de izquierda a derecha. Los paréntesis, se utilizarán para cambiar dicho orden.

- **Cláusulas:** son expresiones con la forma: **cabeza ← cuerpo** resultado del cierre universal de la fórmula **cuerpo → cabeza** que equivale a **not cuerpo or cabeza**.
  - **Cabeza:** es un único predicado *visitar* sobre uno de los ítemes de información.
  - **Cuerpo:** es una expresión que sigue el siguiente patrón:

(*fórmula-temporal-sistema* **and** *fórmula-temporal-autor* **) and previo(relación-conceptual)**

que representa la precondición de la acción de la cabeza.

  - **Fórmula-temporal-sistema:** es una fórmula en la que únicamente pueden aparecer operadores temporales **previo** aplicados a predicados *visitar* sobre ítemes de información unidos por operadores lógicos **and** y/o **or**. Los ítemes a los que se aplique el operador **previo** deben preceder inmediatamente, esto es, en un solo paso, al ítem de la cabeza.
  - **Fórmula-temporal-autor:** es una fórmula en la que únicamente pueden aparecer operadores temporales **antes** y **desde** aplicados a predicados *visitar* sobre ítemes de información unidos por operadores lógicos **and** y/o **or**. El operador temporal **antes** pueden estar precedidos por el operador lógico **not**. Los ítemes a los que se aplique los operadores **antes** y **desde** deben preceder en dos o más pasos, al ítem de la cabeza.

- **Programa Lógico:** programa construido con el lenguaje **N**. Este programa estará formado por un conjunto de cláusulas con la siguiente sintaxis:

concepto.relación-funcional ←
(*fórmula-temporal-sistema* **and** *fórmula-temporal-autor* **) and previo(relación-conceptual)**

Este programa lógico tendrá cómo mucho una Restricción de Orden por cada ítem de información y relación conceptual que llegue al concepto al que está asociado el ítem. Esto es, el número máximo de reglas temporales viene determinado por la propia EC_n. Lo único que puede hacer el autor

es imponer más limitaciones siempre y cuando estos cambios:

a) respeten la integridad de la EC_n y

b) sean consistentes o carezcan de toda contradicción.

Sintaxis del Lenguaje **N**	
Dominios	{II, R_C}
Predicados	{***visitar***(ítem), ***seguir***(relación-conceptual)}
Operadores lógicos	{**and**, **or**, **not**} (operador **not** sólo al operador temporal **antes**)
Operadores temporales	**previo**, **desde**, **antes**
Fórmulas atómicas	**previo**(ítem); **antes**(ítem); **not antes**(ítem)
Fórmulas simples	**previo**(ítem1) **and previo** (ítem2); **previo**(ítem1) **or previo** (ítem2); **antes** (ítem1) **and ante**s (ítem2); **antes** (ítem1) **or antes** (ítem2); (ítem1) **desde** (ítem2)
Fórmulas complejas	f **and** g; f **or** g; donde f y g son una fórmula atómica, simple o compleja
Cláusulas	**concepto.relación-funcional** ← fórmula-temporal **and previo(relación-conceptual)**
Fórmula-temporal	fórmula-temporal-sistema **and** fórmula-temporal-autor
Fórmula-temporal-sistema	Combinación de operador/es **previo** sobre ítem/es unidos con operadores **and** y/o **or**
Fórmula-temporal-autor	Combinación de operador/es **antes** y **desde** sobre ítem/es unidos con operadores **and** y/o **or**

Tabla 5.63 Sintaxis del Lenguaje **N**.

4.3.2 Metalógica para el Lenguaje N

Nuestro lenguaje **N** deberá ser interpretado en el contexto de los sistemas hipermedia. En consecuencia, debemos estudiar cómo funcionará el Sistema de Navegación mediante las cláusulas de este lenguaje. Para ello, es necesario determinar el significado o las asunciones metalógicas que determinan el funcionamiento de dicho lenguaje **N**.

En la navegación de un hiperdocumento bajo nuestro modelo, la interpretación del tiempo es en realidad la secuencia o el orden en que se permiten visitar los distintos ítemes. En cualquier momento tenemos dos instantáneas, una estática y otra dinámica.

La perspectiva estática viene representada por la estructura conceptual de navegación seleccionada. Esto es, sólo pueden formularse operadores temporales sobre ítemes que son predecesores del ítem de la cabeza. Dicho de otro modo, desde los ítemes del cuerpo de la cláusula se debe poder alcanzar el ítem de la cabeza. Esto será posible si existe un camino en nuestro grafo.

Por otro lado, tenemos la perspectiva dinámica que nos la ofrece la navegación en sí o ejecución de la EC_n y que se especifica mediante un programa lógico escrito con nuestro lenguaje **N**. En cada instante de tiempo, en el estado actual de la ejecución de la EC_n —más tarde veremos que este enfoque operacional nos lo ofrece la red de Petri equivalente al programa lógico—, se ha de comprobar si se cumplen o no los cuerpos de las Restricciones de Orden para determinar si es posible o no acceder a los ítemes de la cabeza.

Por tanto, el Sistema de Navegación deberá comprobar en primer lugar que las Restricciones de Orden pueden ser formuladas. Para ello, deberá cotejar que los operadores temporales han sido aplicados correctamente a los ítemes, o lo que es lo mismo, que los ítemes del cuerpo preceden al ítem de la cabeza. Estas pruebas se harán a lo largo del proceso de diseño de la navegación que realice el autor. Más tarde, en tiempo de ejecución, esto es, cuando el lector navegue por el documento, el Sistema de Navegación deberá verificar si se puede alcanzar en cada momento un determinado ítem de información. Esto es, si se puede desencadenar la acción *visitar*(ítem) de la cabeza. Ello dependerá de si la fórmula de la precondición de una acción —el cuerpo— resulta ser *true*.

Hecha esta aclaración, vamos a ver cómo son interpretadas las fórmulas, predicados y cláusulas por el Sistema de Navegación. Basándonos en la definición de Thayse [Thayse, 1989], definimos en qué consiste una interpretación, **I**, en nuestro sistema:

- **Interpretación del Predicado *visitar*:** es una función que devuelve un valor de *true* o *false* dependiendo de si el lector ha inspeccionado o no el ítem sobre el que se formula el predicado.

  I(***visitar***(ítem)) = true.

- **Interpretación del Predicado *seguir*:** es una función que devuelve un valor de *true* o *false* dependiendo de si el lector ha disparado o no un enlace –una relación conceptual— sobre el que se formula el predicado.

  I(***seguir***(relación-conceptual) = true

- **Interpretación o Evaluación de una Fórmula:** es una función que devuelve valores de verdad *true* o *false* de acuerdo a si se ha realizado una interpretación de los predicados ***visitar*** que la componen, y según sea la relación lógica y temporal que se establezca entre los predicados.

  I(*f*) = true.

- **Fórmula Válida:** una fórmula es válida si en el estado de ejecución ε en el que se encuentre la EC_n, su interpretación devuelve el valor *true*.

  $\mathbf{I}_{(\varepsilon)}(f) = true$.

- **Evaluación de un Cláusula:** Evaluar una cláusula implica comprobar si se puede o no inspeccionar —si el Sistema de Navegación puede mostrar— un determinado ítem de información.

  Para ello es necesario interpretar la fórmula de la precondición y concluir que se puede llevar a cabo la acción de la cabeza —***visitar***(ítem)—. El proceso de resolución seguido para evaluar una cláusula es el siguiente:

  > El lector va visitando los ítemes que le ofrece el Sistema de Navegación, obteniéndose acciones —produciéndose hechos desde una perspectiva declarativa—. Los cuerpos de las cláusulas o precondiciones incluyen predicados que se corresponden con esas acciones. Se van evaluando uno a uno los predicados de la precondición hasta

que finalmente se obtenga como resultado el valor *true* de las subfórmulas de ésta y de la precondición completa. Cuando ocurre esto, se concluye que la acción de la cabeza de la cláusula puede realizarse –se puede visitar el ítem de la cabeza—. En este caso, al lector se le mostrarán la/s relacion/es conceptual/es —los enlaces— que les llevan a ese ítem.

Resumiendo, se puede decir que en cada momento el estado de la ejecución ε de la EC_n —el conjunto de ítemes visitados— determina el conjunto de ítemes que se pueden inspeccionar de acuerdo a las dependencias que el autor haya establecido mediante las Restricciones de Orden.

4.3.3 Semántica del Lenguaje N

La semántica del cualquier lenguaje está íntimamente relacionada con la función o uso que se le quiere asociar. En nuestro caso, el lenguaje **N** nos permite expresar el orden en el que deben ser visitados los distintos ítemes que ofrece la estructura conceptual de navegación seleccionada.

Para ello usamos algunos operadores de lógica temporal del pasado —**previo**, **antes**, **desde**— que deben respetar en todo momento la semántica que ya nos impone la propia EC_n. Esto es, en nuestro caso, la posibilidad de visitar o no un ítem depende de si es o no alcanzable desde lugar en el que nos encontramos. Veamos pues, para cada operador temporal y lógico, qué comportamiento del sistema modela y cómo se debe evaluar una precondición que lo contenga.

4.3.3.1 Operadores Temporales

Los únicos operadores de la lógica temporal que usamos son **previo**, **antes** y **desde**. Sus operandos son siempre acciones, es decir, predicados y no fórmulas. Se encargan de evaluar la ocurrencia de acciones y la relación temporal entre sus ocurrencias. En nuestro caso las únicas acciones que se pueden constatar es el haber visitado un determinado conjunto de ítemes y disparar una determinada relación conceptual. La relación temporal queda reducida a cotejar si en un determinado estado de la ejecución se han inspeccionado ya o no determinados ítemes del dominio de información seleccionado.

A continuación, se describen todos los operadores y su semántica dependiendo de la posible cláusula en la que puede aparecer. Para ello, se pondrá un ejemplo concreto, la interpretación semántica, su comprobación mediante la teoría de grafos y una figura en donde se indican los nodos y asociaciones involucrados —marcándolos en gris—.

- ***visitar***(c2.f1) ← **previo**(***seguir***(r1))
  previo cuando se aplica a una relación conceptual expresa si se ha disparado una de las relaciones conceptuales que alcanzan al concepto de la cabeza de manera directa o en un solo paso. Para poder escribir esto —figura 5.13—, el sistema debe comprobar apoyándose en las funciones de alcanzabilidad que:

  $$|\Gamma^{-1}(c2, r1)| \neq 0 \text{ y } |\Gamma^{+1}(c2, f1)| \neq 0.$$

  En realidad, este cotejo no será necesario porque esta parte de la Restricción de Orden es generada automáticamente por el Sistema de Navegación partiendo de la EC_n.

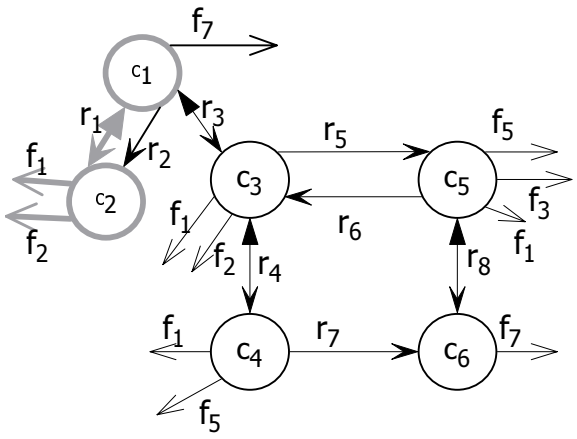


Figura 5.13 Operador **previo** aplicado a relación conceptual: ***visitar***(c2.f1) ← **previo**(***seguir***(r1))

- ***visitar***(c2.f1) ← **previo**(***visitar***(c1.f7))

  previo cuando se aplica a un ítem expresa si se ha visitado un ítem que precede de forma inmediata o en un solo paso al ítem de la cabeza. Para poder escribir esto, figura 5.14, el sistema debe comprobar apoyándose en las funciones de alcanzabilidad que:

$$c1 \subseteq \Gamma^{-1}(c2),\ |\Gamma^{+1}(c2, f1)| \neq 0 \text{ y } |\Gamma^{+1}(c1, f7)| \neq 0.$$

  Nuevamente, esta verificación no será necesaria ya que el Sistema de Navegación es el único que puede formular operadores **previo** sobre ítemes teniendo en cuenta la EC_n. Al autor no le está permitido utilizar este operador temporal.

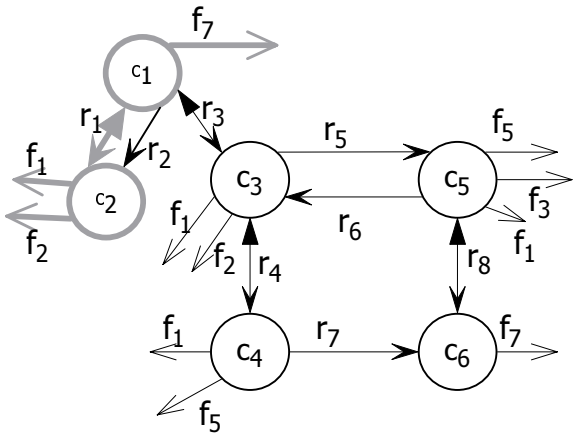


Figura 5.14 Operador **previo** aplicado a un ítem: ***visitar***(c2.f1) ← **previo**(***visitar***(c1.f7))

- ***visitar***(c2.f1) ← ***antes***(***visitar***(c5.f3))

  El operador temporal **antes** sólo se puede aplicar a ítemes de información. En este caso estamos indicando que se puede visitar el ítem de la cabeza c2.f1 si anteriormente el lector ha visitado el ítem c1.f7 que debe preceder, en al menos dos pasos, al ítem de la cabeza. El autor podrá formular ese operador siempre y cuando el sistema verifique que, figura 5.15:

$$c5 \notin \Gamma^{-1}(c2),\ c5 \subseteq \Gamma^{-}(c2),\ |\Gamma^{+1}(c2, f1)| \neq 0 \text{ y } |\Gamma^{+1}(c1, f7)| \neq 0.$$

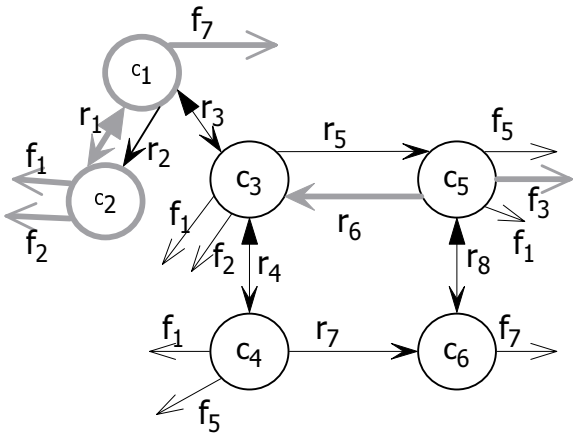


Figura 5.15 Operador **antes** aplicado a un ítem: ***visitar***(c2.f1) ← **antes**(***visitar***(c5.f3))

- ***visitar***(c2.f1) ←***visitar***(c5.f3) **desde** ***visitar***(c4.f1)
  El operador temporal **desde** sólo se puede aplicar a ítemes de información. **desde** se utiliza para establecer una relación de orden entre dos visitas de dos ítemes como condición previa para visitar el ítem de la cabecera. En este caso estamos indicando que se puede inspeccionar el ítem de la cabeza c2.f1 si anteriormente el lector ha visitado el ítem c5.f3 después de haber visto el ítem c4.f1. El autor podrá formular ese operador siempre y cuando el sistema verifique que:

  $c4 \subseteq \Gamma^{-1}(c5)$, $c5 \subseteq \Gamma^{-}(c2)$, $|\Gamma^{+1}(c2, f1)| \neq 0$, $|\Gamma^{+1}(c5, f3)| \neq 0$ y $|\Gamma^{+1}(c4, f1)| \neq 0$.

En el ejemplo —figura 5.16— vemos cómo el autor considera que el lector debe acceder al ítem c5.f3 una vez que ha visitado el c4.f1 para poder alcanzar el ítem c2.f1. Si el autor no hubiese impuesto esta precondición en la Restricción de Orden, el usuario podría directamente ir desde c4.f1 a c2.f1 sin tener que pasar por c5.f3, pues existe un camino más directo entre c4 y c2, {c4→ c3 → c1 → c2} que el que nos plantea el autor {c4 → c6 → c5 → c3 → c1 → c2}. La segunda navegación es mucho más restrictiva pues obliga al lector a pasear antes por todos los conceptos visitando al menos uno de sus ítemes antes de poder alcanzar el ítem c2.f1.

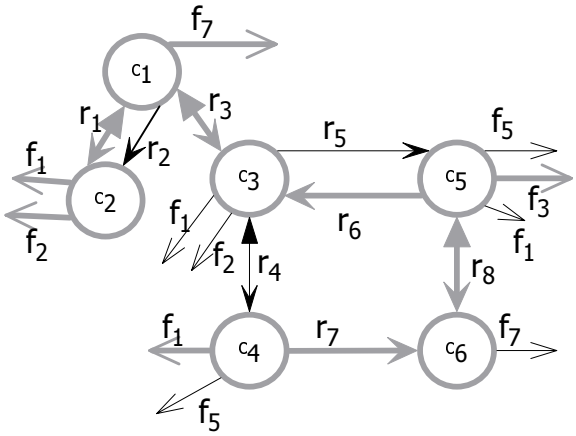


Figura 5.16 Operador **desde** aplicado a ítemes: ***visitar***(c2.f1) ← (***visitar***(c5.f3) **desde** ***visitar***(c4.f1))

4.3.3.2 Operadores Lógicos *and* y *or*

Los únicos operadores lógicos que se pueden usar de manera genérica son **and** y **or**. Estos actúan como conectores lógicos sin ningún valor de orden. Sus operandos deben ser fórmulas. Tienen menor precedencia que los operadores temporales.

- ***visitar***(c3.f1) ←**op-temp**(***visitar***(c4.f1)) **or op-temp**(***visitar***(c4.f5))
  Esta expresión se evalúa a *true* si alguno de sus operandos es evaluado a *true*. Si con anterioridad el lector ha navegado por alguno de esos ítemes, éste podrá visitar el ítem de la cabeza.

- ***visitar***(c3.f1) ←**op-temp**(***visitar***(c4.f1)) **and op-temp**(***visitar***(c4.f5))
  Esta expresión se evalúa a *true* si con anterioridad el lector ha navegado por ambos ítemes. No importa el orden en el que han ocurrido estas acciones, si las dos se han producido el lector puede visitar el ítem de la cabeza.

4.3.3.3 Operador Lógico *not*

El uso del operador **not** es bastante restrictivo. Únicamente puede acompañar de manera simple o directa al operador temporal **antes**. No tiene ningún sentido que el operador lógico acompañe al operador temporal **previo** porque iría en contra de la semántica proporcionada por la EC_n. Tampoco le hemos encontrado ninguna utilidad negar el operador temporal **desde**.

Por tanto, queda reducido su uso al operador **antes** para que el autor pueda expresar situaciones como la siguiente: *en el dominio de información seleccionado existen solapamientos entre la información que contienen algunos de sus ítemes. En tal caso, no queremos que el lector vuelva a consultar la misma información. Por tanto, si el navegante no ha visitado ya esa información en otro ítem entonces tendrá sentido ofrecerle un ítem que contenga esa información.*

- ***visitar***(c3.f1) ←**not antes**(***visitar***(c4.f1))
  Esta expresión se evalúa a *true* si con anterioridad —en el estado en el que se encuentra la ejecución de la EC_n— el lector no ha inspeccionado el ítem del cuerpo.

4.3.3.4 Cláusulas no Permitidas

Indudablemente, el sistema debe garantizar que la modificación de las Restricciones de Orden RT_o por parte del autor no contenga ningún tipo de contradicción. Tenemos pues, que comprobar las posibles inconsistencias —razonamientos lógicos imposibles— derivadas de las combinaciones de los operadores temporales mediante los operadores lógicos **and** y **or**. Para ello, se ha realizado una revisión exhaustiva de los casos que se nos pueden plantear, rechazando aquellos que nos llevan a significados opuestos.

Este estudio que puede parecer *a priori* muy farragoso, no es lo tanto pues la sintaxis del lenguaje **N** nos impone un gran número de limitaciones. Si recordamos toda Restricción de Orden sigue el siguiente formato:

Ítem ← (*fórmula-temporal-sistema* **and** *fórmula-temporal-autor* **) and previo(relación-conceptual)**

Según esto, nuestra precondición esta formada por tres partes unidas por el operador lógico **and**. La tercera de estos fragmentos no nos puede causar ninguna inconsistencia lógica por tratarse de una fórmula atómica: **previo(relación-conceptual)**.

Queda por estudiar las dos fórmulas temporales. La *fórmula temporal del sistema* estará formada únicamente por operadores temporales **previo** unidos mediante operadores lógicos **and** y **or**. Esta fórmula es generada de manera automática por el Sistema de Navegación y no permite que contenga redundancias del tipo: **previo**(ítemA) **and previo**(ítemA) ó **previo**(ítemA) **or previo**(ítemA). Tampoco se puede utilizar el operador lógico **not**, por tanto, esta parte de la Restricción de Orden no contendrá ningún tipo de inconsistencia. La tabla 5.64 resume estas posibilidades.

Fórmula Temporal del Sistema	
Casos Posibles	**Clasificación**
previo(ítemA) **and previo**(ítemB)	**Correcto**
previo(ítemA) **or previo**(ítemB)	**Correcto**

Tabla 5.64 Fórmula Temporal del Sistema: Cláusulas permitidas.

El único fragmento que nos puede causar problemas sería el construido por el autor. En esta fórmula pueden aparecer operadores temporales **antes** y **desde** agrupados con operadores lógicos **and** y **or**. El operador **antes** puede estar precedido del operador **not**.

En la tabla 5.65 se describen todos los casos posibles. Clasificamos las distintas posibilidades como: correcto, incorrecto y mejorables. Las expresiones incorrectas serán rechazadas por el sistema, las mejorables serán reescritas con el fin de simplificar la expresión y eliminar redundancias innecesarias pero manteniendo la semántica. La tabla 5.65 muestra todos los casos indicando su clasificación y, en el caso de ser mejorable, la expresión equivalente en lenguaje **N**.

4.4 Las Acciones Evolutivas del Sistema de Navegación AC_{EN}

Al igual que el resto de los sistemas del Modelo SEM-HP, el sistema de Navegación también puede evolucionar. El autor necesita una serie de operaciones que le permitan especificar y modificar las restricciones de navegación, tanto las de navegabilidad RT_{nb} como las de orden RT_{o}. La ejecución de una de estas operaciones por parte del autor, supone una acción de cambio o evolución. El Sistema de Navegación debe garantizar la integridad y para ello:

1. Hará uso de un conjunto de Meta-Restricciones —ver el subcapítulo 5.1 denominado introducción— que permitan verificar la consistencia del cambio.
2. Propagará el cambio dentro del propio sistema haciendo evolucionar al resto de sus elementos
3. Propagará el cambio al resto de los sistemas haciéndolos evolucionar.

A continuación se describen las operaciones que hacen posible el cambio de las restricciones del Sistema de Navegación, el conjunto de Meta-Restricciones que garantizan la consistencia del cambio y el impacto o propagación del cambio dentro del Sistema.

Fórmula Temporal del Autor		
Casos Posibles	**Clasificación**	**Reescritura**
antes(ítemA) **or antes**(ítemA)	Mejorable	**antes**(ítemA)
not antes(ítemA) **or antes**(ítemA)	**Incorrecto**	
not antes(ítemA) **or not antes**(ítemA)	Mejorable	**not antes**(ítemA)
antes(ítemA) **and antes**(ítemA)	Mejorable	**antes**(ítemA)
not antes(ítemA) **and antes**(ítemA)	**Incorrecto**	
not antes(ítemA) **and not antes**(ítemA)	Mejorable	**not antes**(ítemA)
antes(ítemA) **or antes**(ítemB)	Correcto	
not antes(ítemA) **or antes**(ítemB)	Correcto	
not antes(ítemA) **or not antes**(ítemB)	Correcto	
antes(ítemA) **and antes**(ítemB)	Correcto	
not antes(ítemA) **and antes**(ítemB)	Correcto	
not antes(ítemA) **and not antes**(ítemB)	Correcto	
(ítemA **desde** ítemA)	**Incorrecto**	
(ítemA **desde** ítemB)	Correcto	
(ítemA **desde** ítemB) **or** (ítemB **desde** ítemA)	Mejorable	**antes**(ítemA) **and antes**(ítemB)
(ítemA **desde** ítemB) **and** (ítemB **desde** ítemA)	**Incorrecto**	
(ítemA d**esde** ítemB) **or not antes**(ítemA)	Correcto	
(ítemA **desde** ítemB) **or not antes**(ítemB)	Correcto	
(ítemA **desde** ítemB) **or antes**(ítemA)	Mejorable	**antes**(ítemA)
(ítemA **desde** ítemB) **or antes**(ítemB)	Mejorable	**antes**(ítemB)
(ítemA **desde** ítemB) **and not antes**(ítemA)	**Incorrecto**	
(ítemA **desde** ítemB) **and not antes**(ítemB)	**Incorrecto**	
(ítemA desde ítemB) **and antes**(ítemA)	Mejorable	(ítemA **desde** ítemB)
(ítemA desde ítemB) **and antes**(ítemB)	Mejorable	(ítemA **desde** ítemB)

Tabla 5.65 Fórmula Temporal del Autor: Cláusulas permitidas y no permitidas.

4.4.1 Cambiando las Restricciones de Navegabilidad RT_{NB}

Como se ha definido, por defecto, la navegabilidad de una asociación viene determinada por la conectividad, esto es, es navegable en ambos sentidos. Pero el autor puede limitar la navegación de esa asociación a la dirección o sentido que se estableció en el sistema de memorización —en este caso, la navegación viene determinada por la alcanzabilidad—. En cualquier momento, el autor puede cambiar para cada asociación su restricción de navegabilidad RT_{nb}. Para ello hará uso de dos acciones evolutivas:

1. **Reducir-navegabilidad, RedNB**: se limita la navegación al sentido que se ha establecido en la estructura conceptual. La asociación sólo es navegable desde el concepto origen al destino de la asociación.
2. **Ampliar-navegabilidad, AmpNB**: se amplía la navegación, haciéndola posible en ambos sentidos independientemente de cuál fue la asociación originalmente establecida.

Veamos a continuación, una descripción formal de la aplicación de estas dos acciones evolutivas a la Estructura Conceptual. Para cada una de ellas se especifica su precondición, esto es, las Meta-Restricciones que deben cumplirse para poder llevar acabo estas operaciones. Estas Meta-Restricciones se especifican formalmente mediante Teoría de Grafos.

$\{RedNB, AmpNB\} \in AC_{en}$

Dada la estructura conceptual de navegación que será manipulada, $EC_n = (C, II, A_c, A_f, DC, DR, DF)$, llamaremos, $EC_n' = (C', II', A_c', A_f', DC', DR', DF')$, a la estructura conceptual una vez que haya sido manipulada.

AC[nb1] Se limita la navegación de una asociación conceptual y, por tanto, se elimina la asociación inversa de dicha asociación —figura 5.11—.

RedNB: $(<c_o, r_c, c_d>, EC_n) \rightarrow EC_n'$, donde $<c_o, r_c, c_d> \in A_c$, $c_o, c_d \in C$, $r_c \in R_c$.
Si $<c_o, r_{cinv}, c_d> \in A_c \Rightarrow A_c' = A_c - <c_o, r_{cinv}, c_d>$, $DR' = DR - \{r_{cinv}\}$.

AC[nb2] Se amplía la navegación de una asociación conceptual y, por tanto, se añade la asociación inversa de dicha asociación —figura 5.11—.

AmpNB: $(<c_o, r_c, c_d>, EC_n) \rightarrow EC_n'$, donde $<c_o, r_c, c_d> \in A_c$, $c_o, c_d \in C$ y $r_c \in R_c$.
Si $\exists <c_o, r_{cinv}, c_d> \notin A_c \Rightarrow A_c' = A_c \cup <c_o, r_{cinv}, c_d>$, $DR' = DR \cup \{r_{cinv}\}$.

4.4.2 Cambiando las Restricciones de Orden RT_o

Tal y como se ha visto antes, las Restricciones de Orden, expresadas mediante el lenguaje **N** —en lógica temporal—, siempre van a tener en la cabecera un ítem y en el cuerpo una fórmula, con la siguiente sintáxis:

Ítem ← *fórmula-temporal* **and previo**(*relación-conceptual*)
ó
Ítem ← **(** *fórmula-temporal-sistema* **and** *fórmula-temporal-autor* **) and previo(***relación-conceptual***)**

Donde, el ítem se especifica mediante su asociación funcional, esto es, el concepto al que está unido y la relación funcional que los liga. Por tanto, tendrá la forma *concepto.relaciónfuncional*.

La *fórmula-temporal* es una secuencia de predicados temporales **antes** y **desde** sobre ítemes especificados de igual modo que en la cabecera ligados mediante operadores lógicos **and** y **or**. Esta sintaxis tiene que terminar necesariamente con un operador lógico **and** y un operador temporal **previo** sobre una de las relaciones conceptuales que recibe o alcanza directamente al concepto asociado al ítem de la cabecera. En consecuencia, cada una de estas restricciones se puede identificar indicando el ítem —concepto más relación funcional— de la cabecera y la relación conceptual de acceso al ítem.

El número de restricciones de navegación siempre es fijo. El sistema de navegación las genera automáticamente a partir de la estructura conceptual de navegación. No obstante, el autor puede desactivar por completo una restricción o activar por completo una restricción, añadir o quitar operadores temporales **antes** y **desde** a la *fórmula-temporal-autor* o modificar operadores lógicos en la

fórmulta-temporal mediante una serie de operaciones de evolución. En consecuencia, el Sistema de Navegación debe contar con las siguientes acciones de evolución para la especificación de Restricciones de Orden:

1. **Desactivar-restricción, DactRT:** se borra o desactiva una de las restricciones, para que no sea tenida en cuenta como restricción durante la navegación.
2. **Activar-restricción, ActRT**: se añade o activa una de las restricciones que anteriormente fue desactivada.
3. **Modificar-restricción, ModRT**: se cambia algún **and** por **or** o algún **or** por **and** de la *fórmula-temporal*.
4. **Aumentar-restricción, AumRT**: se añade mediante un **and** u **or** un operador temporal del tipo **antes** o **desde** sobre un ítem a la *fórmula-temporal-autor* de la restricción.
5. **Reducir-restricción, RedRT**: se elimina un operador temporal del tipo antes o desde sobre un ítem junto a su operador lógico —**or** ó **and**— a la *fórmula-temporal-autor* de la restricción.

Estas operaciones deben de cumplir en todo momento una serie de Meta-Restricciones que garanticen la consistencia del programa lógico formado por el conjunto de fórmulas en lenguaje **N**. Necesitamos, pues, de un lenguaje al que denominamos de **M**eta**N**avegación o lenguaje **MN** que al igual que el lenguaje **N** será expresado en lógica temporal pero en este caso de predicados.

4.4.1.1 Definición del Lenguaje de MetaNavegación MN

El lenguaje **MN** permite especificar las precondiciones de las acciones evolutivas destinadas a cambiar las Restricciones de Orden, esto es, las cláusulas en lenguaje **N**. Las acciones en este caso estarán relacionadas con activar, desactivar, modificar, aumentar y reducir una Restricción de Orden. A estas acciones que aparecerán en la cabeza de la cláusula se les asociarán un conjunto de precondiciones —predicados unidos con operadores lógicos— que comprobarán si previamente se verifican —se evalúan a *true*— o no determinadas circunstancias.

Los elementos y la sintaxis del Lenguaje **MN** son muy parecidos al del Lenguaje **N**. Al igual que en el lenguaje **N**, las Meta-Restricciones se escriben mediante cláusulas que modelan la relación entre dos acciones, de forma, que la acción —o el conjunto de las acciones del cuerpo— es una precondición de la acción de la cabeza que se modela con la regla:

$$p(x) \leftarrow previo\ q(y)$$

Sin embargo, las acciones que pueden aparecer en la cabeza serán alguna de las acciones de evolución para las Restricciones de Orden, y en sus cuerpos aparecerán un conjunto de predicados fijos que modelan la verificación de consistencia para cada una de las operaciones de la cabeza. Además, la lógica temporal admite ahora predicados cuyas variables serán los distintos elementos del lenguaje **N** sobre los que se desea preguntar. Por último, en esta lógica temporal el único operador que se utilizará de manera implícita es **previo**. Semánticamente lo único que nos interesa es comprobar si el *estado* en el se encuentra el programa lógico escrito en lenguaje **N** antes y después de realizar la acción evolutiva es consistente. Por tanto, cada una de las Meta-Restricciones de Orden se puede interpretar como: "*se puede realizar la acción evolutiva de la cabeza si se cumplen las condiciones impuestas en el cuerpo de la regla temporal*"

Resumiendo, las Meta-Restricciones de Orden RT_o serán expresadas mediante el lenguaje **MN** que estará compuesto por:

- **Variables:** {VF, VP, VO, VC, VAf, VAc} donde,
  VF = {vF_1, vF_2, ..., vFn} son las fórmulas del lenguaje **N**.
  VP = {vP_1, vP_2, ..., VP_m} son los predicados del lenguaje **N**.
  VO = {vO_1, vO_2, ..., VO_p} son operadores lógicos.
  VC = {vC_1, vC_2, ..., vC_q} son conceptos C.
  VAf = {vAf_1, vAf_2, ..., vAf_r} son asociaciones funcionales Af.
  VAc = {vAc_1, vAc_2, ...,vAc_s} son asociaciones conceptuales Ac.

- **Predicados:** {*DactRT*(vC_1, vAf_1, vAc_1), *ActRT*(vC_1, vAf_1, vAc_1), *ModRT*(vC_1, vAf_1, vAc_1), *AumRT*(vC_1, vAf_1, vAc_1, vO_1, vP_2), *RedRT*(vC_1, vAf_1, vAc_1).

  Estos predicados serán los predicados —acciones— que aparecerán en las cabezas de las Meta-Restricciones de Orden. En todas las acciones es necesario identificar la Restricción de Orden que se pretende cambiar. Si recordamos, nuestro programa lógico puede llegar a tener tantas cláusulas como el producto que resulte de multiplicar el número de ítemes por el número de relaciones conceptuales que nos permiten alcanzarlo. Esto significa que a cada Restricción de Orden se le puede hacer referencia de manera unívoca indicando un ítem y una relación conceptual. Pero como, a su vez, un ítem se identifica con un concepto y una relación funcional, entonces, una restricción se describe con un concepto, una asociación funcional y una asociación conceptual.

- **Predicados auxiliares:** {*isInRT*$_o$(vC_1, vAf_1, vAc_1), *isInAf*(vC_1, vAf_1), *Reachable*(vC_1, vC_2, n), *Equal*(vC_1, vAf_1, vC_2, vAf_2), *Component*(vP_1, vC_1, vAf_1, vAc_1), Semantic(vC_1, vAf_1, vAc_1)}

  Denominamos predicados auxiliares a aquellos que formarán parte del cuerpo de la Meta-Restricción de Orden. Para evaluar estos predicados se hará uso de la teoría de grafos, esto es, la comprobación se realizará directamente sobre la estructura conceptual de navegación EC_n. Estas verificaciones se podrían haber hecho también mediante lógica pero para ello es necesario especificar el grafo en este formalismo. Es mucho más fácil y rápido cotejar los predicados haciendo uso de la representación con la que contamos. A continuación se describe para cada predicado su semántica:

  - *IsInRT*$_o$(vC_1, vAf_1, vAc_1): ¿existe una restricción identificada con esos parámetros?

    cierto si en EC_n se verifica que $vC_1 \in C_n$, $vAf_1 \in A_{fn}$ y $vAc_1 \in A_{cn}$ y $\exists$ $rt_o \in RT_o$ identificada con vC_1, vAf_1, vAc_1.
  - *isInAf*(vC_1, vAf_1): ¿existe dicha asociación funcional?

    cierto si en EC_n se verifica que $vC_1 \in C$ y $vAf_1 \in A_{fn}$ y $|\Gamma^{+1}(vC_1, vAf_1)| \neq 0$
  - *Reachable*(vC_1, vC_2, n): ¿es alcanzable el concepto vC_2 desde vC_1 en *n* pasos?

    cierto si en EC_n se verifica que vC_1, $vC_2 \in C$ y $vC_2 \subseteq \Gamma^{+n}(vC_1)$.
  - *Equal* (vC_1, vAf_1, vC_2, vAf_2): ¿se trata de la misma asociación funcional?

    cierto si en EC_n se verifica que vC_1, $vC_2 \in C$ y vAf_1, $vAf_2 \in A_{fn}$ y $vC_1 = vC_2$ y $vAf_1 = vAf_2$.

- *Component* (vP_1, vC_1, vAf_1, vAc_1): ¿el predicado vP_1 aparece en el cuerpo de la restricción?

  cierto si en EC_n se cumple que c1, c2 ∈ C y rf1, rf2 ∈ Rf y rc ∈ Rc y ∃ $rt_o \in RT_o$ identificada con vC_2, vAf_2, vAc_2 que en su cuerpo aparezca vP_1.

- *Semantic*(vC_1, vAf_1, vAc_1): ¿es correcta semánticamente la restricción?

  cierto si la $rt_o \in RT_o$ identificada con vC_1, vAf_1, vAc_1 es correcta semánticamente, esto es, si todas las fórmulas simples de la fórmula temporal son posibles —vea la tabla 5.65—.

- **Operadores lógicos: and**, **or, not**.
- **Operadores temporales: previo**.

4.4.1.2 Acciones de Evolución para las Restricciones de Orden

Una vez descrito el lenguaje **MN** con el que se especificarán las metarrestricciones de las acciones evolutivas destinadas a cambiar las Restricciones de Orden, pasamos a describir detalladamente su uso.

$\{DactRT(vC_1, vAf_1, vAc_1), ActRT(vC_1, vAf_1, vAc_1), ModRT(vC_1, vAf_1, vAc_1), AumRT(vC_1, vAf_1, vAc_1, vO_1, vP_2), RedRT(vC_1, vAf_1, vAc_1)\} \subseteq AC_{en}$

1. Desactivar-restricción, DactRT: se borra o desactiva una de las Restricciones de Orden que han sido construidas de forma automática por el sistema a partir de la estructura conceptual de navegación EC_n. El autor no puede borrar indiscriminadamente estas restricciones.

El autor no debería poder borrar indiscriminadamente estas restricciones. El sistema debe garantizar que se muestran todos los ítemes y que se pueda navegar por todas las relaciones conceptuales —cualquier otra decisión sería absurda puesto que el autor ya ha realizado su elección y es ése el dominio de información que quiere mostrar y no otro, y quiere que el lector navegue por esas relaciones conceptuales y no otras—.

Luego no se pueden borrar todas las restricciones asociadas a un ítem, ni tampoco todas las restricciones temporales en las que aparezca en el cuerpo una misma relación conceptual. Estas dos condiciones son postcondiciones que deberá cumplir esta acción de evolución de borrado. En el ejemplo anterior —tabla 5.60—, no se puede borrar:

- Restricción1 y Restricción2 ó
- Restricción3 y Restricción4 ó
- Restricción1 y Restricción3 ó
- Restricción2 y Restricción4.

√ **Metarrestricción:** Existencia previa.

$$DactRT(vC_1, vAf_1, vAc_1) \leftarrow isInRT_o(vC_1, vAf_1, vAc_1)$$

√ **Postcondición:** Se puede mostrar el ítem.

$isInRT_o(vC_1, vAf_1, vAc)$, siendo vAc cualquier asociación conceptual

√ **Postcondición:** Se puede navegar por esa relación conceptual.

$isInRT_o(vRT, vC, vAf, vAc_1)$, esto es, $\exists\ rt_o \in RT_o$ que se puede identificar con vC, vAf, vAc_1, donde vC, vAf pueden tomar cualquier valor.

2. Activar-restricción, ActRT: se añade o activa una de las restricciones que anteriormente fue desactivada. En este caso se debe comprobar que no está activa en ese momento esa restricción de navegación.

√ **Metarrestricción:** No existencia previa.

$$ActRT(vC_1, vAf_1, vAc_1) \leftarrow \textbf{not}\ isInRT_o(vC_1, vAf_1, vAc_1)$$

3. Modificar-restricción, ModRT: se cambia algún **and** por **or** o algún **or** por **and** de la *fórmula-temporal*. Como postcondición habrá que comprobar que no existe ninguna contradicción derivadas de las combinaciones de estos operadores lógicos, esto es, que todas las fórmulas simples son semánticamente correctas.

√ **Metarrestricción:** Existencia previa. La restricción sobre la que se va hacer el cambio debe existir.

$$ModRT(vC_1, vAf_1, vAc_1) \leftarrow isInRT_o(vC_1, vAf_1, vAc_1)$$

√ **Postcondición:** La cláusula es semánticamente correcta.

$$Semantic(vC_1, vAf_1, vAc_1)$$

3. Aumentar-restricción, AumRT: se añade a una restricción de navegación mediante un **and** o un **or**, un operador temporal del tipo **antes** o **desde**. Como postcondición habrá que comprobar que no existe ninguna contradicción derivadas de las combinaciones de estos operadores lógicos, esto es, que todas las fórmulas simples que se están cambiando son semánticamente correctas.

√ **Metarrestricción:** Existencia previa. Se debe comprobar que existe el ítem sobre el que se aplica el operador temporal **antes**, o los ítemes sobre los que se aplica el operador **desde** existen. Por tanto, hay que comprobar que existen dichas asociaciones funcionales con un determinado ítem.

$$AumRT(vC_1, vAf_1, vAc_1, vO_1, \textbf{antes}(vC_2, vAf_2)) \leftarrow isInRT_o(vC_1, vAf_1, vAc_1)\ \textbf{and}\ isInAf(vC_1, vAf_1)\ \textbf{and}\ isInAf(vC_2, vAf_2)$$

$$AumRT(vC_1, vAf_1, vAc_1, vO_1, (vC_2, vAf_2\ \textbf{desde}\ vC_2, vAf_2) \leftarrow isInRT_o(vC_1, vAf_1, vAc_1)\ \textbf{and}\ isInAf(vC_1, vAf_1)\ \textbf{and}\ isInAf(vC_2, vAf_2)\ \textbf{and}\ isInAf(vC_3, vAf_3)$$

√ **Metarrestricción:** Alcanzabilidad. Se debe comprobar que efectivamente los operadores temporales que se usan en el cuerpo son posibles, esto es, que desde esos ítemes se puede alcanzar el ítem del cuerpo. Dado que, antes hemos comprobado que efectivamente existen esos ítemes, únicamente debemos comprobar que desde los conceptos del cuerpo se puede alcanzar el concepto de la cabeza.

$$\textit{AumRT}(vC_1, vAf_1, vAc_1, vO_1, \textbf{antes}(vC_2, vAf_2)) \leftarrow \textit{Reachable}(vC_1, vC_2, n), \text{ con } n \geq 2$$

$$\textit{AumRT}(vC_1, vAf_1, vAc_1, vO_1, (vC_2, vAf_2 \text{ desde } vC_2, vAf_2) \leftarrow \textit{Reachable}(vC_3, vC_2, n) \textbf{ and } \textit{Reachable}(vC_2, vC_1, n), \text{ con } n \geq 2$$

√ **Metarrestricción:** Comprobación de recursividad. En el cuerpo de una restricción no puede aparecer el mismo ítem de la cabeza.

$$\textit{AumRT}(vC_1, vAf_1, vAc_1, vO_1, \textbf{antes}(vC_2, vAf_2)) \leftarrow \textbf{not } \textit{Equal}(vC_1, vAf_1, vC_2, vAf_2)$$

$$\textit{AumRT}(vC_1, vAf_1, vAc_1, vO_1, (vC_2, vAf_2 \textbf{ desde } vC_2, vAf_2) \leftarrow \textbf{not } \textit{Equal}(vC_1, vAf_1, vC_2, vAf_2) \textbf{ and not } \textit{Equal}(vC_1, vAf_1, vC_3, vAf_3) \textbf{ and not } \textit{Equal}(vC_2, vAf_2, vC_3, vAf_3)$$

√ **Metarrestricción:** Ciclos recursivos cortos. Sea A precondición de B, si a su vez B es una precondición de A, entonces hablamos de ciclo corto. Esto es, tenemos dos restricciones cuya acción de la cabeza forma parte del cuerpo de la otra restricción. La comprobación de que exista una restricción para un ítem no es necesario hacerla porque todos los ítemes deben tener un camino de acceso tal y como hemos visto en la operación de desactivación de restricciones.

$$\textit{AumRT}(vC_1, vAf_1, vAc_1, vO_1, vP(vC_2, vAf_2)) \leftarrow \textit{isInRT}_o(vC_2, vAf_2, vAc_2) \textbf{ and not } \textit{Component}(vP(vC_1, vAf_1), vC_2, vAf_2, vAc_2)$$

√ **Metarrestricción:** Ciclos recursivos largos. Llamaremos ciclo largo a una situación de precondición en la que intervienen tres o más acciones de forma que, en el caso en que haya tres acciones involucradas, la acción de la cabeza de A es precondición de la acción de B y B es precondición de la acción C, y además, se desea que C sea precondición de A.

$$\textit{AumRT}(vC_1, vAf_1, vAc_1, vO_1, vP(vC_2, vAf_2)) \leftarrow \textit{isInRT}_o(vC_2, vAf_2, vAc_2) \textbf{ and } \textit{isInRT}_o(vC_3, vAf_3, vAc_3) \textbf{ and not } \textit{Component}(vP(vC_3, vAf_3), vC_2, vAf_2, vAc_2) \textbf{ and not } \textit{Component}(vP(vC_1, vAf_1), vC_3, vAf_3, vAc_3)$$

√ **Postcondición:** la cláusula es semánticamente correcta.

$$\textit{Semantic}(vC_1, vAf_1, vAc_1)$$

4. Reducir-restricción, RedRT: se elimina un operador temporal del tipo **antes** o **desde** sobre un ítem junto a su operador lógico —**or** ó **and**— a la restricción de navegación.

√ **Metarrestricción:** Existencia previa. La restricción sobre la que se va hacer el cambio debe existir.

$$RedRT(vC_1, vAf_1, vAc_1) \leftarrow isInRT_o(vRT_1, vC_1, vAf_1, vAc_1)$$

4.4.3 Propagación de los Cambios en el Sistema de Navegación

Cada vez que el autor decide cambiar las restricciones de navegabilidad y/o de orden, el Sistema deberá garantizar la consistencia del mismo. Para ello, se habrá de comprobar que el cambio se puede realizar. En realidad, esto lo garantizan las propias acciones evolutivas puesto que han de respetar en todo momento una serie de metarrestricciones —precondiciones y postcondiciones— expresadas mediante teoría de grafos para las restricciones de navegabilidad y en lógica temporal de predicados para las Restricciones de Orden.

Pero a continuación, el sistema deberá propagar este cambio en el propio sistema para que el resto de los elementos del Sistema de Navegación evolucionen de acuerdo a la modificación establecida. Por último, se deberá considerar la posibilidad de propagar estos cambios al resto de los Sistemas.

4.4.3.1 Cambiando las Restricciones de Navegabilidad

Si cambian las restricciones de navegabilidad, RT_{nb}, limitando o ampliando la navegación —RedNB, AmpNB— de las asociaciones habrá que actualizar las Restricciones de Orden:

1. Si el autor/a reduce la navegabilidad de una asociación se deben eliminar todas las Restricciones de Orden en cuyo cuerpo aparezca esa asociación conceptual.

> Por ejemplo, supongamos que se decide que las asociaciones en las que participa r4 y r8 estén guiadas únicamente por la alcanzabilidad. En este caso, el Sistema de Navegación deberá eliminar cinco Restricciones de Orden —vea figura 5.17 y la tabla 5.66—.

2. Si el autor/a amplía la navegabilidad se debe añadir una Restricción de Orden por cada ítem unido al concepto destino de la asociación.

> Imaginemos que se considera ampliar la navegabilidad de la asociación en la que participa r7, es decir, que esté guiada por la conectividad. Ahora, el Sistema de Navegación deberá añadir dos Restricciones de Orden más —vea figura 5.17 y tabla 5.66—.

Recuerde que el sistema genera una de estas reglas temporales —mediante el lenguaje **N**— por ítem asociado a un determinado concepto y por asociación conceptual que llega a este concepto —un ítem y una asociación conceptual nos permiten identificar cada una de las Restricciones de Orden—
.

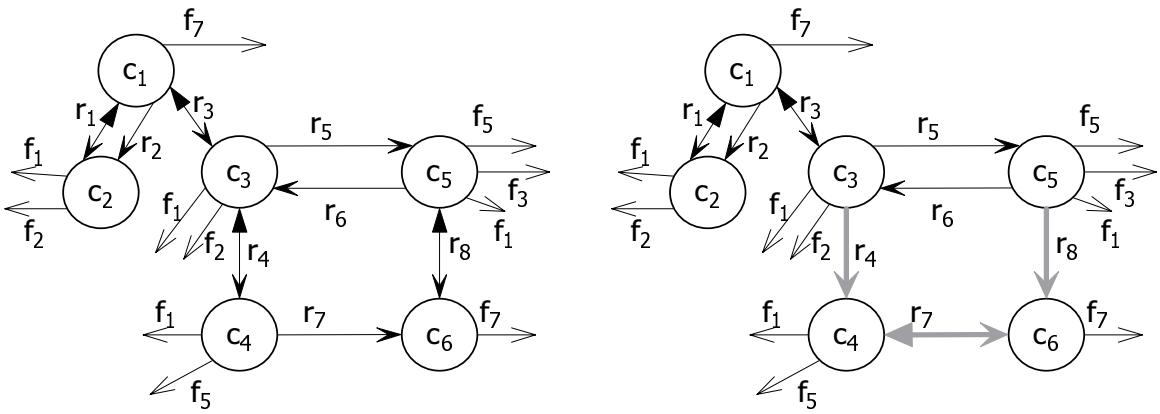


Figura 5.17 Cambiando las Restricciones de Navegabilidad en la EC_n

Restricción de orden explícita		
Fija	**Modificable**	**Fija**
c1.f7 ←	previo(c2.f1) or previo(c2.f2)	and previo(r1inv)
c1.f7 ←	previo(c3.f1) or previo(c3.f2)	and previo(r3inv)
c2.f1 ← c2.f2 ←	previo(c1.f7) previo(c1.f7)	and previo(r1) and previo(r1)
c2.f1 ← c2.f2 ←	previo(c1.f7) previo(c1.f7)	and previo(r2) and previo(r2)
c3.f1 ← c3.f2 ←	previo(c1.f7) previo(c1.f7)	and previo(r3) and previo(r3)
c3.f1 ← c3.f2 ←	previo(c4.f1) or previo(c4.f5) previo(c4.f1) or previo(c4.f5)	and previo(r4inv) and previo(r4inv)
c3.f1 ← c3.f2 ←	previo(c5.f1) or previo(c5.f3) or previo(c5.f5) previo(c5.f1) or previo(c5.f3) or previo(c5.f5)	and previo(r6) and previo(r6)
c4.f1 ← c4.f5 ←	previo(c3.f1) or previo(c3.f2) previo(c3.f1) or previo(c3.f2)	and previo(r4) and previo(r4)
c4.f1 ← **c4.f5 ←**	**previo(c6.f7)** **previo(c6.f7)**	**and previo(r7inv)** **and previo(r7inv)**
c5.f1 ← c5.f3 ← c5.f5 ←	previo(c3.f1) or previo(c3.f2) previo(c3.f1) or previo(c3.f2) previo(c3.f1) or previo(c3.f2)	and previo(r5) and previo(r5) and previo(r5)
c5.f1 ← c5.f3 ← c5.f5 ←	previo(c6.f7) previo(c6.f7) previo(c6.f7)	and previo(r8inv) and previo(r8inv) and previo(r8inv)
c6.f7 ←	previo(c5.f1) or previo(c5.f3) or previo(c5f.5)	and previo(r8)
c6.f7 ←	previo(c4.f1) or previo(c4.f5)	and previo(r7)

Tabla 5.66 Los cambios realizados en la figura 5.17 suponen la activación —en color negro— y la desactivación —en color gris— de restricciones de orden.

4.4.3.2 Cambiando las Restricciones de Orden

De igual modo, cuando el autor redefine, RT_o, una Restricción de Orden entonces el sistema debe comprobar que:

1. El conjunto de Restricciones de Orden especificadas es consistente. Esto está garantizado por las propias acciones de evolución de navegación que han de respetar las metarrestricciones escritas el lenguaje **MN**.

2. La navegación, en orden en el que se recorren los ítemes de información, ha cambiado. Por tanto, de nuevo se debe calcular por completo la red de Petri asociada. Como tenemos una nueva red de Petri también se deberá generar otra vez el Árbol de Navegación. Todos estos procesos son evidentemente propagaciones internas de un cambio en una Restricción de Orden que afectan a la RP y al árbol de navegación.

Resumiendo, tanto las Restricciones de Navegabilidad (1) como las de Orden (2) pueden cambiar. Cada vez que se modifican las primeras se redefine la Estructura Conceptual de Navegación (3). A partir de esta EC_n se generan las Restricciones de Orden (4). Por tanto, si se cambian las RT_{nb}, las RT_o han de evolucionar. Con las Restricciones de Orden se calcula la red de Petri (5). Luego, si el autor/a altera las Restricciones de Orden (2), se vuelve a construir la red de Petri (5) y, a partir de ella, se vuelve a generar el Árbol de Navegación (6).

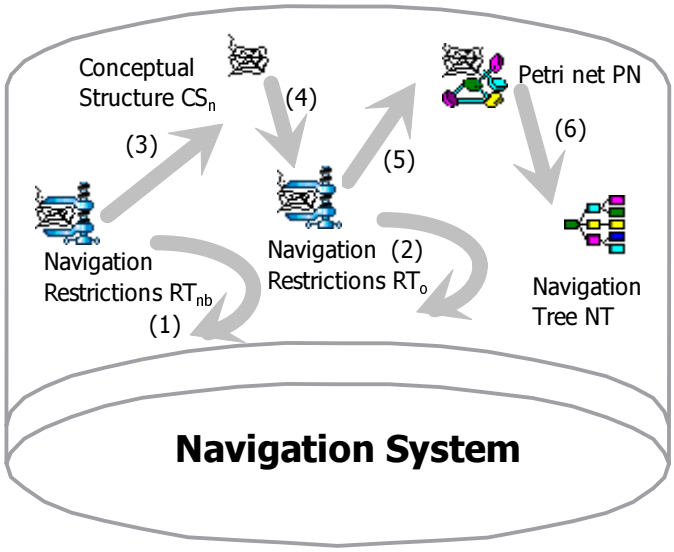


Figura 5.18 Propagación Interna del Cambio en el Sistema de Navegación.

4.5 La Red de Petri RP

A partir de las Restricciones de Orden RT_o especificadas, el Sistema de Navegación construye una Red de Petri **RP** [Perterson, 1981] [Silva, 1985] que determinará, de forma operacional, las posibilidades de navegación que el autor previamente ha establecido. El introducir en este punto un nuevo mecanismo formal para la representación de la navegación no es gratuito. Analicemos, pues, las razones que nos impulsan a transformar las RT_o, expresadas en lenguaje **N**, en una red de Petri:

a) **Semántica de Navegación.** Las Restricciones de Orden pretenden limitar la navegación conceptual que nos ofrece la Estructura Conceptual proporcionada por el Sistema de Navegación. Pero esta navegación debe determinar el orden —los posibles recorridos— en el que queremos mostrar los ítemes de información y no el esquema o dominio conceptual que representa la EC_n. De hecho, el lenguaje **N** aplica directamente sus predicados a ítemes y no a conceptos. Por tanto, es necesario obtener una estructura gráfica que muestre realmente la semántica de navegación.

b) **Navegación Gráfica.** La lógica temporal nos permite expresar esa semántica de navegación pero desgraciadamente no ofrece ningún mecanismo visual. Sin embargo, una **RP** puede ser observada, esto es, los ítemes y sus asociaciones conceptuales son elementos visibles. En consecuencia, una **RP** ofrece de manera directa, un potente mecanismo de ayuda para la navegación muy similar a las herramientas denominadas en la literatura hipermedia *Fish-eye views* [Furnas, 1986].

c) **Ejecución.** La **RP** es capaz de modelar el comportamiento dinámico de la navegación. Es decir, hace posible la ejecución de la navegación de forma gráfica e interactiva para que el autor pueda observar su comportamiento y comprobar si ha especificado la navegación que deseaba.

d) **Análisis de Propiedades.** La **RP** ofrece un formalismo operacional cuya ejecución hace posible el análisis de propiedades como por ejemplo: las rutas que son posibles, qué ítemes de información son mostrados en cada camino, si desde un determinado ítem se puede alcanzar a otro, etc.

e) **Representación del Estado.** La EC_n proporciona la totalidad de la información que puede llegar a ofrecer el sistema. En una red de Petri se puede ver el estado actual en el que se encuentra la ejecución de la navegación y, por tanto, el subconjunto de ítemes de información que en ese instante proporciona el sistema de acuerdo a la información que ya ha sido visitada —la navegación que ha realizado hasta ese momento el lector—.

f) **Concurrencia.** Con un mecanismo formal como el de la **RP** se puede modelar la concurrencia en la realización de recorridos en paralelo. En los navegadores que soporten múltiples ventanas es interesante poder representar:

   - la ejecución simultánea de varias acciones diferentes durante una misma sesión —por ejemplo, después de visitar un ítem, mostrar varios ítemes simultáneamente— y
   - la ejecución simultánea de una misma acción que proviene desde dos sesiones de navegación distintas.

g) **Árboles de Alcanzabilidad.** A partir de una Red de Petri se pueden construir los árboles de alcanzabilidad o de cobertura de la red. Esta posibilidad puede ser muy interesante en un Sistema Hipermedia pues los actuales navegadores no son capaces de ofrecer una versión imprimible del documento completo, sólo página a página. Cada uno de estos árboles constituye un posible recorrido por el dominio de información, esto es, un documento que el usuario puede leer secuencialmente. Por otro lado, el árbol de alcanzabilidad constituye en sí mismo una potente herramienta de análisis.

Stotts y Furuta [Stotts, 1998] también hacen uso de este formalismo para representar los posibles caminos que se pueden seguir en un hiperdocumento. No obstante, su propuesta tiene un enfoque diferente:

- Parten directamente de la red de Petri y no de una representación en lenguaje temporal.
- No trabajan con la red de Petri para hacer las consultas sino sobre su grafo de cobertura.
- Utilizan la lógica temporal ramificada para escribir las preguntas sobre las propiedades del grafo.
- Cuando cambia el hiperdocumento se crea una nueva red. No ofrecen ningún mecanismo para hacer esa evolución ni para garantizar la consistencia del cambio.

4.5.1 Definición Formal de una Red de Petri, RP

Con el objetivo de clarificar el proceso de transcripción de las cláusulas en lenguaje **N** a una red de Petri, introducimos algunas definiciones y conceptos relacionados con dicho formalismo. Asimismo, adaptaremos esta red de Petri al proceso de navegación que queremos modelar estableciendo una correspondencia entre los elementos del hiperdocumento y los componentes de la **RP**.

4.5.1.1 Definiciones Básicas

Las redes de Petri son una herramienta matemática y gráfica que se utiliza para modelar sistemas. El análisis de una red de Petri revela información sobre la estructura y el comportamiento dinámico del sistema modelado y, a partir de él, se puede evaluar el sistema y sugerir mejoras o cambios.

Las redes de Petri permiten considerar los distintos estados por los que va pasando un sistema a lo largo de su historia. Con ellas se puede simular la concurrencia o el paralelismo entre los distintos componentes que forman un sistema, así como la sincronización de estos. Además, su propia estructura define secuencias o caminos posibles entre distintos eventos y una relación de orden parcial entre estos.

Una ***estructura de red de Petri*** [Peterson, 1981] RP se define mediante la cuadrupla : RP= (P, T, I, O) donde:

- $P = \{p_1, p_2, \ldots p_n\}$ es un conjunto finito de lugares,
- $T = \{t_1, t_2, \ldots t_m\}$ es un conjunto finito de transiciones,
- $P \cap T = \emptyset$; es decir, lugares y transiciones son conjuntos disjuntos,
- $I: T \rightarrow P^{\infty}$ es una función de entrada que hace corresponder una transición con una colección de lugares, esto es, los lugares que llegan a la transición,
- $O: T \rightarrow P^{\infty}$ es una función de salida que hace corresponder una transición con una colección de lugares, esto es, los lugares que parten de la transición.

Toda estructura de red de Petri se puede representar gráficamente mediante un grafo bipartito dirigido, G = (V, A), donde:

- $V = \{v_1, v_2, \ldots v_s\}$ es un conjunto de vértices y
- $A = \{a_1, a_2, \ldots a_r\}$ es un conjunto de arcos dirigidos, $a_i = (v_j, v_k)$, con $v_j, v_k \in V$
- El conjunto V se puede particionar en dos conjuntos disjuntos P y T tales que $V = P \cup T$, $P \cap T = \emptyset$, y para cada arco dirigido, $a_i \in A$, si $a_i = (v_j, v_k)$, entonces $v_j \in P$ y $v_k \in T$ o viceversa.

Debido a que una estructura de RP está formada por ***lugares*** y ***transiciones***, el grafo debe contar con dos tipos de nodos: con un círculo se modela un lugar y con una barra o rectángulo una transición. Para conectar los lugares con las transiciones se utilizan arcos dirigidos. Un arco dirigido que vaya desde un lugar hacia una transición representa un ***lugar de entrada*** para la transición. De manera similar, un ***lugar de salida*** se figura con un arco dirigido entre una transición y un lugar. Estos arcos se suelen representar mediante flechas. Existe una equivalencia entre las dos representaciones de una red de Petri, esto es, entre la estructura RP= (P, T, I, O) y el grafo G = (V, A).

La ejecución de una red de Petri está controlada por el número y distribución de las ***marcas*** en la misma. Las marcas residen en los lugares y determinan si se disparan o no las transiciones. Una transición se ejecuta borrando las marcas de sus lugares de entrada y creando nuevas marcas en sus lugares de salida. Una transición se puede disparar si está ***habilitada***. Para que una transición esté habilitada, cada uno de los lugares de entrada tiene que tener al menos tantos marcas como arcos haya desde el lugar a la transición. En el caso de que una transición tenga algún ***arco inhibidor*** —en este caso, la punta del arco termina en un punto negro—, una transición estará habilitada cuando no tenga ninguna marca en sus lugares de entrada.

El ***estado*** de una RP se define por su ***marcado***. En el grafo de la red de Petri las marcas se representan con puntos dentro de los círculos. El disparo de una transición representa un cambio en el estado de una red de Petri por medio de un cambio en el marcado de la red. El cambio en el estado causado por el disparo de una transición se define mediante la función de cambio d llamada función del ***estado-siguiente***. Cuando se aplica a un marcado —estado— μ y a una transición t_j, esta función produce un nuevo marcado o estado μ', resultante de haber disparado t_j en el marcado μ. Puesto que t_j sólo puede dispararse si está habilitada, $\delta(\mu, t_j)$ está indefinida si t_j no está habilitada para ese marcado. Si está habilitada, entonces $\delta(\mu, t_j) = \mu$' donde μ' es el nuevo marcado que resulta de quitar las marcas de las entradas de t_j y añadirlos a las salidas de t_j. Para una RP, C = (P, T, I, O) con marcado μ, un marcado μ' es inmediatamente alcanzable desde μ si existe una transición $t_j \in T$ tal que $\delta(\mu, t_j) = \mu$'.

4.5.1.2 Modelado de la Navegación mediante una RP

Tanto el funcionamiento —la navegación— como la estructura del Sistema de Navegación puede especificarse mediante una red de Petri. El funcionamiento del SN viene dado por la ejecución de la red. Su estructura, esto es, las acciones que se pueden realizar —visitar ítemes y seguir relaciones conceptuales— así como las precondiciones para alcanzar un determinado ítem, se pueden constatar tan sólo con observar el grafo. La red puede ser analizada para estudiar las propiedades de la navegación que se está modelando. El autor puede comprobar si la navegación que ofrece la red de Petri se corresponde con el planteamiento que se había hecho.

La acción ***visitar***(ítem) se modela mediante un lugar en la red. Este lugar estará etiquetado con el concepto al que está asociado el ítem y la relación funcional que los une. La marca de un lugar representa que el ítem ha sido visitado. Las acciones, ***seguir***(relación-conceptual), se representan mediante transiciones.

Sistema de Navegación	Red de Petri
Estructura del SN	Grafo de la red
Navegación o funcionamiento del SN	Ejecución de la red
Estudio de las propiedades	Análisis de la red
visitar(ítem)	Lugar
seguir(relación-conceptual)	Transición
Evento: Alcanzar o visitar un ítem	marca
Realizar un acción	Disparo una transición
Acciones ***visitar***(ítem) del cuerpo de la cláusula	Lugares de entrada de la transición
Acción ***visitar***(ítem) de la cabeza de la cláusula	Lugar de salida

Tabla 5.67 Modelado de la Navegación mediante una **RP**.

La relación que hay entre la acción de la cabeza —***visitar***(ítem)— y su precondición —el cuerpo de la cláusula— se modela mediante arcos que van y parten de la transición —relación-conceptual— que los liga. Los lugares de entrada de una transición —relación-conceptual— son las acciones ***visitar***(ítem) que forman parte de la precondición una determinada acción que se representa mediante un lugar de salida. Para simplificar la construcción de la red a partir de las Restricciones de Orden RT_o, sólo habrá uno o dos lugares de entrada para una determinada transición y un lugar de salida. Si en una precondición intervienen más de dos acciones, se utilizarán transiciones y lugares intermedios entre los lugares que representan estos y la transición que los une con su acción. Es decir, la precondición se modela mediante una subred. En la tabla 5.67 se resume el modelado de la Navegación mediante una RP.

4.5.2 Transformación de Cláusulas en lenguaje **N** a RP

Dado que las Restricciones de Orden están expresadas en lógica temporal —en el lenguaje **N**— necesitamos de algún mecanismo que de manera automática nos transforme las anteriores restricciones en una red de Petri. Este proceso de traducción constará de tres pasos:

1) Descomposición de las cláusulas complejas en cláusulas simples.
2) Modelar cada una de las posibles cláusulas simples que nos puedan aparecer en las restricciones con una red de Petri.
3) Componer las subredes de Petri de manera que la red de Petri resultante sea equivalente a la cláusula compleja que se pretende modelar.

El método conversión que aquí se plantea es una adaptación y simplificación del propuesto por [Rodríguez-Fórtiz, 2000a] para formalización de software evolutivo. En este trabajo se hace una exposición formal y completa de dicho algoritmo de transformación así como la demostración de que ambos formalismos —lenguaje **N** y RP— son equivalentes semánticamente. Por tanto, a continuación nos vamos a limitar a describir de manera intuitiva este algoritmo de transformación y mostrar cómo mediante la RP es posible modelar la navegación en un sistema hipermedia.

4.5.2.1 Descomposición en Cláusulas

Las Restricciones de Orden expresadas mediante cláusulas de la lógica temporal del tipo:

Ítem ← *fórmula—temporal* **and previo**(*relación—conceptual*)

se pueden descomponer en fórmulas simples de manera que *fórmula-temporal* únicamente esté formada por cláusulas simples, esto es, un operador temporal —previo(*ítem*), antes(*ítem*) ó (*ítem* desde *ítem*)— o una combinación simple de estos operadores —(op-temp(ítem) and op-temp(ítem)) ó (op-temp(ítem) or op-temp(ítem))—.

Por ejemplo, la Restricción de Orden:

c1.f7 ← ((**previo**(c3.f1) **or** (**previo**(c3.f2)) **and** (**not antes**(c5.f5) **and antes**(c4.f1))) **and previo(r3inv)**

se puede descomponer en las siguientes Restricciones de Orden haciendo reemplazos:

c1.f7 ← (X and Y) **and previo(r3inv)**

X ← (**previo**(c3.f1) **or previo**(c3.f2)) **and previo(r3inv)**

Y ← (**not antes**(c5.f5) **and antes**(c4.f1)) **and previo(r3inv)**

El proceso de descomposición se puede realizar de manera inversa, esto es, dado un conjunto de cláusulas que comparten predicados y que coinciden en la relación conceptual que dispara la navegación hacia el ítem de la cabeza, se puede obtener una cláusula como resultado de la composición de las anteriores. Para ello, basta con deshacer los reemplazos realizados durante el proceso de composición.

4.5.2.2 Transformación de cláusulas simples en Red de Petri

Por tanto, el primer objetivo es descomponer el cuerpo de la restricción en cláusulas simples. Dado que en este cuerpo únicamente pueden aparecer predicados u operadores lógicos sobre ítemes ligados mediante los operadores lógicos **and** u **or**, éstas serán las cláusulas simples de las que se va a obtener la red de Petri equivalente.

A continuación, se realiza la transformación para cada tipo de cláusula simple teniendo en cuenta el operador o la fórmula temporal del cuerpo.

Operador and

Cada operador temporal sobre un ítem de información, esto es, un predicado de la cláusula, se representa mediante un lugar. Cada operador temporal sobre una relación conceptual se representa con una transición para asociar los predicados temporales del cuerpo con la cabeza. En consecuencia, tenemos dos lugares de entrada para esa transición, uno para cada predicado temporal del cuerpo y otro lugar de salida para el ítem de la cabeza.

Por ejemplo —figura 5.19—,

c.rf ← (**op_temp**(c1.rf1) and **op_temp**(c1.rf2)) **and previo**(rc)

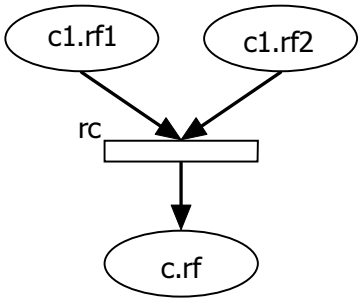


Figura 5.19 Red de Petri para el operador lógico **and**.

Operador or

Nuevamente cada operador temporal sobre un ítem de información o predicado de la cláusula, se representa con un lugar. Cada operador temporal sobre una relación conceptual se representa con dos transiciones para que podamos asociar los predicados temporales del cuerpo con la cabeza. Ahora, cada transición tendrá un lugar de entrada —cada predicado del temporal sobre uno de los ítemes— y, sin embargo, el mismo lugar de salida, el que se corresponde con el ítem de la cabeza.

Por ejemplo —figura 5.20—,

c.rf ← (**op_temp**(c1.rf1) **or op_temp**(c1.rf2)) **and previo**(rc)

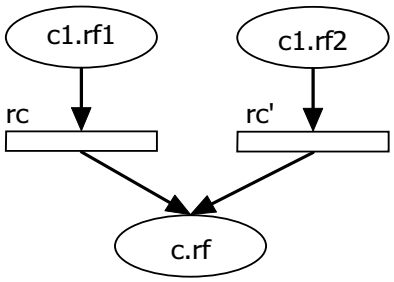


Figura 5.20 Red de Petri para el operador lógico **or**.

Operador desde

En este caso —figura 5.21— se asocia un lugar a cada acción ***visitar*** sobre ítemes del cuerpo y se crean dos transiciones para representar el operador previo aplicado a la relación conceptual. A una de estas transiciones se le llama *transición de la parte positiva* porque recibe un arco de cada lugar. A la otra transición se la denomina *transición de la parte negativa* porque, aunque también recibe un arco de cada lugar, el arco que viene del lugar que se corresponde con el predicado que hay a la derecha del **since** es un arco inhibidor. De la transición negativa sale un arco a un lugar auxiliar que se crea para almacenar las marcas no útiles en la evaluación del operador **desde** y que evitan alterar el orden que éste operador impone.

c.rf ← (c1.rf1 **desde** c1.rf2) **and previo**(rc)

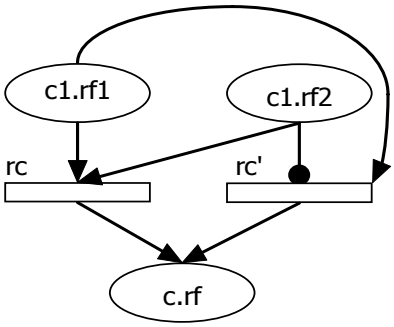


Figura 5.21 Red de Petri para el operador temporal **desde**.

Operador previo/antes

El operador **previo** y **antes** son semánticamente bastante parecidos. El operador **previo** únicamente se puede aplicar a aquellos ítemes que son predecesores en un solo paso del ítem de la cabeza y a la relación conceptual que nos permite llegar al ítem de la cabeza. Este operador sólo puede ser usado por el sistema.

El autor únicamente puede añadir más condiciones con el operador **antes** que podrá ser aplicado a cualquier ítem que sea predecesor del ítem de la cabeza en dos o más pasos. Estas comprobaciones se harán cuando el autor reescriba las Restricciones de Orden, garantizándose que antes de su traducción a red de Petri son correctas semánticamente.

Sin embargo, su transformación a red de Petri es exactamente la misma —figura 5.22—, se crea un lugar para el ítem del cuerpo y una transición para la relación conceptual que lo acompaña, uniéndolo mediante ésta al lugar que se crea para el ítem de la cabeza. Es lógico que así sea pues lo único que hay que verificar es que se cumple la condición impuesta en el cuerpo, esto es, que se han visitado ya los ítemes a los que se aplican los operadores **previo** y **antes**.

c.rf ← (**previo**(c1.rf1)) **and previo**(rc)

c.rf ← (**antes**(c1.rf1)) **and previo**(rc)

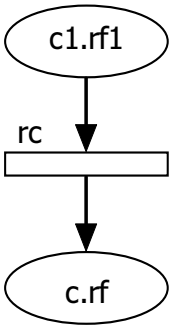


Figura 5.22 Red de Petri para los operadores temporales **previo/antes**.

Operador not

Tal y como se ha descrito en la sintaxis y semántica del lenguaje **N**, el operador **antes** es el único que de forma simple puede estar precedido por el operador lógico **not**. En el caso de que un operador not preceda a un predicado de la cláusula, sobre el que se ha aplicado el operador temporal antes, se creará un lugar para el predicado que será unido mediante un arco inhibidor al lugar que representa al ítem de la cabeza a través de la transición —que representa la relación conceptual que los une—.

Por ejemplo —figura 5.23—,

c.rf ← (**not antes**(c1.rf1)) **and previo**(rc)

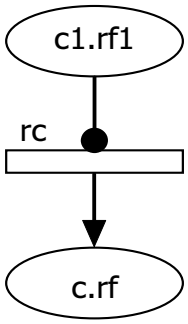


Figura 5.23 Red de Petri para el operador lógico **not**.

4.5.2.3 Composición de Cláusulas Complejas

Anteriormente se ha visto cómo una cláusula se puede descomponer en otras más simples, hasta obtener cláusulas que sólo contengan en su cuerpo una fórmula simple: un operador temporal sobre un ítem, un operador temporal aplicado a un ítem negado o dos operadores temporales unidos con un operador lógico. A cada una de estas fórmulas se le puede asociar una red de Petri obteniendo un conjunto de redes de Petri. Al igual que se pueden componer las cláusulas para obtener una única, se pueden componer las redes de Petri para obtener una sola red de Petri.

En el proceso de composición se usan tres conceptos que pasamos a describir:

- ***Lugar hoja*** de una red de Petri: lugar que sólo recibe arcos de entrada y no tiene arcos de salida.
- ***Lugar raíz*** de una red de Petri: lugar que sólo tiene arcos de salida y no de entrada.
- ***Lugar intermedio*** de una red de Petri: lugar con arcos de entrada y salida, creado de forma artificial para componer redes y que no representa a un predicado sino a una fórmula que forma parte de otra fórmula.
- ***Lugar copia***. En algunos casos, se necesitará expresar que una acción interviene de forma diferente en más de una transición de una red. En este caso, se creará un original para la acción y tantos lugares copia como transiciones tengan como entrada esa acción. El lugar copia estará ligado a sus copias mediante una transición, de tal forma que cada vez que llegue una marca al lugar original, ésta se propagará a todos los lugares copia.

A la hora de componer dos redes para formar una sola, los lugares hoja de éstas pasan a ser lugares intermedios de la nueva red. En el proceso de composición pueden darse dos casos:

a) La Restricción de Orden contiene en el cuerpo una fórmula simple. Por tanto, no habrá descomposición ni composición.

b) La Restricción de Orden rt_{oh} contiene en su cuerpo una fórmula compleja. La restricción puede ser descompuesta en dos Restricciones de Orden rt_{of} y rt_{og} —con cabezas f y g respectivamente— que contengan en su cuerpo fórmulas simples. Para cada una de estas restricciones se obtiene su red de Petri. La *fórmula-temporal* de los cuerpos de estas reglas temporales simples únicamente está formada por un operador temporal —**previo**(*ítem*), **antes**(*ítem*) ó (*ítem* **desde** *ítem*)— o una combinación simple de estos operadores —(**op-temp**(ítem) **and op-temp**(ítem)) ó (**op-temp**(ítem) **or op-temp**(ítem))—. La traducción a red de Petri es inmediata tal y como se ha descrito en el apartado 4.4.2.

Se supone que estas redes de Petri no comparten lugares. En consecuencia, para la composición de las mismas se crea un lugar para h —cabeza de la composición—, para construir el cuerpo se tiene en cuenta el operador lógico que aparece en el cuerpo de la restricción, enlazando los lugares hoja de RP(f) y RP(g) y creando nuevas transiciones dependiendo del operador.

Por ejemplo, para la Restricción de Orden anterior:

c1.f7 ← ((**previo**(c3.f1) **or** (**previo**(c3.f2)) **and** (**not antes**(c5.f5) **and antes**(c4.f1))) **and previo(r3inv)**

se puede calcular la red de Petri a cada una de las Restricciones de Orden simples:

c1.f7 ← (X **and** Y) **and previo(r3inv)** —figura 5.24—

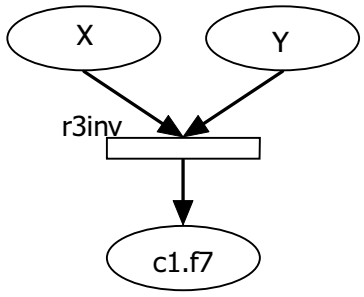


Figura 5.24 Red de Petri para la primera cláusula simple del ejemplo.

X ← (previo(c3.f1) or previo(c3.f2)) **and previo(r3inv)** —figura 5.25—

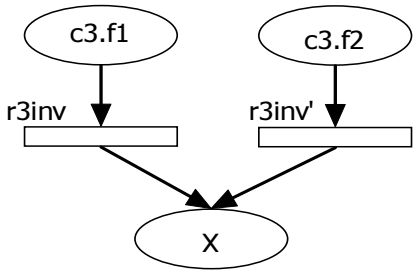


Figura 5.25 Red de Petri para la segunda cláusula simple del ejemplo.

Y ← (**not antes**(c5.f5) **and antes**(c4.f1)) **and previo(r3inv)** —figura 5.26—

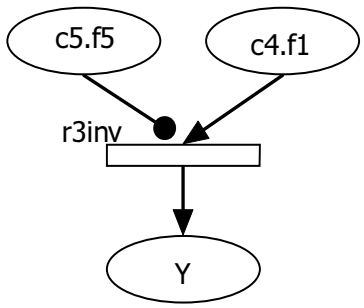


Figura 5.26 Red de Petri para la tercera cláusula simple del ejemplo.

Las redes de Petri anteriores se pueden componer obteniendo la siguente red de Petri —figura 5.27—.

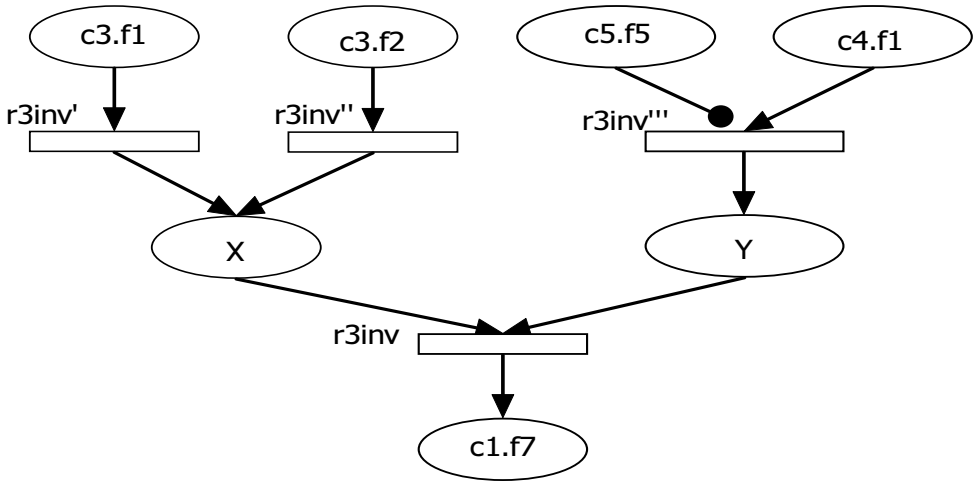


Figura 5.27 Red de Petri compuesta por las redes de las figuras 5.24, 5.25 y 5.26.

4.5.2.4 Guías para Componer Redes de Petri

La composición anterior supone que al combinar dos redes de Petri éstas no coinciden en sus lugares, esto es, son disjuntos. Sin embargo, podemos encontrarnos con Restricciones de Orden en cuyas cláusulas —en los cuerpos— aparezcan predicados sobre los mismos ítemes de información. Por tanto, es necesario establecer algunas reglas más de composición para estos casos:

a) Para cada lugar en común que no sea un lugar copia —definición, en el apartado 4.5.2.3—, se crea un lugar copia en la subred de cada fórmula y se deja sólo un lugar original que se unirá a esos lugares copia mediante una transición. Se nombran los lugares copia con el mismo nombre del lugar original, seguidos por un número correlativo.

b) Para cada lugar en común que sea un lugar copia se renombran para que tengan números correlativos y enlazarlos todos con el lugar original mediante una única transición. En el caso en el que haya un lugar original en la red de la fórmula, se elimina uno de ellos y su transición de salida, quedando el otro lugar enlazado con todos los lugares copia.

c) Para combinar redes con varias cláusulas complejas, se duplican los lugares si se da la circunstancia de que los predicados coinciden. Para simplificar la composición, cada vez que se crea un lugar para una acción de una cláusula simple se creará automáticamente un lugar copia y será a éste al que irán ligadas las transiciones oportunas de la cláusula y no al lugar original. De esta forma siempre estaremos en el segundo caso que implica renombrar los lugares copia y enlazarlos a un único lugar original. Con esta posibilidad facilitaremos también el hecho de que haya referencias a una misma acción en la misma cláusula simple ya que directamente se crean dos copias en lugar de una.

Las acciones, en nuestro caso *visitar ítemes* durante el proceso de navegación, van a estar relacionadas. Indudablemente, haber recorrido determinados ítemes hará posible la visita de uno o varios ítemes. Dicho de otro modo, una acción, ***visitar***(ítem), suele formar parte de una o más precondiciones, al tiempo que ella misma también tiene su propia precondición. El objetivo último que

se pretende en el Sistema de Navegación es modelar y observar, precisamente, las relaciones entre todas esas acciones. Por tanto, el Sistema de Navegación generará una única red de Petri con todas las subredes que se han obtenido al traducir y componer las correspondientes Restricciones de Orden. En el caso de contar con más de dos redes, las guías de composición se aplicarán progresivamente, componiendo en cada paso la red obtenida con una de las redes que queden por integrar.

Por ejemplo, supongamos que se tienen las siguientes Restricciones de Orden:

(1) c1.f7 ← ((**previo**(c3.f1) **or** (**previo**(c3.f2)) **and** (**not antes**(c5.f5) **and antes**(c4.f1))) **and previo(r3inv)**

(2) c3.f1 ← ((**previo**(c4.f1) **or** (**previo**(c4.f5)) **and previo(r4inv)**

(3) c5.f5 ← **previo**(c6.f7) **and previo(r8inv)**

Las reglas (2) y (3) no son disjuntas con la regla (1), tienen predicados sobre los mismos ítemes de información. Por tanto, se deben crear lugares copias para obtener su composición —figura 5.28—.

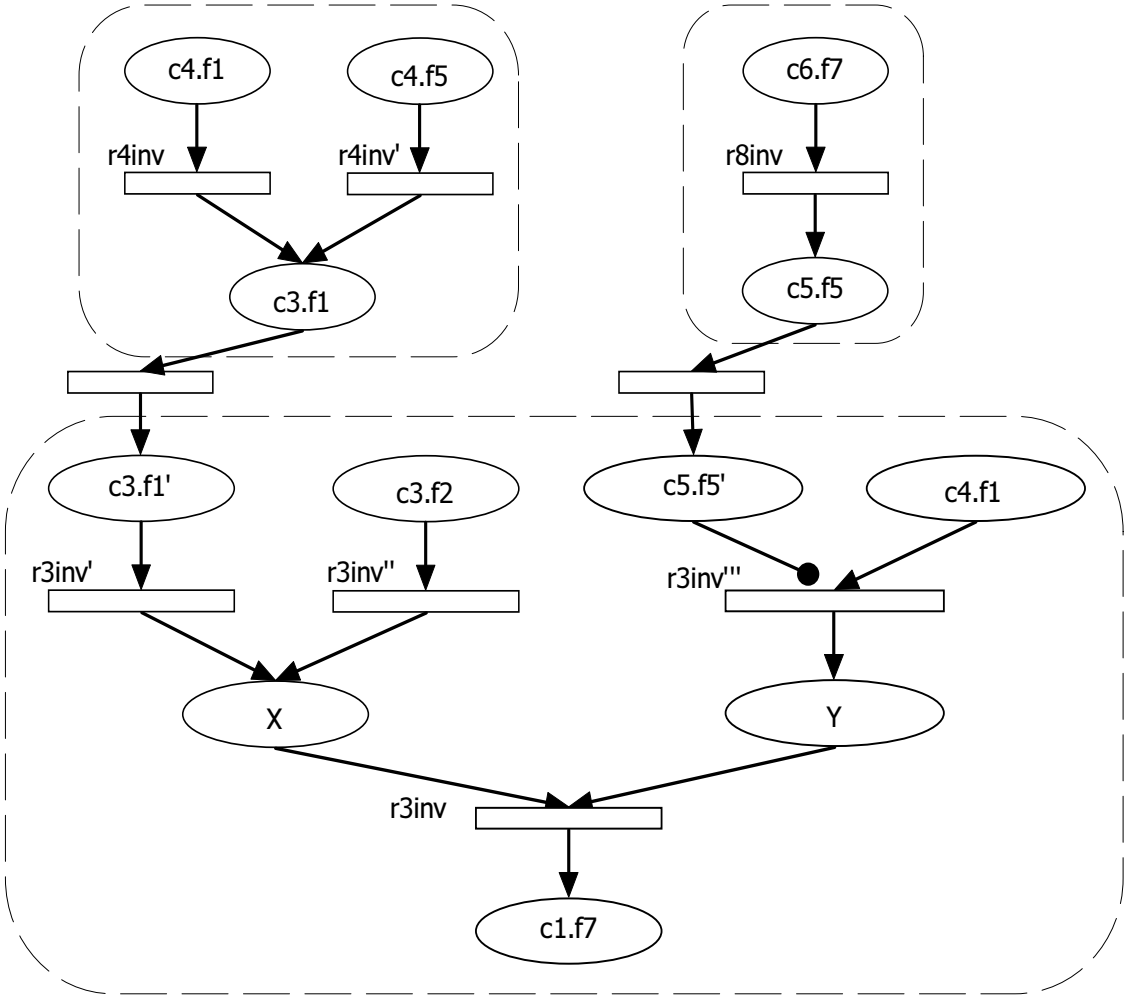


Figura 5.28 Red de Petri compuesta por redes que tienen lugares en común.

4.6 Árboles de Alcanzabilidad o de Navegación

Los árboles de alcanzabilidad o grafos de cobertura son una de las técnicas de análisis más utilizadas en la teoría sobre redes de Petri. Estos árboles permiten estudiar más fácilmente el comportamiento del sistema modelado y comprobar su validez. A esta tradicional ventaja se le ha encontrado otra, de orden práctico, en la navegación de un sistema hipermedia. Los árboles de alcanzabilidad nos permiten calcular todos los recorridos secuenciales que se pueden establecer a través del dominio de información seleccionado. Dos aplicaciones directas de esta posibilidad tal y como luego se describirá más detalladamente serán:

a) navegación indizada o mediante *tour*, a los que denominamos Árboles de Navegación y,

b) la posibilidad de generar documentos secuenciales directamente imprimibles.

4.6.1 Definiciones Previas

Un árbol de alcanzabilidad [Peterson, 1981] representa el conjunto de lugares que son alcanzables para una determinada red de Petri. En este árbol, los nodos son los marcados de la red y los arcos las transiciones que se disparan entre cada marcado. El nodo raíz es el marcado inicial de la red.

Por ejemplo, supongamos que el marcado inicial de la red de Petri de la figura 5.29 es (1, 0, 0). Es este caso, dos transiciones de la red están habilitadas: t_1 y t_2. Por tanto, se puede construir dos nuevos nodos en el árbol de alcanzabilidad con el marcado que resulte de disparar ambas transiciones. De nuevo, se puede calcular todos los marcados que son alcanzables desde esos dos nuevos estados. En la figura 5.30 se muestra el árbol de alcanzabilidad que se puede construir después de haberse repetido este proceso tres veces.

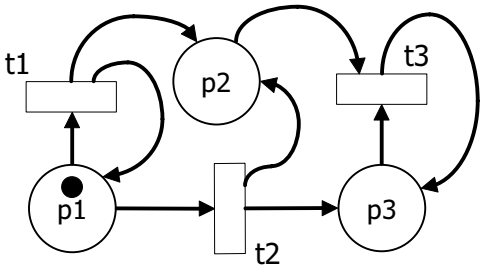


Figura 5.29 Ejemplo de red de Petri con un marcado inicial.

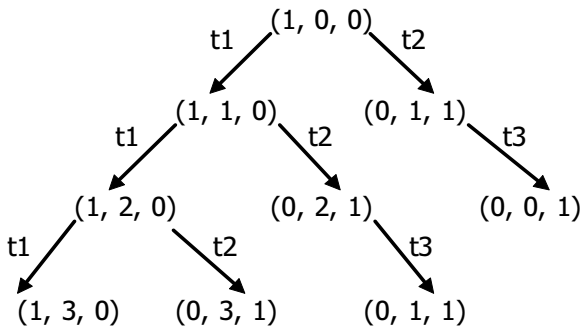


Figura 5.30 Árbol de Alcanzabilidad de la red de Petri de la figura 5.29 después de tres *disparos*.

No obstante, el árbol de alcanzabilidad puede ser muy grande e incluso infinito. Este problema se planteará cuando en la red de Petri existan ciclos entre los lugares. Para resolver esta situación se detectan los marcados duplicados o que tienen una componente periódica con otros marcados debido a que estos son los que conducen a un árbol infinito. A continuación se construye el árbol reducido, esto es, un árbol en el que se han eliminado los nodos duplicados y dando un valor ω —representa un número arbitrariamente largo— a la componente con posible valor infinito.

En consecuencia, se puede plantear un algoritmo que calcula automáticamente este árbol de alcanzabilidad. A cada nodo i del árbol, se le asociará un marcado extendido μ[i], en el que el número de marcas en un determinado lugar i puede ser un número no negativo o el valor ω. Asimismo, cada nodo se puede clasificar como nodo *frontera*, *terminal*, *duplicado* o *interno*. Los nodos frontera serán aquellos que son susceptibles de ser procesados por el algoritmo, convirtiéndolos en terminales, duplicados o interiores. El algoritmo comienza definiendo un marcado inicial que será el nodo raíz del árbol, en un primer momento, un nodo frontera. Mientras queden nodos frontera, el algoritmo seguirá procesándolos.

Para cada nodo x frontera:

1. Si existe otro nodo y en el árbol que no es nodo frontera y que tiene el mismo marcado asociado, $\mu[x] = \mu[y]$, entonces x es un nodo duplicado.

2. Si la transición no está habilitada por el marcado $\mu[x]$, esto es, la función del *estado-siguiente*, $\delta(\mu[x], t_j)$, es indefinida para todo $t_j \in T$, entonces x es un nodo terminal.

3. Para todas las transiciones $t_j \in T$ que esté habilitada por el marcado $\mu[x]$, esto es, $\delta(\mu[x], t_j)$ está definida, se crea un nuevo nodo z en el árbol de alcanzabilidad. El marcado asociado $\mu[z]$, a este nuevo nodo z para cada lugar p_i es:

   (a) Si $\mu[x]_i = \omega$, entonces $\mu[z]_i = \omega$.

   (b) Si existe un nodo y en el camino que va desde el nodo raíz al nodo x tal que $\mu[y] < \delta(\mu[x], t_j)$ y $\mu[y]_i < \delta(\mu[x], t_j)_i$, entonces $\mu[z]_i = \omega$.

   (c) En otro caso, $\mu[z]_i = \delta(\mu[x], t_j)_i$.

   Se dibuja un arco, etiquetado con t_j, que va desde el nodo x al nodo z. El nodo x se redefine como nodo interior y el nodo z se convierte en nodo frontera.

El algoritmo termina cuando todos los nodos se hayan clasificado como terminales, duplicados o interiores. Una propiedad muy importante de este algoritmo es el hecho de que acaba —la demostración la puede encontrar en [Peterson, 1981, página 97].

En la figura 5.31 se muestra el árbol de alcanzabilidad que calcula este algoritmo al ser aplicado a la red de Petri del ejemplo de la figura 5.29.

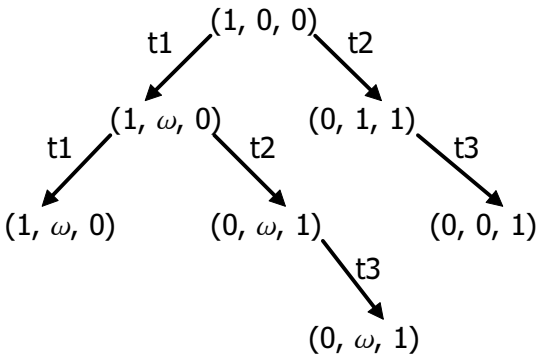


Figura 5.31 Árbol de Alcanzabilidad de la red de Petri de la figura 5.29.

4.6.2 Aplicaciones del Árbol de Alcanzabilidad o Navegación

El cálculo del árbol de alcanzabilidad dota al Sistema de Navegación de una nueva funcionalidad de la que puede beneficiarse tanto al lector como al autor. A partir de la red de Petri se pueden generar todos los posibles recorridos —ramas del árbol— que admite su ejecución —navegación—.

Por tanto, el lector dispone de una navegación indizada, ordenada a través de los ítemes que se seleccionaron en el Sistema de Presentación. Además, es posible generar para cada uno de estos caminos un documento secuencial que sea legible —de acuerdo a los criterios establecidos por el autor mediante las RT_o— y, directamente, imprimible.

Al autor le va a proporcionar una herramienta muy útil a la hora de analizar la red de Petri. En última instancia, le permitirá comprobar si la navegación que ha diseñado es la deseada, esto es, si la ejecución de la red de Petri modela el comportamiento que había proyectado. Veamos detalladamente qué tipo de comprobaciones se pueden realizar mediante el árbol de alcanzabilidad.

Seguridad y Limitación. Una red de Petri es segura si el número de marcas de cada lugar no puede ser superior a uno. De igual forma, una RP estará limitada si el número de marcas de cada lugar no puede exceder de un número k. Ambas propiedades se pueden comprobar fácilmente mediante el árbol de alcanzabilidad. La red de Petri estará limitada si y sólo si el símbolo w nunca aparece en dicho árbol. En este caso, también se puede contar y verificar el número de marcas de cada lugar para cada uno de los marcados posibles. Con esta comprobación el autor podrá saber si un ítem se puede visitar un número de veces determinado.

Conservación. Una red de Petri es conservativa si no se pierden o ganan marcas, esto es, el número de marcas en la red es siempre constante. En un árbol de alcanzabilidad se pueden sumar las marcas de cada marcado y comprobar que para todo el árbol este valor no cambia. Si aparece el símbolo ω pero la red está limitada, se puede dar inicialmente un valor a ω y ponderarlo con ese valor a la hora de hacer la suma de las marcas. De esta manera, el desarrollador podrá verificar si existen o no ciclos en su hiperdocumento.

Cobertura o alcanzabilidad. Esta será la propiedad más valorada por el autor. Mediante el árbol de navegación se puede estudiar si a partir de un marcado m existe una secuencia de transiciones que permita llegar a un estado μ', esto es, si μ' es alcanzable desde μ o que μ cubre un marcado μ'. De manera directa se puede comprobar con tan sólo inspeccionar el árbol de alcanzabilidad. En el caso de que aparezca el símbolo ω simplemente se estará

indicando que la rama es más larga y que cíclicamente deberán dispararse transiciones hasta llegar al marcado deseado. El único problema es que no podemos conocer, a priori, cuántas veces debe ejecutar una transición de forma repetida.

No todo son ventajas, el árbol de alcanzabilidad presenta algunos problemas. Al trabajar con marcados completos y con transiciones entre marcados no son posibles algunas consultas y comprobaciones como: saber qué lugares son alcanzables a partir de un lugar concreto o, dado un lugar, qué transiciones hay que disparar para llegar a otro lugar distante de la red. Pero este tipo de consultas se resolverán ejecutando directamente la red de Petri.

Para finalizar, se muestra un ejemplo completo que parte de una serie de Restricciones de Orden:

c6.f7 ← (**previo**(c4.f1) **and** (**previo**(c4.f5)) **and previo(r7)**

c4.f1 ← (**previo**(c6.f7) **and previo(r7inv)**

c4.f5 ← (**previo**(c6.f7) **and** (**antes**(c4.f1) **and antes**(c4.f5)) **previo(r7inv)**

A partir de estas reglas se construye su red de Petri compuesta —figura 5.32—, se le da un marcado inicial (1, 1, 0, 0, 0, 0, 0) y, se genera su árbol de navegación —figura 5.33—. Se han numerado los lugares con objeto de indicar su posición en el vector de marcado.

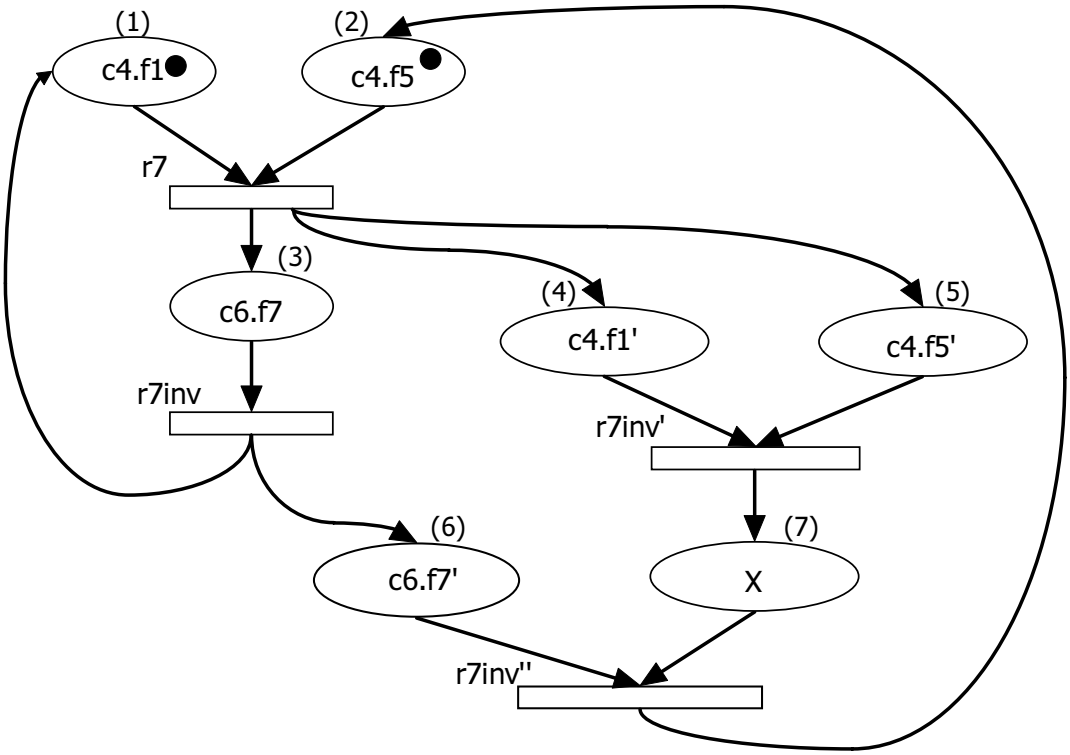


Figura 5.32 Red de Petri con marcado inicial (1, 1, 0, 0, 0, 0, 0).

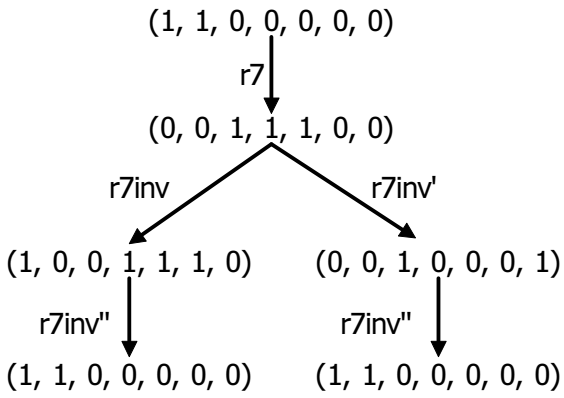


Figura 5.33 Árbol de Alcanzabilidad de la red de Petri de la figura 5.32.

Este árbol de Navegación está diciendo que si se ha visitado c4.f1 y c4.f5 entonces se puede llegar a c6.f7 disparando r7. Que si antes ocurrió esto, se puede regresar desde c6.f7 a esos conceptos —c4.f1 y c4.f5— mediante r7inv. Por último, que también se puede ir directamente desde c6.f7 a c4.f1 —pero no a c4.f5— mediante r7inv. Los posibles recorridos son, por tanto, los que muestra la figura 5.34.

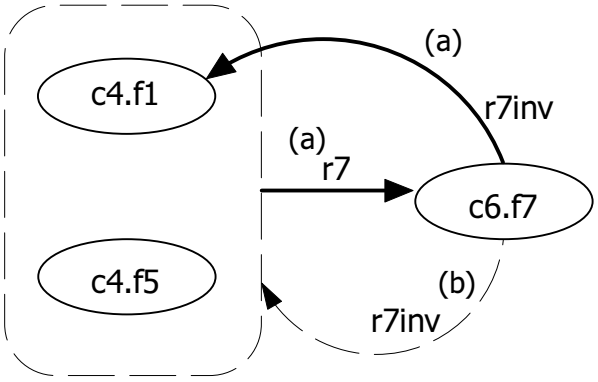


Figura 5.34 Posibles navegaciones para el lector de acuerdo al árbol de Navegación de la figura 5.33.

Subcapítulo 5.5

Consideraciones Generales

«Lo mismo es nuestra vida que una comedia; no se atiende a si es larga, sino a si la han representado bien. Concluye donde quieras, con tal de que pongas un buen final.»

Séneca.
Escritor hispanolatino (54 a.C.-39 d.C.).

«Toda ciencia se convierte en poesía, después de haberse convertido en filosofía.»

Novalis.
Poeta romántico alemán (1772-1801).

«El intelecto busca; pero es el corazón el que halla».

George Sand.
Novelista francesa (1804-1876).

Resumen

Con este apartado se acaba la exposición de este capítulo dedicado a la formalización del modelo SEM-HP. En él se describe la propagación externa del cambio, se justifican los formalismos empleados y, finalmente se pone de manifiesto la flexibilidad y versatilidad de una herramienta concebida bajo SEM-HP.

5 Consideraciones Generales 217
5.1 Propagación Externa del Cambio 217
5.1.1 SubSistema de Memorización: Propagación Externa del Cambio 217
5.1.2 SubSistema de Presentación: Propagación Externa del Cambio 218
5.1.3 Sistema de Navegación: Propagación Externa del Cambio 218
5.2 Prestaciones de la Formalización Usada 219
5.2.1 ESTRUCTURA y Formalismos 219
5.2.2 Evolución y Formalismos 220
5.3 Flexibilidad y Versatilidad de SEM-HP 221
5.4 Referencias 222

5 Consideraciones Generales

Una vez especificados formalmente cada uno de los Sistemas —del *Subsistema de Memorización* y *Presentación* y del *Sistema de Navegación*— de SEM-HP, se dedica este apartado a aspectos de colofón y reflexión que atañen a todo el Modelo tales como:

- la propagación externa del cambio, es decir, entre los distintos Subsistemas y Sistemas de SEM-HP.
- la cogitación y justificación de los formalismos empleados en SEM-HP, y
- la valoración de la funcionalidad de SEM-HP.

5.1 Propagación Externa del Cambio

En la descripción de cada una de las especificaciones formales de los Sistemas, se ha visto cómo las Acciones de Evolución nos permiten cambiar sus distintos componentes y cómo las Restricciones, en su papel de precondición, consiguen mantener la consistencia, no sólo del elemento cambiado, sino del resto de los componentes del Sistema garantizando su integridad. Esta situación no es suficiente, realizar una modificación en algún Sistema puede tener algún tipo de repercusión sobre el resto de los Sistemas, esto es, el Hipermedia debe también garantizar la propagación externa o evolución de los demás Sistemas que lo componen.

5.1.1 SubSistema de Memorización: Propagación Externa del Cambio

Desde el Subsistema de Memorización se deberá comprobar si las modificaciones realizadas en cada uno de sus elementos —la Estructrua Conceptual EC_m, el diccionario D y las Restricciones de Autor RT_a—tiene algún tipo de impacto sobre el resto de los Sistemas —el Subsistema de Presentación y el Sistema de Navegación—.

Dado que el Subsistema de Presentación selecciona un subconjunto de la Estructura Conceptual EC_m definida en el Subsistema de Memorización, esta subestructura no puede contener ningún concepto, relación conceptual, relación funcional o ítem que no esté en la EC_m. Esto significa que, estos objetos deben desaparecer en la EC_p del subsistema de presentación, si previamente han sido eliminados de EC_m.

Por otro lado, ese subdominio conceptual y de información por el que ha optado el autor debe respetar las restricciones relacionales definidas en el Subsistema de Memorización. Por tanto, si se redefine el conjunto de restricciones asociado a una relación conceptual, también habrá que comprobar, de nuevo, que el subgrafo del Subsistema de Presentación lo verifica. Para ello, se volverá aplicar el algoritmo 1 descrito en el apartado 2.5.2 de este capítulo —tabla 5.68— .

Algoritmo 1: Redefinición de las Restricciones relacionales en el SM
R_{ci} = Conjunto de relaciones conceptuales que no cumplen las Restricciones de Autor $\forall$ <c_o, r_{ci}, c_d> tal que $r_{ci} \in R_{ci}$ Borra-asociación-conceptual(<c_o, r_{ci}, c_d>, EC_m) R_{ci} = Conjunto de relaciones conceptuales que no cumplen las Restricciones de Autor

Tabla 5.68 Algoritmo 1 que redefine las Restricciones Relacionales Conceptuales.

Resumiendo, por un lado, el cambio de la Estructura Conceptual del Subsistema de Memorización EC_m desencadena la actualización de las subestructuras conceptuales del Subsistema de Presentación EC_p que se basen en ella. Indudablemente, si éstas cambian también tendrán que hacerlo las Estructuras Virtuales que se cimentan en estas presentaciones. Sin embargo, esto no se puede considerar un propagación externa sino interna.

Por otro lado, la redefinición de las Restricciones RT_a de autor en el Subsistema de Memorización supone cotejar de nuevo la validez de las subestructuras conceptuales seleccionadas. El único componente cuya actualización no provoca ningún tipo de propagación externa es el Diccionario.

5.1.2 SubSistema de Presentación: Propagación Externa del Cambio

Cambiar una presentación, esto es, ofrecer una nuevo subdominio conceptual y de información, redefinir EC_p, supone definir de nuevo y por completo sus posibilidades de navegación. El autor/a deberá volver a proporcionar las Restricciones de Navegación RT_n. Este cambio, no puede considerarse como una verdadera evolución ya que el creador/a está diseñando una nueva navegación. No obstante, dado que el proceso de construcción será definido de manera incremental, el sistema asistirá al autor recordándole que debe volver a especificar sus criterios de navegación.

El modificar las Estructuras Virtuales desde el Subsistema de Presentación no implica ningún tipo de repercusión, puesto que, la especificación de recorridos se hace siempre sobre la Subestructura Conceptual suministrada por el Subsistema de Presentación y sobre sus agrupaciones conceptuales virtuales.

5.1.3 Sistema de Navegación: Propagación Externa del Cambio

En el caso del *Sistema de Navegación* no cabe la posibilidad de realizar ningún tipo de propagación externa. El cambio de las Restricciones de Navegación no afecta a los elementos de ninguno de los Subsistemas del Sistema de Conocimiento. El Sistema de Navegación parte de un subdominio conceptual y de información que se le suministra desde Subsistema de Presentación, no existe transferencia de conocimiento en sentido inverso.

Seguramente, estas premisas cambiarán cuando se complete el modelo SEM-HP con la especificación conceptual y formal del Sistema de Aprendizaje. Presumiblemente, la actualización de este Sistema podría representar información de retroalimentación para el Sistema de Navegación e incluso para el Sistema de Conocimiento. Es lógico que así sea. El lector recorrerá la información de acuerdo a sus propios esquemas conceptuales. Estos criterios pueden o no coincidir con la concepción del autor que podrá verse rebatido y vencido por el uso real realizado por parte de los lectores.

En pocas palabras —figura 5.35—, se puede decir que las modificaciones realizadas desde el Subsistema de Memorización, en concreto, en la EC_m y en las RT_a, tienen un impacto directo sobre la EC_p. El cambio de la EC_p desde el Subsistema de Presentación implica la redefinición de las Restricciones de Navegación RT_n desde el Sistema de Navegación

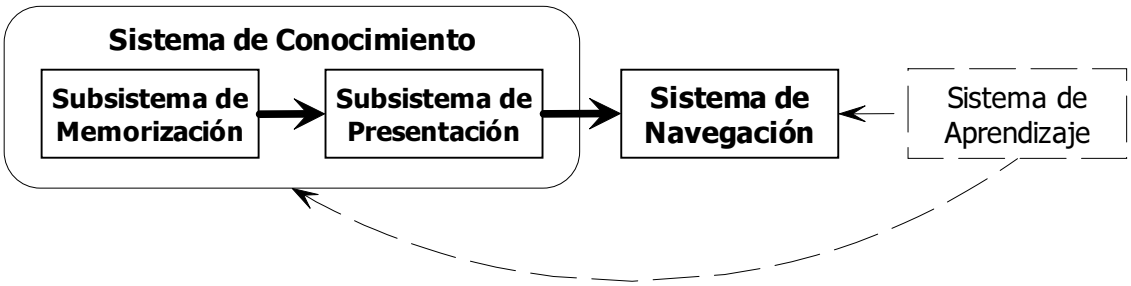


Figura 5.35 Propagación externa del cambio entre los Sistemas de SEM-HP.

5.2 Prestaciones de la Formalización Usada

Los distintos formalismos empleados —grafos, redes de Petri, lógica temporal proposicional y de predicados— permiten modelar las distintas fases del proceso de desarrollo de un Sistema Hipermedia consistentes en: crear dominios conceptuales y de información, seleccionar presentaciones y crear navegaciones.

5.2.1 Estructura y Formalismos

En particular, desde el Sistema de Conocimiento se realiza el diseño de la semántica de estructuración y de sus presentaciones, por tanto, un grafo etiquetado parece ser el formalismo más apropiado para su representación.

Para la especificación de las Restricciones de Sistema también se ha acudido a la Teoría de Grafos, dada la facilidad con la que se pueden indicar funciones de correspondencia que calculan los conceptos que son alcanzables, los que pueden ser alcanzados, si dos conceptos están simplemente conectados y, por último, las relaciones que los mantienen unidos. Por su parte, las Restricciones de Autor encaminadas a determinar cuándo una asociación es válida en la Estructura Conceptual han sido reducidas a un conjunto de dieciséis posibles caracterizaciones basadas en este mismo formalismo. Aunque esta decisión simplifica y unifica en gran medida su especificación e implementación, al tiempo que ofrece un amplio abanico de expresiones, también limita la posibilidad de poder expresar condiciones más complejas en las que se tengan en cuenta:

- los conceptos que se asocian,
- los conceptos que preceden o alcanzan a un determinado concepto.

En definitiva, el mecanismo de especifición de Restricciones de Autor hubiese sido más rico si se hubiera utilizado algún lenguaje basado en lógica temporal. Sin embargo, dada la gran capacidad de expresión de este formalismo, plantea dos retos importantes:

- construir un Meta-lenguaje con el que comprobar que el conjunto de Restricciones de Autor es consistente y
- especificar acciones con las que se pueda hacer evolucionar a estas Restricciones.

Se está trabajando en ello y se propone como trabajo futuro.

Por su parte, en el Sistema de Navegación, el principal objetivo es restringir los posibles caminos que, en principio, determina una determinada presentación. En este caso, se está intentando especificar relaciones de orden entre los ítemes. En consecuencia, el formalismo más adecuado es la lógica temporal proposicional y para convertirlo en un mecanismo operacional que se pueda ejecutar, se transforma en redes de Petri. Con ellas, se podrán ejecutar las navegaciones diseñadas y analizar sus propiedades. La tabla 5.69 resume todos estos formalismos.

Subsistemas de Memorización y Presentación	**Grafos**	**Lógica Temporal**
Concepto (C)	Nodo etiquetado	Proposición
Ítem (II)	Nodo hoja etiquetado	Proposición
Asociaciones entre conceptos (A_c) o entre conceptos e ítemes (A_f)	Arco Etiquetado	Fórmula con operadores temporales y lógicos
Sistema de Navegación	**Redes de Petri**	**Lógica Temporal**
Concepto o Ítem	Lugar	Proposición
Relaciones de órden entre conceptos o ítemes	Transiciones y arcos	Fórmula con operadores temporales
Navegación	Disparo de transiciones	Instanciación de fórmulas
Relaciones de dependencia entre conceptos o ítemes	Transiciones y arcos	Fórmula con operadores lógicos

Tabla 5.69 Formalismos empleados en la especificación de la ESTRUCTURA de los Sistemas.

5.2.2 Evolución y Formalismos

Los formalismos que se ha usado para especificar la evolución de los Sistemas —Teoría de Grafos y el lenguaje **MN** basado en lógica temporal de predicados, tabla 5.70— admiten fácilmente los cambios y su propagación. Como se ha mostrado en las apartados precedentes, los Sistemas suministran Acciones de Evolución con las que se pueden modificar los ítemes, la Estructura Conceptual y las Restricciones de Autor con indenpendencia. Pero, al mismo tiempo, el Sistema consigue hacer propagar estos cambios con el fin de mantenerlo en un estado íntegro.

Concretamente, la Teoría de Grafos se basa en la la Teoría de Conjuntos, por tanto, muchas de las operaciones de evolución se expresan mediante simples operaciones sobre conjuntos. Las Restricciones de Autor del Sistema de Conocimiento también se pueden modificar fácilmente pues se sabe de antemano las combinaciones que son válidas y las que no —apartado 2.4.2.1 de este capítulo y apéndice B—.

El lenguaje **MN** —basado en lógica temporal de predicados— garantiza la consistencia de los cambios de las Restricciones de Navegación expresadas en lenguaje **N** —especificado en lógica temporal proposicional—. La lógica temporal de predicados permite subir un nivel de abstracción —Meta— y manejar las proposiciones de las restricciones como variables, modificándolas y, por tanto, consiguiendo su evolución.

En suma, todos los formalismos tienen un mismo objetivo: representar la evolución de cada componente de aquella manera que mejor resuelva el problema de cotejar su consistencia y propagar el cambio. Obviamente, estos formalismos estarán ocultos al autor por las herramientas que implementan los Sistemas. El desarrollador interactúa con las interfaces visuales de estos —capítulo 6— con las que desarrolla sus ítemes, estructuras conceptuales, restricciones de autor, presentaciones y navegaciones.

Subsistemas de Memorización y Presentación	**Grafos**
Operación	Operación sobre conjuntos
Meta-Restricción	Funciones de Alcanzabilidad
Elemento modificado	Conjunto
Sistema de Navegación	**Lógica Temporal**
Operación	Predicado
Meta-Restricción	Fórmula
Elemento modificado	Instanciación en la variable de un predicado

1) Consistencia
2) Propagación interna
3) Propagación externa

Tabla 5.70 Formalismos empleados en la especificación de la Evolución de los Sistemas.

5.3 Flexibilidad y Versatilidad de SEM-HP

La especificación formal de cada uno de los Sistemas del Modelo SEM-HP pone de manifiesto la versatilidad y flexibilidad que ofrecería una herramienta que lo implementase.

Sobre un mismo Sistema de Información el/la autor/a podrá crear tantas Estructuras Conceptuales como desee mediante el *Subsistema de Memorización*. Cada una de éstas define un dominio conceptual y de información. A partir de cada uno de estos dominios, el/la modelador/a podrá ofrecer un conjunto de posibles presentaciones o subdominios conceptuales y de información haciendo uso del *Subsistema de Presentación*. A su vez, para cada uno de estos subdominios representados por subestructuras conceptuales se podrán definir un grupo de navegaciones desde el *Sistema de Navegación* —figura 5.36—.

En consecuencia, en cada momento, el/la lector/a del hiperdocumento estará recorriendo un determinado subdominio de información con ciertas restricciones de navegación perteneciente a una presentación concreta; una de las posibles que se podría haber hecho con la Estructura Conceptual y de Información en la que se incluye. Por supuesto, este dominio conceptual y de información ha sido uno de los posibles espacios ontológicos que nos puede ofrecer nuestro Sistema de Información. Evidentemente si el/la diseñador/a no organiza la información mediante las acciones de evolución y, no impone ningún tipo de restricción desde alguno o varios de los Sistemas se estará desaprovechando la capacidad de la aplicación para mantener, reusar y evolucionar sus dominios conceptuales y de información.

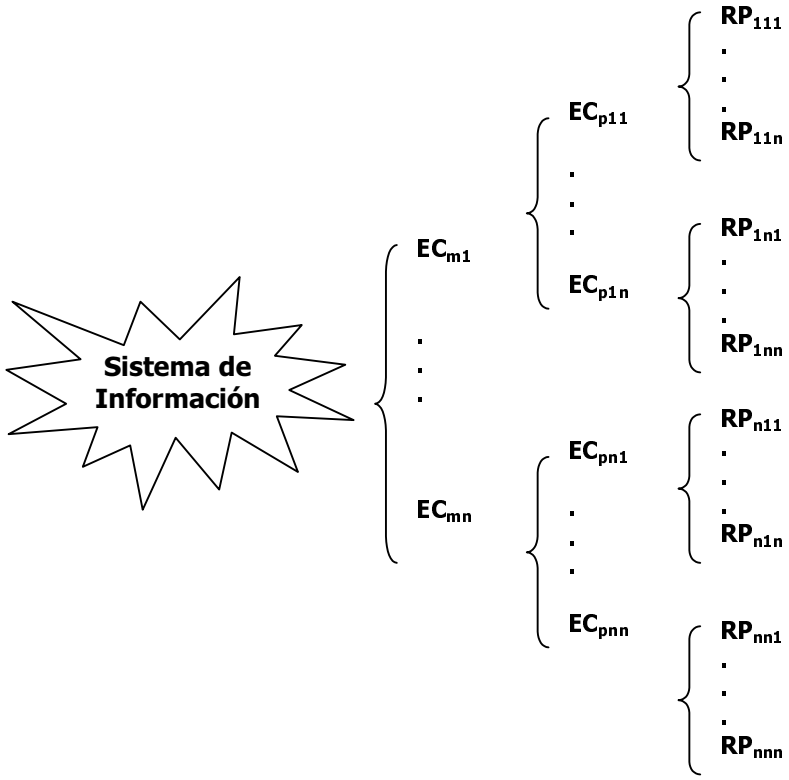


Figura 5.36 Flexibilidad y versatilidad de SEM-HP.

5.4 Referencias

[Afrati, 1990] 107
[Barrow, 1993] 144
[Bench-Capon, 1989] 107
[Carre 1979] 108
[Christofides, 1975] 108
[Christofides, 1975, página 12] 157
[Christofides, 1975, página 21] 157
[Christofides, 1975, página 23] 159
[Christofides, 1975, página 79] 160
[Chua, 1991] 107
[Furnas, 1986] 198
[Garg, 1988] 107
[Halasz, 1990] 107
[Ince, 1990] 107
[Koo, 1989] 107
[Mosterín, 2000, página 11] 107
[Parunak 1991] 107
[Perterson, 1981] 197
[Peterson, 1981, página 97] 210
[Peterson, 1981] 199, 209
[Rodríguez-Fórtiz, 2000a, página 99] 178
[Rodríguez-Fórtiz, 2000a] 173, 201
[Silva, 1985] 197
[Spivey, 1989] 107
[Sttots, 1989, 1998] 107
[Stotts, 1998] 198
[Tochtermann, 1995] 107
[Thayse, 1989] 182
[Wang, 1995, 1998] 107

Capítulo 6

Consideraciones sobre Diseño e Implementación

«El precio de las cosas debería depender de su mérito, jamás de su epíteto.»

William Shakespeare.
Autor, escritor y empresario teatral inglés (1564-1616).

«Las ideas no duran mucho. Hay que hacer algo con ellas.»

Santiago Ramón y Cajal.
Histólogo español (1852-1934).

Resumen

Se finaliza la parte dedicada a la descripción del modelo SEM-HP exponiendo aquellos aspectos que están relacionados con la implementación del mismo.

En particular, se incluye en este capítulo una especificación de SEM-HP en UML, una presentación del prototipo de las distintas interfaces con las que interactuaría tanto el autor como el lector de los hiperdocumentos y, por último, un análisis de las posibles implicaciones que pueden tener la familia de lenguajes XML en la representación e implementación del modelo propuesto.

1 Introducción 225
2 Especificación de SEM-HP mediante UML 226
2.1 Diagramas de Clases 226
2.2 Diagramas de Secuencia 234
3 Prototipos de las Interfaces de SEM-HP 242
3.1 El Sistema de Memorización 242
3.2 El Sistema de Presentación 247
3.3 El Sistema de Navegación 247
3.4 El Navegador 247
4 El Papel de XML en SEM-HP 256
4.1 XML 256
4.2 El lenguaje de enlazado de XML (XML Linking) 257
4.3 SEM-HP y XML 258
5 Referencias 259

1 Introducción

La herramienta que implemente SEM-HP pretenderá cubrir todas las fases —el ciclo de vida— de una aplicación hipermedia. Además, esta aplicación deberá incorporar interfaces gráficas muy versátiles que faciliten la interacción del creador y del usuario de los hiperdocumentos. Por tanto, el desarrollo completo de un producto software real —no un prototipo— que se base en este modelo requiere de un esfuerzo y dedicación que queda fuera de los objetivos de esta tesis, pero no de los propósitos del grupo de investigación GEDES —Grupo de Especificación, Desarrollo y Evolución de Software—. Seguramente, la incorporación de una becaria de investigación y la contratación de un programador —asociado a un proyecto I+D— para el próximo curso 2001/2002 hará realidad esta intención.

Sin embargo, sí se ha logrado alcanzar la aspiración de la mayoría de las implementaciones que se realizan en una investigación, enfocadas más a demostrar la viabilidad de la tesis que a ofrecer una aplicación comercial:

- En primer lugar, todos los aspectos relacionados con la evolución y, en particular, los que concomitan con la prueba lógica de las Restricciones y Meta-Restricciones expresadas en un lenguaje temporal, han sido ya incorporados y probados en la herramienta HEDES, en la que lleva trabajando varios años el grupo GEDES [Rodríguez-Fórtiz, 1997, 2000a, 2000b, 2000c] [Paderewski, 1999]. En concreto, esta aplicación:
  - incorpora los invariantes del Metasistema;
  - realiza una verificación de los mismos;
  - admite la especificación de Restricciones en lógica temporal;
  - verifica las sintaxis de las reglas temporales; y
  - evalúa las reglas temporales.
- En segundo lugar, la formalización rigurosa y completa del modelo ofrece un mecanismo formal, flexible y homogéneo que asegura la integridad del sistema y, al mismo tiempo, establece un diseño preliminar del que se puede derivar fácilmente su especificación en alguna de las metodologías conducentes a la elaboración de una aplicación software.

No obstante, no se pretende en este trabajo soslayar, en absoluto, el planteamiento de la implementación del modelo SEM-HP propuesto y, por ello, se detallan en este capítulo algunas consideraciones relacionadas con la misma. En particular, se comienza por realizar una especificación de la herramienta mediante el Lenguaje de Modelado Unificado —UML [Booch, 1999] [Booch, 2000]—. Posteriormente, con objeto de demostrar la viabilidad y mostrar la funcionalidad de la aplicación, se ha construido un prototipo de las distintas interfaces con las que interactuaría tanto el autor como el lector de los hiperdocumentos. Por último, era de obligado cumplimiento añadir algunas disquisiciones sobre el que pretende convertirse en el lenguaje estándar de la WWW —World Wide Web—, el popular pero en realidad desconocido XML —eXtensible Markup Language—.

2 Especificación de SEM-HP mediante UML

Con objeto de visualizar, documentar y facilitar la implementación del modelo SEM-HP, se presenta en esta sección la especificación explícita del mismo mediante el lenguaje gráfico UML. En este caso, el proceso de desarrollo de *software* estará centrado en la arquitectura, y su representación capta los aspectos más relevantes y de mayor nivel de abstracción del mismo. Concretamente, se describirán los conceptos del dominio de la aplicación mediante diagramas de clases, y el modo en el que interaccionan mediante diagramas de secuencia. Mediante el primer tipo de reprentación se mostrará la vista estática; la estructura de objetos, y con el segundo tipo de especificación; el comportamiento dinámico del sistema.

2.1 Diagramas de Clases

El *software* que se pretende construir estará destinado al diseño de hiperdocumentos desde un entorno gráfico e interactivo. Para su especificación se ha seguido el paradigma propuesto por [Golberg, 1983a, 1983b], denominado *Model-View-Controller* —MVC—, por estar enfocado precisamente al desarrollo de aplicaciones de software interactivo. La estrategia MVC consiste en percibir una herramienta desde tres puntos de vista:

- Los Modelos: los componentes que hacen el trabajo y simulan el dominio de aplicación.
- Las Vistas, que captan y muestran algunos de los aspectos del modelo.
- Los Controladores, que se encarga de enviar mensajes al modelo y mediar entre el modelo y sus vistas asociadas.

El diseño e implementación de herramientas mediante esta filosofía de trabajo aporta grandes ventajas con respecto a la modularidad y a la reutilización de software. MVC impone una separación, por un lado, entre el modelo del dominio de la aplicación y las vistas que reflejan el estado del mismo y, por otro, entre la edición o control del modelo y las propias vistas. Hecha esta aclaración, el sistema hipermedia contará con cuatro componentes fundamentales —figura 6.1—:

- El Modelo SEM-HP: la clase que contiene el dominio del problema con el que los usuarios del *software* tendrán que interaccionar.
- Un conjunto de Vistas del modelo SEM-HP —la clase *View*—: la clase que recibe las entradas del usuario y la que le muestra los resultados.
- El Evaluador —la clase *Evaluator*—: la clase a la que se enviarán todas las operaciones que se realicen desde las vistas. Desde este componente se decidirá si se lleva a cabo o no una determinada acción de evolución.
- El Propagador de cambios —la clase *Propagator*—: cada vez que el Evaluador decida llevar a cabo una operación deberá informar al Propagador para que éste lleve a cabo las acciones de evolución que se deriven del cambio realizado.

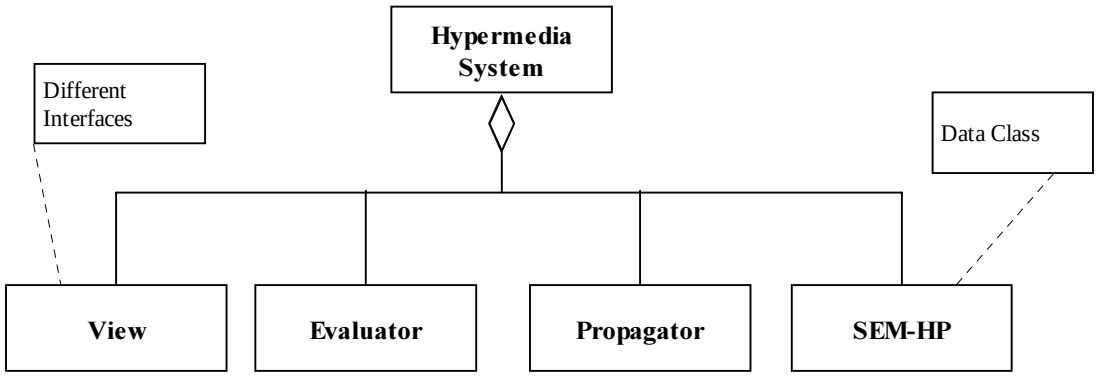


Figura 6.1 Diagrama de Clases del Sistema Hipermedia.

A su vez, la clase que representa el modelo SEM-HP estará formado por los consabidos sistemas de Conocimiento —compuesto por los Subsistemas de Memorización y Presentación—, de Navegación y de Aprendizaje —figura 6.2—.

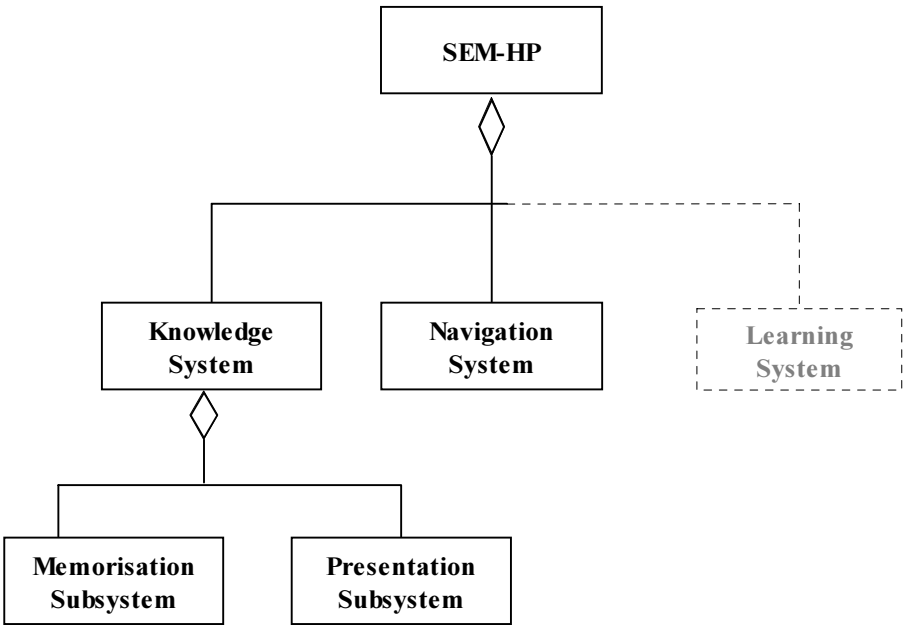


Figura 6.2 Diagrama de Clases del Modelo SEM-HP.

A continuación se detallan los diagramas de clases correspondientes al Subsistema de Memorización —figura 6.3—, Subsistema de Presentación —figura 6.4— y Sistema de Navegación —figura 6.5—, que de forma clara siguen la formalización realizada —capítulo 5—.

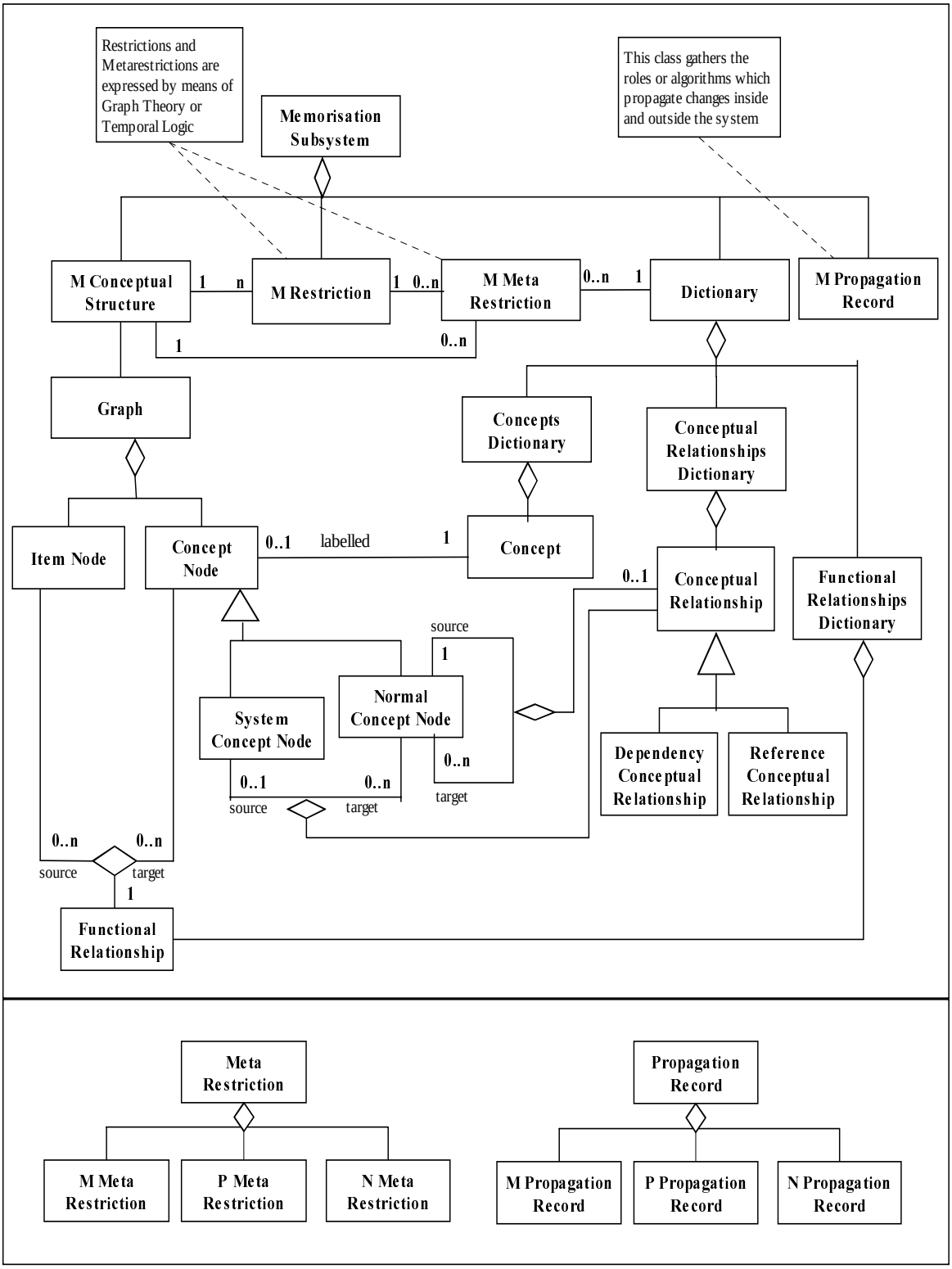


Figura 6.3 Diagrama de Clases del Subsistema de Memorización.

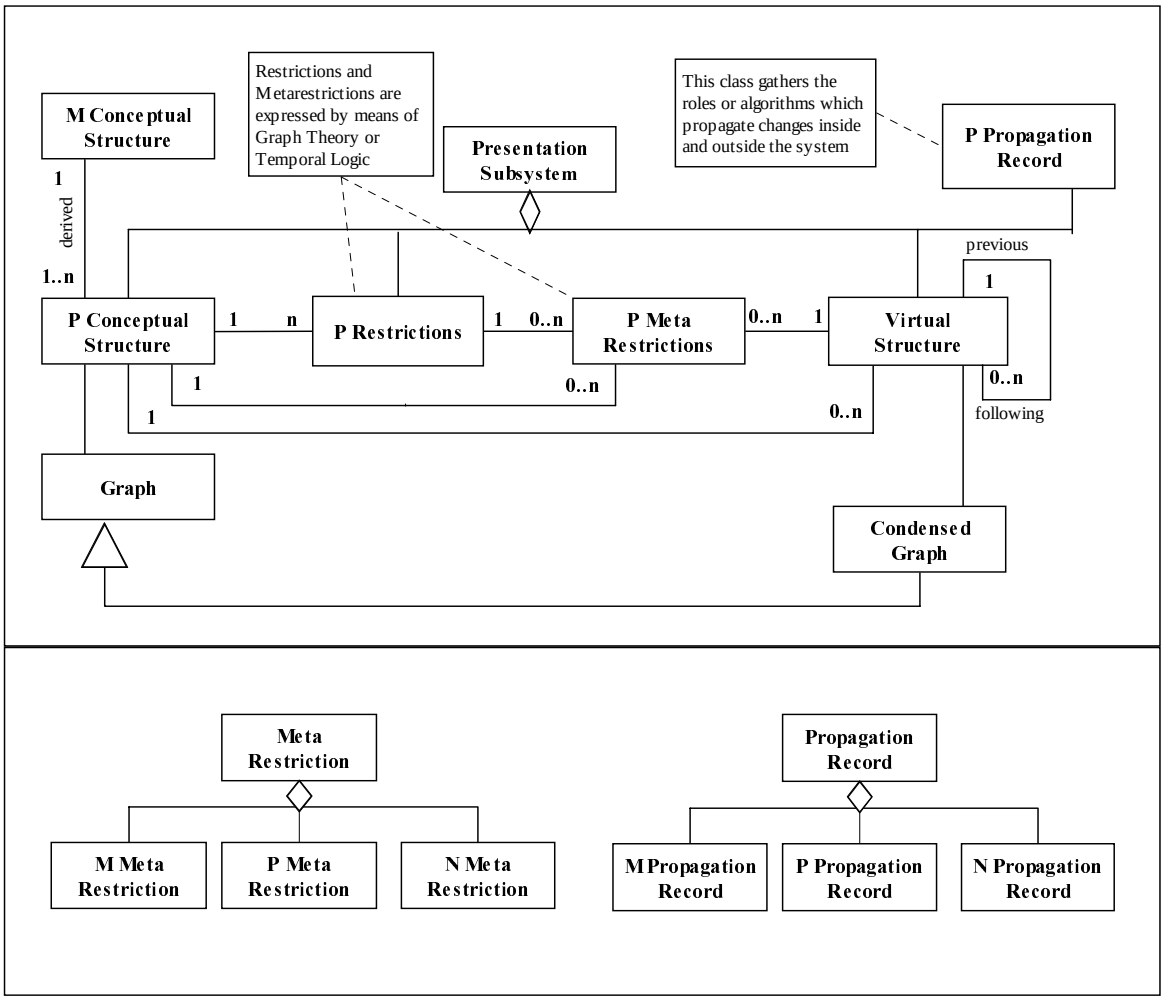


Figura 6.4 Diagrama de Clases del Subsistema de Presentación.

Como puede observarse —figuras 6.3, 6.4 y 6.5—, las Restricciones, las Meta-Restricciones y el registro de la propagación son clases dinámicas que no pueden ser especificadas mediante UML. Las Restricciones y Meta-Restricciones son reglas que estarán expresadas en algún formalismo —teoría de grafos o lógica temporal—. Las reglas de propagación son los algoritmos que indican cómo deben propagarse los cambios dentro y fuera del sistema.

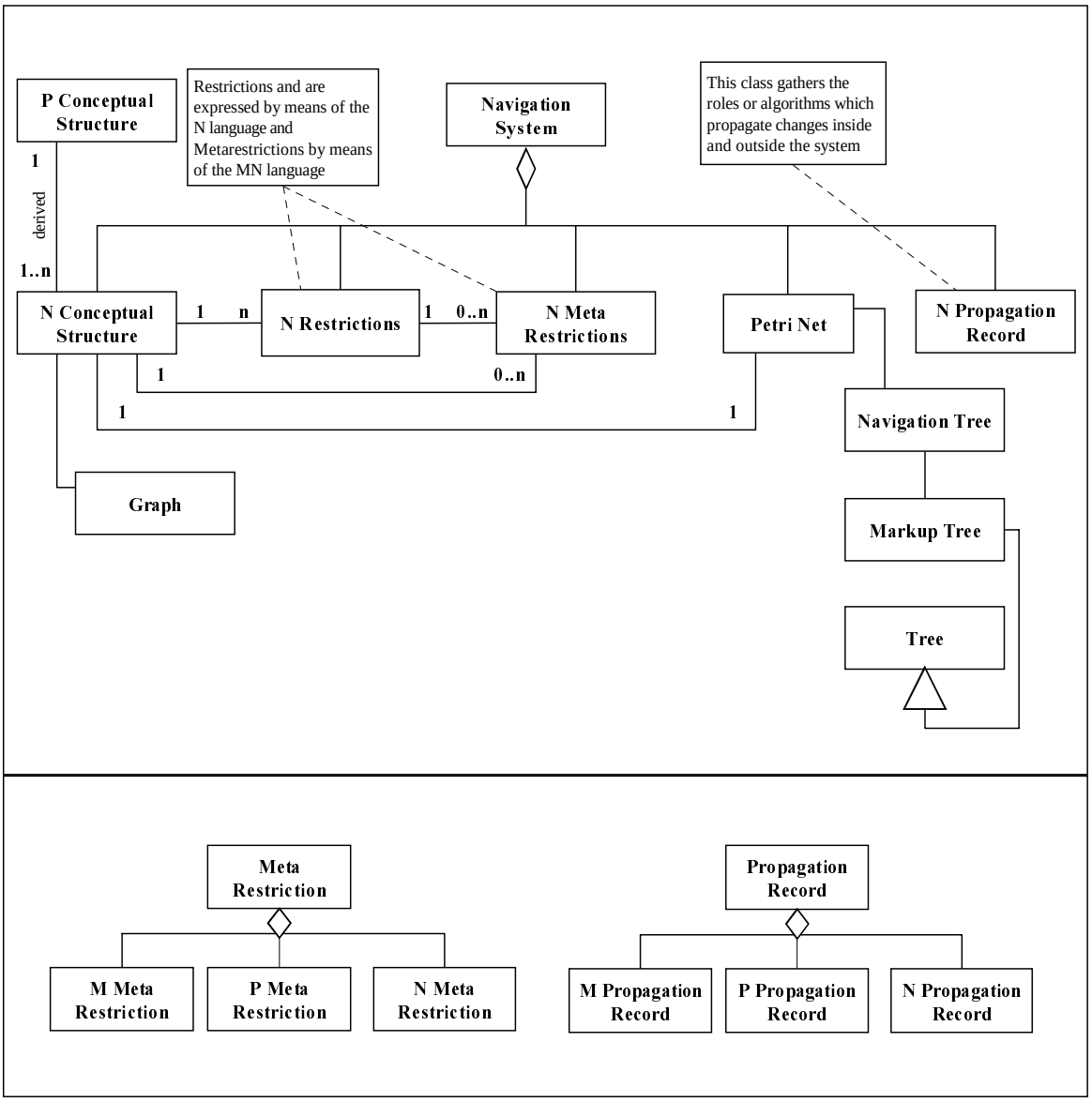


Figura 6.5 Diagrama de Clases del Sistema de Navegación.

2.1.1 Descripción de las Clases

En este apartado se detallan la descripción de las clases que son relevantes en el modelo de datos SEM-HP. Esta especificación se centra en aquellos atributos, valores y operaciones que están directamente relacionados con la gestión y evolución de SEM-HP. Quedan sin determinar, por tanto, todos aquellos aspectos vinculados a las interfaces gráficas destinados a facilitar la interacción del usuario con la aplicación.

M Conceptual Structure / Estructura Conceptual M
Nombre: Cadena autores: Cadena creado: Fecha modificado: Fecha accedido: Fecha número-conceptos: Número número-relaciones: Número
accesibles-desde(n, c, ECm): conceptos alcanzan-a(n, c, ECm): conceptos conectados-a(n, c, ECm): conceptos relacionados-mediante(nodo, nodo, ECm): relaciones-conceptuales Crea-concepto(etiqueta-c, ECm): c Crea-asociación-conceptual(origen, destino, etiqueta-rc, ECm): asociación-conceptual Crea-asociación-funcional(origen, destino, etiqueta-rf, ECm): asociación-funcional Borra-asociación-conceptual(asociación-conceptual, ECm): boolean Borra-asociación-funcional(asociación-funcional, ECm): boolean Crea-Ac-y-concepto(concepto, etiqueta-c, etiqueta-rc, ECm): asociación-conceptual Modifica-asociación-conceptual(origen,destino, ECm): asociación-conceptual Modifica-asociación-funcional(origen, destino, ECm): asociación-funcional Marca-concepto-raíz(concepto, ECm): boolean Cambia-etiqueta(objeto, etiqueta, ECm): boolean

Figura 6.6 Clase: Estructura Conceptual de Memorización.

Graph / Grafo
crea-nodo(G): nodo borra-nodo(nodo, G): boolean crea-arco(G): arco borra-arco(arco, G): boolean pon-etiqueta-nodo(nodo, etiqueta, G): boolean pon-etiqueta-arco(arco, etiqueta, G): boolean quita_etiqueta-nodo(nodo, G): boolean quita-etiqueta-arco(arco, G): boolean nodos-accesibles-desde(n, nodo, G): nodos nodos-alcanzan-a(n, nodo, G): nodos nodos-conectados-a(n, nodo, G): nodos relacionados-mediante(origen, destino, G): arcos marca-nodo-raíz(nodo, G): boolean quita-nodo-raíz(nodo, G): boolean

Figura 6.7 Clase: Grafo.

Item Node / Nodo Item
identificador: numii.numvar lenguaje: {texto, sonido, gráfico, imagen, animación, ejecución, hipermedia} aplicación: Aplicación formato: Formato idioma: Idioma autor/es: Conjunto de Nombre organismo/s: Conjunto de Nombre calidad: Nombre número-visitantes: Número tipo-visitantes: Conjunto de Nombre organismos-recomiendan: Conjunto de Nombre personas-recomiendan: Conjunto de Nombre
Crea-ítem(ECm): item Edita-contenido-ítem(item, ECm): boolean Edita-propiedades-ítem(item, propiedad, ECm): boolean Borra-ítem(item, ECm): boolean Crea-variante(item, ECm): boolean Convierte-en-variante(item1, item2, ECm): boolean Crea-ítem-copia(item1, ECm): item2 Crea-ítem-copia-contenido(item1, ECm): item2 Crea-ítem-copia-propiedades (item1, ECm): item2 Crea-ítem-compuesto(relación-conceptual, conj-funciones, ECm): item

Figura 6.8 Clase: Nodo Item.

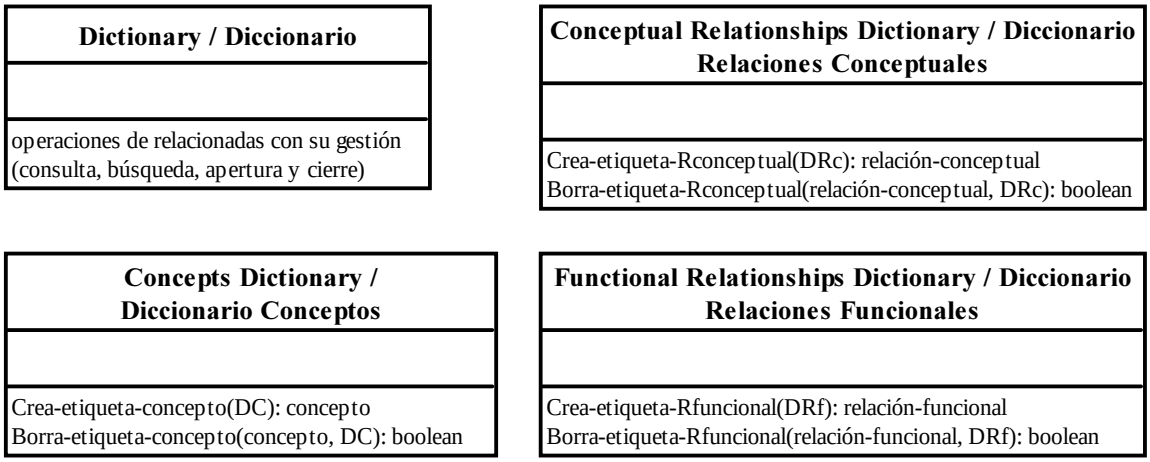


Figura 6.9 Clases relacionadas con los Diccionarios.

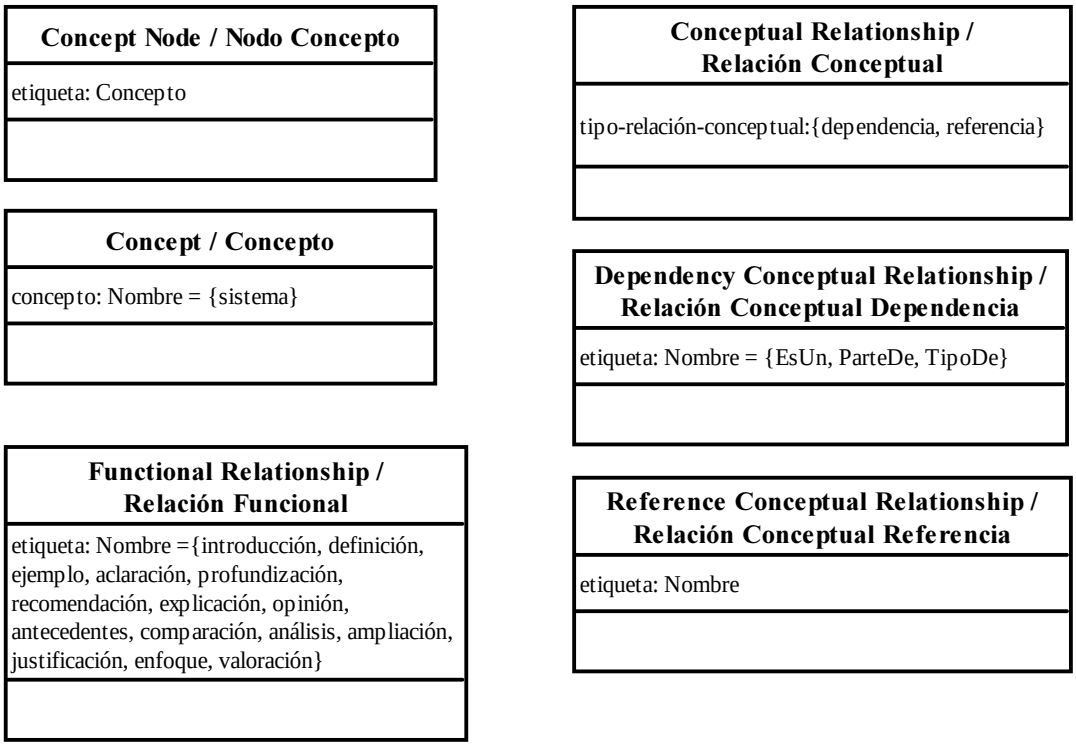


Figura 6.10 Clases relacionadas con los Conceptos y las Relaciones.

P Conceptual Structure / Estructura Conceptual P
nombre: Cadena deriva-de: Cadena autores: Cadena creado: Fecha modificado: Fecha accedido: Fecha número-conceptos: Número número-relaciones: Número
Oculta-RC(asociación-conceptual, ECp-visible, ECp-oculta): boolean Oculta-RF-e-ítem(asociación-funcional, ECp-visible, ECp-oculta): boolean Muestra-RC-y-concepto(asociación-conceptual, ECp-visible, ECp-oculta): boolean Muestra-RF-e-ítem(asociación-funcional, ECp-visible, ECp-oculta): boolean

Figura 6.11 Clase: Estructura Conceptual de Presentación.

Virtual Structure / Estructura Virtual
nombre: Cadena deriva-de: Cadena autores: Cadena creado: Fecha modificado: Fecha accedido: Fecha numero-conceptos: Número numero-relaciones: Número
Calcular-Componentes-fuertes(G): componentes Localizar-Centros(G): centros Calcular-Grafo-Condensado(G): grafo-condensado Asignar-nombres-conceptos(grafo-condensado): boolean Asignar-nombres-relaciones-conceptuales(grafo-condensado): boolean

Figura 6.12 Clase: Estructura Virtual.

N Conceptual Structure / Estructura Conceptual N
nombre: Cadena deriva-de: Cadena autores: Cadena creado: Fecha modificado: Fecha accedido: Fecha numero-conceptos: Número numero-relaciones: Número
RedNB(asociación-conceptual, ECn): boolean AmpNB(asociación-conceptual, ECn): boolean Calcular-Red-Petri(ECn, RTn): Red-Petri

Figura 6.13 Clase: Estructura Conceptual de Navegación.

Petri Net / Red Petri
nombre: Cadena deriva-de: Cadena autores: Cadena creado: Fecha modificado: Fecha accedido: Fecha número-conceptos: Número número-relaciones: Número
Calcular-Arbol-Navegación(Red-Petri): Arbol-Navegación

Figura 6.14 Clase: Red de Petri.

2.2 Diagramas de Secuencia

Los diagramas de secuencia que se han realizado prentenden mostrar la interación básica y genérica del sistema hipermedia completo y de cada uno de los sistemas que forman el modelo SEM-HP. Dado que todas las operaciones o acciones de evolución presentan un mismo patrón dinámico, se ha optado por mostrar diagramas de interacción de la operación *realizar-acción-evolutiva(parámetros)* o, de forma resumida, *acción-evolutiva(parámetros)*, entendiendo por *acción-evolutiva*, cualquiera de las acciones que se pueden realizar en el sistema y, por *parámetros*, los correspondientes a cada una de las operaciones en concreto.

La interacción de la figura 6.15 describe cómo el autor realiza una operación de evolución de las disponibles a traves de una de las vistas. La acción evolutiva se envía al Evaluador que, en primer lugar, consulta el modelo SEM-HP para capturar las Restricciones y Meta-Restricciones asociadas a dicha operación. El Evaluador comprueba que las verifica y decide si lleva a cabo o no la acción demandada. En el caso de llevarla a efecto, el Evaluador consulta al Propagador para saber cuál será el impacto del cambio realizado y desencadenar nuevas operaciones de evolución que mantengan al sistema en una situación de integridad.

Los siguientes diagramas de secuencia mostrarán la propagación del cambio para cada uno de los componentes de cada uno de los sistemas del modelo SEM-HP. Esta evolución fue descrita detalladamente en el capítulo 5 dedicado a la formalización. Las abreviaturas que se usan para nombrar los distintos elementos de los sistemas provienen de la nomenclatura usada en ese documento; en cualquier caso puede consultar la tabla correspondiente —figuras 6.16-21—.

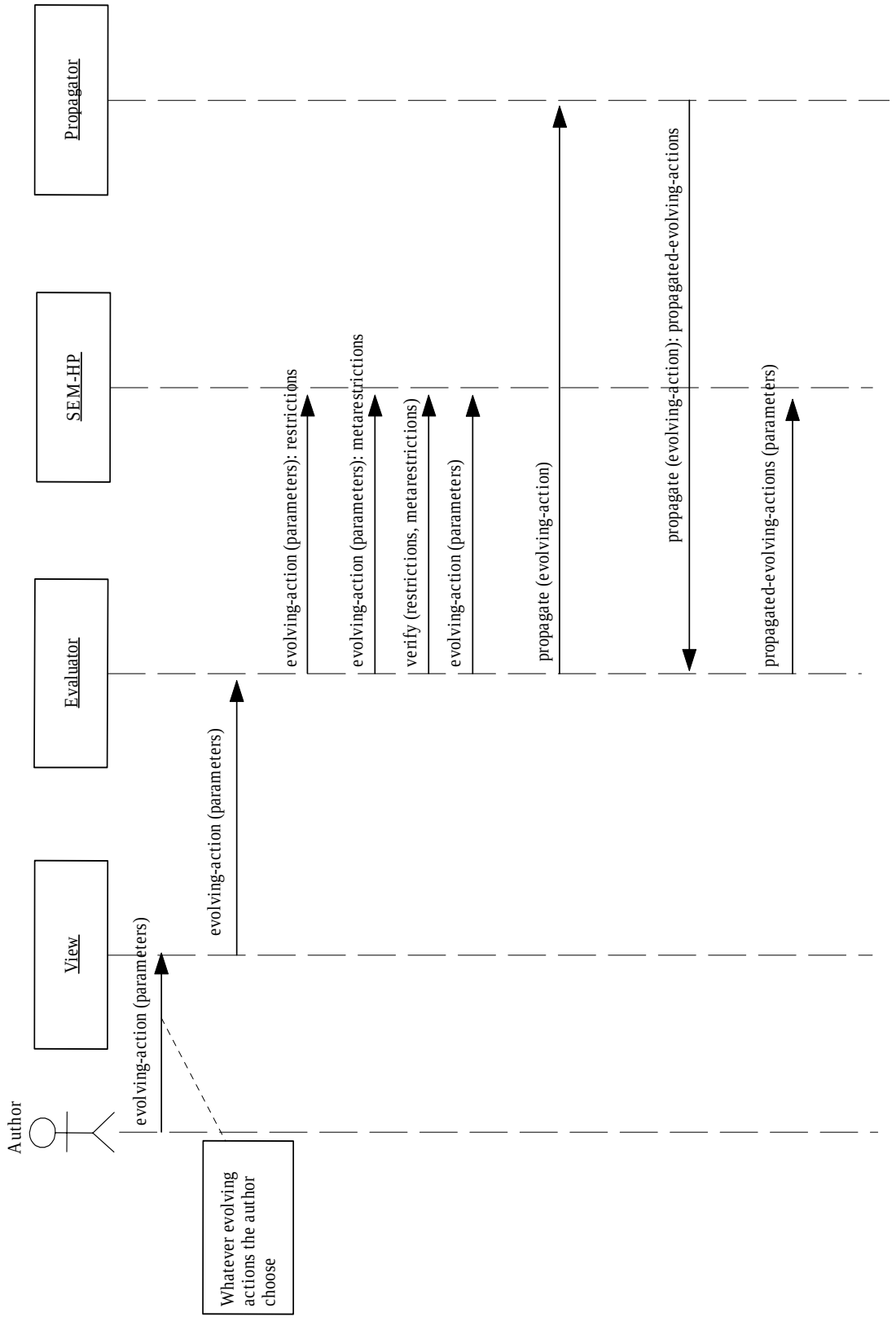


Figura 6.15 Clase: Diagrama de Secuencia del Sistema Hipermedia.

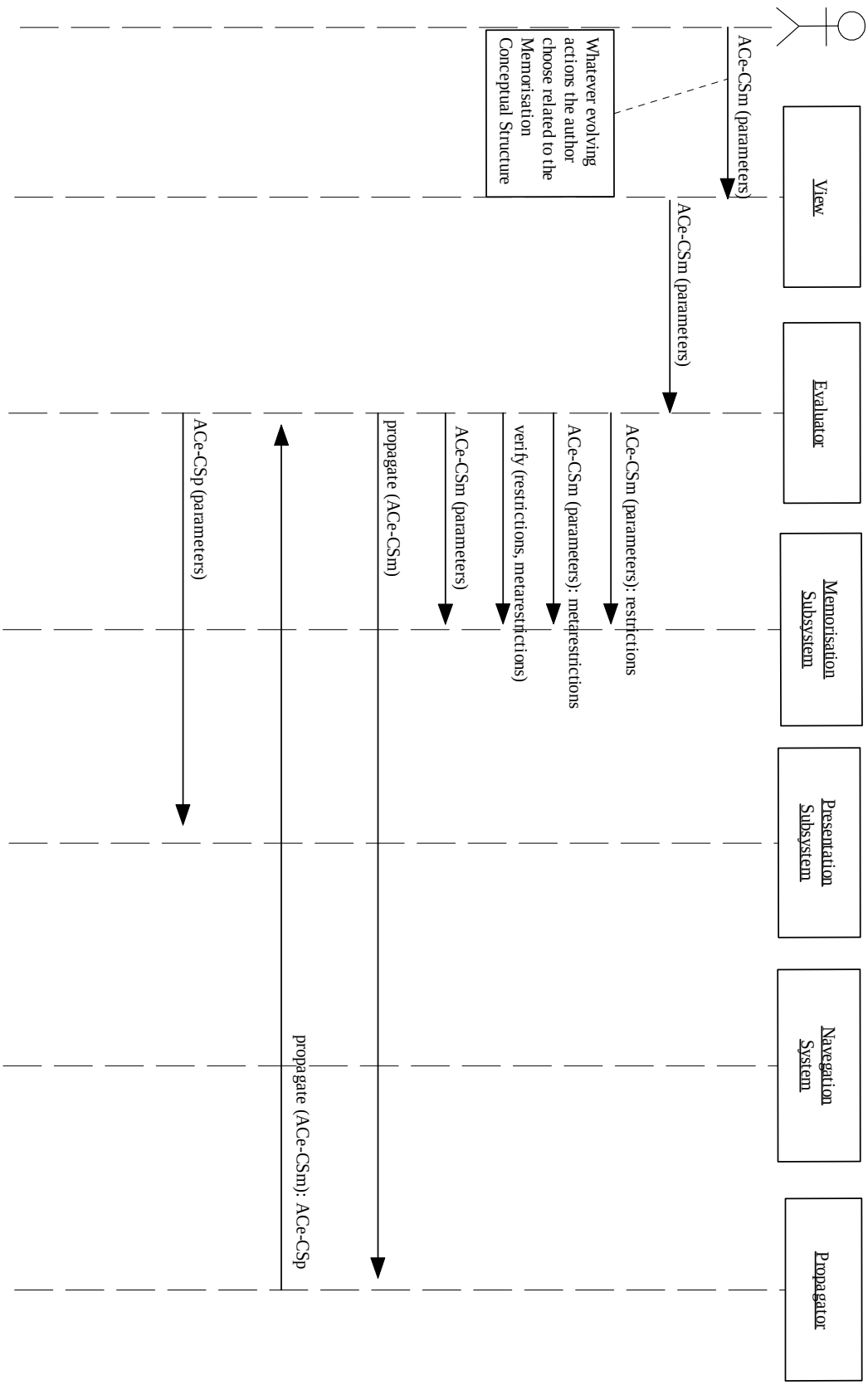


Figura 6.16 Diagrama de Secuencia del Subsistema de Memorización: Cambiando la Estructura de Memorización.

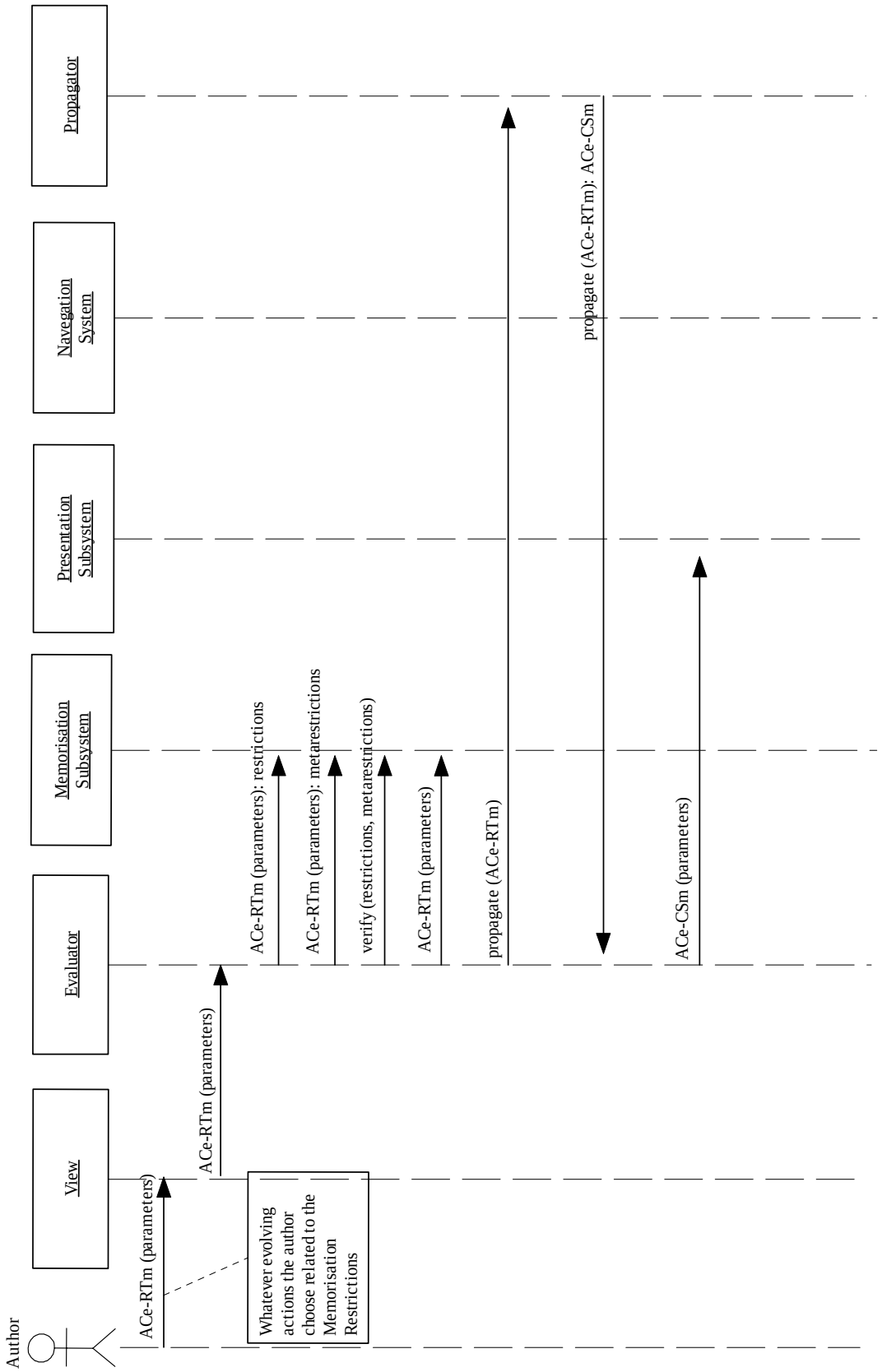


Figura 6.17 Diagrama de Secuencia del Subsistema de Memorización: Cambiando las Restricciones de Memorización.

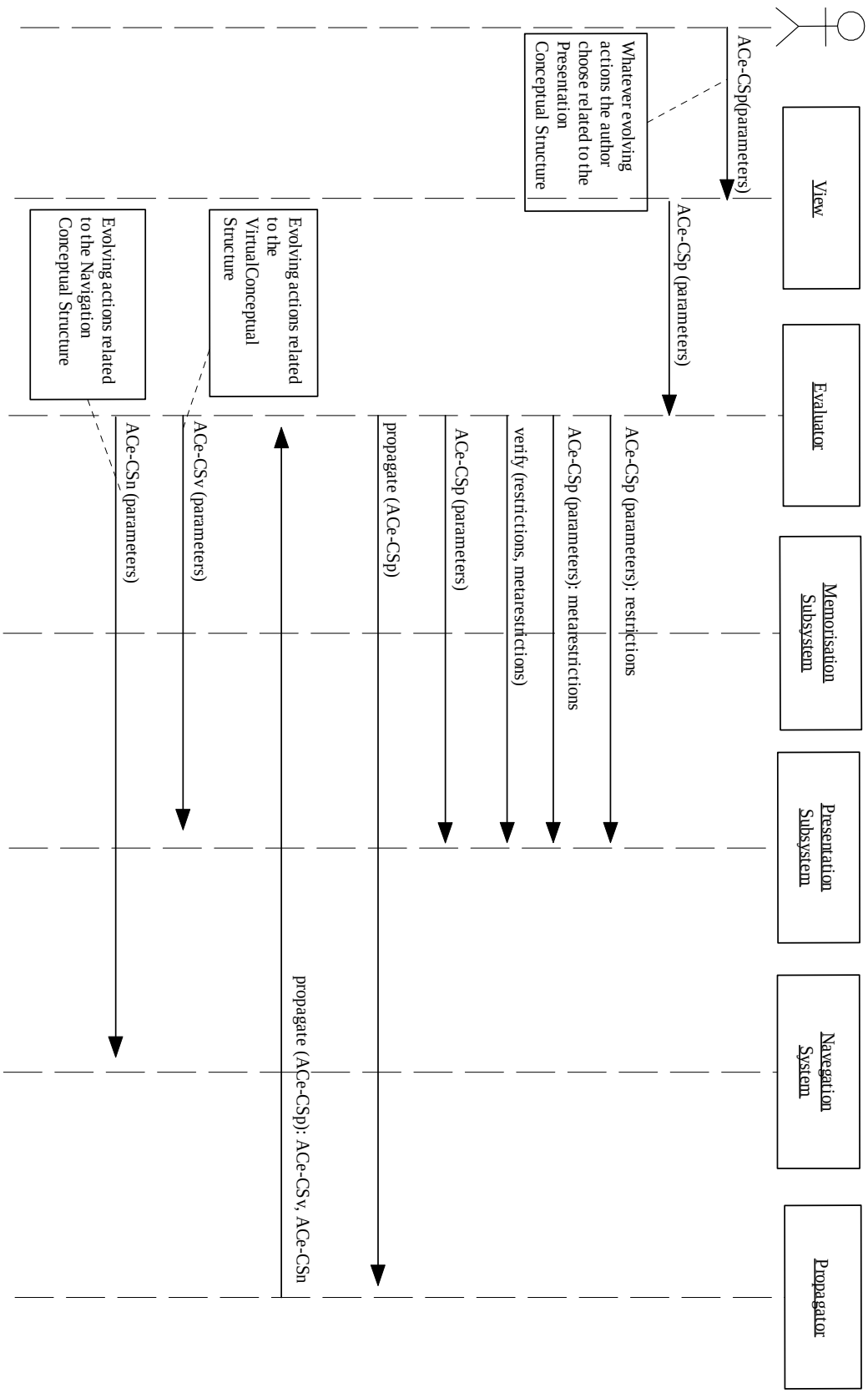


Figura 6.18 Diagrama de Secuencia del Subsistema de Presentación: Cambiando la Estructura Conceptual de Presentación.

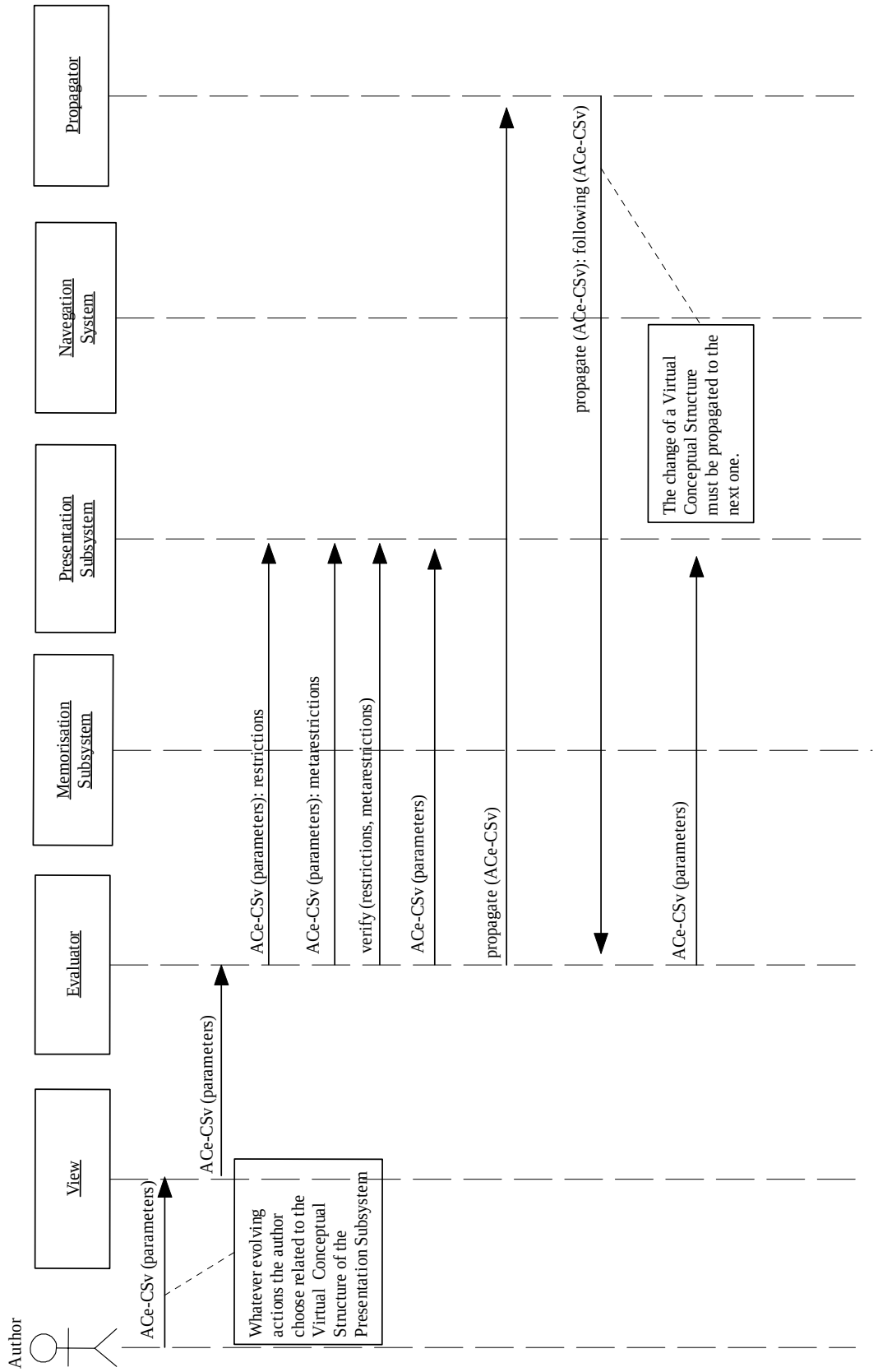


Figura 6.19 Diagrama de Secuencia del Subsistema de Presentación: Cambiando la Estructura Conceptual Virtual.

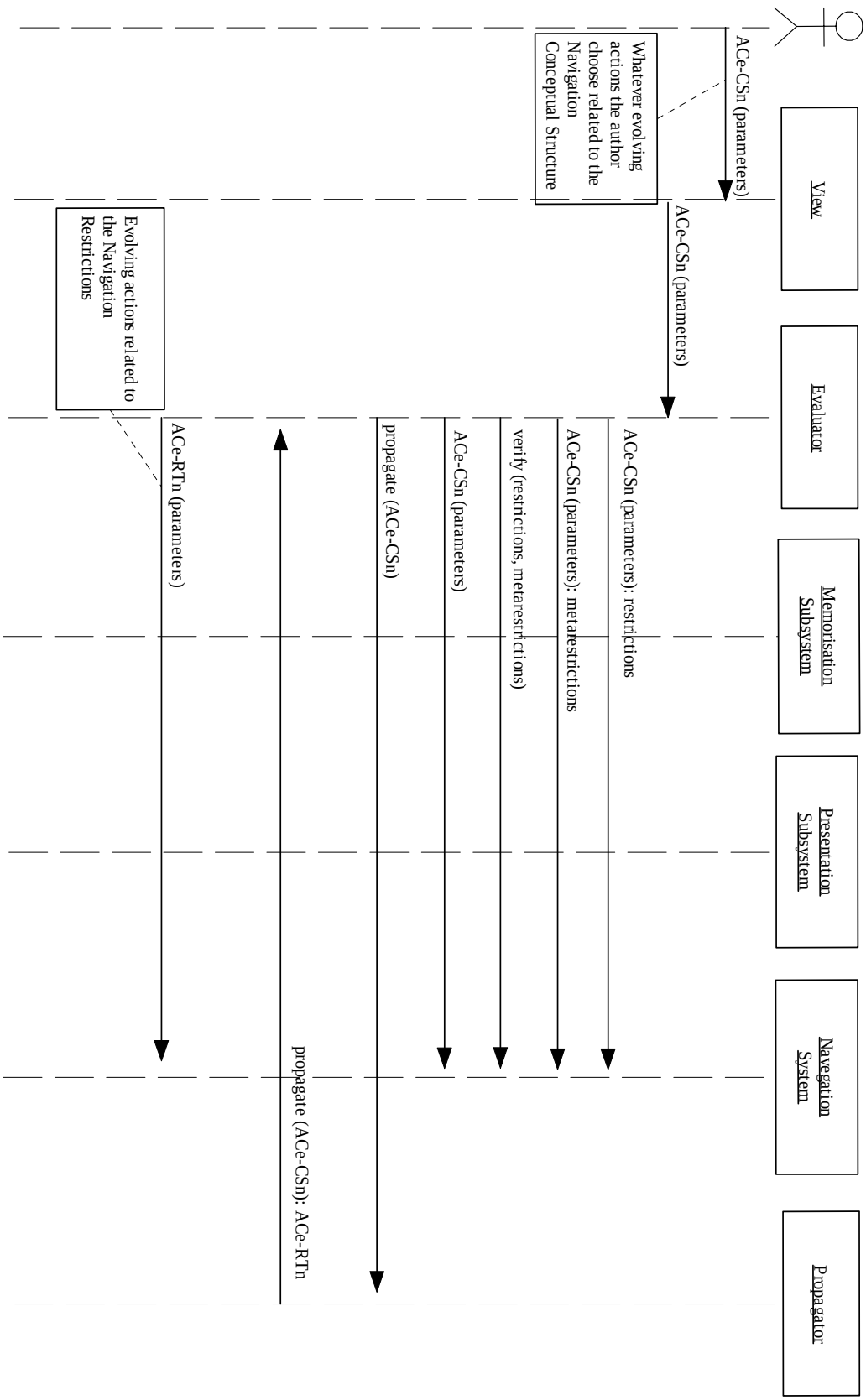


Figura 6.20 Diagrama de Secuencia del Subsistema de Navegación: Cambiando la Estructura Conceptual de Navegación.

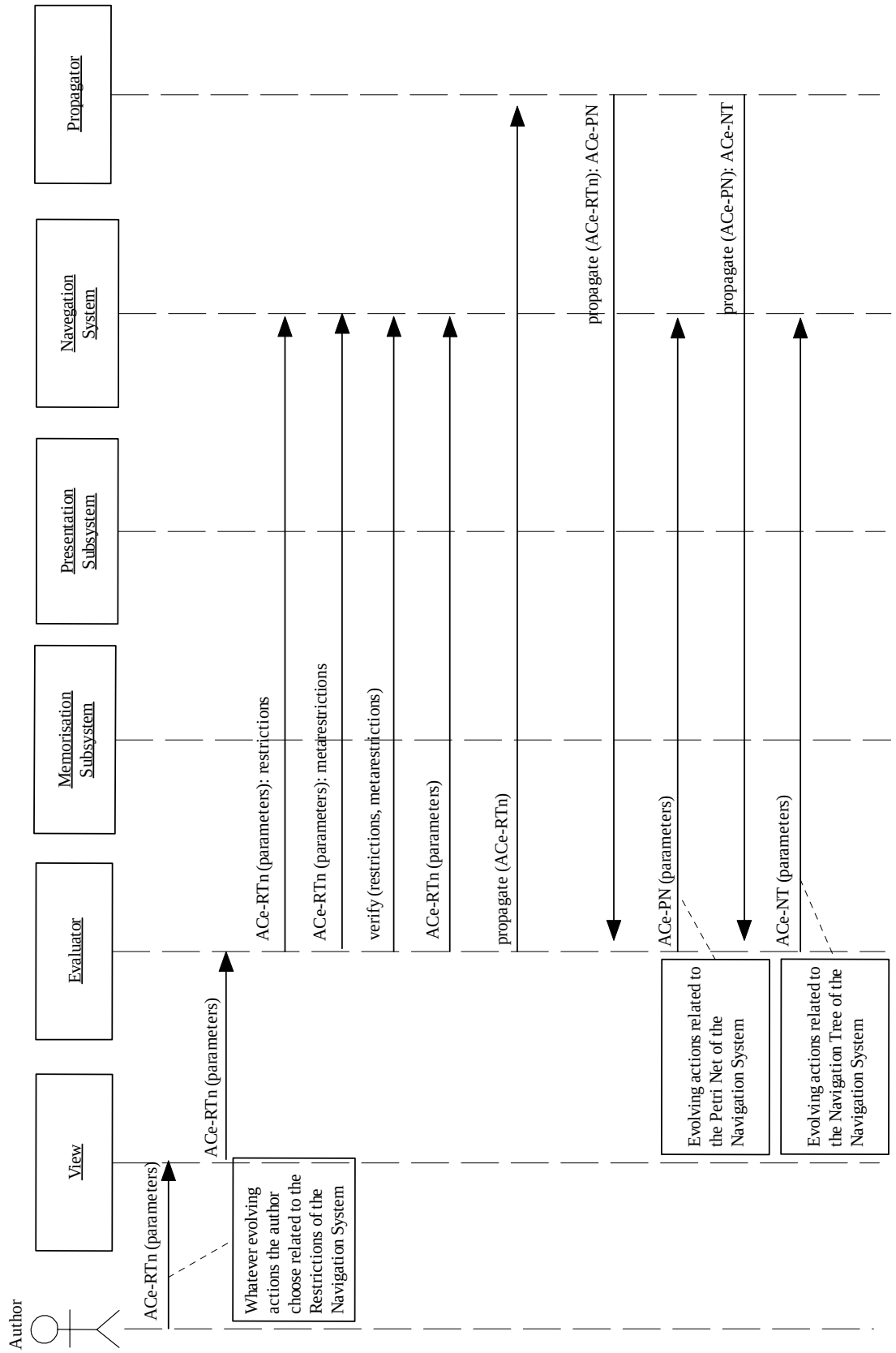


Figura 6.21 Diagrama de Secuencia del Subsistema de Navegación: Cambiando las Restricciones de Navegación.

3 Prototipos de las Interfaces de SEM-HP

La herramienta SEM-HP es un entorno gráfico e interactivo compuesto por cuatro aplicaciones. Las tres primeras —en el prototipo se han denominado Sistemas— implementan el Subsistema de Memorización, el Subsistema de Presentación y el Sistema de Navegación; la cuarta es un navegador con el que se podrá recorrer los hiperdocumentos diseñados con los anteriores Sistemas.

Todas las ventanas tienen el mismo diseño. Cuentan con una barra de operaciones y otra barra de botones con las que se pueden activar y desactivar las distintas interfaces de forma independiente. En la figura 6.22 se puede ver la herramienta SEM-HP en el momento en el que autor ha arrancado todas las aplicaciones pulsando sus botones. En los siguientes apartados se describen la funcionalidad de cada una de las mismas.

3.1 El Sistema de Memorización

Este Sistema cuenta con cuatro botones que se corresponden con los cuatro componentes básicos del mismo: la Estructura Conceptual, los Ítemes de Información, las Restricciones de Autor y los Diccionarios. La barra de operaciones cambiará en función de la ventana que esté activa en cada momento. El autor podrá tener abiertos o cerrados los componentes que considere oportunos en cada momento.

La pulsación del botón dedicado a la Estructura Conceptual despliega una interfaz gráfica con la que el autor podrá construir sus estructuras conceptuales. En la figura 6.23 se muestra una estructura conceptual sobre el *Sistema Solar*. Con el siguiente botón —de izquierda a derecha— se activa la aplicación dedicada a la edición de los ítemes de información. El Sistema de Memorización ejecutará la herramienta de edición con la que se ha construido el ítem. Junto al contenido del ítem se despliegan las propiedades del mismo que han sido agrupadas en cuatro categorías: General, Contenido, Edición y Navegación. En la figura 6.24 se está diseñando un ítem sobre los planetas —en formato html— del que podemos ver su contenido, sus propiedades de edición y el estilo final del mismo.

Con el botón de las Restricciones —el que está junto al de ítem de información— el autor puede asociar restricciones a las relaciones conceptuales de la Estructura Conceptual. En la figura 6.25 se puede ver las Restricciones asociadas a las relaciones *PartOf*, *aKindOf*, *IsA* y *rotate*. Finalmente con el botón de los diccionarios el autor puede crear y borrar etiquetas de conceptos, relaciones conceptuales y relaciones funcionales. La figura 6.25 muestra los diccionarios correspondientes a la estructura conceptual *Sistema Solar*.

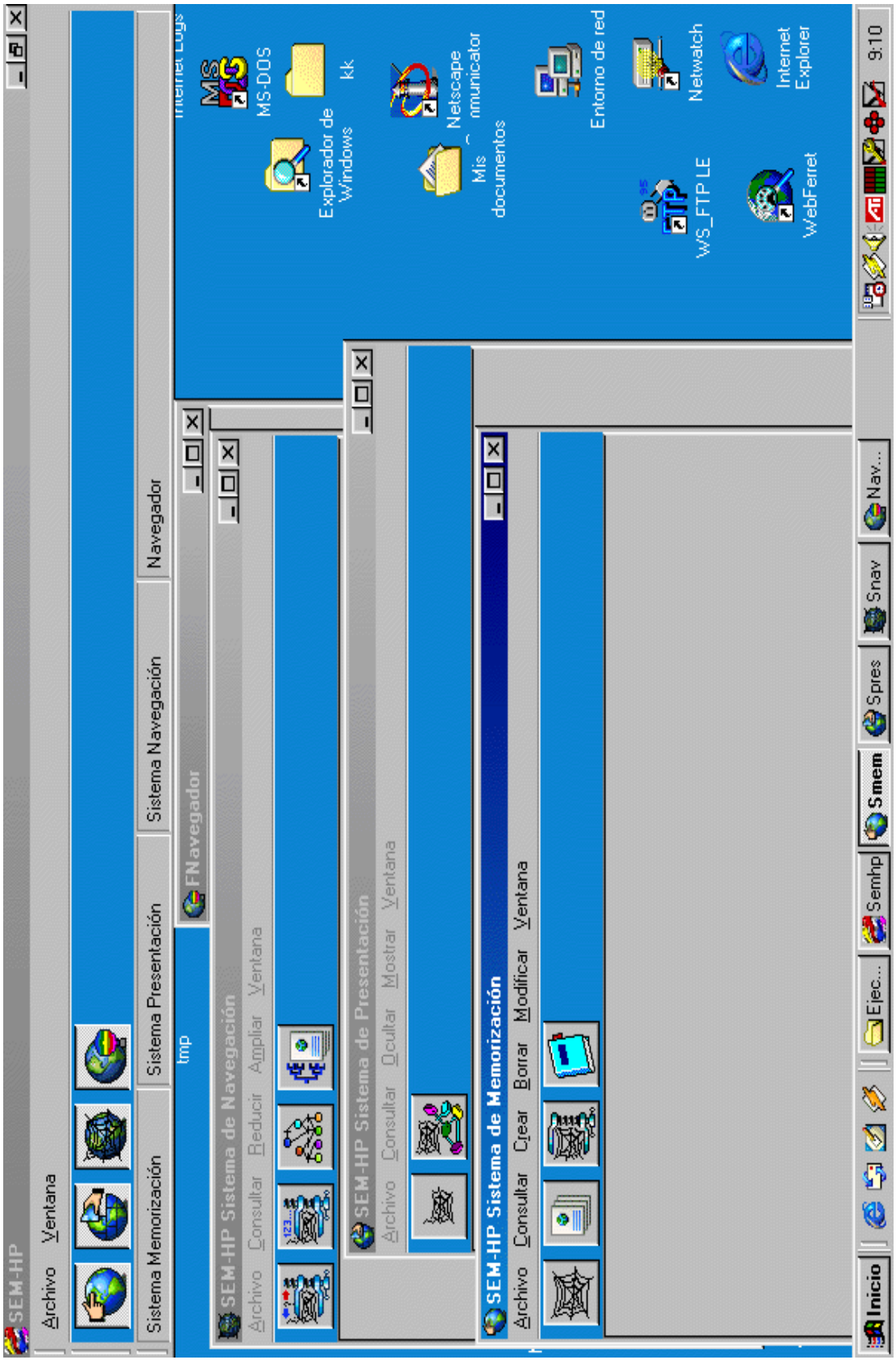


Figura 6.22 La herramienta SEM-HP: ejecución de todas sus aplicaciones.

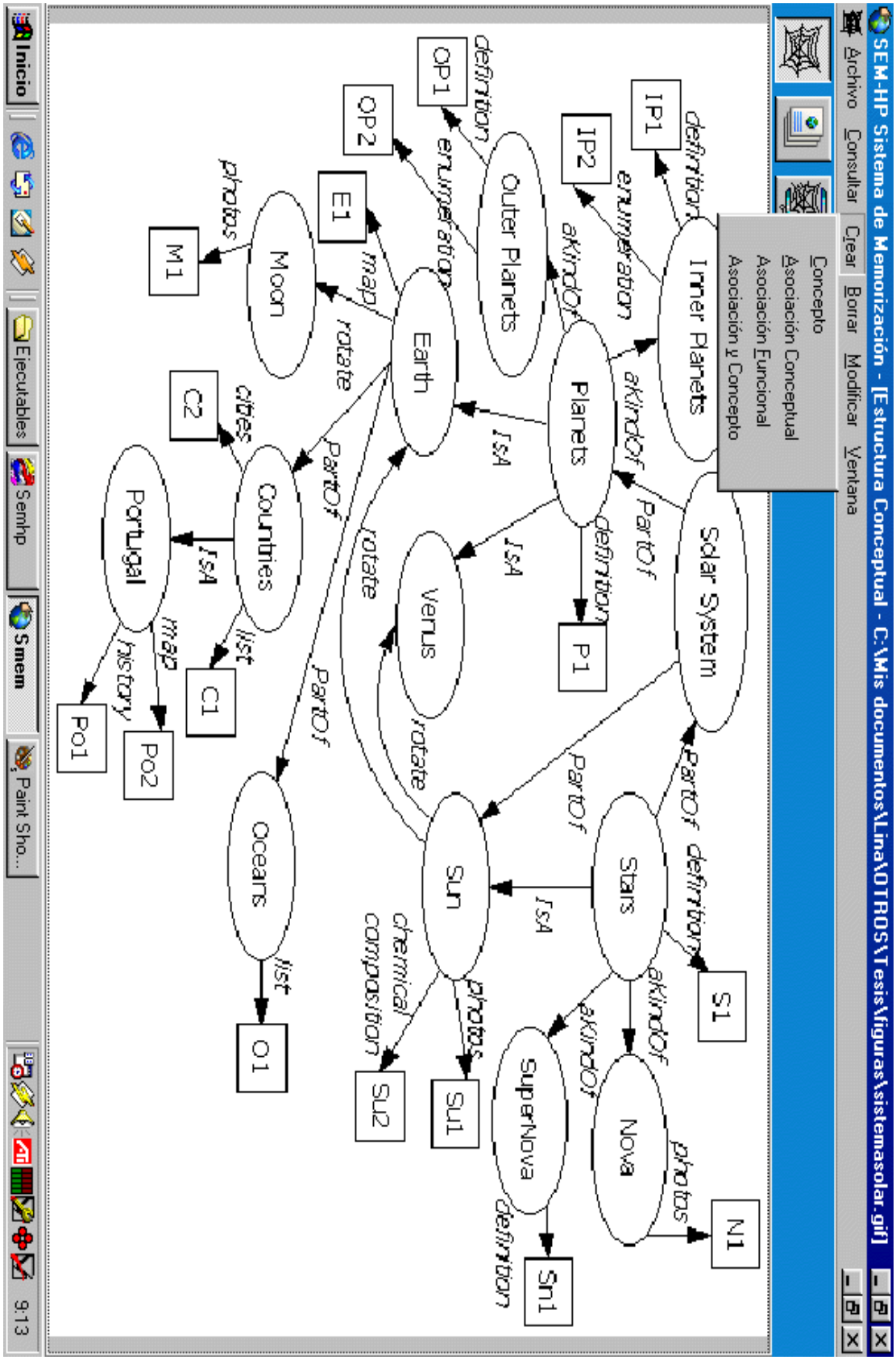


Figura 6.23 El Sistema de Memorización: Creando la Estructura Conceptual.

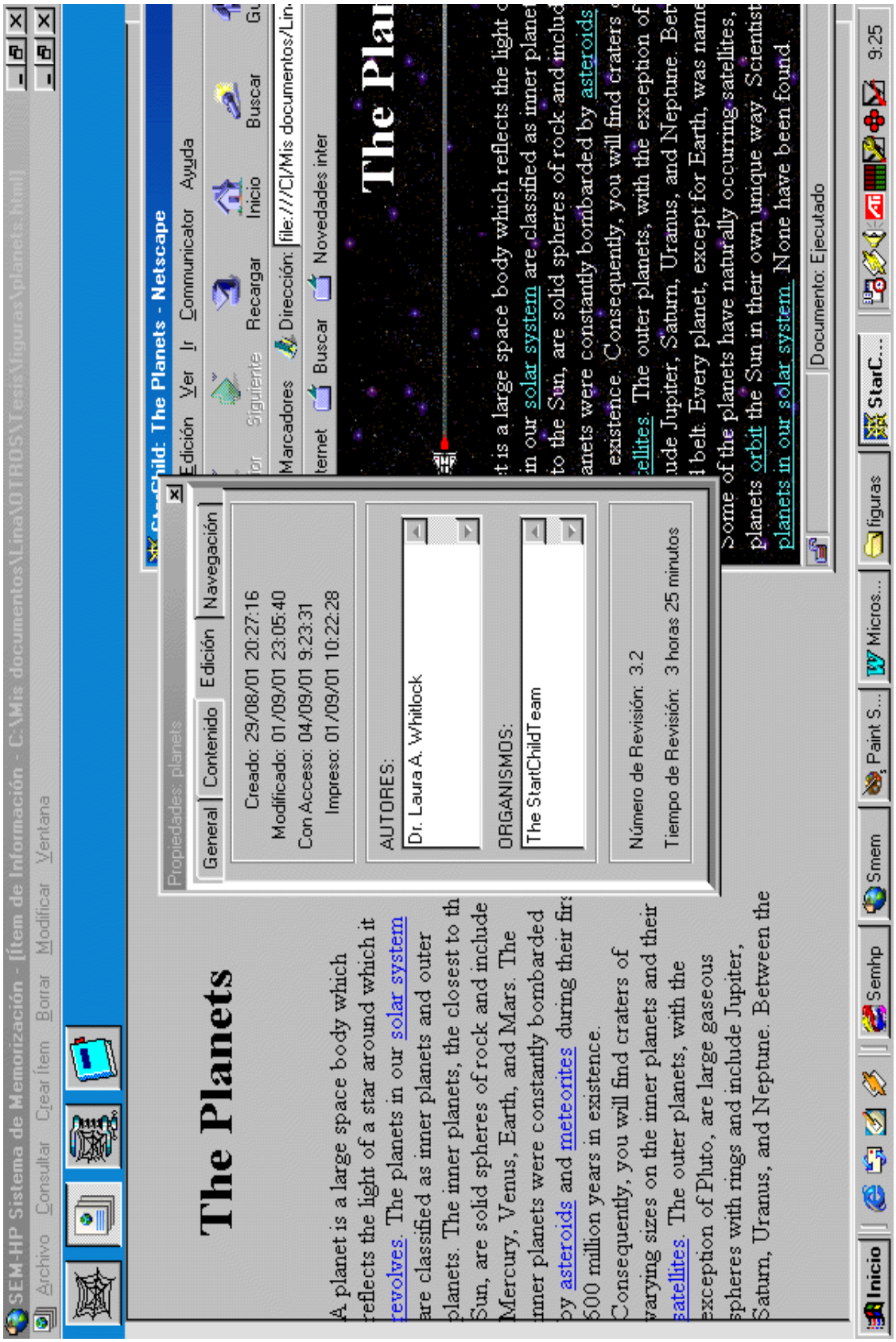


Figura 6.24 El Sistema de Memorización: Editanto el contenido y las propiedades de los Ítemes de Información.

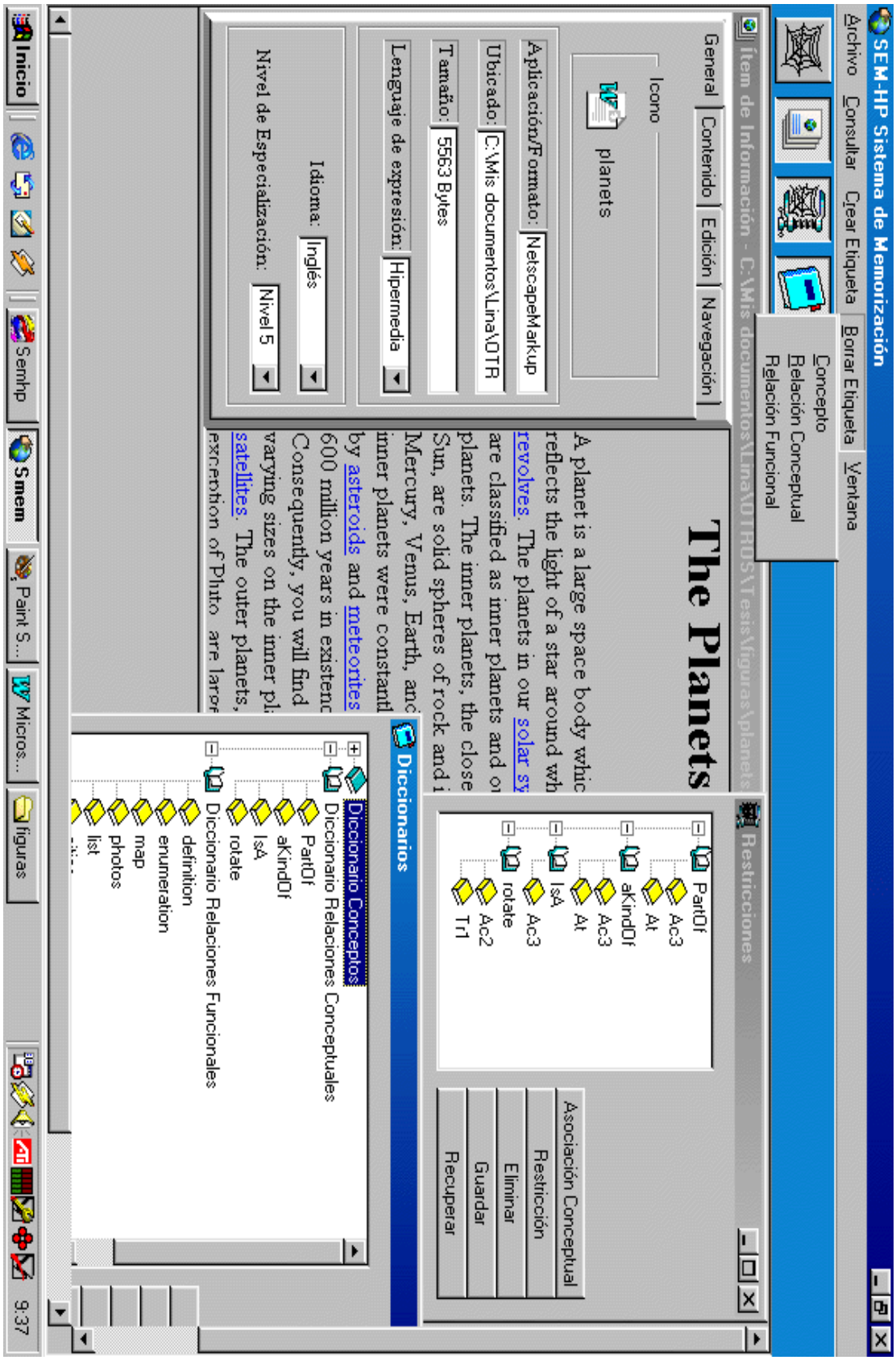


Figura 6.25 El Sistema de Memorización: Asociando Restricciones de Autor y creando los Diccionarios.

3.2 El Sistema de Presentación

El sistema de Presentación cuenta con dos botones: el primero dedicado a la selección de una presentación a partir de un determinado ítem de información, y el segundo es el que utilizará el autor para confeccionar las Estructuras Virtuales.

En la figura 6.26 se muestra la subestructura seleccionada por el autor de la estructura conceptual Sistema Solar. En la figura 6.27 se muestra una posible simplificación de la estructura conceptual Sistema Solar mediante el cálculo de los componentes fuertes del grafo.

3.3 El Sistema de Navegación

Finalmente, el desarrollador cuenta con el Sistema de Navegación con el que puede ampliar o reducir la navegación de las relaciones conceptuales —primer botón—, especificar restricciones de orden —segundo botón—, ejecutar la navegación —mediante su red de Petri asociada, tercer botón— o construir el árbol de navegación correspondiente —cuarto botón—.

En la figura 6.28 se puede ver que para el autor sólo son bidireccionales las asociaciones conceptuales: <*Earth*, *rotate*, *Moon*> y <*Earth*, *PartOf*, *Countries*>. Por su parte, la figura 6.29 muestra las Restricciones de Orden que genera la presentación de la figura 6.28. En la figura 6.30 se modifican estas restricciones de orden y se calcula la red de Petri correspondiente. Por último, en la figura 6.31 se muestra el árbol de navegación que se puede calcular a partir de la red de Petri anterior.

3.4 El Navegador

El último botón de la herramienta SEM-HP ejecuta un navegador con el que el lector y el autor pueden recorrer los hiperdocumentos de acuerdo a la navegación diseñada por el desarrollador. La novedad con respecto a otros hipermedia es que ahora se tienen tres posibilidades de interacción: con la Estructura Conceptual de presentación, con el árbol conceptual y con los ítemes de información.

El navegador, por tanto, aparece dividido —figura 6.32— en tres marcos. El marco superior izquierdo muestra la estructura conceptual, la ventana inferior izquierda el árbol de navegación y, por último, el cuadro derecho el ítem de información que está visitando ahora el lector.

Mediante la Estructura Conceptual el lector podrá:

- Hacer un *clic* en uno de los conceptos. En este caso, el sistema le ofrecerá los ítemes de información catalogados por sus propiedades. El usuario selecciona todos o un subconjunto de ellos, y se mostrarán en la pantalla. Junto a cada ítem se muestran los otros conceptos con los que está etiquetado.
- Seleccionar una subestructura conceptual y generar un árbol de conceptos.
- En el caso de que se muestre un estructura virtual, el lector podrá moverse un nivel arriba o abajo dentro del rango de niveles virtuales previstos por el autor.

Si el lector navega sobre un árbol conceptual generado a partir de la estructura conceptual podrá:

- Hacer un *clic* en uno de los conceptos. En este caso, el sistema le ofrecerá los ítemes de información catalogados por sus propiedades. El usuario selecciona todos o un subconjunto de ellos, y se mostrarán en la pantalla. Junto a cada ítem se muestran los otros conceptos con los que está etiquetado.
- En el caso de que se muestre el árbol de una estructura virtual, el lector podrá moverse un nivel arriba o abajo dentro del rango de niveles virtuales previstos por el autor.

Por último, el usuario podrá interactuar de forma tradicional con el hiperdocumento pudiendo:

- Navegar por los otros conceptos con los que está etiquetado el ítem de información.
- Seleccionar otros ítemes de información catalogados con el mismo concepto pero seleccionando otras propiedades.
- Navegar por hiperenlaces como en una página web normal.

Los tres fragmentos pueden estar sincronizados o no, de modo que un cambio en alguna de las ventanas supondrá la actualización o no del resto de las navegaciones.

Las figuras —6.26-6.32— correspondientes a los ejemplosr relacionados con las interfaces de las aplicaciones sobre el Sistema de Presentacion, el Sistema de Navegación y el Navegador se encuentran en las páginas 249-255 de este capítulo.

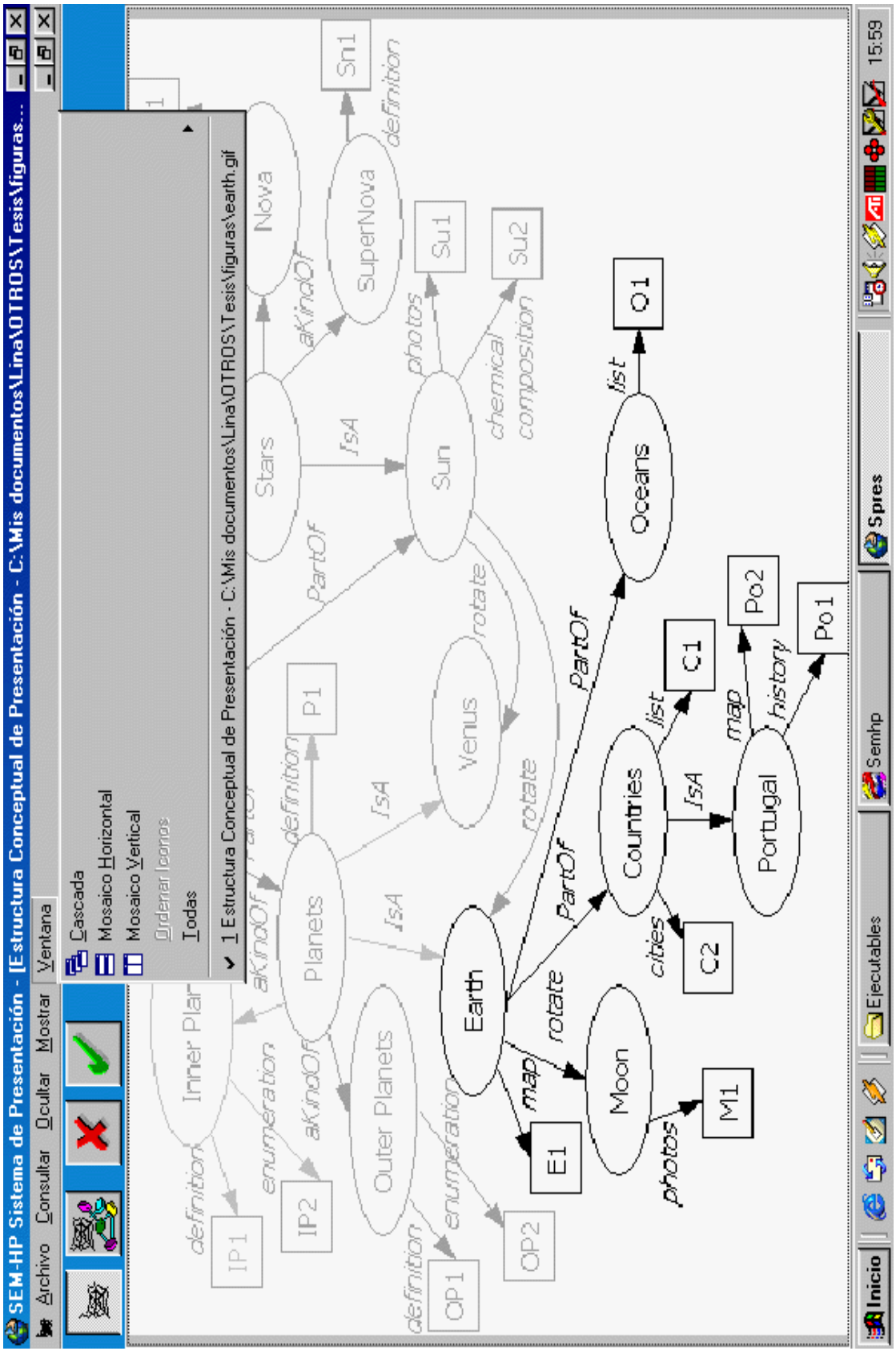


Figura 6.26 El Sistema de Presentación: Seleccionando la Estructura Conceptual de presentación.

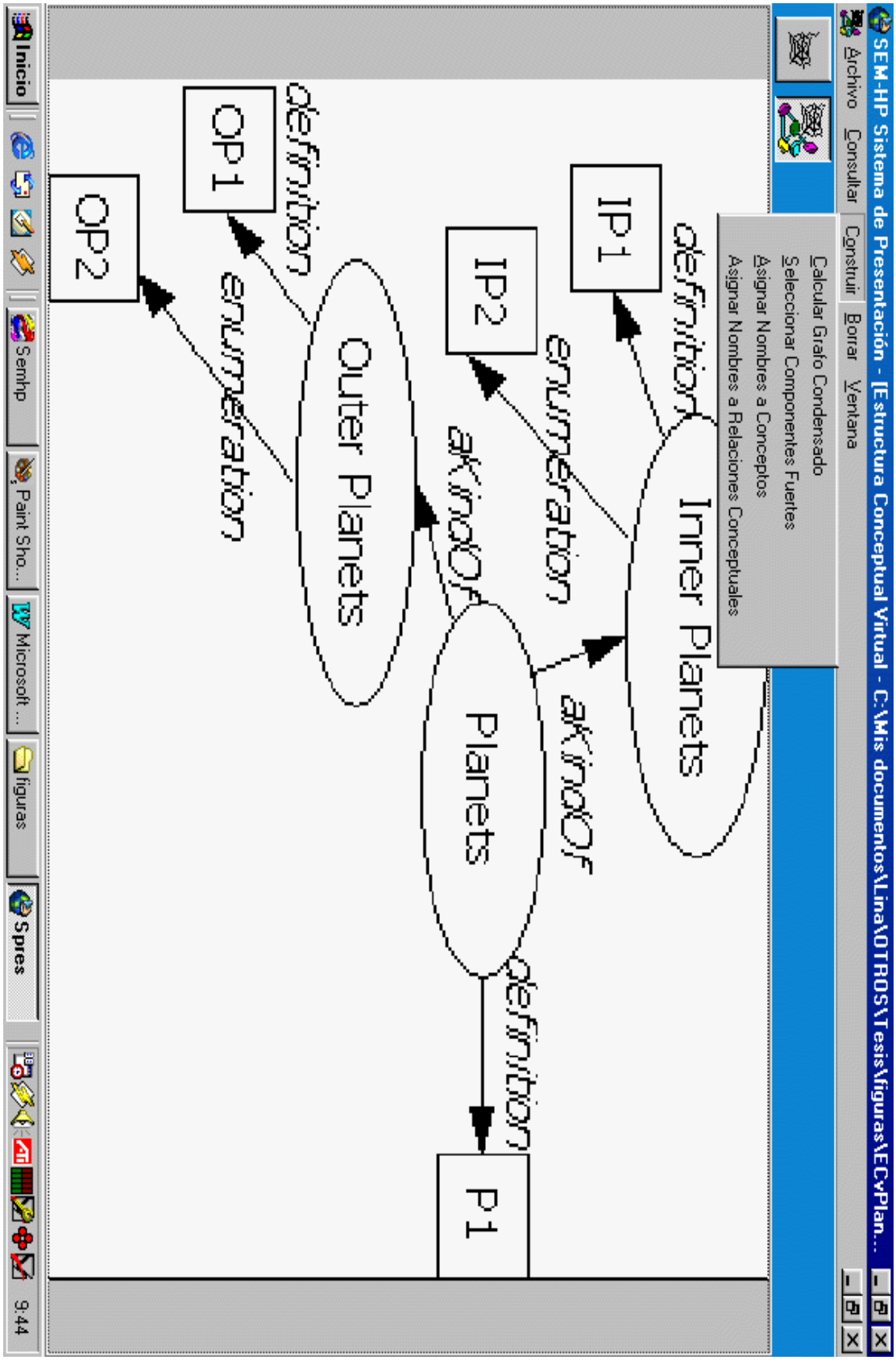


Figura 6.27 El Sistema de Presentación: Creando Estructuras Conceptuales Virtuales.

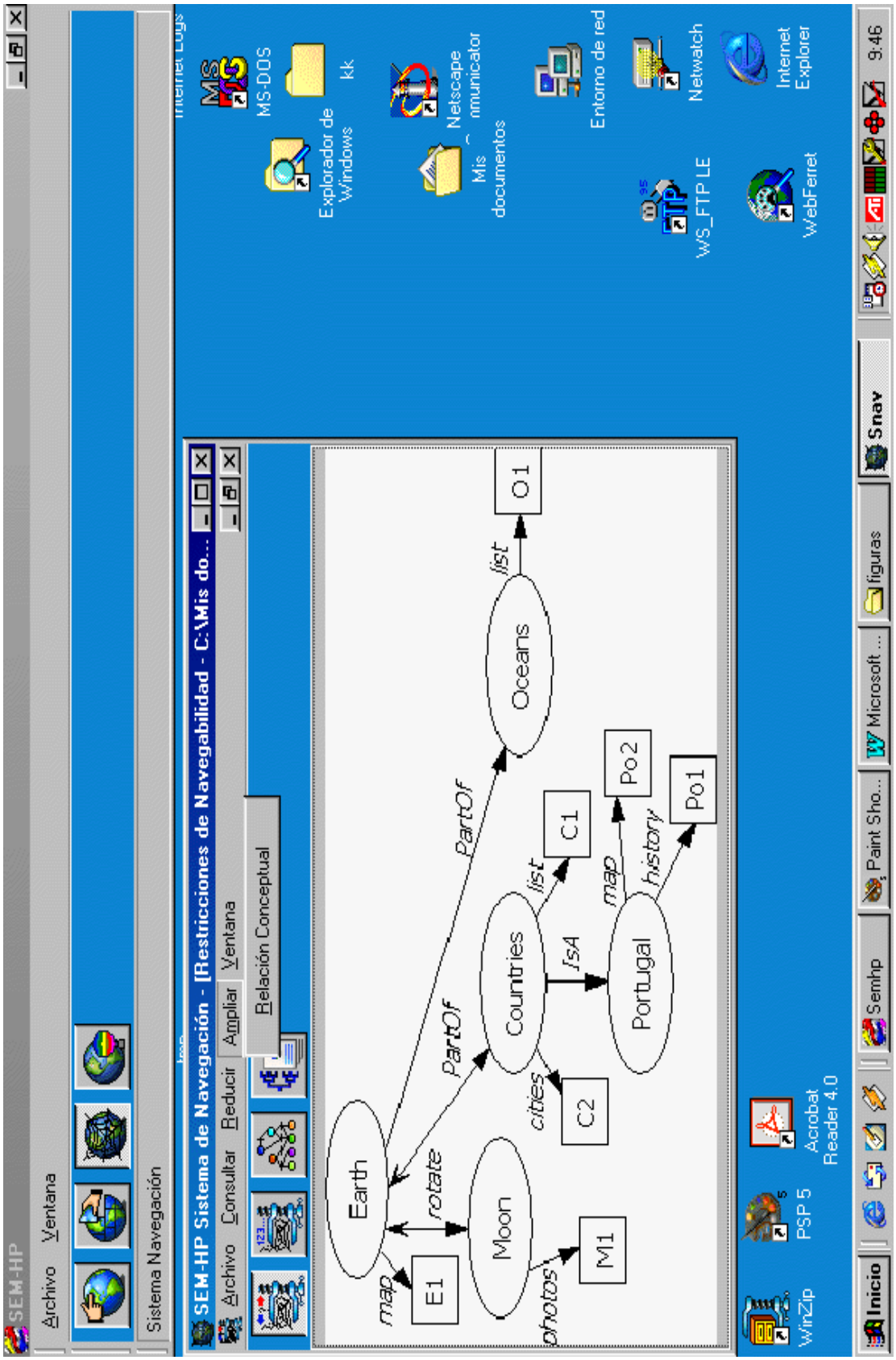


Figura 6.28 El Sistema de Navegación: Ampliando y reduciendo la navegabilidad de las relaciones conceptuales.

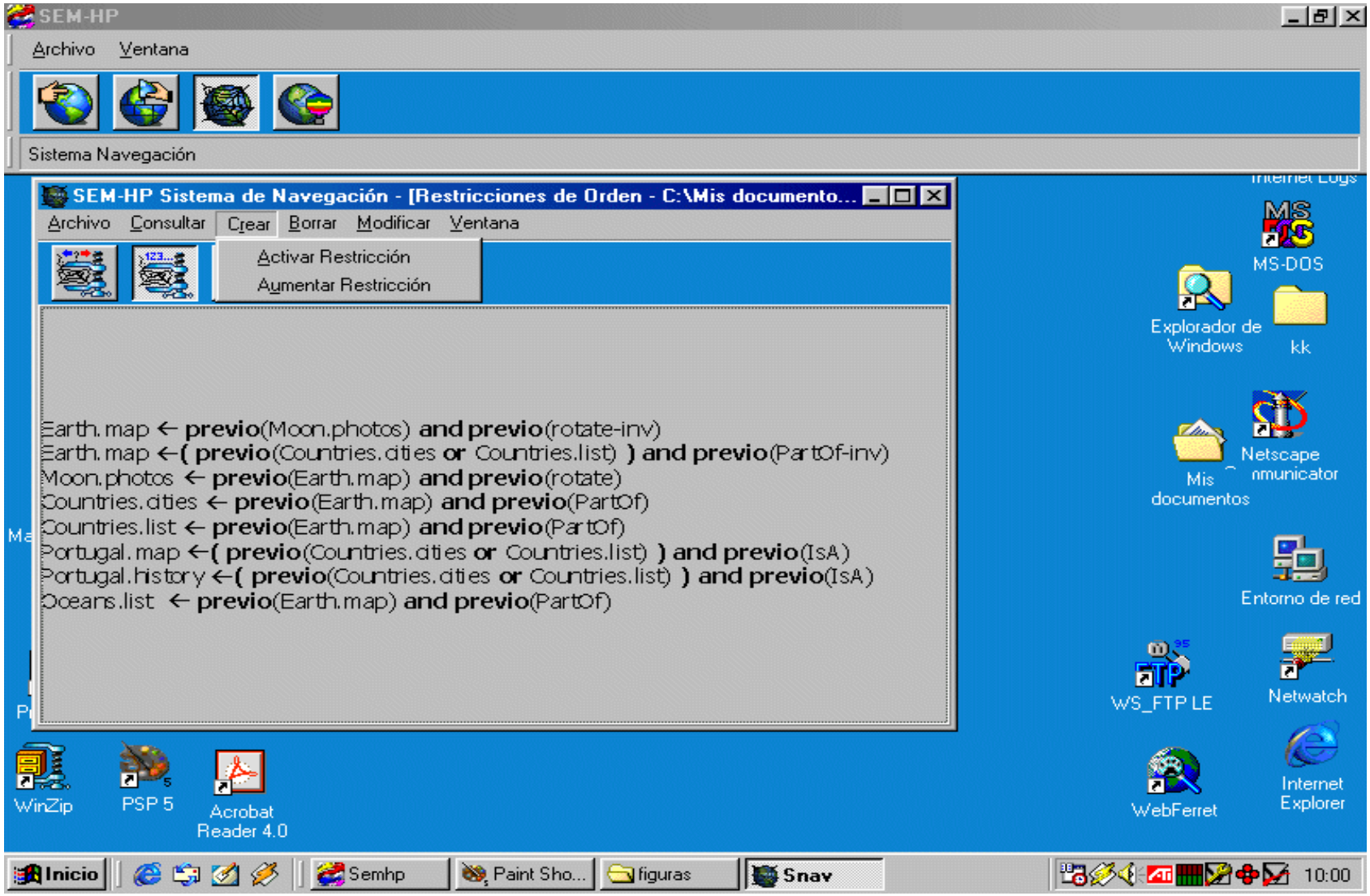


Figura 6.29 El Sistema de Navegación: Generando las Restricciones de Navegación.

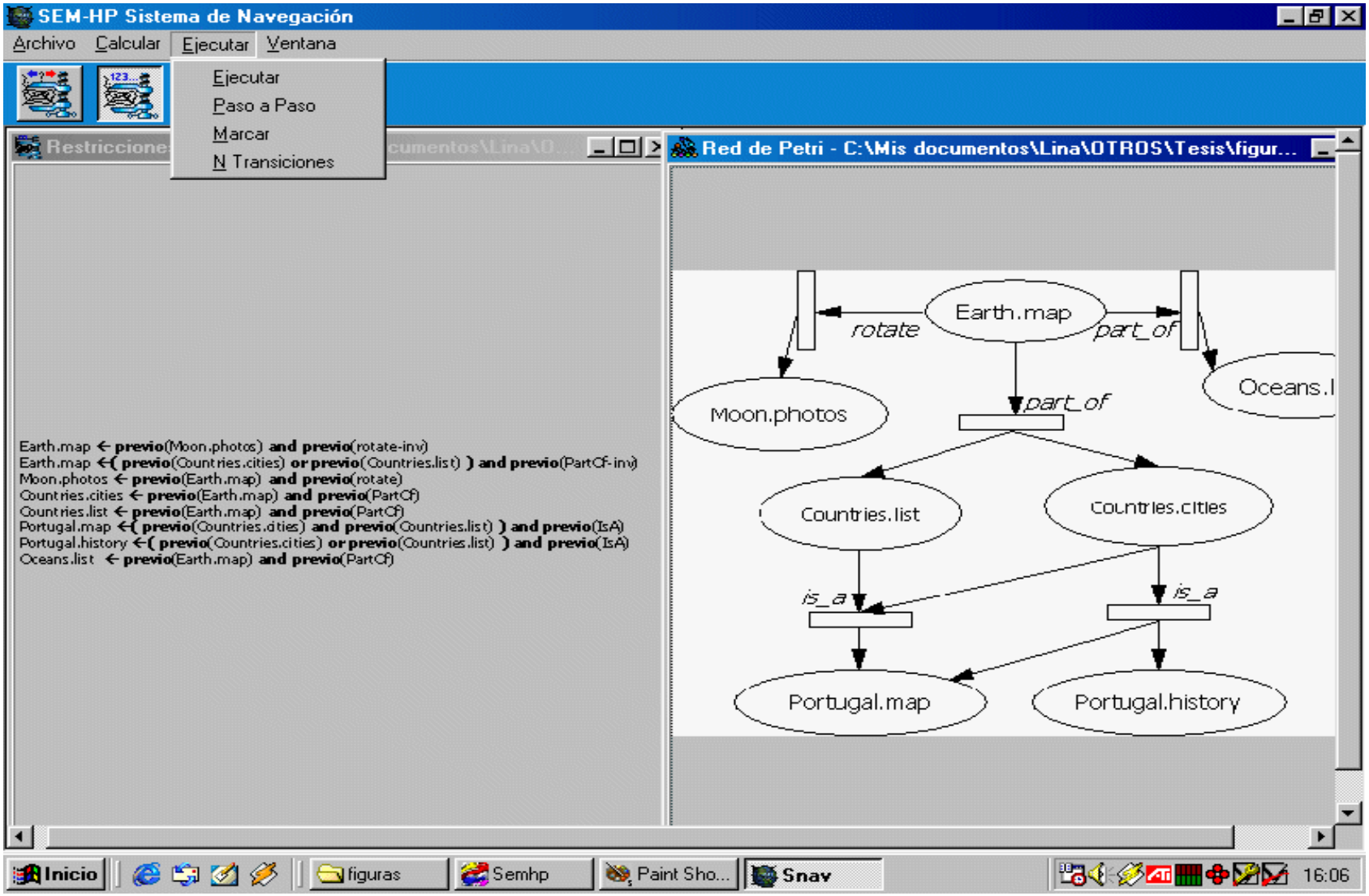


Figura 6.30 El Sistema de Navegación: Cambiando las Restricciones de Navegación y generando su red de Petri.

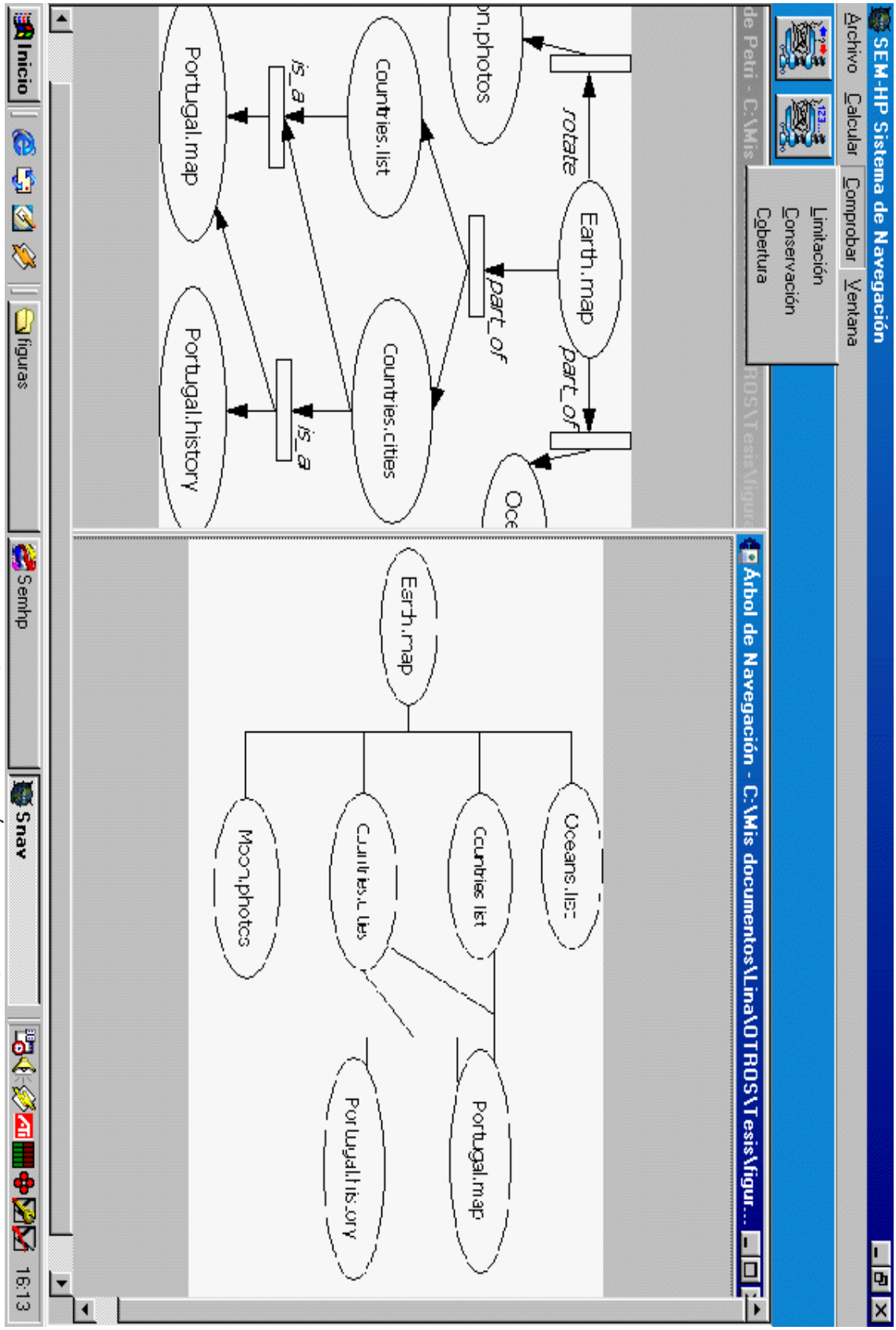


Figura 6.31 El Sistema de Navegación: Generando el Árbol de Navegación.

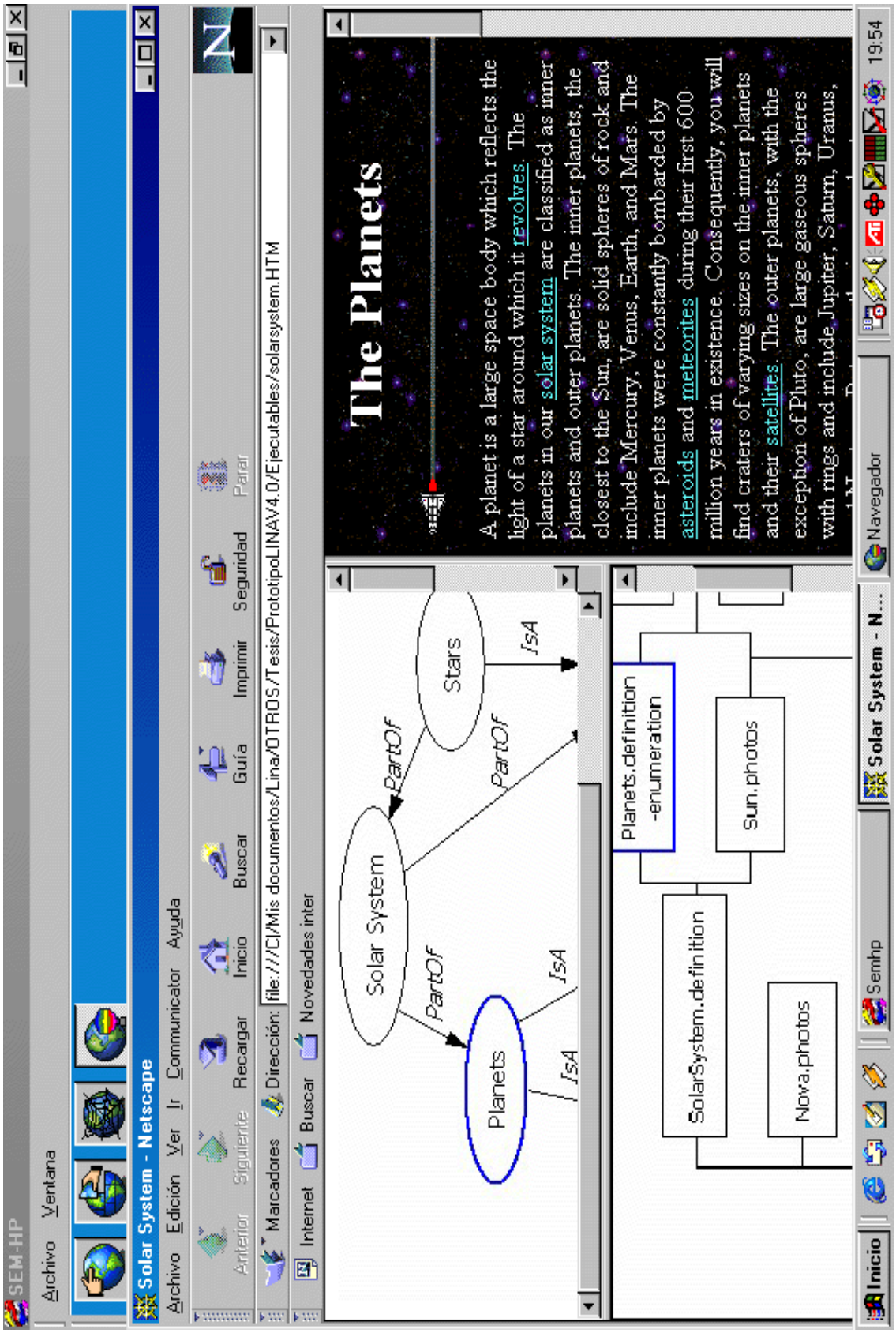


Figura 6.32 El Navegador: Tres posibles modos de interacción.

4 El Papel de XML en SEM-HP

XML —eXtensible Markup Language— [Bray, 1998] [Harold, 1998] [Goldfarb, 2000] es el nuevo talismán de las ciencias de la computación y, por supuesto, de la web. Este lenguaje de marcas con tan solo unos pocos años de vida —es un estándar del World Wide Web Consortium [W3C] desde febrero de 1998— fue desarrollado para salvar la distancia entre sus dos predecesores: HTML —Hypertext Markup Language [Gulbransen, 1998] y SGML —Standard Generalized Markup Language— [Goldfarb, 1990] y para convertirse en el nuevo lenguaje estándar de la web.

XLink, XPointer y XSL son otros lenguajes relacionados íntimamente con XML. XLink y XPointer definen enlaces y estructuras de direccionamiento para aplicarlas a documentos XML, y XSL define un lenguaje de estilo para especificar cómo se deben mostrar al usuario los documentos XML. Precisamente, el añadir estos lengujes separando el marcado, el hiperenlazado y los estilos es uno de los mayores logros de XML con respecto de HTML —tabla 6.1—.

Lenguajes de Marcas	Familia HTML	Familia XML	Familia SGML
Marcado	HTML	XML	SGML
Hiperenlazado	HTML	XLink + XPointer	HyTime, TEI
Estilo y presentaciones	CSS	XSL	DSSSL

Tabla 6.1 Equivalencias entre las familias de lenguajes HTML, XML y SGML.

En consecuencia, es necesario plantear las posibles implicaciones que puede tener esta familia de lenguajes XML en la representación e implementación del modelo que aquí se defiende. Y para ello, se dará respuesta, en primer lugar a las siguientes preguntas: ¿qué es XML? y ¿con qué tipo de enlazado cuenta XML?, para luego determinar cuál puede ser el papel que puede desempeñar XML en el modelo SEM-HP propuesto.

4.1 XML

Formalmente XML es un lenguaje de metamarcas con el que se pueden crear lenguajes de marcas para documentos concretos. Con este lenguaje se pueden estructurar datos en un fichero de texto y crear tipos de documentos —DTD, Document Type Definition— tales como una hoja de cálculo, un memorándum, un libro, una carta, un manual técnico, etc. Una definición de tipo de documento declara lo que es legal e ilegal dentro de un documento y en qué lugar es legal.

Dicho de otro modo, XML es un conjunto de reglas con el que se pueden diseñar *formatos de textos* que incluyen esos datos, esto es, formula etiquetas semánticas que dividen un documento en

partes y las identifica. A cada una de estas partes se las llama *elemento*. Los componentes de un documento se organizan mediante una estructura arborescente. El *elemento documento* o *elemento raíz* hace referencia al documento completo. Los elementos pueden tener información adicional con la que se describe sus propiedades a las que se denomina atributos. A veces no está claro qué aspectos de un documento deben representarse como elementos y cuáles como atributos.

En XML, a diferencia de HTML, las etiquetas sólo sirven para delimitar elementos de datos; la interpretación de los mismos se deja a la aplicación que los lee. Por ejemplo, si en un documento XML encontramos un texto etiquetado con <p> no se puede asumir que sea un párrafo, puede ser un «precio», un «parámetro», una «persona» o cualquier cosa cuya palabra comience por «p». XML puede incluir otros componentes multimedia pero no en formato XML sino en sus formatos nativos GIF, JPEG ó MPEG.

En consecuencia, XML supera a HTML porque con él se puede especificar cualquier dominio y se pueden estructurar los datos a los que se da formato, convirtiéndose así en un excelente mecanismo de intercambio de datos entre diferentes aplicaciones. Además, a diferencia de HTML, XML únicamente especifica el formato y no el estilo del texto —negrita, tamaño, color, etc— que queda a cargo del XSL —the eXtensible Style Language—.

4.2 El lenguaje de enlazado de XML (XML Linking)

El lenguaje de enlazado de XML [De Rose, 1999] pretende incrementar la funcionalidad hipertetextual de la Web. Intenta superar las limitaciones evidentes del enlazado simple, fijo y incrustado en la definición de HTML, convirtiéndolo en esencialmente estático. La especificación de la familia de lenguajes de enlazado —XPath, XPointer y XLink— no está aún terminada. XLink es el lenguaje que se ocupará del enlazado propiamente dicho, tiene la determinación de:

- crear e identificar distintos elementos de enlazado;
- adjuntar enlaces a los documentos sin tener que modificarlos;
- proporcionar enlaces externos, simétricos y multidestino; y
- separar la semántica de seguimiento de un enlace del tipo de enlace.

Por tanto, con XLink se podrán tener enlaces bidireccionales y alcanzar múltiples destinos. Estas direcciones podrán almacenarse en bases de datos y agruparse independientemente de los documentos a los que hacer referencia. Además, el atravesar un enlace puede implicar diferentes presentaciones, como por ejemplo:

- La sustitución de la página por la nueva a la que se salta.
- La apertura de una nueva ventana donde se muestra la nueva página.
- El incrustar la información dentro de la página

El despliegue de un enlace puede hacerse cuando el usuario lo requiera —tal y como sucede ahora— o bien ser automático, de manera que una página puede abrir automáticamente otra página. Estas dos últimas características se especifican en XLink mediante atributos de comportamiento.

4.3 SEM-HP y XML

Tal y como se ha descrito anteriormente, XML viene a superar las dos limitaciones más importantes de HTML:

- La imposibilidad de añadir nuevos tipos de elementos al lenguaje.
- La simplicidad y escasa versatilidad de sus enlaces.

¿Serán estas características suficientes para concluir que XML ha de ser el lenguaje de representación de SEM-HP?. Puede ser prematuro constestar a esta pregunta debido al estado de desarrollo en el que se encuentra XML, en especial los lenguajes de enlazado y estilos que aún no están ni siquiera especificados. La ausencia de aplicaciones de procesamiento adecuadas y de navegadores de XML en el mercado, plantea un claro inconveniente. En estos momentos, los documentos XML deben ser convertidos a HTML antes de ser distribuidos o bien ser traducidos sobre la marcha. La táctica más usual es escribir rutinas de analizador sintáctico en DHTML —Dynamic HTML— o Java para analizar los documentos XML, aplicarles reglas de formato mediante una hoja de estilo y convertirlos todo finalmente a HTML. La ventaja de esta limitación es que se superará con el tiempo.

Una segunda dificultad más seria que la anterior es que con XML sólo se pueden describir estructuras estrictamente jerárquicas en forma de árbol. Es lógico que sea así, por otra parte, ya que XML se ha creado para diseñar formatos de documentos y no para representar mundos conceptuales. XML estructura datos y no conceptos. Esto es, con XML podemos decir que un documento que tiene un elemento autor, un elemento dirección, un elemento e-mail y uno o más elementos párrafos es una carta, pero también podría ser un artículo. Se podría argumentar que entonces ese algo cuenta con una introducción y unas conclusiones, pero esto sólo es verdad si el artículo es científico o técnico. En cualquier caso, aunque por su formato se sepa el tipo de documento, no dirá necesariamente de qué trata, esto es, qué información nos está proporcionando y con qué propósito.

En definitiva, XML se centra más en aquellos metadatos que describen la estructura de edición del contenido —las distintas componentes lógicas de las que consta un ítem: encabezado, título, capítulo sección, etc— y en los que reflejan las propiedades de estos elementos —los autores, la fecha de creación, etc—. Sin embargo, no parece estar interesado por la semántica de estructuración, por los conceptos que representa un ítem de información —deporte, campeonato, jugadores, etc— y por la función o papel que puede desempeñar en el contexto —definición, introducción, explicación, enumeración, etc—.

Se podría argüir que esto no es necesario, que se puede encontrar cualquier documento con algún motor de búsqueda. Sin embargo, la recuperación basada en el contenido no siempre funciona. El contenido no está explícito, esto es evidente cuando el lenguaje de expresión no es el texto. Pero, aun siendo un texto escrito a veces ninguna palabra o frase del mismo realmente dice de qué trata el documento.

El problema fundamental es que normalmente no estamos interesados estrictamente en el contenido. Las palabras, imágenes o animaciones son representación de objetos, ideas y conceptos del mundo real. Una cosa es la representación, y otra su significado o semántica. El mismo concepto se puede representar o expresar de diferentes modos.

En consecuencia, hay que extraer el significado implícito de los ítemes de información y relacionarlos de acuerdo a las asociaciociones conceptuales que el autor considere necesarias. En nuestra opinión, la recuperación y la navegación debe estar más relacionada con los conceptos que con su

representación. En palabras de [Carr, 1998]: «*Link following is more than a user navigation issue: it is a fundamental mechanism for document construction*».

De acuerdo a este razonamiento, parece ser que el lenguaje más idóneo para representar las estructuras conceptuales sería XML pero acompañado de XLink del que desafortunadamente aún no se cuenta con su especificación final.

No obstante, en el modelo SEM-HP existen otros componentes tan importantes o más que las propias estructuras conceptuales: las Restricciones, Meta-Restricciones y las acciones evolutivas. Estos lenguajes no cuentan con ningún mecanismo con el que se puedan representar estas reglas lógicas, ni tampoco prevén ningún medio para especificar la propagación de los cambios para asegurar la integridad de los sistemas.

Resumiendo, no hay ningún inconveniente para que los ítemes que se ofrezcan estén especificados en XML y que sus enlaces lo estén en XLink. Pero serían una representación final, internamente se ha de trabajar con los formalismos propuestos para mantener y hacer evolucionar al sistema hipermedia.

5 Referencias

[Booch, 1999] 225
[Booch, 2000] 225
[Bray, 1998] 256
[Carr, 1998] 259
[De Rose, 1999] 257
[Golberg, 1983a, 1983b] 226
[Goldfarb, 1990] 256
[Goldfarb, 2000] 256
[Gulbransen, 1998] 256
[Harold, 1998] 256
[Paderewski, 1999] 225
[Rodríguez-Fórtiz, 1997, 2000a, 2000b, 2000c] 225
[W3C] 256

Parte III

Capítulo 7

Recapitulación

«La vida es el arte de sacar conclusiones suficientes a partir de datos insuficientes.»

Samuel Butler.
Escritor británico (1835-1902).

«La sabiduría muchas veces consiste en saber cuál es el siguiente paso que hay que dar.»

Herbert Hoover.
Político estadounidense (1874-1964).

«El futuro no pertenece a nadie. No hay precursores, sólo hay rezagados.»

Jean Cocteau.
Escritor y director francés (1889-1963).

Resumen

Se cierra esta memoria dedicando este capítulo a resumir de forma sucinta el trabajo realizado, haciendo mención a:

- las aportaciones que guardan ciertas similitudes con este trabajo,
- las conclusiones que se han extraído de la revisión realizada y las aportaciones del modelo SEM-HP al proceso de desarrollo de Sistemas Hipermedia, y
- finalmente, las líneas de trabajos y futuras proyecciones que se pueden derivar de esta aportación.

1 Trabajos Relacionados 263
1.1 El Sistema Hipertexto RICH 263
1.1.1 Representación de la Red Semántica 263
1.1.2 Comparativa 265
1.2 El Modelo de Referencia AHAM 266
1.2.1 Arquitectura de AHAM 266
1.2.2 Comparativa 268
1.3 Hipermedia como Autómatas 268
1.3.1 Comparativa 269
2 Conclusiones del trabajo 269
3 Problemas Abiertos 271
4 Referencias 273

1 Trabajos Relacionados

Aunque no se ha encontrado en la literatura técnica ningún grupo de investigación o trabajo que se dedique a contemplar todo el proceso de desarrollo de Sistemas Hipermedia desde una perspectiva evolutiva, sí existen algunos campos y propuestas concretas de investigación próximas que:

- Comparten algunos de los objetivos, soluciones y aplicaciones del modelo SEM-HP.
- Hacen uso de algunos de los formalismos empleados en SEM-HP.

Se analizan en este apartado estos trabajos relacionados, indicando en cada caso las analogías y las carencias que muestran con respecto al modelo SEM-HP. Estas propuestas no fueron estudiadas en el capítulo 2 porque no son modelos de referencia de carácter general sino aproximaciones con un ámbito de aplicación muy concreto.

1.1 El Sistema Hipertexto RICH

El Sistema Hipertexto RICH [Wang, 1998] —Reusable Intelligent Collaborative Hypermedia— es la propuesta de trabajo con la que más se puede identificar la aportación que aquí se presenta. El principal objetivo que persiguen Wang y Rada con este sistema es escribir documentos científico-técnicos y reutilizarlos para confeccionar otros nuevos.

Para ello plantean un modelo hipertexto estructurado basado en una red semántica que captura las caraterísticas estructurales y semánticas de los documentos técnicos con el que se puede llevar a cabo procesos de:

- Creación de un documento consistente en la captura, organización y representación del conocimiento sobre el dominio, para luego usar ese conocimiento en desarrollar componentes de documento reusables para producir nuevos documentos.
- Gestión del documento, esto es, realizar una taxonomía de los documentos.
- Recuperación del documento mediante navegación o búsqueda con filtros.
- Reorganización del documento para confeccionar con los componentes encontrados un nuevo documento.

1.1.1 Representación de la Red Semántica

Estos autores representan el modelo de datos semántico hipertexto mediante una red semántica o grafo dirigido acíclico con raíz, un conjunto de esquemas de composición, un conjunto de reglas para describir los conceptos del dominio y unas operaciones para construir y reutilizar documentos.

La red semántica que organiza el hipertexto hace uso de un conjunto de tipos de nodos, enlaces y reglas de asociación de alto nivel predeterminadas —tabla 7.1—. El usuario puede clasificar sus nodos con alguno de los subtipos que se agrupan en tres clases: *Tesauro*, *Documento* y *Anotación*. También puede definir sus propios subtipos para los nodos tipo *Documento* y tipo *Anotación*.

Los enlaces se pueden clasificar en dos grupos: de organización —*Tesauro*, *Documento* y *Anotación*— y de referencia —*RT*, *see-also* y subtipos definidos por el usuario—. Los primeros definen un grafo dirigido acíclico con una sóla raíz; para los segundos no existen restricciones estructurales. Sin embargo, RICH ignora por completo los enlaces de referencia y sólo hace uso de los jerárquicos para la creación, gestión, recuperación y reorganización de documentos.

Estos nodos y enlaces se pueden asociar de acuerdo a un conjunto de reglas generales que determina qué tripletas, <tipo de nodo, tipo de enlace, tipo de nodo> son admisibles. Según se puede observar —tabla 7.1— estas reglas construyen una red semántica en la se pueden distinguir tres niveles estructurales ligeramente relacionados:

- Una jerarquía de conceptos que forman un *tesauro* con el que se puede catalogar los documentos —enlazándolos con un término-preferido—.
- Una jerarquía de documentos a los que se puede enlazar comentarios.
- Una jerarquía de comentarios.

Nodos	**Enlaces**
Tipo TESAURO ■ Subtipo Término-preferido ■ Subtipo Término-no-preferido ■ Subtipo Espacio-de-trabajo WS	**Tipo TESAURO** ■ UF -desde término preferido a no preferido- ■ NT -entre términos preferidos- ■ WS -entre espacios de trabajo
Reglas	
<Término-preferido, UF, Término-no-preferido> <Término-preferido, NT, Término-preferido> <WS, WS ,WS>	
Tipo DOCUMENTO ■ Por defecto, Documento ■ Subtipos definido-usuario	**Tipo DOCUMENTO** ■ Documento -entre documentos- ■ DT -entre término preferido y documento-
Reglas	
<Documento, Documento, Documento> <Término-preferido, DT, Documento>	
Tipo ANOTACIÓN ■ Por defecto, Anotación ■ Subtipos definido-usuario	**Tipo ANOTACIÓN** ■ Comentario -entre documento y anotación o entre anotación y anotación- ■ Subtipos definido-usuario
Reglas	
<Documento, Comentario, Anotación> <Anotación, Comentario, Anotación>	
	Tipo REFERENCIA ■ see-also ■ Tesauro RT ■ Subtipos definido-usuario
Reglas	
<TESAURO, RT, TESAURO> <Documento, RT, Documento> <Anotación, RT, Anotación> <---, see-also, ---->	

Tabla 7.1 Tipos de nodos, enlaces y reglas generales del Sistema RICH.

1.1.2 Comparativa

El sistema RICH, aunque utiliza la tecnología hipermedia como mecanismo para la administración y reutilización de documentos científicos-técnicos, no parece tener como objetivo principal la construcción de hiperdocumentos. Sin duda por esta razón, no contempla todo el proceso de desarrollo de un sistema hipermedia, y muestra fuertes carencias tales como:

- No se distingue claramente entre dominio conceptual y de información. En la red semántica están representados tanto los índices estructurados de los documentos —que no corresponden a su estructuración conceptual sino a su edición final—, como sus comentarios y su clasificación con *tesauros*.
- Parten del texto para transformarlo en hipertexto. Para ello primero lo estructuran mediante un índice y referencias cruzadas entre estos índices. Luego, añaden anotaciones para facilitar la tarea de la comunicación entre los usuarios que comparten la base documental. Después, o al tiempo, clasifican estos documentos, o componentes del documento, mediante *tesauros*. Sin embargo, no utilizan estos *tesauros* para generar documentos o presentaciones de éstos ni, en absoluto, consideran la posibilidad de navegar estos documentos teniendo en cuenta su estructura semántica.

  El punto de vista de SEM-HP es muy distinto: todo ítem de información puede pertenecer a uno o más dominios conceptuales. Estas piezas son catalogadas mediante tales mundos conceptuales, se pueden componer y seleccionar para que formen parte de una determinada presentación que puede ser recorrida de acuerdo a la especificación que haya establecido el autor. En consecuencia, la estructuración semántica es crucial en SEM-HP, y es la que determina sus posibilidades de asociación entre los ítemes y de navegación.
- Las reglas estructurales y semánticas son muy simples y poco flexibles. Los enlaces de organización tienen que formar siempre un árbol acíclico con raíz. Las reglas semánticas son tripletas con la forma <tipo-nodo, tipo-enlace, tipo-nodo> en las que no se considera su entorno conceptual, esto es, los conceptos que alcanzan y que son alcanzados desde un determinado nodo. Por último, estas reglas sólo son tenidas en cuenta en la construcción; el autor no puede especificar distintos recorridos como sucede en SEM-HP con las denominadas Restricciones de Navegación.
- Existe cierta confusión entre las nociones *concepto* y *función* de un documento en un contexto. Los nodos se catalogan como *tesauro*, *documento* y *anotación*. En SEM-HP sólo existen dos tipos de nodos: conceptos e ítemes de información. Cada etiqueta de concepto para SEM-HP es un nuevo tipo. Los ítemes pueden ser asociados mediante funciones a los ítemes de información. De esta forma se puede:
  - Catalogar con más de un concepto a un ítem.
  - Tener más de un ítem asociado al mismo concepto pero con distinta función. Un ítem puede desempeñar el papel anotación para un concepto, pero también introducción, definición, explicación, etc.
- No definen propiedades para los ítemes de información. En SEM-HP se pueden tener variantes sobre un mismo ítem, esto es, cada ítem puede estar catalogado con los mismos conceptos y funciones pero expresados en otro idioma, con diferente nivel de especialización, etc.

- Finalmente, no contemplan el ciclo de vida completo de un Sistema Hipermedia, que consiste en el diseño, desarrollo y mantenimiento de dominios conceptuales y de información, presentaciones y navegaciones. En consecuencia, tampoco se ocupan de sus implicaciones evolutivas.

1.2 El Modelo de Referencia AHAM

AHAM —Adaptive Hypermedia Application Model— [Wu, 2001] pretende convertirse en el Modelo de Referencia para el desarrollo de Hipermedias Adaptativos. Estos sistemas tratan de ajustar la información al usuario y guiarlo en el hiperespacio presentándole únicamente el material que pueda resultarle más interesante teniendo en cuenta sus objetivos y preferencias.

Tradicionalmente, esta adaptación se ha basado en el *modelo de usuario* cuya misión es la representación de aspectos relevantes para el lector, como sus preferencias, conocimientos e intereses. Para ello, el sistema recoge información sobre el usuario observando el uso que hace de la aplicación y, en concreto, de la navegación. Por tanto, los Sistemas Adaptativos se encargarían de describir la funcionalidad que se le ha asignado al denominado Sistema de Aprendizaje de SEM-HP —que queda fuera de los objetivos de esta tesis—. No obstante, el modelo AHAM —cuya investigación está en curso— contiene interesantes innovaciones que convergen en cierto sentido con el resto de los Sistemas de SEM-HP.

1.2.1 Arquitectura de AHAM

Wu considera que la posible utilidad de la adaptación de una aplicación depende de que:

- la aplicación esté basada en un *modelo de dominio* que describa cómo se estructuran los contenidos de los hiperdocumentos mediante conceptos y relaciones entre conceptos.
- el sistema construya y mantenga un *modelo de usuario* detallado que represente sus preferencias, conocimientos, metas, historia de navegación, etc.
- el sistema permita realizar adaptaciones de los contenidos y de su estructura de enlazado —la navegación— basándose en los dos anteriores modelos. Para ello el autor debe contar con un *modelo de adaptación* que consiste en reglas que generan presentaciones personalizadas y que actualizan el *modelo de usuario*.

En este trabajo, Wu propone una arquitectura que describe la funcionalidad de los Hipermedia Adaptativos en el nivel conceptual. Esta aproximación se basa en el modelo Dexter [Halasz, 1990] —capítulo 2, página 43— para el que redefine su nivel de Almacenamiento. Considera que debe estar integrado por los anteriores modelos: de dominio DM, de usuario UM y de adaptación AM—figura 7.1—.

El *modelo de dominio* está formado por conceptos y relaciones entre conceptos:

- Un concepto representa un ítem de información del dominio de aplicación y puede ser simple o compuesto. Un concepto simple es un fragmento de información. Un concepto compuesto es una secuencia de subconceptos y un constructor que describe cómo se combinan. Estas composiciones deben formar un grafo dirigido acíclico.

- Las relaciones entre conceptos son objetos que unen dos o más conceptos. Cada relación tiene un tipo, que puede ser un enlace hipertexto normal o enlaces que describen relaciones que desempeñan un papel en la adaptación como: *prerrequisito*, *inhibición* y *parte-de*. Estos últimos tipos de relaciones no suponen la existencia de enlace entre esos conceptos, sólo que, por ejemplo, para visitar C_2 es necesario haber visitado C_1.

El *modelo de usuario* se representa mediante una tabla con parejas de entidades sobre los conceptos y valores. Este modelo asigna valores relacionados con los conocimientos, intereses y comportamiento del usuario durante su navegación por los conceptos. Wu y De Bra sugieren como ejemplos de entidades sobre un concepto: *conocimiento*, *leído* o *listo-para-leer*.

El *modelo de adaptación* incluye reglas para adaptar el contenido y los enlaces, y para actualizar el modelo de usuario. Pueden existir reglas genéricas que se apliquen a todos los conceptos o a todos los conceptos que estén relacionados con algún tipo determinado. Finalmente, también se pueden especificar reglas específicas que afectan a un concepto, conjunto de conceptos o relaciones entre conceptos concretas. Los autores en este artículo representan estas reglas de adaptación mediante reglas de Acción-Condición [Widom, 1996], que escriben siguiendo la sintaxis de SQL.

Un *motor de adaptación* es el responsable de realizar las adaptaciones en la presentación. La ejecución de las reglas de adaptación se activan cuando el usuario navega y provoca la actualización de valores en el *modelo de usuario* y el disparo de otras reglas de adaptación.

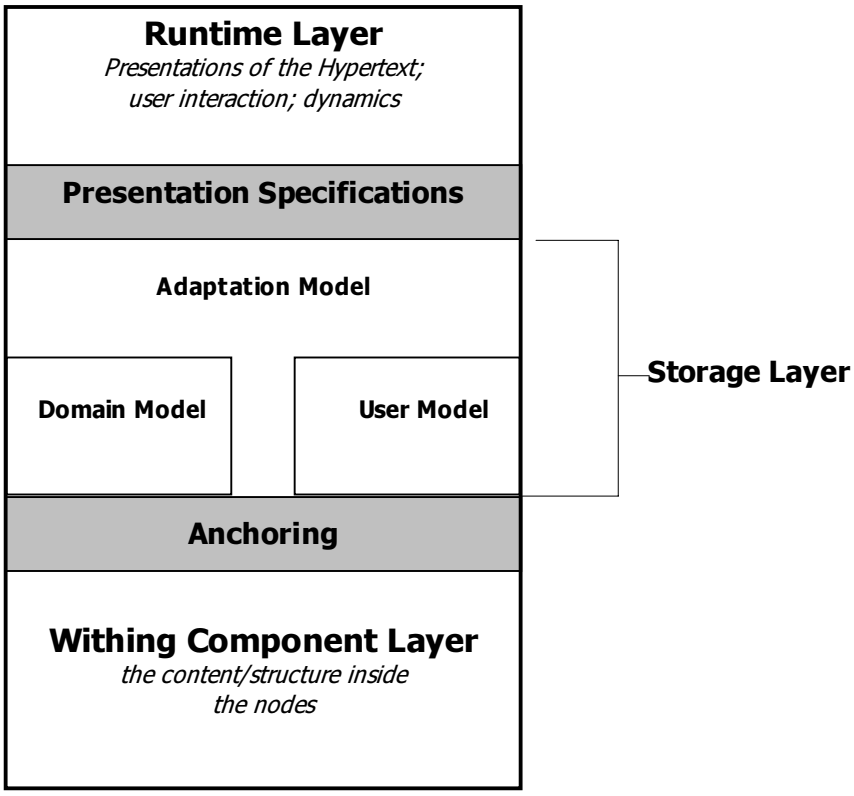


Figura 7.1 Arquitectura del modelo AHAM [Wu, 2001].

1.2.2 Comparativa

El modelo AHAM reconoce la bondad de incluir un *modelo de dominio* —que podría asemejarse a la Estructura Conceptual de SEM-HP—. Sin embargo, nuevamente, esta propuesta está mediatizada por un objetivo muy concreto: el lograr la personalización de las páginas al lector de forma automática durante el proceso de navegación. En consecuencia, no ofrece un entorno que dé soporte a todo el proceso de desarrollo de Sistemas Hipermedia. En particular, se pueden señalar los siguientes inconvenientes:

- El *modelo de dominio* mezcla en la misma representación los enlaces que existen entre las páginas o fragmentos de páginas y las posibles restricciones de navegación que ellos expresan con tipos de relaciones como *prerrequisito*, *inhibe* o *parte-de*.
- Los enlaces normales no reflejan en absoluto la estructuración conceptual del documento sino su estructura de edición.
- No se incluyen guías o controles —Restricciones de Autor en terminología SEM-HP— para garantizar la consistencia del grafo que representa el modelo de dominio.
- No ofrecen tampoco ningún tipo de mecanismo para seleccionar una parte del dominio conceptual y de información que se tiene en el Sistema. Seguramente esto sea así porque parten de sistemas hipermedia concretos y no de mundos conceptuales que determinan dominios de información.
- Los mismos autores reconocen que existe un gran riesgo de definir reglas —Restricciones de Navegación en terminología SEM-HP— que desencadenen ciclos infinitos de evaluación y ejecución de reglas. Para paliar este problema, proponen un par de limitaciones y órdenes a la hora de evaluar las reglas. No obstante, admiten que no es suficiente y que necesitan algún mecanismo que detecte la no terminación y la falta de confluencia, esto es, la posibilidad de que se generen resultados impredecibles. En definitiva, no han incorporado Meta-Restricciones que garanticen la consistencia de las reglas de adaptación.

En resumen, aunque en este trabajo los autores coinciden con nosotros en separar la distintas funciones en distintos modelos, no incorporan mecanismos para controlarlos y hacerlos evolucionar.

1.3 Hipermedia como Autómatas

Stotts y Furuta [Stotts, 1991] también han contribuido significativamente en el campo de los Sistemas Dinámicos o Adaptativos. Proponen el uso de estructuras virtuales que describen como adaptaciones dinámicas de la estructura hipertexto. Para calcularlas, se debe recoger la información de interacción del usuario con el sistema hipertexto. Con esta información se hacen inferencias y decisiones y se crean los cambios físicos apropiados en el documento en el momento adecuado.

Estos autores [Stotts, 1998], consideran que un hipermedia es un *documento interactivo* que proporciona una *estructura dinámica*. En consecuencia, se puede asociar a un documento un grafo dirigido —una máquina de estados finitos— que muestre sus diferentes enlaces y que determine las páginas que se van a mostrar y los botones —enlaces— que se pueden seleccionar en cada momento o estado de su recorrido.

Para hacer consultas sobre el estado actual y los estados que se pueden alcanzar a partir de uno dado generan un árbol de cobertura al que denominan *comprobación-del-modelo —model-checking—*. Las preguntas sobre el hiperdocumento se realizan en un lenguaje de la lógica temporal ramificada llamado HTL —Hypertext Temporal Logic—.

1.3.1 Comparativa

Al igual que SEM-HP, en este trabajo se proponen las redes de Petri y la lógica temporal como formalismos para comprobar las propiedades de navegación del hipermedia, esto es, si los recorridos diseñados se comportan tal y como espera el diseñador. Sin embargo, su aproximación tiene un enfoque diferente:

- Parten directamente de la red de Petri y no, como en SEM-HP, de una representación en lenguaje temporal —los lenguajes **N** y **MN**—.
- No se trabaja con la red de Petri para hacer las consultas, sino sobre su grafo de cobertura.
- Utilizan la lógica temporal ramificada para escribir las preguntas sobre las propiedades del grafo. En SEM-HP se utiliza lógica temporal proposicional para hacer consultas y también para generar la estructura de la red de Petri.
- Cuando se cambia el hiperdocumento se crea una nueva red de Petri. No ofrecen ningún mecanismo para hacer esa evolución, para garantizar la consistencia del cambio ni para propagarlo.

2 Conclusiones del trabajo

Se resumen en este apartado las principales conclusiones y aportaciones que se pueden colegir del trabajo realizado, destacando, por un lado, las reflexiones que se derivan del análisis del marco teórico y, por otro, las características y bondades del modelo SEM-HP propuesto.

1. Se ha realizado una profunda revisión de los modelos tradicionales de Referencia, poniendo de manifiesto su incapacidad para la representación de los procesos de desarrollo, mantenimiento y navegación de un Sistema de Información en continua evolución.
2. Se han buscado las causas de este fracaso, llegando a la conclusión de que:
    a) estos modelos se ocupan de crear hiperespacios y no de espacios conceptuales, desasistiendo con su falta de estructuración semántica el proceso de desarrollo de un Sistema Hipermedia.
    b) carecen por completo de mecanismos que den soporte al proceso de evolución inherente a todo Sistema Hipermedia.
3. Se han identificado tres fases principales en el proceso de desarrollo de Sistemas Hipermedia, dedicadas a la construcción de dominios conceptuales y de información, a la selección de presentaciones y al diseño de navegaciones.

4. Se ha propuesto un modelo al que se ha denominado SEM-HP —un modelo **S**istémico, **E**volutivo y **SEM**ántico para el diseño de Sistemas **HiP**ermedia— compuesto por un conjunto de Sistemas interrelacionados y en interacción a los que se ha llamado Sistema de Conocimiento —formado por los Subsistemas de Memorización y Presentación—, Sistema de Navegación y Sistema de Aprendizaje, con la siguientes características:
   - El modelo facilita y hace flexible la construcción, el mantenimiento y la navegación de los sistemas hipermedia. Estos tres procesos clave estarán expuestos a continuos cambios y adaptaciones y el modelo es capaz de integrarlos.
   - La representación explícita de la semántica de estructuración impregna el modelo, ya que es la que posibilita su estructuración y la que determina su posible cambio, adaptación y evolución. Esto es, el proceso de diseño y construcción está basado en un modelo cognitivo [García-Cabrera, 2000] [García-Cabrera, 2001b].
   - Beneficia a sus usuarios —autor y lector—, durante el desarrollo y actividades de uso: construcción, mantenimiento y navegación, puesto que:
     - El autor puede realizar un proceso de diseño incremental de sus hiperdocumentos.
     - La colaboración entre autores se convierte en una realidad.
     - Se puede realizar un buen mantenimiento debido a que el proceso de diseño y razonamiento subyacente llevado a cabo puede ser representado.
     - El lector cuenta con un acceso contextualizado que facilita la comprensión del hiperdocumento.
   - El modelo ofrece una representación semántica flexible con la que el autor puede caracterizar sus propios dominios de información mediante sus propias ontologías.
   - Cada uno de los Sistemas del modelo SEM-HP sigue la arquitectura de dos niveles de abstracción de MEDES integrada por:
     - un Meta-Sistema o Sistema de Desarrollo que se dedica a construir la ESTRUCTURA del Sistema Hipermedia, y
     - el Sistema Hipermedia que ofrece una determinada FUNCIONALIDAD mediante la ejecución de los componentes de su ESTRUCTURA.
   - El Sistema de Desarrollo asiste al autor durante el proceso de evolución —construcción, adaptación y mejora— de los Sistemas mediante cuatro mecanismos fundamentales: Acciones Evolutivas, Restricciones, Propagación del Cambio y Meta-Restricciones.
5. Se ha realizado una representación formal del Subsistema de Memorización, del Subsistema de Presentación y del Sistema de Navegación del modelo SEM-HP. Los distintos formalismos permiten modelar el sistema de información, la estructura conceptual de ese dominio de información, seleccionar una determinada presentación y, por último, construir la navegación que se quiere realizar sobre la presentación elegida. En particular se ha empleado:
   - la teoría de grafos —y la teoría de conjuntos en la que se basa— para formalizar la Estructura Conceptual. Un grafo es el mecanismo más adecuado para representarlo porque ofrece una gran versatilidad debido a sus posibilidades de navegabilidad e interacción visual.

- la lógica temporal proposicional para describir las Restricciones, debido a que con este formalismo es muy fácil expresar relaciones de orden.
- lógica temporal de predicados para especificar las Meta-Restricciones, porque necesitamos subir un nivel de abstracción.
- la teoría sobre redes de Petri para convertir las Restricciones de Navegación en un formalismo operacional con el que se pueda ejecutar una navegación y analizar sus propiedades.

Con estos formalismos se han podido especificar los mecanimos de evolución que nos permiten hacer cambiar la ESTRUCTURA de los Sistemas y mantener su integridad.

6. Se ha desarrollado una especificación de la herramienta basada en el modelo SEM-HP mediante el Lenguaje de Modelado Unificado —UML—.
7. Con objeto de demostrar la viabilidad y mostrar la funcionalidad de la aplicación, se ha construido un prototipo de las distintas interfaces de la herramienta basada en el modelo SEM-HP.

3 Problemas Abiertos

Un hombre es sabio mientras busca la sabiduría; si llega a creer que lo ha entendido, se convierte en idiota.

Proverbio árabe

Ninguna investigación que se precie puede darse por terminada, y mucho menos en los dos campos en los que se enmarca esta Tesis Doctoral: la Evolución y los Sistemas Hipermedia. Antes al contrario, ambas áreas demandan un continuo proceso de indagación, desarrollo y maduración.

Se enumeran en este apartado mejoras y problemas abiertos que pueden tener un futuro desarrollo, pues pueden originar aplicaciones prácticas o estudios teóricos de interés.

Implementación

Uno de los objetivos más inmediatos, en el que se está ya trabajando, es el desarrollo completo de un producto software real que se base en el modelo SEM-HP. Seguramente, la confección de la herramienta nos permitirá evaluar de forma práctica el modelo y comprobar si los autores consiguen ser más productivos y construir hipermedias de más calidad.

Además, la implementación abre la posibilidad de realizar estudios estadísticos de uso de la aplicación y de comparar el Sistema SEM-HP cuantitativa y cualitativamente con otros productos.

7 Recapitulación

El Sistema de Aprendizaje

Una de las posibles continuaciones de esta aportación sería completar el modelo SEM-HP con el Sistema de Aprendizaje. Habría que acometer la completa especificación y formalización de este Sistema. Para su desarrollo se debería tener en cuenta, sin duda, las aportaciones realizadas en los Sistemas Adaptativos, y cómo no, la de sus precursores, los sistemas de usuario adaptativos. La incorporación del Sistema de Aprendizaje ampliaría sin duda la funcionalidad de SEM-HP:

- La presentación de un dominio conceptual implicaría imponer restricciones al dominio conceptual y a los ítemes de información. Dependiendo de las restricciones que impongamos se ofrecería una u otra presentación. Si se contara con un Sistema de Aprendizaje, el autor podría ofrecer un conjunto de presentaciones para un mismo dominio conceptual, pudiéndose caracterizar un tipo de navegación con unas restricciones específicas o hacerlas depender de la categoría o puesto del usuario, de sus conocimientos, de su edad, etc.
- La navegación se podría adaptar haciéndola depender de las metas y de los logros que el lector va comunicando durante una sesión de trabajo.
- El comportamiento del lector podría servir de información de retroalimentación para el Sistema de Navegación, Subsistema de Presentación y Memorización provocando la reconsideración de las navegaciones, presentaciones y estructuras conceptuales diseñadas.

Restricciones Asociativas

Las Restricciones relacionadas con las relaciones conceptuales han sido expresadas mediante la Teoría de Grafos. Estas Restricciones han sido reducidas a un total de dieciséis posibles caracterizaciones, pudiendo el autor asociar un subconjunto de las mismas a cada relación conceptual.

Estas Restricciones de Autor no tienen en cuenta las conceptos que relacionan. Se podría ofrecer un lenguaje basado en lógica temporal parecido a los empleados en las Restricciones de Navegación. Con este lenguaje se proporcionaría un mecanismo capaz de expresar cualquier tipo de limitación relacionada con las asociaciones. En este caso, el autor podría indicar qué conceptos o conjuntos de conceptos pueden asociarse y mediante qué relaciones conceptuales.

Si bien la especificación de este lenguaje no entraña ningún tipo de inconveniente —en el Apéndice D se hace una primera aproximación—, sí es un reto acometer el diseño de las Meta-Restricciones que han de garantizar, en todo momento, la consistencia de estas Restricciones Asociativas.

Composición de Estructuras Conceptuales

Un interesante reto teórico y práctico sería aumentar la funcionalidad del Subsistema de Memorización para que dé soporte a la unión, división y mezcla de Estructuras Conceptuales. Los ítemes de información pueden pertenecer a más de una Estructura Conceptual y un mismo concepto puede aparecer en más de una Estructura Conceptual. Esta posibilidad plantea interesantes preguntas:

- Los conceptos con igual nombre en dos Estructuras Conceptuales, ¿son distintos?, ¿son iguales?. Por ejemplo, si son iguales, un ítem que esté asociado a un concepto debe estar asociado a todas sus apariciones de cualquier EC. Entonces, tendríamos repetida la misma tripleta <c, r_f, ii> en el Sistema, pero cada una de ellas pertenece a una Estructura Conceptual distinta.

- ¿cómo se puede crear una nueva Estructura Conceptual a partir de otra?, ó ¿cómo fundir dos ECs que tengan conceptos comunes? Esto es, habría que reflexionar detenidamente sobre cómo se mezclan los dominios conceptuales.

Recuperación de Información

Todos hemos sufrido las búsquedas en la Web. Devuelven un volumen desmesurado de desaciertos, de informaciones que ni lejanamente tienen algo que ver con lo que se está demandando. El usuario se ve obligado a realizar continuos filtrados manuales que le suponen demasiado esfuerzo.

Por tanto, otra propuesta de trabajo sería la elaboración de un lenguaje de consulta y de mecanismos de recuperación —mediante agentes cooperativos— que se basen en los dominios conceptuales a los que pertenecen. Esto es, proporcionar al lector una herramienta que sea capaz de encontrar aquellos ítemes de información cuya semántica —representación del conocimiento— se ajusta de modo preciso al tipo de información que anda buscando.

Un modelo SEM-HP Cooperativo

Una vez que se completen las investigaciones de otros miembros del grupo GEDES sobre Sistemas de Soporte a la Decisión y Sistemas Cooperativos, se pueden aplicar estas aportaciones al modelo SEM-HP para convertirlo en un modelo válido para construir herramientas que den soporte a la coordinación de Sistemas Hipermedia Cooperativos.

4 Referencias

[García-Cabrera, 2000] 270
[García-Cabrera, 2001b] 270
[Halasz, 1990] 266
[Stotts, 1991] 268
[Stotts, 1998] 268
[Wang, 1998] 263
[Widom, 1996] 267
[Wu, 2001] 266

Apéndice A

Glosario de Términos de SEM-HP

Las **Acciones Evolutivas** son el conjunto de operaciones que puede cambiar los artefactos y restricciones de los distintos Sistemas. Una acción de evolución debe verificar un conjunto de Restricciones y, de esta forma, garantizar la consistencia del Sistema y llevar a cabo la propagación del cambio dentro y fuera del Sistema.

Una **Asociación Conceptual** es una relación etiquetada entre dos conceptos.

Una **Asociación Conceptual de Dependencia** es una asociación etiquetada entre dos conceptos que es independiente del Dominio Conceptual considerado {*PartOf*, *aKindOf*, *IsA*}.

Una **Asociación Conceptual de Referencia** es una asociación etiquetada entre dos conceptos, dependiente de un Dominio Conceptual y, por tanto, son definidas por el autor.

Una **Asociación Funcional** es una relación etiquetada con un *rol* o función entre un concepto y un ítem de información. De este ítem se dirá que pertenece al Dominio de Información que determina el Dominio Conceptual que incluye a ese concepto.

Un **Concepto** es una idea, pensamiento o abstracción que puede ser etiquetado por el autor con el fin de hacer explícito su conocimiento y hacerlo comprensible. De manera breve, es una idea etiquetada.

Un **Dominio Conceptual** es el conjunto conceptos con los que se pueden identificar los distintos ítemes de información de un sistema hipermedia y el conjunto de asociaciones que se pueden establecer entre estos conceptos, esto es, el conjunto de Asociaciones Conceptuales de una Estructura Conceptual.

Un **Dominio de Información** es el conjunto de ítemes de información identificados con conceptos que pertenecen a un Dominio Conceptual concreto, esto es, el conjunto de Asociaciones Funcionales de una Estructura Conceptual.

Un **Entorno Conceptual** es el conjunto de conceptos relacionados con un determinado concepto, esto es, que alcanzan o son alcanzados por el mismo.

El **Entorno de un Ítem de Información** es el conjunto de restricciones que se cumplen en el momento de alcanzar dicho ítem y que delimita o restringe el conjunto de ítemes de información que se puede alcanzar desde el mismo.

La **Estructura Conceptual** de un Dominio Conceptual es una red o grafo etiquetado cuyos nodos son conceptos e ítemes, y en la que se mantiene información sobre:

a) las asociaciones entre conceptos, y

b) las asociaciones entre conceptos e ítemes de información o asociaciones funcionales.

Un ítem de información puede desempeñar diferentes ***Roles*** o **Funciones** en el contexto de un sistema de información. Desde el punto de vista del autor, un ítem puede realizar o desempeñar un determinado *rol* en el contexto. Desde la perspectiva del lector, éste sigue un enlace con la intención de alcanzar un determinado tipo de información sobre un concepto concreto.

Un Documento Hipermedia o **Hiperdocumento** construido con SEM-HP es un subconjunto de ítemes de información y de asociaciones posibles entre los mismos que determina el Sistema de Navegación, de acuerdo al conjunto de restricciones que se verifican en cada instante.

Un **Ítem de Información** es cualquier trozo de información identificable en un sistema hipermedia y que representa una unidad conceptual del sistema de información. Cada uno de estos ítemes de información tiene una serie de propiedades que describen el tipo y funcionalidad de la información que contienen.

Un **Logro** es el conjunto de conocimientos que quiere adquirir o aprehender un lector, y que pueden ser definidos a partir de la Estructura Conceptual.

Una **Meta** es el conjunto de ítemes de información que pretende alcanzar el lector.

Las **Meta-Restricciones** son restricciones definidas *a priori* y que garantizan la consistencia de las Acciones Evolutivas que cambian las Restricciones de Autor.

El **Nivel de Especialización** o versión es una de las propiedades asociadas a un ítem de información. Determina el grado o nivel de especialización con el que se expresa o presenta la información contenida en el ítem.

Se utiliza la expresión **Propagación del Cambio** para hacer referencia a los tres tipos de actuaciones —comprobación de la consistencia del cambio, propagación interna y propagación externa— que se desencadenan cuando el autor realiza una Acción Evolutiva.

Una **Propiedad** de un ítem de información es aquella información que está asociada a un ítem de información y que describe el tipo de información que contiene, su función y su comportamiento.

Las **Restricciones** son el conjunto de condiciones o reglas que limitan las asociaciones conceptuales y de navegación. Garantizan la consistencia de los distintos componentes de los Sistemas y realizan la función de precondiciones para las acciones de evolución.

Las **Restricciones de Autor**, RT_a, son el subconjunto de restricciones establecidas por el autor del Sistema Hipermedia y que pueden ser modificadas en cualquier momento.

Las **Restricciones Estructurales** son aquellas que delimitan las asociaciones entre conceptos que se pueden efectuar en la Estructura Conceptual. Pueden ser impuestas tanto por el sistema como por el autor desde el Sistema de Conocimiento.

Las **Restricciones de Navegación** son aquellas que restringen las asociaciones entre ítemes de información que nos podemos encontrar durante la navegación. Pueden ser especificadas por el autor desde el Sistema de Navegación y deben respetar la Estructura Conceptual en la que se basan.

Las **Restricciones de Sistema**, RT_s, son el subconjunto de restricciones impuesto a priori y que no cambia.

El **Sistema de Aprendizaje** es uno de los Sistemas del Modelo SEM-HP que modelan un sistema hipermedia. Permite la evaluación y la modificación del Sistema de Navegación teniendo en cuenta, en cada instante, las metas o los logros que fijan los lectores.

El **Sistema de Conocimiento** es uno de los Sistemas del modelo SEM-HP que modelan un sistema hipermedia. Proporciona los ítemes de información, su catalogación y las reglas básicas que establecen sus posibles asociaciones. Está formado por los *Subsistemas de Memorización* y *Presentación*.

Un **Sistema Hipermedia** en SEM-HP, es un conjunto de sistemas interrelacionados y en interacción denominados Sistema de Conocimiento —formado por los *subsistemas de Memorización* y *Presentación*—, Sistema de Navegación y Sistema de Aprendizaje que permiten:

a) realizar una construcción y mantenimiento flexible de documentos hipermedia;

b) proporcionar una representación de la estructura conceptual del mismo;

c) ofrecer más de una presentación del sistema de información del conjunto de las posibles; y

d) una navegación dinámica en dónde es posible una navegación multidestino, multipropósito y con contextualización estructural.

El **Subsistema de Memorización** es uno de los dos subsistemas del *Sistema de Conocimiento*. Establece la materia prima con la que se ha de construir el sistema hipermedia con un componente fundamental: la Estructura Conceptual.

El **Sistema de Navegación** es uno de los Sistemas del Modelo SEM-HP que modelan un sistema hipermedia. Restringe o filtra el conjunto de las presentaciones posibles para optar por un subconjunto de las mismas, haciendo uso de las restricciones aportadas por el *Subsistema de Presentación* y las suministradas por el propio *Sistema de Navegación*.

El **Subsistema de Presentación** es uno de los dos subsistemas que modelan el *Sistema de Conocimiento* de un sistema hipermedia. Permite seleccionar un subconjunto de una Estructura Conceptual con el fin de determinar el documento hipermedia que se mostrará mediante el *Sistema de Navegación*.

Apéndice B

Demostración de la Consistencia de RT_A

1 Introducción

En todo momento, el conjunto de restricciones asociado a una relación conceptual debe ser consistente. Cualquier combinación de restricciones no tiene por qué ser válida. Por tanto, hay que determinar el conjunto de combinaciones válidas y, en consecuencia, consistentes. Antes de añadir o borrar una de las restricciones que caracterizan a una relación conceptual se debe comprobar nuevamente que el conjunto de restricciones es consistente. Para ello, únicamente deberemos comprobar que es una de las combinaciones válidas.

Se ha optado por una demostración computacional que ha consistido en los siguientes pasos:

1. Generar con un algoritmo todas las combinaciones posibles que se pueden generar con las 16 restricciones relacionales —se pueden hacer hasta 2^{16} caracterizaciones posibles a una relación conceptual—.
2. Rechazar aquellas combinaciones que son inconsistentes —se ha conseguido reducir el anterior número a un total de 290 combinaciones a priori válidas—.
3. Demostrar que las combinaciones que han quedado después de eliminar las no válidas son consistentes.

Para alcanzar el objetivo del paso 2:

1. Se han establecido reglas de inclusión e incompatibilidad entre algunas de las restricciones.
2. Se ha demostrado la veracidad de estas reglas.
3. Se ha escrito un algoritmo que elimina aquellas combinaciones no válidas, esto es, que contienen estas restricciones incluyentes o incompatibles.

Finalmente, para demostrar que las combinaciones que han resultado del anterior proceso son válidas se ha actuado de la siguiente manera:

1. Dado el gran número de combinaciones —un total de 290— que hace inviable en la práctica la demostración individual, se ha optado por agruparlas en representantes.
2. Se han demostrado las reglas representantes que permiten resumir las combinaciones válidas.
3. Con los representantes se han establecido reglas o axiomas de consistencia.
4. Se han demostrado estas reglas de consistencia.
5. Se ha escrito un algoritmo que comprueba que todas las combinaciones válidas no eliminadas contienen un subconjunto de estas reglas válidas.
6. Por último, se ha comprobado que este conjunto de reglas de consistencia es completo. Para ello, se ha elaborado un algoritmo en Prolog el cual, a partir de estos axiomas, genera las mismas 90 combinaciones válidas del paso 2.

2 Combinaciones Posibles

A cada una de las relaciones que se defina en el sistema se le asociará un subconjunto del conjunto de las dieciséis restricciones que proporciona el sistema. En todo momento, el subconjunto de restricciones deberá ser válido, esto es, no debe suponer ningún tipo de inconsistencia.

$$\{Ac1, Ac2, Ac3, At, S1, S2, S3, Ll1, Ll2, Ll3, Tr1, Tr2, Tr3, As1, As2, As3\} = RT_a$$

Dado que el sistema es capaz de ofrecer dieciséis restricciones, el autor podría seleccionar, en principio, cualquier combinación de las mismas. Por tanto, con estas dieciséis restricciones se pueden hacer hasta 2^{16} caracterizaciones posibles de una relación conceptual. Esto quiere decir, que deberíamos demostrar cuáles de las 2^{16} combinaciones son válidas o posibles. Esto puede parecer inviable o, al menos, bastante laborioso. En realidad, vamos a ver que se puede reducir de manera cuantiosa esta cifra.

RT_a	Acíclicas			AT	Salida			Llegada			Transmisión			Asociación		
	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As1	As2	As3
Ac1	■■■■															
Ac2		■■■■														
Ac3	⊂	⊂	■■■■													
At				■■■■												
S1				⊂	■■■■	><	><							⊂		
S2					><	■■■■	><								⊂	⊂
S3				⊂	⊂	⊂	■■■■							⊂	⊂	⊂
Ll1				⊂				■■■■	><	><				⊂		
Ll2								><	■■■■	><					⊂	⊂
Ll3				⊂				⊂	⊂	■■■■				⊂	⊂	⊂
Tr1	⊂	⊂	⊂	⊂							■■■■	><	><			
Tr2											><	■■■■	><			
Tr3	⊂	⊂	⊂	⊂							⊂	⊂	■■■■			
As1														■■■■	><	><
As2														><	■■■■	><
As3														⊂	⊂	■■■■

Tabla B.1 Inclusiones e Incompatibilidades entre las Restricciones sobre las Relaciones Conceptuales.

En primer lugar, la restricción As1 —ver tabla B.1— no es elegible; es una restricción universal que tiene que cumplir cualquier relación. Luego el sistema ofrece tan sólo quince reglas, y no dieciséis. Esto quiere decir que las combinaciones quedan automáticamente reducidas a la mitad —2^{15}—.

Por otro lado, no tiene sentido decir que una relación debe cumplir la restricción As2 porque, dado que ya cumple As1, esto sería equivalente a decir que cumple directamente la restricción As3. En consecuencia, la regla As2 no se debe ofrecer. Esto significa, que el sistema va a ofrecer únicamente catorce restricciones o lo que es lo mismo hasta un total de 2^{14} combinaciones.

3 Combinaciones Inconsistentes

Se ha logrado reducir el número de posibilidades a una cuarta parte. No parece nada alentador pero aún se puede disminuir más el número de combinaciones válidas. Si nos fijamos en estas restricciones muchas de ellas se solapan; por tanto, no tiene sentido asociar a una relación una restricción cuando ya tiene asociada otra que incluye a ésta. Por otro lado, las restricciones pertenecientes a un mismo grupo —Salida, Llegada, Transmisión, ver tabla B.1— son incompatibles y, en consecuencia, inconsistentes.

En la tabla B.1 se resumen todas estas inclusiones e incompatibilidades. Para interpretarla, nos situamos en una fila, suponemos que la restricción de la fila se cumple, e intentamos añadir cada una de las restricciones de las columnas. Para aquellas restricciones cuyo añadido suponga una inclusión se marcará la casilla con el símbolo $\subset$; para aquellas otras que suponga una incompatibilidad se marcará la casilla con el símbolo $><$.

En general, diremos que una restricción rt_1 *incluye* a otra rt_2, cuando se cumpla que si $rt_1 \Rightarrow rt_2$. Así, la restricción rt_1 es más fuerte que rt_2, de forma que sólo imponiendo rt_1 también se está obligando a cumplir rt_2. Diremos que rt_1 es *incompatible* con respecto a rt_2, si $rt_1 \Rightarrow \neg\ rt_2$, esto es, la restricción rt_1 implica que no se dé rt_2. La razón por la que obligamos a que no se cumpla esa otra restricción es porque, en ese caso, existe otra restricción que incluye a ambas. Por tanto, únicamente debemos demostrar las reglas de inclusión, porque la incompatibilidad de algunas de las reglas tiene su origen en la existencia de una restricción que incluye a ambas —en concreto, la restricción tercera de los grupos Salida, Llegada, Transmisión y Asociación es la unión de las restricciones uno y dos de su grupo—.

La tabla B.2 recoge el conjunto de reglas de inclusión e incompatibilidad de la tabla B.1. La demostración de estas reglas se incluyen en el siguente apartado.

Nº	Reglas de Inclusión	Reglas de Incompatibilidad
1.	Ac3 ⇒ Ac1 ∧ Ac2	
2.	S1 ⇒ At	S1 ⇒ ¬ S2
3.	S2 ⇒ As3	S2 ⇒ ¬ S1
4.	S3 ⇒ At; S3 ⇒ As3; S3 ⇒ S1 ∧ S2	
5.	Ll1 ⇒ At	Ll1 ⇒ ¬ Ll2
6.	Ll2 ⇒ As3	Ll2 ⇒ ¬ Ll1
7.	Ll3 ⇒ At; Ll3 ⇒ As3; Ll3 ⇒ Ll1 ∧ Ll2	
8.	Tr1 ⇒ Ac1; Tr1 ⇒ Ac2; Tr1 ⇒ Ac3; Tr1 ⇒ At	Tr1 ⇒ ¬ Tr2
9.		Tr2 ⇒ ¬Tr1
10.	Tr3 ⇒ Ac1; Tr3 ⇒ Ac2; Tr3 ⇒ Ac3; Tr3 ⇒ At; Tr3 ⇒ Tr1 ∧ Tr2	

Tabla B.2 Reglas de inclusión o incompatibilidad entre las Restricciones sobre Relaciones Conceptuales.

3.1 Demostración de las Reglas de Inclusión e Incompatibilidad de la Tabla B.2

1.	**Ac3 ⇒ Ac1 ∧ Ac2** Si una relación es acíclica quiere decir que, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow c_1 \neq c_n$ Sea n, el número de tripletas que tiene el ciclo. Con n=1, $\forall < c_1, r_i, c_2 > \in A \Rightarrow c_1 \neq c_2$, definición de antirreflexiva. Con n=2, $\forall < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \in A \Rightarrow c_1 \neq c_3$, esto es, $\forall < c_1, r_i, c_2 > \in A \Rightarrow < c_2, r_i, c_1 > \notin A$, definición de antisimétrica. Por tanto, las reglas Ac1 y Ac2 son casos particulares de Ac3. No es necesario decir de una relación que es acíclica que además es antirreflexiva o antisimétrica.

Tabla B.3 DemoI1: Demostración de la regla de Inclusión/Incompatibilidad 1.

2.	**S1 ⇒ At** Si una relación presenta una incompatibilidad de salida consigo misma, quiere decir que, $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$ Supongamos que una relación es transitiva, entonces $\forall\ \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^+(c_1, r_i)| = 2$, contradice la definición de S1 Por tanto, si una relación cumple S1, también es antitransitiva. **S1 ⇒ ¬ S2** Supongamos que una relación cumple S2, entonces $\forall\ c_i \in C$ y $r_c \in R_c$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r_q \in R_c - \{r_c\}$, $|\Gamma^+(c_i, r_q)| = 0$ como además cumple S1, entonces $\forall\ c_i \in C$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$, definición de S3 Por tanto, no se puede decir que una relación cumple S1 y S2 porque es lo mismo que decir que cumple S3.

Tabla B.4 DemoI2: Demostración de la regla de Inclusión/Incompatibilidad 2.

3.	**S2 ⇒ As3** Si una relación presenta una incompatibilidad de salida con el resto de las relaciones, quiere decir que, $\forall\ c_i \in C$ y $r_c \in R_c$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r_q \in R_c - \{r_c\}$, $|\Gamma^+(c_i, r_q)| = 0$ Por otro lado, dado que se cumple As1 por ser universal $\forall < c_i, r_i, c_d > \in A_c, \nexists < c_i, s_i, c_d > \in A_c$, tal que $r_c.nombre = s_c.nombre$ esto es, $|\Gamma^-(c_i, r_i)| = 1$ como además se cumple S2, entonces $\Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$, definición As3 Por tanto, si una relación cumple S2, también cumple As3. **S2 ⇒ ¬ S1** Supongamos que una relación cumple S1, entonces $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$ como además cumple S2, entonces $\forall\ c_i \in C$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$, definición de S3 Por tanto, no se puede decir que una relación cumple S2 y S1 porque es lo mismo que decir que cumple S3.

Tabla B.5 DemoI3: Demostración de la regla de Inclusión/Incompatibilidad 3.

4.	**S3 ⇒ At** Si una relación presenta una incompatibilidad de salida total, quiere decir que, $\forall\ c_i \in C,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ Supongamos que una relación es transitiva, entonces $\forall\ \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^+(c_1, r_i)| = 2,$ Por tanto, si una relación cumple S1, también es antitransitiva. **S3 ⇒ As3** Si una relación cumple S3, entonces $\forall\ c_i, c_d \in C,\ \Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$, definición As3 Por tanto, si una relación cumple S3, también cumple As3. **S3 ⇒ S1 ∧ S2** Si una relación cumple S3, entonces $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$, definición S1 y $\forall\ c_i \in C$ y $r_c \in R_c,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^+(c_i, r_q)| = 0$, definición S2 esto es S3 = S1 + S2, luego si ya cumple S3 no tiene sentido decir que cumple S1 y S2.

Tabla B.6 DemoI4: Demostración de la regla de Inclusión/Incompatibilidad 4.

5.	**LI1 ⇒ At** Si una relación presenta una incompatibilidad de llegada consigo misma, quiere decir que, $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 1$ Supongamos que una relación es transitiva, entonces $\forall\ \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^-(c_n, r_i)| = 2,$ Por tanto, si una relación cumple S1, también es antitransitiva. **LI1 ⇒ ¬ LI2** Supongamos que una relación cumple LI2, entonces $\forall\ c_i \in C$ y $r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^-(c_i, r_q)| = 0$ como además cumple LI1, entonces $\forall\ c_i \in C,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^-(c_i, r)| = 0$, definición de LI3 Por tanto, no se puede decir que una relación cumple LI1 y LI2 porque es lo mismo que decir que cumple LI3.

Tabla B.7 DemoI5: Demostración de la regla de Inclusión/Incompatibilidad 5.

6.	**LI2 ⇒ As3** Dado que se cumple As1 por ser universal $\forall <c_i, r_i, c_d> \in A_c \Rightarrow \nexists <c_i, s_i, c_d> \in A_c$, tal que r_c.nombre = s_c.nombre esto es, $\lvert\Gamma^-(c_d, r_i)\rvert = 1$ como además se cumple LI2, entonces $\Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$, definición As3 Por tanto, si una relación cumple LI2, también cumple As3. **LI2 ⇒ ¬ LI1** Si una relación presenta una incompatibilidad de llegada con el resto de las relaciones, quiere decir que, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^-(c_i, r_q)\rvert = 0$ Supongamos que una relación cumple LI1, entonces $\forall c_i \in C$ y $r_c \in R_c \Rightarrow \lvert\Gamma^-(c_i, r_c)\rvert \leq 1$ como además cumple LI2, entonces $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$, definición de LI3 Por tanto, no se puede decir que un relación cumple LI2 y LI1 porque es lo mismo que decir que cumple LI3.

Tabla B.8 DemoI6: Demostración de la regla de Inclusión/Incompatibilidad 6.

7.	**LI3 ⇒ At** Si una relación presenta una incompatibilidad de llegada total, quiere decir que, $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$ Supongamos que una relación es transitiva, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n> \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $\lvert\Gamma^-(c_n, r_i)\rvert = 2$, Por tanto, si una relación cumple LI3, también es antitransitiva. **LI3 ⇒ As3** Si una relación cumple LI3, entonces $\forall c_i, c_d \in C$, $\Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$, definición As3 Por tanto, si una relación cumple LI3, también cumple As3. **LI3 ⇒ LI1 ∧ LI2** Si una relación cumple LI3, entonces $\forall c_i \in C$ y $r_c \in R_c \Rightarrow \lvert\Gamma^-(c_i, r_c)\rvert \leq 1$, definición LI1 y $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$, definición LI2 esto es LI3 = LI1 + LI2, luego si ya cumple LI3 no tiene sentido decir que cumple LI1 ó LI2.

Tabla B.9 DemoI7: Demostración de la regla de Inclusión/Incompatibilidad 7.

8.	**Tr1 ⇒ Ac1; Tr1 ⇒ Ac2; Tr1 ⇒ Ac3** Si una relación presenta una incompatibilidad de transmisión consigo misma quiere decir que, $\forall c_i \in C, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_i)| = 0$ Supongamos que una restricción es cíclica, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n> \in EC_m \Rightarrow c_1 = c_n$ esto significa que, $\forall c_i \in C \Rightarrow |\Gamma^-(c_i, r_i)| \neq 0, |\Gamma^+(c_i, r_i)| \neq 0$ que contradice a Tr1. Por tanto, tiene que ser acíclica. Como antes hemos visto que Ac2 y Ac3 son casos particulares de Ac1, también las incluye. **Tr1 ⇒ At** Supongamos que una relación es transitiva, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n> \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^+(c_1, r_i)| = 2$, que contradice la definición de Tr1 Por tanto, si una relación cumple Tr1, también es antitransitiva. **Tr1 ⇒ ¬Tr2** Supongamos que una relación cumple Tr2, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}, |\Gamma^+(c_i, r_q)| = 0$ como además cumple Tr1, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c, |\Gamma^+(c_i, r)| = 0$, definición de Tr3 Por tanto, no se puede decir que un relación cumple Tr1 y Tr2 porque es lo mismo que decir que cumple Tr3.

Tabla B.10 DemoI8: Demostración de la regla de Inclusión/Incompatibilidad 8.

9.	**Tr2 ⇒ ¬ Tr1** Si una relación presenta una incompatibilidad de transmisión Tr2 con el resto de las relaciones, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}, |\Gamma^+(c_i, r_q)| = 0$ Supongamos que una relación cumple Tr1, entonces $\forall c_i \in C, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_i)| = 0$ como además cumple Tr2, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c, |\Gamma^+(c_i, r)| = 0$, definición de Tr3 Por tanto, no se puede decir que un relación cumple Tr2 y Tr1 porque es lo mismo que decir que cumple Tr3.

Tabla B.11 DemoI9: Demostración de la regla de Inclusión/Incompatibilidad 9.

10.	**Tr3 ⇒ Ac1; Tr3 ⇒ Ac2; Tr3 ⇒ Ac3** Si una relación presenta una incompatibilidad de transmisión total quiere decir que, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_i)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^+(c_i, r)\rvert = 0$ Supongamos que una restricción es cíclica, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow c_1 = c_n$ esto significa que, $\forall c_i \in C \Rightarrow \lvert\Gamma^-(c_i, r_i)\rvert \neq 0$, $\lvert\Gamma^+(c_i, r_i)\rvert \neq 0$ que contradice a Tr3. Por tanto, tiene que ser acíclica. Como antes hemos visto que Ac2 y Ac3 son casos particulares de Ac1, también incluye a estas restricciones. **Tr3 ⇒ At** Supongamos que una relación es transitiva, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $\lvert\Gamma^+(c_1, r_i)\rvert = 2$, que contradice la definición de Tr3 Por tanto, si una relación cumple Tr3, también es antitransitiva. **Tr3 ⇒ Tr1 ∧ Tr2** Si una relación cumple Tr3, entonces $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_i)\rvert \neq 0 \Rightarrow \lvert\Gamma^+(c_i, r_i)\rvert = 0$, definición Ll1 y $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^+(c_i, r_q)\rvert = 0$, definición Ll2 esto es Tr3 = Tr1 + Tr2, luego si ya cumple Tr3 no tiene sentido decir que cumple Tr1 ó Tr2.

Tabla B.12 DemoI10: Demostración de la regla de Inclusión/Incompatibilidad 10.

3.2 Programa1 : Elimina Combinaciones No Válidas

La demostración de las reglas de la tabla B.2 prueba que no son posibles las combinaciones en las que aparezcan las restricciones de la derecha de la implicación si en esa misma combinación se encuentra la restricción de la izquierda. Las causas por las que no es posible su combinación será porque, o bien la restricción de la derecha incluye a la de la izquierda, o bien porque es incompatible con respecto a la restricción de la izquierda.

Se ha escrito un algoritmo —ver apéndice C— que elimina las combinaciones no válidas y calcula el conjunto de combinaciones que, en el caso de ser consistentes, serían válidas en nuestro sistema. Este algoritmo elimina un total de 16094 combinaciones, siendo *a priori* válidas un total de 290 (2^{14} = 16384 - 16094 = 290). Este programa ha sido desarrollado en ANSI C.

```
Programa1 Elimina-no-válidas;
Comb = Genera-todas-combinaciones-restricciones-relacionales;
for  all (combinación ∈ Comb) {
      if (Ac3 and  (Ac1 or Ac2)) quita (combinación);
      if (S1 and (At or S2 or S3)) quita (combinación);
      if (S2 and (S1 or S3 or As3)) quita (combinación);
      if (S3 and (At or S1 or S2 or As3)) quita (combinación);
      if (Ll1 and (At or Ll2 or Ll3)) quita (combinación);
      if (Ll2 and (Ll1 or Ll3 or As3)) quita (combinación);
      if (Ll3 and (At or Ll1 or Ll2 or As3)) quita (combinación);
      if (Tr1 and (Ac1 or Ac2 or Ac3 or At or Tr2 or Tr3)) quita (combinación);
      if (Tr2 and (Tr1 or Tr3)) quita (combinación);
      if (Tr3 and (Ac1 or Ac2 or Ac3 or At or Tr1 or Tr2)) quita (combinación);
}
```

4 Combinaciones Consistentes

Queda, por tanto, demostrar que estas 290 posibilidades son consistentes. Sin embargo, se va a intentar reducir el número de demostraciones. Para ello, se hace uso de la posibilidad de agrupar las restricciones en las seis categorías anteriormente mencionadas: Acíclico —Ac1, Ac2 y Ac3—, Antitransitiva —At—, Salida —S1, S2 y S3—, Llegada —Ll1, Ll2 y Ll3—, Transmisión —Tr1, Tr2 y Tr3—, y Asociación —As3—. Y esto es posible porque una de las restricciones puede representar al resto. Por ejemplo, tal y como se ha demostrado en DemoI1 la restricción que niega la posibilidad de existencia de ciclos Ac3 incluye a la antirreflexiva Ac1 y a la antisimétrica Ac2. La antirreflexiva y la antisimétrica son la negación de ciclos de uno y dos pasos respectivamente. En la tabla B.13 se muestra la regla que representa al grupo.

Esto significa que, demostrando la consistencia para estos representantes estaríamos demostrando la validez de aquellas otras combinaciones en las que aparezcan algunas de las restricciones del grupo.

Ac	Ac3 ⊃ Ac1, Ac2
At	At
S	S3 ⊃ S1, S2
Ll	Ll3 ⊃ Ll1, Ll2
Tr	Tr3 ⊃ Tr1, Tr2
As	As3 ⊃ As1, As2

Tabla B.13 Representantes de las Restricciones sobre las Relaciones Conceptuales.

4.1 Demostración de los Representantes

La mayoría de las inclusiones de la tabla B.13 han sido demostradas anteriormente. Para Ac consulte la demostración de la regla de inclusión/incompatibilidad número 1 —tabla Demo1—, para S la demostración de la regla número 4 —tabla Demo4—, para Ll la demostración de la regla número 7 —tabla Demo7— y, para Tr la demostración de la regla número 10 —tabla Demo10—. Únicamente, queda por probar As, tal y como se describe a continuación.

	As3 ⊃ As1, As2
As	Si una relación cumple As3, entonces $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ esto quiere decir que, $\forall <c_o, r_c, c_d> \in A_c, \nexists <c_o, s_c, c_d> \in A_c$, tal que r_c.nombre = s_c.nombre., definición As1 y $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall\ r_q \in R_c - \{r_c\}, <c_o, r_q, c_d> \notin A$, definición As2 esto es As3 incluye a As1 y a As2.

Tabla B.14 DemoAs: Demostración del Representante As.

4.2 Reglas Representantes

Estas seis reglas representantes —tabla B.13— permiten resumir varias combinaciones válidas y demostrar su consistencia. En la siguiente —tabla B.15—, se indica cuándo dos restricciones —o sus representantes— se pueden dar al mismo tiempo, esto es, son consistentes. Al igual que antes, para interpretarla nos situamos en una fila, suponemos que la restricción de la fila se cumple e intentamos añadir cada una de las restricciones de las columnas. Para aquellas restricciones cuyo añadido no suponga ninguna inconsistencia se marcará la casilla con una C —con || estamos indicando lo misma pero leída por columnas—.

Estas mismas consistencias se recogen en la tabla B.16, y vienen a decir que aquellas combinaciones en las que aparezcan estas reglas son válidas y consistentes. Para ello, se debe demostrar que las 290 combinaciones cumplen un subconjunto de estas aserciones. Pero, además, se debe probar que estos axiomas son capaces de generar las 290 combinaciones mediante deducción lógica —es decir, que el sistema formal es semánticamente completo—.

Para demostrar lo primero se ha elaborado un programa —en C— que comprueba que, efectivamente, todas las combinaciones supuestamente válidas verifican una o varias de estas reglas. Para probar lo segundo se ha construido un programa lógico —en Prolog— que verdaderamente es capaz de inferir todas las combinaciones de restricciones relacionales que son válidas en nuestro sistema. Ergo, si se demuestran las 16 aserciones, se está demostrando la consistencia de las 290 combinaciones válidas.

RT_a	Acíclicas			AT	Salida			Llegada			Transmisión			As
	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Ac1	▪▪▪▪	c	--	C	C			C			--	C	--	C
Ac2	| |	▪▪▪▪	--								--		--	
Ac3	x	x	▪▪▪▪								--		--	
At	| |			▪▪▪▪	--	c	--	--	c	--	--	c	--	c
S1	| |			x	▪▪▪▪	x	x	C			C			c
S2				| |	x	▪▪▪▪	x						x
S3				x	x	x	▪▪▪▪							x
Ll1	| |			x	| |		▪▪▪▪	x	x	C			c
Ll2				| |			x	▪▪▪▪	x				x
Ll3				x				x	x	▪▪▪▪				x
Tr1	x	x	x	x	| |			| |			▪▪▪▪	x	x	C
Tr2	| |			| |						x	▪▪▪▪	x	
Tr3	x	x	x	x							x	x	▪▪▪▪	
As3	| |			| |	| |	--	--	| |	--	--	| |		▪▪▪▪

Tabla B.15 Consistencias entre las Restricciones sobre las Relaciones Conceptuales.

1.	1.	Ac1 and Ac2
2.	2.	Ac and At
	3.	Ac and S
	4.	Ac and Ll
	5.	Ac and Tr2
	6.	Ac and As
3.	7.	At and S2
	8.	At and Ll2
	9.	At and Tr2
	10.	At and As
4.	11.	S and Ll
	12.	S and Tr
5.	13.	S1 and As
6.	14.	Ll and Tr
7.	15.	Ll1 and As
8.	16.	Tr and As

Tabla B.16 Axiomas de consistencia entre Restricciones sobre las Relaciones Conceptuales.

4.3 Demostración de las Combinaciones Consistentes

A continuación se demuestra —basándonos en las propias definiciones de las restricciones conceptuales— que son consistentes las combinaciones que se generen a partir de los axiomas de la tabla B16. Estas demostraciones se han agrupado en ocho bloques debido a que con frecuencia una misma restricción aparece combinada con varias de las otras.

1.	**Ac1 and Ac2** Si una relación es antirreflexiva Ac1 quiere decir que, $\forall < c_1, r_i, c_2 > \in A \Rightarrow c_1 \neq c_2$ Supongamos que la relación es antisimétrica Ac2, entonces, $\forall < c_1, r_i, c_2 > \in A \Rightarrow < c_2, r_i, c_1 > \notin A$ No nos conduce a ninguna incompatibilidad, ni inclusión, decir que no se permiten ciclos de un paso ni de dos pasos. Por otro lado, no hay ningún tipo de inconsistencia en decir que se permiten ciclos de un paso pero no de dos o viceversa.

Tabla B.17 DemoC1: Demostración de la regla de consistencia 1.

2.	**Ac and At** Si una relación es acíclica Ac quiere decir que, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow c_1 \neq c_n$ luego $\Gamma^-(c_1, r_i) \neq c_n$, $\Gamma^+(c_n, r_i) \neq c_1$ Supongamos que la relación es antitransitiva At, entonces, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow < c_1, r_i, c_n > \notin A$ luego $\Gamma^+(c_1, r_i) \neq c_n$, $\Gamma^-(c_n, r_i) \neq c_1$ En consecuencia, una relación puede ser al mismo tiempo acíclica y antitransitiva, o cíclica y antitransitiva, o acíclica y transitiva. **Ac and S** Supongamos que la relación presenta incompatibilidad de salida total S, entonces, $\forall c_i \in C, |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ac y S, o cumplir Ac y no S, o no cumplir Ac y sí S. **Ac and Ll** Supongamos que la relación presenta incompatibilidad de llegada total Ll, entonces, $\forall c_i \in C, |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^-(c_i, r)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ac y Ll, o cumplir Ac y no Ll, o no cumplir Ac y sí Ll. **Ac and Tr2** Supongamos que la relación presenta incompatibilidad de transmisión Tr2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $|\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}, |\Gamma^+(c_i, r_q)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ac y Tr2, o cumplir Ac y no Tr2, o no cumplir Ac y sí Tr2. **Ac and As** Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$ En consecuencia, una relación puede cumplir al tiempo Ac y As, o cumplir Ac y no As, o no cumplir Ac y sí As.

Tabla B.18 DemoC2: Demostración de la regla de consistencia 2-6.

3.	**At and S2** Si una relación es antitransitiva At quiere decir que, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow < c_1, r_i, c_n > \notin A$ luego $\Gamma^+(c_1, r_i) \neq c_n$, $\Gamma^-(c_n, r_i) \neq c_1$ Supongamos que la relación presenta incompatibilidad de salida S2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^+(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^+(c_i, r_q)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo At y S2, o cumplir At y no S2, o no cumplir At y sí S2. **At and Ll2** Supongamos que la relación presenta incompatibilidad de llegada Ll2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^-(c_i, r_q)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo At y Ll2, o cumplir At y no Ll2, o no cumplir At y sí Ll2. **At and Tr2** Supongamos que la relación presenta incompatibilidad de transmisión Tr2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^+(c_i, r_q)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo At y Tr2, o cumplir At y no Tr2, o no cumplir At y sí Tr2. **At and As** Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$ En consecuencia, una relación puede cumplir al tiempo At y As, o cumplir At y no As, o no cumplir At y sí As.

Tabla B.19 DemoC3: Demostración de la regla de consistencia 7-10.

4.	**S and Ll** Si una relación presenta incompatibilidad de salida total S quiere decir que, $\forall c_i \in C$, $\lvert\Gamma^+(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^+(c_i, r)\rvert = 0$ Supongamos que la relación presenta incompatibilidad de llegada total Ll, entonces, $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo S y Ll, o cumplir S y no Ll, o no cumplir S y sí Ll. **S and Tr** Supongamos que la relación presenta incompatibilidad de transmisión total Tr, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_i)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^+(c_i, r)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo S y Tr, o cumplir S y no Tr, o no cumplir S y sí Tr.

Tabla B.20 DemoC4: Demostración de la regla de consistencia 11-12.

5.	**S1 and As** Si una relación presenta incompatibilidad de salida S1 quiere decir que, $\forall c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$ Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ luego $\forall c_i \in C$, $|\Gamma^+(c_i, r_i)| = 1$ y , $|\Gamma^-(c_i, r_i)| = 1$ En consecuencia, una relación puede cumplir al tiempo S1 y As, o cumplir S1 y no As, o no cumplir S1 y sí As.

Tabla B.21 DemoC5: Demostración de la regla de consistencia 13.

6.	**Ll and Tr** Si una relación presenta incompatibilidad de llegada total Ll quiere decir que, $\forall c_i \in C$, $|\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^-(c_i, r)| = 0$ Supongamos que la relación presenta incompatibilidad de transmisión total Tr, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $|\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ll y Tr, o cumplir Ll y no Tr, o no cumplir Ll y sí Tr.

Tabla B.22 DemoC6: Demostración de la regla de consistencia 14.

7.	**Ll1 and As** Si una relación presenta incompatibilidad de salida Ll1 quiere decir que, $\forall c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 1$ Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ luego $\forall c_i \in C$, $|\Gamma^+(c_i, r_c)| = 1$ y , $|\Gamma^-(c_i, r_c)| = 1$ En consecuencia, una relación puede cumplir al tiempo Ll1 y As, o cumplir Ll1 y no As, o no cumplir Ll1 y sí As.

Tabla B.23 DemoC7: Demostración de la regla de consistencia 15.

8.	**Tr and As** Si una relación presenta incompatibilidad de transmisión total Tr quiere decir que, $\forall c_i \in C$ y $r_c \in R_c$, $|\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ luego $\forall c_i \in C$, $|\Gamma^+(c_i, r_c)| = 1$ y , $|\Gamma^-(c_i, r_c)| = 1$ En consecuencia, una relación puede cumplir al tiempo Ll1 y As, o cumplir Ll1 y no As, o no cumplir Ll1 y sí As.

Tabla B.24 DemoC8: Demostración de la regla de consistencia 16.

4.4 Programa 2: Comprueba que las Combinaciones Válidas cumplen los Axiomas

Este programa comprueba que todas las combinaciones válidas cumplen los axiomas de la tabla B.16. Para ello se recorren todas ellas y se va cotejando si verifica uno o más de los 16 axiomas posibles. Este programa ha sido desarrollado en ANSI C.

```
Programa2 Cumple-axiomas;
Genera-combinaciones-restricciones-consistentes;
for  all (combinación-consistente) {
      if ( Ac1 and Ac2) cumple (axioma1, combinación-consistente);
      if (Ac and At) cumple (axioma2, combinación-consistente);
      if (Ac and S) cumple (axioma3, combinación-consistente);
      if (Ac and Ll) cumple (axioma4, combinación-consistente);
      if (Ac and Tr2) cumple (axioma5, combinación-consistente);
      if (Ac and As) cumple (axioma6, combinación-consistente);
      if (At and S2) cumple (axioma7, combinación-consistente);
      if (At and Ll2) cumple (axioma8, combinación-consistente);
      if (At and Tr2) cumple (axioma9, combinación-consistente);
      if (At and As) cumple (axioma10, combinación-consistente);
      if (S and Ll) cumple (axioma11, combinación-consistente);
      if (S and Tr) cumple (axioma12, combinación-consistente);
      if (S1 and As) cumple (axioma13, combinación-consistente);
      if (Ll and Tr) cumple (axioma14, combinación-consistente);
      if (Ll1 and As) cumple (axioma15, combinación-consistente);
      if (Tr and As) cumple (axioma16, combinación-consistente);
}
```

4.5 Programa 3: Genera a partir de los Axiomas las Combinaciones Válidas

Este programa lógico ha sido implementado en Prolog —con el intérprete JIP, Jave Internet en Prolog—. Su ejecución genera las combinaciones válidas a partir de los axiomas de la tabla B.16, las mismas se calcularon eliminando las inconsistentes.

```
//Programa3 GeneraCombinaciones

aciclico(ac1). aciclico(ac2). aciclico(ac3). anti(at). salida(s1). salida(s2). salida(s3).
llegada(ll1). llegada(ll2). llegada(ll3). transmision(tr1). transmision(tr2). transmision(tr3). asociacion(as).

posible2(ac1, ac2).
posible2(A,B):- aciclico(A), anti(B).
posible2(A,B):- aciclico(A), salida(B).
posible2(A,B):- aciclico(A), llegada(B).
posible2(A,tr2):- aciclico(A).
posible2(A,as):- aciclico(A).
posible2(at,s2).
posible2(at,ll2).
```

```
posible2(at,tr2).
posible2(at,as).
posible2(A,B):- salida(A), llegada(B).
posible2(A,B):- salida(A), transmision(B).
posible2(s1,as).
posible2(A,B):- llegada(A), transmision(B).
posible2(ll1,as).
posible2(A,as):- transmision(A).

posible3(A, B, C):- posible2(A, B), posible2(A, C), posible2(B, C).
posible4(A,B,C,D):- posible3(A,B,C), posible3(A,B,D), posible3(A,C,D), posible3(B,C,D).
posible5(A,B,C,D,E):-
   posible4(A,B,C,D), posible4(A,B,C,E), posible4(A,B,D,E), posible4(A,C,D,E), posible4(B,C,D,E).
posible6(A,B,C,D,E,F):-
   posible5(A,B,C,D,E), posible5(A,B,C,D,F), posible5(A,B,C,E,F), posible5(A,B,D,E,F),
posible5(A,C,D,E,F), posible5(B,C,D,E,F).
```

En conclusión, las demostraciones de las tablas B.2, B13 y B16, y los programas 1, 2 y 3 prueban la inconsistencia de las combinaciones no válidas y la consistencia de las combinaciones válidas. En el apéndice C se puede consultar los códigos de los programas y los resultados obtenidos con su ejecución.

Apéndice B

Demostración de la Consistencia de RT_A

1 Introducción

En todo momento, el conjunto de restricciones asociado a una relación conceptual debe ser consistente. Cualquier combinación de restricciones no tiene por qué ser válida. Por tanto, hay que determinar el conjunto de combinaciones válidas y, en consecuencia, consistentes. Antes de añadir o borrar una de las restricciones que caracterizan a una relación conceptual se debe comprobar nuevamente que el conjunto de restricciones es consistente. Para ello, únicamente deberemos comprobar que es una de las combinaciones válidas.

Se ha optado por una demostración computacional que ha consistido en los siguientes pasos:

1. Generar con un algoritmo todas las combinaciones posibles que se pueden generar con las 16 restricciones relacionales —se pueden hacer hasta 2^{16} caracterizaciones posibles a una relación conceptual—.
2. Rechazar aquellas combinaciones que son inconsistentes —se ha conseguido reducir el anterior número a un total de 290 combinaciones a priori válidas—.
3. Demostrar que las combinaciones que han quedado después de eliminar las no válidas son consistentes.

Para alcanzar el objetivo del paso 2:

1. Se han establecido reglas de inclusión e incompatibilidad entre algunas de las restricciones.
2. Se ha demostrado la veracidad de estas reglas.
3. Se ha escrito un algoritmo que elimina aquellas combinaciones no válidas, esto es, que contienen estas restricciones incluyentes o incompatibles.

Finalmente, para demostrar que las combinaciones que han resultado del anterior proceso son válidas se ha actuado de la siguiente manera:

1. Dado el gran número de combinaciones —un total de 290— que hace inviable en la práctica la demostración individual, se ha optado por agruparlas en representantes.
2. Se han demostrado las reglas representantes que permiten resumir las combinaciones válidas.
3. Con los representantes se han establecido reglas o axiomas de consistencia.
4. Se han demostrado estas reglas de consistencia.
5. Se ha escrito un algoritmo que comprueba que todas las combinaciones válidas no eliminadas contienen un subconjunto de estas reglas válidas.
6. Por último, se ha comprobado que este conjunto de reglas de consistencia es completo. Para ello, se ha elaborado un algoritmo en Prolog el cual, a partir de estos axiomas, genera las mismas 90 combinaciones válidas del paso 2.

2 Combinaciones Posibles

A cada una de las relaciones que se defina en el sistema se le asociará un subconjunto del conjunto de las dieciséis restricciones que proporciona el sistema. En todo momento, el subconjunto de restricciones deberá ser válido, esto es, no debe suponer ningún tipo de inconsistencia.

$$\{Ac1, Ac2, Ac3, At, S1, S2, S3, Ll1, Ll2, Ll3, Tr1, Tr2, Tr3, As1, As2, As3\} = RT_a$$

Dado que el sistema es capaz de ofrecer dieciséis restricciones, el autor podría seleccionar, en principio, cualquier combinación de las mismas. Por tanto, con estas dieciséis restricciones se pueden hacer hasta 2^{16} caracterizaciones posibles de una relación conceptual. Esto quiere decir, que deberíamos demostrar cuáles de las 2^{16} combinaciones son válidas o posibles. Esto puede parecer inviable o, al menos, bastante laborioso. En realidad, vamos a ver que se puede reducir de manera cuantiosa esta cifra.

RT_a	Acíclicas			AT	Salida			Llegada			Transmisión			Asociación		
	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As1	As2	As3
Ac1	■■■■															
Ac2		■■■■														
Ac3	⊂	⊂	■■■■													
At				■■■■												
S1				⊂	■■■■	><	><							⊂		
S2					><	■■■■	><								⊂	⊂
S3				⊂	⊂	⊂	■■■■							⊂	⊂	⊂
Ll1				⊂				■■■■	><	><				⊂		
Ll2								><	■■■■	><					⊂	⊂
Ll3				⊂				⊂	⊂	■■■■				⊂	⊂	⊂
Tr1	⊂	⊂	⊂	⊂							■■■■	><	><			
Tr2											><	■■■■	><			
Tr3	⊂	⊂	⊂	⊂							⊂	⊂	■■■■			
As1														■■■■	><	><
As2														><	■■■■	><
As3														⊂	⊂	■■■■

Tabla B.1 Inclusiones e Incompatibilidades entre las Restricciones sobre las Relaciones Conceptuales.

En primer lugar, la restricción As1 —ver tabla B.1— no es elegible; es una restricción universal que tiene que cumplir cualquier relación. Luego el sistema ofrece tan sólo quince reglas, y no dieciséis. Esto quiere decir que las combinaciones quedan automáticamente reducidas a la mitad —2^{15}—.

Por otro lado, no tiene sentido decir que una relación debe cumplir la restricción As2 porque, dado que ya cumple As1, esto sería equivalente a decir que cumple directamente la restricción As3. En consecuencia, la regla As2 no se debe ofrecer. Esto significa, que el sistema va a ofrecer únicamente catorce restricciones o lo que es lo mismo hasta un total de 2^{14} combinaciones.

3 Combinaciones Inconsistentes

Se ha logrado reducir el número de posibilidades a una cuarta parte. No parece nada alentador pero aún se puede disminuir más el número de combinaciones válidas. Si nos fijamos en estas restricciones muchas de ellas se solapan; por tanto, no tiene sentido asociar a una relación una restricción cuando ya tiene asociada otra que incluye a ésta. Por otro lado, las restricciones pertenecientes a un mismo grupo —Salida, Llegada, Transmisión, ver tabla B.1— son incompatibles y, en consecuencia, inconsistentes.

En la tabla B.1 se resumen todas estas inclusiones e incompatibilidades. Para interpretarla, nos situamos en una fila, suponemos que la restricción de la fila se cumple, e intentamos añadir cada una de las restricciones de las columnas. Para aquellas restricciones cuyo añadido suponga una inclusión se marcará la casilla con el símbolo $\subset$; para aquellas otras que suponga una incompatibilidad se marcará la casilla con el símbolo $><$.

En general, diremos que una restricción rt_1 *incluye* a otra rt_2, cuando se cumpla que si $rt_1 \Rightarrow rt_2$. Así, la restricción rt_1 es más fuerte que rt_2, de forma que sólo imponiendo rt_1 también se está obligando a cumplir rt_2. Diremos que rt_1 es *incompatible* con respecto a rt_2, si $rt_1 \Rightarrow \neg\ rt_2$, esto es, la restricción rt_1 implica que no se dé rt_2. La razón por la que obligamos a que no se cumpla esa otra restricción es porque, en ese caso, existe otra restricción que incluye a ambas. Por tanto, únicamente debemos demostrar las reglas de inclusión, porque la incompatibilidad de algunas de las reglas tiene su origen en la existencia de una restricción que incluye a ambas —en concreto, la restricción tercera de los grupos Salida, Llegada, Transmisión y Asociación es la unión de las restricciones uno y dos de su grupo—.

La tabla B.2 recoge el conjunto de reglas de inclusión e incompatibilidad de la tabla B.1. La demostración de estas reglas se incluyen en el siguente apartado.

Nº	Reglas de Inclusión	Reglas de Incompatibilidad
1.	Ac3 ⇒ Ac1 ∧ Ac2	
2.	S1 ⇒ At	S1 ⇒ ¬ S2
3.	S2 ⇒ As3	S2 ⇒ ¬ S1
4.	S3 ⇒ At; S3 ⇒ As3; S3 ⇒ S1 ∧ S2	
5.	Ll1 ⇒ At	Ll1 ⇒ ¬ Ll2
6.	Ll2 ⇒ As3	Ll2 ⇒ ¬ Ll1
7.	Ll3 ⇒ At; Ll3 ⇒ As3; Ll3 ⇒ Ll1 ∧ Ll2	
8.	Tr1 ⇒ Ac1; Tr1 ⇒ Ac2; Tr1 ⇒ Ac3; Tr1 ⇒ At	Tr1 ⇒ ¬ Tr2
9.		Tr2 ⇒ ¬Tr1
10.	Tr3 ⇒ Ac1; Tr3 ⇒ Ac2; Tr3 ⇒ Ac3; Tr3 ⇒ At; Tr3 ⇒ Tr1 ∧ Tr2	

Tabla B.2 Reglas de inclusión o incompatibilidad entre las Restricciones sobre Relaciones Conceptuales.

3.1 Demostración de las Reglas de Inclusión e Incompatibilidad de la Tabla B.2

1.	**Ac3 ⇒ Ac1 ∧ Ac2** Si una relación es acíclica quiere decir que, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow c_1 \neq c_n$ Sea n, el número de tripletas que tiene el ciclo. Con n=1, $\forall < c_1, r_i, c_2 > \in A \Rightarrow c_1 \neq c_2$, definición de antirreflexiva. Con n=2, $\forall < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \in A \Rightarrow c_1 \neq c_3$, esto es, $\forall < c_1, r_i, c_2 > \in A \Rightarrow < c_2, r_i, c_1 > \notin A$, definición de antisimétrica. Por tanto, las reglas Ac1 y Ac2 son casos particulares de Ac3. No es necesario decir de una relación que es acíclica que además es antirreflexiva o antisimétrica.

Tabla B.3 DemoI1: Demostración de la regla de Inclusión/Incompatibilidad 1.

2.	**S1 ⇒ At** Si una relación presenta una incompatibilidad de salida consigo misma, quiere decir que, $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$ Supongamos que una relación es transitiva, entonces $\forall\ \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^+(c_1, r_i)| = 2$, contradice la definición de S1 Por tanto, si una relación cumple S1, también es antitransitiva. **S1 ⇒ ¬ S2** Supongamos que una relación cumple S2, entonces $\forall\ c_i \in C$ y $r_c \in R_c$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r_q \in R_c - \{r_c\}$, $|\Gamma^+(c_i, r_q)| = 0$ como además cumple S1, entonces $\forall\ c_i \in C$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$, definición de S3 Por tanto, no se puede decir que una relación cumple S1 y S2 porque es lo mismo que decir que cumple S3.

Tabla B.4 DemoI2: Demostración de la regla de Inclusión/Incompatibilidad 2.

3.	**S2 ⇒ As3** Si una relación presenta una incompatibilidad de salida con el resto de las relaciones, quiere decir que, $\forall\ c_i \in C$ y $r_c \in R_c$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r_q \in R_c - \{r_c\}$, $|\Gamma^+(c_i, r_q)| = 0$ Por otro lado, dado que se cumple As1 por ser universal $\forall < c_i, r_i, c_d > \in A_c, \nexists < c_i, s_i, c_d > \in A_c$, tal que $r_c.nombre = s_c.nombre$ esto es, $|\Gamma^-(c_i, r_i)| = 1$ como además se cumple S2, entonces $\Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$, definición As3 Por tanto, si una relación cumple S2, también cumple As3. **S2 ⇒ ¬ S1** Supongamos que una relación cumple S1, entonces $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$ como además cumple S2, entonces $\forall\ c_i \in C$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$, definición de S3 Por tanto, no se puede decir que una relación cumple S2 y S1 porque es lo mismo que decir que cumple S3.

Tabla B.5 DemoI3: Demostración de la regla de Inclusión/Incompatibilidad 3.

4.	**S3 ⇒ At** Si una relación presenta una incompatibilidad de salida total, quiere decir que, $\forall\ c_i \in C,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ Supongamos que una relación es transitiva, entonces $\forall\ \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^+(c_1, r_i)| = 2,$ Por tanto, si una relación cumple S1, también es antitransitiva. **S3 ⇒ As3** Si una relación cumple S3, entonces $\forall\ c_i, c_d \in C,\ \Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$, definición As3 Por tanto, si una relación cumple S3, también cumple As3. **S3 ⇒ S1 ∧ S2** Si una relación cumple S3, entonces $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$, definición S1 y $\forall\ c_i \in C$ y $r_c \in R_c,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^+(c_i, r_q)| = 0$, definición S2 esto es S3 = S1 + S2, luego si ya cumple S3 no tiene sentido decir que cumple S1 y S2.

Tabla B.6 DemoI4: Demostración de la regla de Inclusión/Incompatibilidad 4.

5.	**LI1 ⇒ At** Si una relación presenta una incompatibilidad de llegada consigo misma, quiere decir que, $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 1$ Supongamos que una relación es transitiva, entonces $\forall\ \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^-(c_n, r_i)| = 2,$ Por tanto, si una relación cumple S1, también es antitransitiva. **LI1 ⇒ ¬ LI2** Supongamos que una relación cumple LI2, entonces $\forall\ c_i \in C$ y $r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^-(c_i, r_q)| = 0$ como además cumple LI1, entonces $\forall\ c_i \in C,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^-(c_i, r)| = 0$, definición de LI3 Por tanto, no se puede decir que una relación cumple LI1 y LI2 porque es lo mismo que decir que cumple LI3.

Tabla B.7 DemoI5: Demostración de la regla de Inclusión/Incompatibilidad 5.

6.	**LI2 ⇒ As3** Dado que se cumple As1 por ser universal $\forall <c_i, r_i, c_d> \in A_c \Rightarrow \nexists <c_i, s_i, c_d> \in A_c$, tal que r_c.nombre = s_c.nombre esto es, $\lvert\Gamma^-(c_d, r_i)\rvert = 1$ como además se cumple LI2, entonces $\Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$, definición As3 Por tanto, si una relación cumple LI2, también cumple As3. **LI2 ⇒ ¬ LI1** Si una relación presenta una incompatibilidad de llegada con el resto de las relaciones, quiere decir que, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^-(c_i, r_q)\rvert = 0$ Supongamos que una relación cumple LI1, entonces $\forall c_i \in C$ y $r_c \in R_c \Rightarrow \lvert\Gamma^-(c_i, r_c)\rvert \leq 1$ como además cumple LI2, entonces $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$, definición de LI3 Por tanto, no se puede decir que un relación cumple LI2 y LI1 porque es lo mismo que decir que cumple LI3.

Tabla B.8 DemoI6: Demostración de la regla de Inclusión/Incompatibilidad 6.

7.	**LI3 ⇒ At** Si una relación presenta una incompatibilidad de llegada total, quiere decir que, $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$ Supongamos que una relación es transitiva, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n> \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $\lvert\Gamma^-(c_n, r_i)\rvert = 2$, Por tanto, si una relación cumple LI3, también es antitransitiva. **LI3 ⇒ As3** Si una relación cumple LI3, entonces $\forall c_i, c_d \in C$, $\Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$, definición As3 Por tanto, si una relación cumple LI3, también cumple As3. **LI3 ⇒ LI1 ∧ LI2** Si una relación cumple LI3, entonces $\forall c_i \in C$ y $r_c \in R_c \Rightarrow \lvert\Gamma^-(c_i, r_c)\rvert \leq 1$, definición LI1 y $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$, definición LI2 esto es LI3 = LI1 + LI2, luego si ya cumple LI3 no tiene sentido decir que cumple LI1 ó LI2.

Tabla B.9 DemoI7: Demostración de la regla de Inclusión/Incompatibilidad 7.

8.	**Tr1 ⇒ Ac1; Tr1 ⇒ Ac2; Tr1 ⇒ Ac3** Si una relación presenta una incompatibilidad de transmisión consigo misma quiere decir que, $\forall c_i \in C, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_i)| = 0$ Supongamos que una restricción es cíclica, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n> \in EC_m \Rightarrow c_1 = c_n$ esto significa que, $\forall c_i \in C \Rightarrow |\Gamma^-(c_i, r_i)| \neq 0, |\Gamma^+(c_i, r_i)| \neq 0$ que contradice a Tr1. Por tanto, tiene que ser acíclica. Como antes hemos visto que Ac2 y Ac3 son casos particulares de Ac1, también las incluye. **Tr1 ⇒ At** Supongamos que una relación es transitiva, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n> \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^+(c_1, r_i)| = 2$, que contradice la definición de Tr1 Por tanto, si una relación cumple Tr1, también es antitransitiva. **Tr1 ⇒ ¬Tr2** Supongamos que una relación cumple Tr2, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}, |\Gamma^+(c_i, r_q)| = 0$ como además cumple Tr1, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c, |\Gamma^+(c_i, r)| = 0$, definición de Tr3 Por tanto, no se puede decir que un relación cumple Tr1 y Tr2 porque es lo mismo que decir que cumple Tr3.

Tabla B.10 DemoI8: Demostración de la regla de Inclusión/Incompatibilidad 8.

9.	**Tr2 ⇒ ¬ Tr1** Si una relación presenta una incompatibilidad de transmisión Tr2 con el resto de las relaciones, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}, |\Gamma^+(c_i, r_q)| = 0$ Supongamos que una relación cumple Tr1, entonces $\forall c_i \in C, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_i)| = 0$ como además cumple Tr2, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c, |\Gamma^+(c_i, r)| = 0$, definición de Tr3 Por tanto, no se puede decir que un relación cumple Tr2 y Tr1 porque es lo mismo que decir que cumple Tr3.

Tabla B.11 DemoI9: Demostración de la regla de Inclusión/Incompatibilidad 9.

10.	**Tr3 ⇒ Ac1; Tr3 ⇒ Ac2; Tr3 ⇒ Ac3** Si una relación presenta una incompatibilidad de transmisión total quiere decir que, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_i)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^+(c_i, r)\rvert = 0$ Supongamos que una restricción es cíclica, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow c_1 = c_n$ esto significa que, $\forall c_i \in C \Rightarrow \lvert\Gamma^-(c_i, r_i)\rvert \neq 0$, $\lvert\Gamma^+(c_i, r_i)\rvert \neq 0$ que contradice a Tr3. Por tanto, tiene que ser acíclica. Como antes hemos visto que Ac2 y Ac3 son casos particulares de Ac1, también incluye a estas restricciones. **Tr3 ⇒ At** Supongamos que una relación es transitiva, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $\lvert\Gamma^+(c_1, r_i)\rvert = 2$, que contradice la definición de Tr3 Por tanto, si una relación cumple Tr3, también es antitransitiva. **Tr3 ⇒ Tr1 ∧ Tr2** Si una relación cumple Tr3, entonces $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_i)\rvert \neq 0 \Rightarrow \lvert\Gamma^+(c_i, r_i)\rvert = 0$, definición Ll1 y $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^+(c_i, r_q)\rvert = 0$, definición Ll2 esto es Tr3 = Tr1 + Tr2, luego si ya cumple Tr3 no tiene sentido decir que cumple Tr1 ó Tr2.

Tabla B.12 DemoI10: Demostración de la regla de Inclusión/Incompatibilidad 10.

3.2 Programa1 : Elimina Combinaciones No Válidas

La demostración de las reglas de la tabla B.2 prueba que no son posibles las combinaciones en las que aparezcan las restricciones de la derecha de la implicación si en esa misma combinación se encuentra la restricción de la izquierda. Las causas por las que no es posible su combinación será porque, o bien la restricción de la derecha incluye a la de la izquierda, o bien porque es incompatible con respecto a la restricción de la izquierda.

Se ha escrito un algoritmo —ver apéndice C— que elimina las combinaciones no válidas y calcula el conjunto de combinaciones que, en el caso de ser consistentes, serían válidas en nuestro sistema. Este algoritmo elimina un total de 16094 combinaciones, siendo *a priori* válidas un total de 290 (2^{14} = 16384 - 16094 = 290). Este programa ha sido desarrollado en ANSI C.

```
Programa1 Elimina-no-válidas;
Comb = Genera-todas-combinaciones-restricciones-relacionales;
for  all (combinación ∈ Comb) {
      if (Ac3 and  (Ac1 or Ac2)) quita (combinación);
      if (S1 and (At or S2 or S3)) quita (combinación);
      if (S2 and (S1 or S3 or As3)) quita (combinación);
      if (S3 and (At or S1 or S2 or As3)) quita (combinación);
      if (Ll1 and (At or Ll2 or Ll3)) quita (combinación);
      if (Ll2 and (Ll1 or Ll3 or As3)) quita (combinación);
      if (Ll3 and (At or Ll1 or Ll2 or As3)) quita (combinación);
      if (Tr1 and (Ac1 or Ac2 or Ac3 or At or Tr2 or Tr3)) quita (combinación);
      if (Tr2 and (Tr1 or Tr3)) quita (combinación);
      if (Tr3 and (Ac1 or Ac2 or Ac3 or At or Tr1 or Tr2)) quita (combinación);
}
```

4 Combinaciones Consistentes

Queda, por tanto, demostrar que estas 290 posibilidades son consistentes. Sin embargo, se va a intentar reducir el número de demostraciones. Para ello, se hace uso de la posibilidad de agrupar las restricciones en las seis categorías anteriormente mencionadas: Acíclico —Ac1, Ac2 y Ac3—, Antitransitiva —At—, Salida —S1, S2 y S3—, Llegada —Ll1, Ll2 y Ll3—, Transmisión —Tr1, Tr2 y Tr3—, y Asociación —As3—. Y esto es posible porque una de las restricciones puede representar al resto. Por ejemplo, tal y como se ha demostrado en DemoI1 la restricción que niega la posibilidad de existencia de ciclos Ac3 incluye a la antirreflexiva Ac1 y a la antisimétrica Ac2. La antirreflexiva y la antisimétrica son la negación de ciclos de uno y dos pasos respectivamente. En la tabla B.13 se muestra la regla que representa al grupo.

Esto significa que, demostrando la consistencia para estos representantes estaríamos demostrando la validez de aquellas otras combinaciones en las que aparezcan algunas de las restricciones del grupo.

Ac	Ac3 ⊃ Ac1, Ac2
At	At
S	S3 ⊃ S1, S2
Ll	Ll3 ⊃ Ll1, Ll2
Tr	Tr3 ⊃ Tr1, Tr2
As	As3 ⊃ As1, As2

Tabla B.13 Representantes de las Restricciones sobre las Relaciones Conceptuales.

4.1 Demostración de los Representantes

La mayoría de las inclusiones de la tabla B.13 han sido demostradas anteriormente. Para Ac consulte la demostración de la regla de inclusión/incompatibilidad número 1 —tabla Demo1—, para S la demostración de la regla número 4 —tabla Demo4—, para Ll la demostración de la regla número 7 —tabla Demo7— y, para Tr la demostración de la regla número 10 —tabla Demo10—. Únicamente, queda por probar As, tal y como se describe a continuación.

	As3 ⊃ As1, As2
As	Si una relación cumple As3, entonces $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ esto quiere decir que, $\forall <c_o, r_c, c_d> \in A_c, \nexists <c_o, s_c, c_d> \in A_c$, tal que r_c.nombre = s_c.nombre., definición As1 y $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall\ r_q \in R_c - \{r_c\}, <c_o, r_q, c_d> \notin A$, definición As2 esto es As3 incluye a As1 y a As2.

Tabla B.14 DemoAs: Demostración del Representante As.

4.2 Reglas Representantes

Estas seis reglas representantes —tabla B.13— permiten resumir varias combinaciones válidas y demostrar su consistencia. En la siguiente —tabla B.15—, se indica cuándo dos restricciones —o sus representantes— se pueden dar al mismo tiempo, esto es, son consistentes. Al igual que antes, para interpretarla nos situamos en una fila, suponemos que la restricción de la fila se cumple e intentamos añadir cada una de las restricciones de las columnas. Para aquellas restricciones cuyo añadido no suponga ninguna inconsistencia se marcará la casilla con una C —con || estamos indicando lo misma pero leída por columnas—.

Estas mismas consistencias se recogen en la tabla B.16, y vienen a decir que aquellas combinaciones en las que aparezcan estas reglas son válidas y consistentes. Para ello, se debe demostrar que las 290 combinaciones cumplen un subconjunto de estas aserciones. Pero, además, se debe probar que estos axiomas son capaces de generar las 290 combinaciones mediante deducción lógica —es decir, que el sistema formal es semánticamente completo—.

Para demostrar lo primero se ha elaborado un programa —en C— que comprueba que, efectivamente, todas las combinaciones supuestamente válidas verifican una o varias de estas reglas. Para probar lo segundo se ha construido un programa lógico —en Prolog— que verdaderamente es capaz de inferir todas las combinaciones de restricciones relacionales que son válidas en nuestro sistema. Ergo, si se demuestran las 16 aserciones, se está demostrando la consistencia de las 290 combinaciones válidas.

RT_a	Acíclicas			AT	Salida			Llegada			Transmisión			As
	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Ac1	▪▪▪▪	c	--	C	C			C			--	C	--	C
Ac2	| |	▪▪▪▪	--								--		--	
Ac3	x	x	▪▪▪▪								--		--	
At	| |			▪▪▪▪	--	c	--	--	c	--	--	c	--	c
S1	| |			x	▪▪▪▪	x	x	C			C			c
S2				| |	x	▪▪▪▪	x						x
S3				x	x	x	▪▪▪▪							x
Ll1	| |			x	| |		▪▪▪▪	x	x	C			c
Ll2				| |			x	▪▪▪▪	x				x
Ll3				x				x	x	▪▪▪▪				x
Tr1	x	x	x	x	| |			| |			▪▪▪▪	x	x	C
Tr2	| |			| |						x	▪▪▪▪	x	
Tr3	x	x	x	x							x	x	▪▪▪▪	
As3	| |			| |	| |	--	--	| |	--	--	| |		▪▪▪▪

Tabla B.15 Consistencias entre las Restricciones sobre las Relaciones Conceptuales.

1.	1.	Ac1 and Ac2
2.	2.	Ac and At
	3.	Ac and S
	4.	Ac and Ll
	5.	Ac and Tr2
	6.	Ac and As
3.	7.	At and S2
	8.	At and Ll2
	9.	At and Tr2
	10.	At and As
4.	11.	S and Ll
	12.	S and Tr
5.	13.	S1 and As
6.	14.	Ll and Tr
7.	15.	Ll1 and As
8.	16.	Tr and As

Tabla B.16 Axiomas de consistencia entre Restricciones sobre las Relaciones Conceptuales.

4.3 Demostración de las Combinaciones Consistentes

A continuación se demuestra —basándonos en las propias definiciones de las restricciones conceptuales— que son consistentes las combinaciones que se generen a partir de los axiomas de la tabla B16. Estas demostraciones se han agrupado en ocho bloques debido a que con frecuencia una misma restricción aparece combinada con varias de las otras.

1.	**Ac1 and Ac2** Si una relación es antirreflexiva Ac1 quiere decir que, $\forall < c_1, r_i, c_2 > \in A \Rightarrow c_1 \neq c_2$ Supongamos que la relación es antisimétrica Ac2, entonces, $\forall < c_1, r_i, c_2 > \in A \Rightarrow < c_2, r_i, c_1 > \notin A$ No nos conduce a ninguna incompatibilidad, ni inclusión, decir que no se permiten ciclos de un paso ni de dos pasos. Por otro lado, no hay ningún tipo de inconsistencia en decir que se permiten ciclos de un paso pero no de dos o viceversa.

Tabla B.17 DemoC1: Demostración de la regla de consistencia 1.

2.	**Ac and At** Si una relación es acíclica Ac quiere decir que, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow c_1 \neq c_n$ luego $\Gamma^-(c_1, r_i) \neq c_n$, $\Gamma^+(c_n, r_i) \neq c_1$ Supongamos que la relación es antitransitiva At, entonces, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow < c_1, r_i, c_n > \notin A$ luego $\Gamma^+(c_1, r_i) \neq c_n$, $\Gamma^-(c_n, r_i) \neq c_1$ En consecuencia, una relación puede ser al mismo tiempo acíclica y antitransitiva, o cíclica y antitransitiva, o acíclica y transitiva. **Ac and S** Supongamos que la relación presenta incompatibilidad de salida total S, entonces, $\forall c_i \in C, |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ac y S, o cumplir Ac y no S, o no cumplir Ac y sí S. **Ac and Ll** Supongamos que la relación presenta incompatibilidad de llegada total Ll, entonces, $\forall c_i \in C, |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^-(c_i, r)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ac y Ll, o cumplir Ac y no Ll, o no cumplir Ac y sí Ll. **Ac and Tr2** Supongamos que la relación presenta incompatibilidad de transmisión Tr2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $|\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}, |\Gamma^+(c_i, r_q)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ac y Tr2, o cumplir Ac y no Tr2, o no cumplir Ac y sí Tr2. **Ac and As** Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$ En consecuencia, una relación puede cumplir al tiempo Ac y As, o cumplir Ac y no As, o no cumplir Ac y sí As.

Tabla B.18 DemoC2: Demostración de la regla de consistencia 2-6.

3.	**At and S2** Si una relación es antitransitiva At quiere decir que, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow < c_1, r_i, c_n > \notin A$ luego $\Gamma^+(c_1, r_i) \neq c_n$, $\Gamma^-(c_n, r_i) \neq c_1$ Supongamos que la relación presenta incompatibilidad de salida S2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^+(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^+(c_i, r_q)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo At y S2, o cumplir At y no S2, o no cumplir At y sí S2. **At and Ll2** Supongamos que la relación presenta incompatibilidad de llegada Ll2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^-(c_i, r_q)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo At y Ll2, o cumplir At y no Ll2, o no cumplir At y sí Ll2. **At and Tr2** Supongamos que la relación presenta incompatibilidad de transmisión Tr2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^+(c_i, r_q)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo At y Tr2, o cumplir At y no Tr2, o no cumplir At y sí Tr2. **At and As** Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$ En consecuencia, una relación puede cumplir al tiempo At y As, o cumplir At y no As, o no cumplir At y sí As.

Tabla B.19 DemoC3: Demostración de la regla de consistencia 7-10.

4.	**S and Ll** Si una relación presenta incompatibilidad de salida total S quiere decir que, $\forall c_i \in C$, $\lvert\Gamma^+(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^+(c_i, r)\rvert = 0$ Supongamos que la relación presenta incompatibilidad de llegada total Ll, entonces, $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo S y Ll, o cumplir S y no Ll, o no cumplir S y sí Ll. **S and Tr** Supongamos que la relación presenta incompatibilidad de transmisión total Tr, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_i)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^+(c_i, r)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo S y Tr, o cumplir S y no Tr, o no cumplir S y sí Tr.

Tabla B.20 DemoC4: Demostración de la regla de consistencia 11-12.

5.	**S1 and As** Si una relación presenta incompatibilidad de salida S1 quiere decir que, $\forall c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$ Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ luego $\forall c_i \in C$, $|\Gamma^+(c_i, r_i)| = 1$ y , $|\Gamma^-(c_i, r_i)| = 1$ En consecuencia, una relación puede cumplir al tiempo S1 y As, o cumplir S1 y no As, o no cumplir S1 y sí As.

Tabla B.21 DemoC5: Demostración de la regla de consistencia 13.

6.	**Ll and Tr** Si una relación presenta incompatibilidad de llegada total Ll quiere decir que, $\forall c_i \in C$, $|\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^-(c_i, r)| = 0$ Supongamos que la relación presenta incompatibilidad de transmisión total Tr, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $|\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ll y Tr, o cumplir Ll y no Tr, o no cumplir Ll y sí Tr.

Tabla B.22 DemoC6: Demostración de la regla de consistencia 14.

7.	**Ll1 and As** Si una relación presenta incompatibilidad de salida Ll1 quiere decir que, $\forall c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 1$ Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ luego $\forall c_i \in C$, $|\Gamma^+(c_i, r_c)| = 1$ y , $|\Gamma^-(c_i, r_c)| = 1$ En consecuencia, una relación puede cumplir al tiempo Ll1 y As, o cumplir Ll1 y no As, o no cumplir Ll1 y sí As.

Tabla B.23 DemoC7: Demostración de la regla de consistencia 15.

8.	**Tr and As** Si una relación presenta incompatibilidad de transmisión total Tr quiere decir que, $\forall c_i \in C$ y $r_c \in R_c$, $|\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ luego $\forall c_i \in C$, $|\Gamma^+(c_i, r_c)| = 1$ y , $|\Gamma^-(c_i, r_c)| = 1$ En consecuencia, una relación puede cumplir al tiempo Ll1 y As, o cumplir Ll1 y no As, o no cumplir Ll1 y sí As.

Tabla B.24 DemoC8: Demostración de la regla de consistencia 16.

4.4 Programa 2: Comprueba que las Combinaciones Válidas cumplen los Axiomas

Este programa comprueba que todas las combinaciones válidas cumplen los axiomas de la tabla B.16. Para ello se recorren todas ellas y se va cotejando si verifica uno o más de los 16 axiomas posibles. Este programa ha sido desarrollado en ANSI C.

```
Programa2 Cumple-axiomas;
Genera-combinaciones-restricciones-consistentes;
for  all (combinación-consistente) {
      if ( Ac1 and Ac2) cumple (axioma1, combinación-consistente);
      if (Ac and At) cumple (axioma2, combinación-consistente);
      if (Ac and S) cumple (axioma3, combinación-consistente);
      if (Ac and Ll) cumple (axioma4, combinación-consistente);
      if (Ac and Tr2) cumple (axioma5, combinación-consistente);
      if (Ac and As) cumple (axioma6, combinación-consistente);
      if (At and S2) cumple (axioma7, combinación-consistente);
      if (At and Ll2) cumple (axioma8, combinación-consistente);
      if (At and Tr2) cumple (axioma9, combinación-consistente);
      if (At and As) cumple (axioma10, combinación-consistente);
      if (S and Ll) cumple (axioma11, combinación-consistente);
      if (S and Tr) cumple (axioma12, combinación-consistente);
      if (S1 and As) cumple (axioma13, combinación-consistente);
      if (Ll and Tr) cumple (axioma14, combinación-consistente);
      if (Ll1 and As) cumple (axioma15, combinación-consistente);
      if (Tr and As) cumple (axioma16, combinación-consistente);
}
```

4.5 Programa 3: Genera a partir de los Axiomas las Combinaciones Válidas

Este programa lógico ha sido implementado en Prolog —con el intérprete JIP, Jave Internet en Prolog—. Su ejecución genera las combinaciones válidas a partir de los axiomas de la tabla B.16, las mismas se calcularon eliminando las inconsistentes.

```
//Programa3 GeneraCombinaciones

aciclico(ac1). aciclico(ac2). aciclico(ac3). anti(at). salida(s1). salida(s2). salida(s3).
llegada(ll1). llegada(ll2). llegada(ll3). transmision(tr1). transmision(tr2). transmision(tr3). asociacion(as).

posible2(ac1, ac2).
posible2(A,B):- aciclico(A), anti(B).
posible2(A,B):- aciclico(A), salida(B).
posible2(A,B):- aciclico(A), llegada(B).
posible2(A,tr2):- aciclico(A).
posible2(A,as):- aciclico(A).
posible2(at,s2).
posible2(at,ll2).
```

```
posible2(at,tr2).
posible2(at,as).
posible2(A,B):- salida(A), llegada(B).
posible2(A,B):- salida(A), transmision(B).
posible2(s1,as).
posible2(A,B):- llegada(A), transmision(B).
posible2(ll1,as).
posible2(A,as):- transmision(A).

posible3(A, B, C):- posible2(A, B), posible2(A, C), posible2(B, C).
posible4(A,B,C,D):- posible3(A,B,C), posible3(A,B,D), posible3(A,C,D), posible3(B,C,D).
posible5(A,B,C,D,E):-
   posible4(A,B,C,D), posible4(A,B,C,E), posible4(A,B,D,E), posible4(A,C,D,E), posible4(B,C,D,E).
posible6(A,B,C,D,E,F):-
   posible5(A,B,C,D,E), posible5(A,B,C,D,F), posible5(A,B,C,E,F), posible5(A,B,D,E,F),
posible5(A,C,D,E,F), posible5(B,C,D,E,F).
```

En conclusión, las demostraciones de las tablas B.2, B13 y B16, y los programas 1, 2 y 3 prueban la inconsistencia de las combinaciones no válidas y la consistencia de las combinaciones válidas. En el apéndice C se puede consultar los códigos de los programas y los resultados obtenidos con su ejecución.

Apéndice B

Demostración de la Consistencia de RT_A

1 Introducción

En todo momento, el conjunto de restricciones asociado a una relación conceptual debe ser consistente. Cualquier combinación de restricciones no tiene por qué ser válida. Por tanto, hay que determinar el conjunto de combinaciones válidas y, en consecuencia, consistentes. Antes de añadir o borrar una de las restricciones que caracterizan a una relación conceptual se debe comprobar nuevamente que el conjunto de restricciones es consistente. Para ello, únicamente deberemos comprobar que es una de las combinaciones válidas.

Se ha optado por una demostración computacional que ha consistido en los siguientes pasos:

1. Generar con un algoritmo todas las combinaciones posibles que se pueden generar con las 16 restricciones relacionales —se pueden hacer hasta 2^{16} caracterizaciones posibles a una relación conceptual—.
2. Rechazar aquellas combinaciones que son inconsistentes —se ha conseguido reducir el anterior número a un total de 290 combinaciones a priori válidas—.
3. Demostrar que las combinaciones que han quedado después de eliminar las no válidas son consistentes.

Para alcanzar el objetivo del paso 2:

1. Se han establecido reglas de inclusión e incompatibilidad entre algunas de las restricciones.
2. Se ha demostrado la veracidad de estas reglas.
3. Se ha escrito un algoritmo que elimina aquellas combinaciones no válidas, esto es, que contienen estas restricciones incluyentes o incompatibles.

Finalmente, para demostrar que las combinaciones que han resultado del anterior proceso son válidas se ha actuado de la siguiente manera:

1. Dado el gran número de combinaciones —un total de 290— que hace inviable en la práctica la demostración individual, se ha optado por agruparlas en representantes.
2. Se han demostrado las reglas representantes que permiten resumir las combinaciones válidas.
3. Con los representantes se han establecido reglas o axiomas de consistencia.
4. Se han demostrado estas reglas de consistencia.
5. Se ha escrito un algoritmo que comprueba que todas las combinaciones válidas no eliminadas contienen un subconjunto de estas reglas válidas.
6. Por último, se ha comprobado que este conjunto de reglas de consistencia es completo. Para ello, se ha elaborado un algoritmo en Prolog el cual, a partir de estos axiomas, genera las mismas 90 combinaciones válidas del paso 2.

2 Combinaciones Posibles

A cada una de las relaciones que se defina en el sistema se le asociará un subconjunto del conjunto de las dieciséis restricciones que proporciona el sistema. En todo momento, el subconjunto de restricciones deberá ser válido, esto es, no debe suponer ningún tipo de inconsistencia.

$$\{Ac1, Ac2, Ac3, At, S1, S2, S3, Ll1, Ll2, Ll3, Tr1, Tr2, Tr3, As1, As2, As3\} = RT_a$$

Dado que el sistema es capaz de ofrecer dieciséis restricciones, el autor podría seleccionar, en principio, cualquier combinación de las mismas. Por tanto, con estas dieciséis restricciones se pueden hacer hasta 2^{16} caracterizaciones posibles de una relación conceptual. Esto quiere decir, que deberíamos demostrar cuáles de las 2^{16} combinaciones son válidas o posibles. Esto puede parecer inviable o, al menos, bastante laborioso. En realidad, vamos a ver que se puede reducir de manera cuantiosa esta cifra.

RT_a	Acíclicas			AT	Salida			Llegada			Transmisión			Asociación		
	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As1	As2	As3
Ac1	■■■■															
Ac2		■■■■														
Ac3	⊂	⊂	■■■■													
At				■■■■												
S1				⊂	■■■■	><	><							⊂		
S2					><	■■■■	><								⊂	⊂
S3				⊂	⊂	⊂	■■■■							⊂	⊂	⊂
Ll1				⊂				■■■■	><	><				⊂		
Ll2								><	■■■■	><					⊂	⊂
Ll3				⊂				⊂	⊂	■■■■				⊂	⊂	⊂
Tr1	⊂	⊂	⊂	⊂							■■■■	><	><			
Tr2											><	■■■■	><			
Tr3	⊂	⊂	⊂	⊂							⊂	⊂	■■■■			
As1														■■■■	><	><
As2														><	■■■■	><
As3														⊂	⊂	■■■■

Tabla B.1 Inclusiones e Incompatibilidades entre las Restricciones sobre las Relaciones Conceptuales.

En primer lugar, la restricción As1 —ver tabla B.1— no es elegible; es una restricción universal que tiene que cumplir cualquier relación. Luego el sistema ofrece tan sólo quince reglas, y no dieciséis. Esto quiere decir que las combinaciones quedan automáticamente reducidas a la mitad —2^{15}—.

Por otro lado, no tiene sentido decir que una relación debe cumplir la restricción As2 porque, dado que ya cumple As1, esto sería equivalente a decir que cumple directamente la restricción As3. En consecuencia, la regla As2 no se debe ofrecer. Esto significa, que el sistema va a ofrecer únicamente catorce restricciones o lo que es lo mismo hasta un total de 2^{14} combinaciones.

3 Combinaciones Inconsistentes

Se ha logrado reducir el número de posibilidades a una cuarta parte. No parece nada alentador pero aún se puede disminuir más el número de combinaciones válidas. Si nos fijamos en estas restricciones muchas de ellas se solapan; por tanto, no tiene sentido asociar a una relación una restricción cuando ya tiene asociada otra que incluye a ésta. Por otro lado, las restricciones pertenecientes a un mismo grupo —Salida, Llegada, Transmisión, ver tabla B.1— son incompatibles y, en consecuencia, inconsistentes.

En la tabla B.1 se resumen todas estas inclusiones e incompatibilidades. Para interpretarla, nos situamos en una fila, suponemos que la restricción de la fila se cumple, e intentamos añadir cada una de las restricciones de las columnas. Para aquellas restricciones cuyo añadido suponga una inclusión se marcará la casilla con el símbolo $\subset$; para aquellas otras que suponga una incompatibilidad se marcará la casilla con el símbolo $><$.

En general, diremos que una restricción rt_1 *incluye* a otra rt_2, cuando se cumpla que si $rt_1 \Rightarrow rt_2$. Así, la restricción rt_1 es más fuerte que rt_2, de forma que sólo imponiendo rt_1 también se está obligando a cumplir rt_2. Diremos que rt_1 es *incompatible* con respecto a rt_2, si $rt_1 \Rightarrow \neg\ rt_2$, esto es, la restricción rt_1 implica que no se dé rt_2. La razón por la que obligamos a que no se cumpla esa otra restricción es porque, en ese caso, existe otra restricción que incluye a ambas. Por tanto, únicamente debemos demostrar las reglas de inclusión, porque la incompatibilidad de algunas de las reglas tiene su origen en la existencia de una restricción que incluye a ambas —en concreto, la restricción tercera de los grupos Salida, Llegada, Transmisión y Asociación es la unión de las restricciones uno y dos de su grupo—.

La tabla B.2 recoge el conjunto de reglas de inclusión e incompatibilidad de la tabla B.1. La demostración de estas reglas se incluyen en el siguente apartado.

Nº	Reglas de Inclusión	Reglas de Incompatibilidad
1.	Ac3 ⇒ Ac1 ∧ Ac2	
2.	S1 ⇒ At	S1 ⇒ ¬ S2
3.	S2 ⇒ As3	S2 ⇒ ¬ S1
4.	S3 ⇒ At; S3 ⇒ As3; S3 ⇒ S1 ∧ S2	
5.	Ll1 ⇒ At	Ll1 ⇒ ¬ Ll2
6.	Ll2 ⇒ As3	Ll2 ⇒ ¬ Ll1
7.	Ll3 ⇒ At; Ll3 ⇒ As3; Ll3 ⇒ Ll1 ∧ Ll2	
8.	Tr1 ⇒ Ac1; Tr1 ⇒ Ac2; Tr1 ⇒ Ac3; Tr1 ⇒ At	Tr1 ⇒ ¬ Tr2
9.		Tr2 ⇒ ¬Tr1
10.	Tr3 ⇒ Ac1; Tr3 ⇒ Ac2; Tr3 ⇒ Ac3; Tr3 ⇒ At; Tr3 ⇒ Tr1 ∧ Tr2	

Tabla B.2 Reglas de inclusión o incompatibilidad entre las Restricciones sobre Relaciones Conceptuales.

3.1 Demostración de las Reglas de Inclusión e Incompatibilidad de la Tabla B.2

1.	**Ac3 ⇒ Ac1 ∧ Ac2** Si una relación es acíclica quiere decir que, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow c_1 \neq c_n$ Sea n, el número de tripletas que tiene el ciclo. Con n=1, $\forall < c_1, r_i, c_2 > \in A \Rightarrow c_1 \neq c_2$, definición de antirreflexiva. Con n=2, $\forall < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \in A \Rightarrow c_1 \neq c_3$, esto es, $\forall < c_1, r_i, c_2 > \in A \Rightarrow < c_2, r_i, c_1 > \notin A$, definición de antisimétrica. Por tanto, las reglas Ac1 y Ac2 son casos particulares de Ac3. No es necesario decir de una relación que es acíclica que además es antirreflexiva o antisimétrica.

Tabla B.3 DemoI1: Demostración de la regla de Inclusión/Incompatibilidad 1.

2.	**S1 ⇒ At** Si una relación presenta una incompatibilidad de salida consigo misma, quiere decir que, $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$ Supongamos que una relación es transitiva, entonces $\forall\ \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^+(c_1, r_i)| = 2$, contradice la definición de S1 Por tanto, si una relación cumple S1, también es antitransitiva. **S1 ⇒ ¬ S2** Supongamos que una relación cumple S2, entonces $\forall\ c_i \in C$ y $r_c \in R_c$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r_q \in R_c - \{r_c\}$, $|\Gamma^+(c_i, r_q)| = 0$ como además cumple S1, entonces $\forall\ c_i \in C$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$, definición de S3 Por tanto, no se puede decir que una relación cumple S1 y S2 porque es lo mismo que decir que cumple S3.

Tabla B.4 DemoI2: Demostración de la regla de Inclusión/Incompatibilidad 2.

3.	**S2 ⇒ As3** Si una relación presenta una incompatibilidad de salida con el resto de las relaciones, quiere decir que, $\forall\ c_i \in C$ y $r_c \in R_c$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r_q \in R_c - \{r_c\}$, $|\Gamma^+(c_i, r_q)| = 0$ Por otro lado, dado que se cumple As1 por ser universal $\forall < c_i, r_i, c_d > \in A_c, \nexists < c_i, s_i, c_d > \in A_c$, tal que $r_c.nombre = s_c.nombre$ esto es, $|\Gamma^-(c_i, r_i)| = 1$ como además se cumple S2, entonces $\Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$, definición As3 Por tanto, si una relación cumple S2, también cumple As3. **S2 ⇒ ¬ S1** Supongamos que una relación cumple S1, entonces $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$ como además cumple S2, entonces $\forall\ c_i \in C$, $|\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$, definición de S3 Por tanto, no se puede decir que una relación cumple S2 y S1 porque es lo mismo que decir que cumple S3.

Tabla B.5 DemoI3: Demostración de la regla de Inclusión/Incompatibilidad 3.

4.	**S3 ⇒ At** Si una relación presenta una incompatibilidad de salida total, quiere decir que, $\forall\ c_i \in C,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ Supongamos que una relación es transitiva, entonces $\forall\ \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^+(c_1, r_i)| = 2,$ Por tanto, si una relación cumple S1, también es antitransitiva. **S3 ⇒ As3** Si una relación cumple S3, entonces $\forall\ c_i, c_d \in C,\ \Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$, definición As3 Por tanto, si una relación cumple S3, también cumple As3. **S3 ⇒ S1 ∧ S2** Si una relación cumple S3, entonces $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$, definición S1 y $\forall\ c_i \in C$ y $r_c \in R_c,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^+(c_i, r_q)| = 0$, definición S2 esto es S3 = S1 + S2, luego si ya cumple S3 no tiene sentido decir que cumple S1 y S2.

Tabla B.6 DemoI4: Demostración de la regla de Inclusión/Incompatibilidad 4.

5.	**LI1 ⇒ At** Si una relación presenta una incompatibilidad de llegada consigo misma, quiere decir que, $\forall\ c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 1$ Supongamos que una relación es transitiva, entonces $\forall\ \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^-(c_n, r_i)| = 2,$ Por tanto, si una relación cumple S1, también es antitransitiva. **LI1 ⇒ ¬ LI2** Supongamos que una relación cumple LI2, entonces $\forall\ c_i \in C$ y $r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^-(c_i, r_q)| = 0$ como además cumple LI1, entonces $\forall\ c_i \in C,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $|\Gamma^-(c_i, r)| = 0$, definición de LI3 Por tanto, no se puede decir que una relación cumple LI1 y LI2 porque es lo mismo que decir que cumple LI3.

Tabla B.7 DemoI5: Demostración de la regla de Inclusión/Incompatibilidad 5.

6.	**LI2 ⇒ As3** Dado que se cumple As1 por ser universal $\forall <c_i, r_i, c_d> \in A_c \Rightarrow \nexists <c_i, s_i, c_d> \in A_c$, tal que r_c.nombre = s_c.nombre esto es, $\lvert\Gamma^-(c_d, r_i)\rvert = 1$ como además se cumple LI2, entonces $\Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$, definición As3 Por tanto, si una relación cumple LI2, también cumple As3. **LI2 ⇒ ¬ LI1** Si una relación presenta una incompatibilidad de llegada con el resto de las relaciones, quiere decir que, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^-(c_i, r_q)\rvert = 0$ Supongamos que una relación cumple LI1, entonces $\forall c_i \in C$ y $r_c \in R_c \Rightarrow \lvert\Gamma^-(c_i, r_c)\rvert \leq 1$ como además cumple LI2, entonces $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$, definición de LI3 Por tanto, no se puede decir que un relación cumple LI2 y LI1 porque es lo mismo que decir que cumple LI3.

Tabla B.8 DemoI6: Demostración de la regla de Inclusión/Incompatibilidad 6.

7.	**LI3 ⇒ At** Si una relación presenta una incompatibilidad de llegada total, quiere decir que, $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$ Supongamos que una relación es transitiva, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n> \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $\lvert\Gamma^-(c_n, r_i)\rvert = 2$, Por tanto, si una relación cumple LI3, también es antitransitiva. **LI3 ⇒ As3** Si una relación cumple LI3, entonces $\forall c_i, c_d \in C$, $\Lambda(c_i, c_d) = r_j$ esto es, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$, definición As3 Por tanto, si una relación cumple LI3, también cumple As3. **LI3 ⇒ LI1 ∧ LI2** Si una relación cumple LI3, entonces $\forall c_i \in C$ y $r_c \in R_c \Rightarrow \lvert\Gamma^-(c_i, r_c)\rvert \leq 1$, definición LI1 y $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$, definición LI2 esto es LI3 = LI1 + LI2, luego si ya cumple LI3 no tiene sentido decir que cumple LI1 ó LI2.

Tabla B.9 DemoI7: Demostración de la regla de Inclusión/Incompatibilidad 7.

8.	**Tr1 ⇒ Ac1; Tr1 ⇒ Ac2; Tr1 ⇒ Ac3** Si una relación presenta una incompatibilidad de transmisión consigo misma quiere decir que, $\forall c_i \in C, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_i)| = 0$ Supongamos que una restricción es cíclica, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n> \in EC_m \Rightarrow c_1 = c_n$ esto significa que, $\forall c_i \in C \Rightarrow |\Gamma^-(c_i, r_i)| \neq 0, |\Gamma^+(c_i, r_i)| \neq 0$ que contradice a Tr1. Por tanto, tiene que ser acíclica. Como antes hemos visto que Ac2 y Ac3 son casos particulares de Ac1, también las incluye. **Tr1 ⇒ At** Supongamos que una relación es transitiva, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n> \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $|\Gamma^+(c_1, r_i)| = 2$, que contradice la definición de Tr1 Por tanto, si una relación cumple Tr1, también es antitransitiva. **Tr1 ⇒ ¬Tr2** Supongamos que una relación cumple Tr2, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}, |\Gamma^+(c_i, r_q)| = 0$ como además cumple Tr1, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c, |\Gamma^+(c_i, r)| = 0$, definición de Tr3 Por tanto, no se puede decir que un relación cumple Tr1 y Tr2 porque es lo mismo que decir que cumple Tr3.

Tabla B.10 DemoI8: Demostración de la regla de Inclusión/Incompatibilidad 8.

9.	**Tr2 ⇒ ¬ Tr1** Si una relación presenta una incompatibilidad de transmisión Tr2 con el resto de las relaciones, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}, |\Gamma^+(c_i, r_q)| = 0$ Supongamos que una relación cumple Tr1, entonces $\forall c_i \in C, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_i)| = 0$ como además cumple Tr2, entonces $\forall c_i \in C$ y $r_c \in R_c, |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c, |\Gamma^+(c_i, r)| = 0$, definición de Tr3 Por tanto, no se puede decir que un relación cumple Tr2 y Tr1 porque es lo mismo que decir que cumple Tr3.

Tabla B.11 DemoI9: Demostración de la regla de Inclusión/Incompatibilidad 9.

10.	**Tr3 ⇒ Ac1; Tr3 ⇒ Ac2; Tr3 ⇒ Ac3** Si una relación presenta una incompatibilidad de transmisión total quiere decir que, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_i)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^+(c_i, r)\rvert = 0$ Supongamos que una restricción es cíclica, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow c_1 = c_n$ esto significa que, $\forall c_i \in C \Rightarrow \lvert\Gamma^-(c_i, r_i)\rvert \neq 0$, $\lvert\Gamma^+(c_i, r_i)\rvert \neq 0$ que contradice a Tr3. Por tanto, tiene que ser acíclica. Como antes hemos visto que Ac2 y Ac3 son casos particulares de Ac1, también incluye a estas restricciones. **Tr3 ⇒ At** Supongamos que una relación es transitiva, entonces $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, \ldots, < c_{n-1}, r_i, c_n > \in EC \Rightarrow < c_1, r_i, c_n > \in A$ esto quiere decir que, $\lvert\Gamma^+(c_1, r_i)\rvert = 2$, que contradice la definición de Tr3 Por tanto, si una relación cumple Tr3, también es antitransitiva. **Tr3 ⇒ Tr1 ∧ Tr2** Si una relación cumple Tr3, entonces $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_i)\rvert \neq 0 \Rightarrow \lvert\Gamma^+(c_i, r_i)\rvert = 0$, definición Ll1 y $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^+(c_i, r_q)\rvert = 0$, definición Ll2 esto es Tr3 = Tr1 + Tr2, luego si ya cumple Tr3 no tiene sentido decir que cumple Tr1 ó Tr2.

Tabla B.12 DemoI10: Demostración de la regla de Inclusión/Incompatibilidad 10.

3.2 Programa1 : Elimina Combinaciones No Válidas

La demostración de las reglas de la tabla B.2 prueba que no son posibles las combinaciones en las que aparezcan las restricciones de la derecha de la implicación si en esa misma combinación se encuentra la restricción de la izquierda. Las causas por las que no es posible su combinación será porque, o bien la restricción de la derecha incluye a la de la izquierda, o bien porque es incompatible con respecto a la restricción de la izquierda.

Se ha escrito un algoritmo —ver apéndice C— que elimina las combinaciones no válidas y calcula el conjunto de combinaciones que, en el caso de ser consistentes, serían válidas en nuestro sistema. Este algoritmo elimina un total de 16094 combinaciones, siendo *a priori* válidas un total de 290 (2^{14} = 16384 - 16094 = 290). Este programa ha sido desarrollado en ANSI C.

```
Programa1 Elimina-no-válidas;
Comb = Genera-todas-combinaciones-restricciones-relacionales;
for  all (combinación ∈ Comb) {
      if (Ac3 and  (Ac1 or Ac2)) quita (combinación);
      if (S1 and (At or S2 or S3)) quita (combinación);
      if (S2 and (S1 or S3 or As3)) quita (combinación);
      if (S3 and (At or S1 or S2 or As3)) quita (combinación);
      if (Ll1 and (At or Ll2 or Ll3)) quita (combinación);
      if (Ll2 and (Ll1 or Ll3 or As3)) quita (combinación);
      if (Ll3 and (At or Ll1 or Ll2 or As3)) quita (combinación);
      if (Tr1 and (Ac1 or Ac2 or Ac3 or At or Tr2 or Tr3)) quita (combinación);
      if (Tr2 and (Tr1 or Tr3)) quita (combinación);
      if (Tr3 and (Ac1 or Ac2 or Ac3 or At or Tr1 or Tr2)) quita (combinación);
}
```

4 Combinaciones Consistentes

Queda, por tanto, demostrar que estas 290 posibilidades son consistentes. Sin embargo, se va a intentar reducir el número de demostraciones. Para ello, se hace uso de la posibilidad de agrupar las restricciones en las seis categorías anteriormente mencionadas: Acíclico —Ac1, Ac2 y Ac3—, Antitransitiva —At—, Salida —S1, S2 y S3—, Llegada —Ll1, Ll2 y Ll3—, Transmisión —Tr1, Tr2 y Tr3—, y Asociación —As3—. Y esto es posible porque una de las restricciones puede representar al resto. Por ejemplo, tal y como se ha demostrado en DemoI1 la restricción que niega la posibilidad de existencia de ciclos Ac3 incluye a la antirreflexiva Ac1 y a la antisimétrica Ac2. La antirreflexiva y la antisimétrica son la negación de ciclos de uno y dos pasos respectivamente. En la tabla B.13 se muestra la regla que representa al grupo.

Esto significa que, demostrando la consistencia para estos representantes estaríamos demostrando la validez de aquellas otras combinaciones en las que aparezcan algunas de las restricciones del grupo.

Ac	Ac3 ⊃ Ac1, Ac2
At	At
S	S3 ⊃ S1, S2
Ll	Ll3 ⊃ Ll1, Ll2
Tr	Tr3 ⊃ Tr1, Tr2
As	As3 ⊃ As1, As2

Tabla B.13 Representantes de las Restricciones sobre las Relaciones Conceptuales.

4.1 Demostración de los Representantes

La mayoría de las inclusiones de la tabla B.13 han sido demostradas anteriormente. Para Ac consulte la demostración de la regla de inclusión/incompatibilidad número 1 —tabla Demo1—, para S la demostración de la regla número 4 —tabla Demo4—, para Ll la demostración de la regla número 7 —tabla Demo7— y, para Tr la demostración de la regla número 10 —tabla Demo10—. Únicamente, queda por probar As, tal y como se describe a continuación.

	As3 ⊃ As1, As2
As	Si una relación cumple As3, entonces $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall\ r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ esto quiere decir que, $\forall <c_o, r_c, c_d> \in A_c, \nexists <c_o, s_c, c_d> \in A_c$, tal que r_c.nombre = s_c.nombre., definición As1 y $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall\ r_q \in R_c - \{r_c\}, <c_o, r_q, c_d> \notin A$, definición As2 esto es As3 incluye a As1 y a As2.

Tabla B.14 DemoAs: Demostración del Representante As.

4.2 Reglas Representantes

Estas seis reglas representantes —tabla B.13— permiten resumir varias combinaciones válidas y demostrar su consistencia. En la siguiente —tabla B.15—, se indica cuándo dos restricciones —o sus representantes— se pueden dar al mismo tiempo, esto es, son consistentes. Al igual que antes, para interpretarla nos situamos en una fila, suponemos que la restricción de la fila se cumple e intentamos añadir cada una de las restricciones de las columnas. Para aquellas restricciones cuyo añadido no suponga ninguna inconsistencia se marcará la casilla con una C —con || estamos indicando lo misma pero leída por columnas—.

Estas mismas consistencias se recogen en la tabla B.16, y vienen a decir que aquellas combinaciones en las que aparezcan estas reglas son válidas y consistentes. Para ello, se debe demostrar que las 290 combinaciones cumplen un subconjunto de estas aserciones. Pero, además, se debe probar que estos axiomas son capaces de generar las 290 combinaciones mediante deducción lógica —es decir, que el sistema formal es semánticamente completo—.

Para demostrar lo primero se ha elaborado un programa —en C— que comprueba que, efectivamente, todas las combinaciones supuestamente válidas verifican una o varias de estas reglas. Para probar lo segundo se ha construido un programa lógico —en Prolog— que verdaderamente es capaz de inferir todas las combinaciones de restricciones relacionales que son válidas en nuestro sistema. Ergo, si se demuestran las 16 aserciones, se está demostrando la consistencia de las 290 combinaciones válidas.

RT_a	Acíclicas			AT	Salida			Llegada			Transmisión			As
	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Ac1	▪▪▪▪	c	--	C	C			C			--	C	--	C
Ac2	| |	▪▪▪▪	--								--		--	
Ac3	x	x	▪▪▪▪								--		--	
At	| |			▪▪▪▪	--	c	--	--	c	--	--	c	--	c
S1	| |			x	▪▪▪▪	x	x	C			C			c
S2				| |	x	▪▪▪▪	x						x
S3				x	x	x	▪▪▪▪							x
Ll1	| |			x	| |		▪▪▪▪	x	x	C			c
Ll2				| |			x	▪▪▪▪	x				x
Ll3				x				x	x	▪▪▪▪				x
Tr1	x	x	x	x	| |			| |			▪▪▪▪	x	x	C
Tr2	| |			| |						x	▪▪▪▪	x	
Tr3	x	x	x	x							x	x	▪▪▪▪	
As3	| |			| |	| |	--	--	| |	--	--	| |		▪▪▪▪

Tabla B.15 Consistencias entre las Restricciones sobre las Relaciones Conceptuales.

1.	1.	Ac1 and Ac2
2.	2.	Ac and At
	3.	Ac and S
	4.	Ac and Ll
	5.	Ac and Tr2
	6.	Ac and As
3.	7.	At and S2
	8.	At and Ll2
	9.	At and Tr2
	10.	At and As
4.	11.	S and Ll
	12.	S and Tr
5.	13.	S1 and As
6.	14.	Ll and Tr
7.	15.	Ll1 and As
8.	16.	Tr and As

Tabla B.16 Axiomas de consistencia entre Restricciones sobre las Relaciones Conceptuales.

4.3 Demostración de las Combinaciones Consistentes

A continuación se demuestra —basándonos en las propias definiciones de las restricciones conceptuales— que son consistentes las combinaciones que se generen a partir de los axiomas de la tabla B16. Estas demostraciones se han agrupado en ocho bloques debido a que con frecuencia una misma restricción aparece combinada con varias de las otras.

1.	**Ac1 and Ac2** Si una relación es antirreflexiva Ac1 quiere decir que, $\forall < c_1, r_i, c_2 > \in A \Rightarrow c_1 \neq c_2$ Supongamos que la relación es antisimétrica Ac2, entonces, $\forall < c_1, r_i, c_2 > \in A \Rightarrow < c_2, r_i, c_1 > \notin A$ No nos conduce a ninguna incompatibilidad, ni inclusión, decir que no se permiten ciclos de un paso ni de dos pasos. Por otro lado, no hay ningún tipo de inconsistencia en decir que se permiten ciclos de un paso pero no de dos o viceversa.

Tabla B.17 DemoC1: Demostración de la regla de consistencia 1.

2.	**Ac and At** Si una relación es acíclica Ac quiere decir que, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow c_1 \neq c_n$ luego $\Gamma^-(c_1, r_i) \neq c_n$, $\Gamma^+(c_n, r_i) \neq c_1$ Supongamos que la relación es antitransitiva At, entonces, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow < c_1, r_i, c_n > \notin A$ luego $\Gamma^+(c_1, r_i) \neq c_n$, $\Gamma^-(c_n, r_i) \neq c_1$ En consecuencia, una relación puede ser al mismo tiempo acíclica y antitransitiva, o cíclica y antitransitiva, o acíclica y transitiva. **Ac and S** Supongamos que la relación presenta incompatibilidad de salida total S, entonces, $\forall c_i \in C, |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ac y S, o cumplir Ac y no S, o no cumplir Ac y sí S. **Ac and Ll** Supongamos que la relación presenta incompatibilidad de llegada total Ll, entonces, $\forall c_i \in C, |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^-(c_i, r)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ac y Ll, o cumplir Ac y no Ll, o no cumplir Ac y sí Ll. **Ac and Tr2** Supongamos que la relación presenta incompatibilidad de transmisión Tr2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $|\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}, |\Gamma^+(c_i, r_q)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ac y Tr2, o cumplir Ac y no Tr2, o no cumplir Ac y sí Tr2. **Ac and As** Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$ En consecuencia, una relación puede cumplir al tiempo Ac y As, o cumplir Ac y no As, o no cumplir Ac y sí As.

Tabla B.18 DemoC2: Demostración de la regla de consistencia 2-6.

3.	**At and S2** Si una relación es antitransitiva At quiere decir que, $\forall \mu = < c_1, r_i, c_2 >, < c_2, r_i, c_3 >, ..., < c_{n-1}, r_i, c_n > \in EC_m \Rightarrow < c_1, r_i, c_n > \notin A$ luego $\Gamma^+(c_1, r_i) \neq c_n$, $\Gamma^-(c_n, r_i) \neq c_1$ Supongamos que la relación presenta incompatibilidad de salida S2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^+(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^+(c_i, r_q)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo At y S2, o cumplir At y no S2, o no cumplir At y sí S2. **At and Ll2** Supongamos que la relación presenta incompatibilidad de llegada Ll2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^-(c_i, r_q)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo At y Ll2, o cumplir At y no Ll2, o no cumplir At y sí Ll2. **At and Tr2** Supongamos que la relación presenta incompatibilidad de transmisión Tr2, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r_q \in R_c - \{r_c\}$, $\lvert\Gamma^+(c_i, r_q)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo At y Tr2, o cumplir At y no Tr2, o no cumplir At y sí Tr2. **At and As** Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d > \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $< c_o, r, c_d > \notin A$ En consecuencia, una relación puede cumplir al tiempo At y As, o cumplir At y no As, o no cumplir At y sí As.

Tabla B.19 DemoC3: Demostración de la regla de consistencia 7-10.

4.	**S and Ll** Si una relación presenta incompatibilidad de salida total S quiere decir que, $\forall c_i \in C$, $\lvert\Gamma^+(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^+(c_i, r)\rvert = 0$ Supongamos que la relación presenta incompatibilidad de llegada total Ll, entonces, $\forall c_i \in C$, $\lvert\Gamma^-(c_i, r_c)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^-(c_i, r)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo S y Ll, o cumplir S y no Ll, o no cumplir S y sí Ll. **S and Tr** Supongamos que la relación presenta incompatibilidad de transmisión total Tr, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $\lvert\Gamma^-(c_i, r_i)\rvert \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $\lvert\Gamma^+(c_i, r)\rvert = 0$ En consecuencia, una relación puede cumplir al tiempo S y Tr, o cumplir S y no Tr, o no cumplir S y sí Tr.

Tabla B.20 DemoC4: Demostración de la regla de consistencia 11-12.

5.	**S1 and As** Si una relación presenta incompatibilidad de salida S1 quiere decir que, $\forall c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^+(c_i, r_c)| \leq 1$ Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ luego $\forall c_i \in C$, $|\Gamma^+(c_i, r_i)| = 1$ y , $|\Gamma^-(c_i, r_i)| = 1$ En consecuencia, una relación puede cumplir al tiempo S1 y As, o cumplir S1 y no As, o no cumplir S1 y sí As.

Tabla B.21 DemoC5: Demostración de la regla de consistencia 13.

6.	**Ll and Tr** Si una relación presenta incompatibilidad de llegada total Ll quiere decir que, $\forall c_i \in C$, $|\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^-(c_i, r)| = 0$ Supongamos que la relación presenta incompatibilidad de transmisión total Tr, entonces, $\forall c_i \in C$ y $r_c \in R_c$, $|\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ En consecuencia, una relación puede cumplir al tiempo Ll y Tr, o cumplir Ll y no Tr, o no cumplir Ll y sí Tr.

Tabla B.22 DemoC6: Demostración de la regla de consistencia 14.

7.	**Ll1 and As** Si una relación presenta incompatibilidad de salida Ll1 quiere decir que, $\forall c_i \in C$ y $r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 1$ Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ luego $\forall c_i \in C$, $|\Gamma^+(c_i, r_c)| = 1$ y , $|\Gamma^-(c_i, r_c)| = 1$ En consecuencia, una relación puede cumplir al tiempo Ll1 y As, o cumplir Ll1 y no As, o no cumplir Ll1 y sí As.

Tabla B.23 DemoC7: Demostración de la regla de consistencia 15.

8.	**Tr and As** Si una relación presenta incompatibilidad de transmisión total Tr quiere decir que, $\forall c_i \in C$ y $r_c \in R_c$, $|\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $|\Gamma^+(c_i, r)| = 0$ Supongamos que la relación presenta incompatibilidad de asociación total As, entonces, $\forall < c_o, r_c, c_d> \in A \Rightarrow \forall r \in R_c$, con $r \neq r_c$, $<c_o, r, c_d> \notin A$ luego $\forall c_i \in C$, $|\Gamma^+(c_i, r_c)| = 1$ y , $|\Gamma^-(c_i, r_c)| = 1$ En consecuencia, una relación puede cumplir al tiempo Ll1 y As, o cumplir Ll1 y no As, o no cumplir Ll1 y sí As.

Tabla B.24 DemoC8: Demostración de la regla de consistencia 16.

4.4 Programa 2: Comprueba que las Combinaciones Válidas cumplen los Axiomas

Este programa comprueba que todas las combinaciones válidas cumplen los axiomas de la tabla B.16. Para ello se recorren todas ellas y se va cotejando si verifica uno o más de los 16 axiomas posibles. Este programa ha sido desarrollado en ANSI C.

```
Programa2 Cumple-axiomas;
Genera-combinaciones-restricciones-consistentes;
for  all (combinación-consistente) {
      if ( Ac1 and Ac2) cumple (axioma1, combinación-consistente);
      if (Ac and At) cumple (axioma2, combinación-consistente);
      if (Ac and S) cumple (axioma3, combinación-consistente);
      if (Ac and Ll) cumple (axioma4, combinación-consistente);
      if (Ac and Tr2) cumple (axioma5, combinación-consistente);
      if (Ac and As) cumple (axioma6, combinación-consistente);
      if (At and S2) cumple (axioma7, combinación-consistente);
      if (At and Ll2) cumple (axioma8, combinación-consistente);
      if (At and Tr2) cumple (axioma9, combinación-consistente);
      if (At and As) cumple (axioma10, combinación-consistente);
      if (S and Ll) cumple (axioma11, combinación-consistente);
      if (S and Tr) cumple (axioma12, combinación-consistente);
      if (S1 and As) cumple (axioma13, combinación-consistente);
      if (Ll and Tr) cumple (axioma14, combinación-consistente);
      if (Ll1 and As) cumple (axioma15, combinación-consistente);
      if (Tr and As) cumple (axioma16, combinación-consistente);
}
```

4.5 Programa 3: Genera a partir de los Axiomas las Combinaciones Válidas

Este programa lógico ha sido implementado en Prolog —con el intérprete JIP, Jave Internet en Prolog—. Su ejecución genera las combinaciones válidas a partir de los axiomas de la tabla B.16, las mismas se calcularon eliminando las inconsistentes.

```
//Programa3 GeneraCombinaciones

aciclico(ac1). aciclico(ac2). aciclico(ac3). anti(at). salida(s1). salida(s2). salida(s3).
llegada(ll1). llegada(ll2). llegada(ll3). transmision(tr1). transmision(tr2). transmision(tr3). asociacion(as).

posible2(ac1, ac2).
posible2(A,B):- aciclico(A), anti(B).
posible2(A,B):- aciclico(A), salida(B).
posible2(A,B):- aciclico(A), llegada(B).
posible2(A,tr2):- aciclico(A).
posible2(A,as):- aciclico(A).
posible2(at,s2).
posible2(at,ll2).
```

```
posible2(at,tr2).
posible2(at,as).
posible2(A,B):- salida(A), llegada(B).
posible2(A,B):- salida(A), transmision(B).
posible2(s1,as).
posible2(A,B):- llegada(A), transmision(B).
posible2(ll1,as).
posible2(A,as):- transmision(A).

posible3(A, B, C):- posible2(A, B), posible2(A, C), posible2(B, C).
posible4(A,B,C,D):- posible3(A,B,C), posible3(A,B,D), posible3(A,C,D), posible3(B,C,D).
posible5(A,B,C,D,E):-
   posible4(A,B,C,D), posible4(A,B,C,E), posible4(A,B,D,E), posible4(A,C,D,E), posible4(B,C,D,E).
posible6(A,B,C,D,E,F):-
   posible5(A,B,C,D,E), posible5(A,B,C,D,F), posible5(A,B,C,E,F), posible5(A,B,D,E,F),
posible5(A,C,D,E,F), posible5(B,C,D,E,F).
```

En conclusión, las demostraciones de las tablas B.2, B13 y B16, y los programas 1, 2 y 3 prueban la inconsistencia de las combinaciones no válidas y la consistencia de las combinaciones válidas. En el apéndice C se puede consultar los códigos de los programas y los resultados obtenidos con su ejecución.

Apéndice C

Programas y Resultados

1 Introducción

En este apéndice se muestran los códigos de los programas que se han elaborado para la demostración de la consistencia de las Restricciones sobre las relaciones conceptuales descrita en el apéndice A y los resultados que se han obtenido con su ejecución.

1.1 Programa 1: Elimina Combinaciones No Válidas

```
/* Este programa «inconsistentes.c» elimina las combinaciones no válidas
por ser incluyentes o incompatibles */

#include <stdio.h>
#define casos 16384 /* número de casos posibles */
#define res 14      /* número de restricciones */

int main (){

unsigned short int Rs[res][casos]; /* matriz con todas las posibles com-
binaciones */
unsigned short int valido[casos]; /* matriz con las combinaciones validas
*/
int c,i,numc;
unsigned short int p0, p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11, p12,
p13;


c = 0;
/* se generan todas las posibles combinaciones */
for (p0=0; p0<=1; p0++){
  for (p1=0; p1<=1; p1++){
    for (p2=0; p2<=1; p2++){
       for (p3=0; p3<=1; p3++){
          for (p4=0; p4<=1; p4++){
             for (p5=0; p5<=1; p5++){
                for (p6=0; p6<=1; p6++){
                   for (p7=0; p7<=1; p7++){
                      for (p8=0; p8<=1; p8++){
                         for (p9=0; p9<=1; p9++){
                            for (p10=0; p10<=1; p10++){
                               for (p11=0; p11<=1; p11++){
                                  for (p12=0; p12<=1; p12++){
                                     for (p13=0; p13<=1; p13++){
```

```
          Rs[0][c] = p0; Rs[1][c] = p1;
          Rs[2][c] = p2; Rs[3][c] = p3;
          Rs[4][c] = p4; Rs[5][c] = p5;
          Rs[6][c] = p6; Rs[7][c] = p7;
          Rs[8][c] = p8; Rs[9][c] = p9;
          Rs[10][c] = p10; Rs[11][c] = p11;
          Rs[12][c] = p12; Rs[13][c] = p13;

                   c++;

      }}}}}}}}}}}}}}
/* se listan todas las combinaciones posibles */
      c=0;
      while (c <casos){
        for(i=0; i<res; i++){
          printf(«%i «, Rs[i][c]);
          }
        puts(«»);
        c++;}

/* se incializa a 1 la matriz con los casos a priori validos */
      for(i=0;i<casos;i++){
        valido[i]=1;}

/* se elimina aquellos casos que son incluyentes o incompatibles */
     for (i=0; i<casos; i++){

        if ((Rs[2][i])&&((Rs[0][i])||(Rs[1][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[4][i])&&((Rs[3][i])||(Rs[5][i])||(Rs[6][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[5][i])&&((Rs[4][i])||(Rs[6][i])||(Rs[13][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if
((Rs[6][i])&&((Rs[3][i])||(Rs[4][i])||(Rs[5][i])||(Rs[13][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[7][i])&&((Rs[3][i])||(Rs[8][i])||(Rs[9][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[8][i])&&((Rs[7][i])||(Rs[9][i])||(Rs[13][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if
((Rs[9][i]&&((Rs[3][i])||(Rs[7][i])||(Rs[8][i])||(Rs[13][i]))))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if
((Rs[10][i])&&((Rs[0][i])||(Rs[1][i])||(Rs[2][i])||(Rs[3][i])||(Rs[11][i])||(Rs[12][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[11][i])&&((Rs[10][i])||(Rs[12][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if
((Rs[12][i])&&((Rs[0][i])||(Rs[1][i])||(Rs[2][i])||(Rs[3][i])||(Rs[10][i])||(Rs[11][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
      }
```

```
/* se lista todos los casos que son a priori válidos */
    for (i=0; i<casos;i++){
       if (valido[i]) {
         printf(«columna %i   «, i);
         for(c=0; c<res;c++){
            printf(«%i «, Rs[c][i]);
            }
         puts(«»);
       }
     }
  printf(«combinaciones quitadas %i «, numc);
}
```

1.2 Resultados 1: Elimina Combinaciones No Válidas

Las tablas C1-C9 muestran la salida que se obtiene al ejecutar el programa 1. Como puede observarse, el número de combinaciones *a priori* válidas son un total de 290. Para cada combinación se indica con 1 si cumple la Restricción, y con 0 si el autor no ha asociado esa Restricción a la relación conceptual. Para facilitar la comparación de estas combinaciones con las salidas de los otros programas se incluye también el número decimal que corresponde a la codificación binaria de la combinación.

Por tanto, para cada entrada de la tabla se muestra su número de combinación, su identificación decimal y las Restricciones con las que se caracteriza esa combinación.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
1.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2.	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
3.	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0
4.	3	0	0	0	0	0	0	0	0	0	0	0	0	1	1
5.	4	0	0	0	0	0	0	0	0	0	0	0	1	0	0
6.	5	0	0	0	0	0	0	0	0	0	0	0	1	0	1
7.	8	0	0	0	0	0	0	0	0	0	0	1	0	0	0
8.	9	0	0	0	0	0	0	0	0	0	0	1	0	0	1
9.	16	0	0	0	0	0	0	0	0	0	1	0	0	0	0
10.	18	0	0	0	0	0	0	0	0	0	1	0	0	1	0
11.	20	0	0	0	0	0	0	0	0	0	1	0	1	0	0
12.	24	0	0	0	0	0	0	0	0	0	1	1	0	0	0
13.	32	0	0	0	0	0	0	0	0	1	0	0	0	0	0
14.	34	0	0	0	0	0	0	0	0	1	0	0	0	1	0
15.	36	0	0	0	0	0	0	0	0	1	0	0	1	0	0
16.	40	0	0	0	0	0	0	0	0	1	0	1	0	0	0
17.	64	0	0	0	0	0	0	0	1	0	0	0	0	0	0
18.	65	0	0	0	0	0	0	0	1	0	0	0	0	0	1
19.	66	0	0	0	0	0	0	0	1	0	0	0	0	1	0
20.	67	0	0	0	0	0	0	0	1	0	0	0	0	1	1
21.	68	0	0	0	0	0	0	0	1	0	0	0	1	0	0
22.	69	0	0	0	0	0	0	0	1	0	0	0	1	0	1
23.	72	0	0	0	0	0	0	0	1	0	0	1	0	0	0
24.	73	0	0	0	0	0	0	0	1	0	0	1	0	0	1
25.	128	0	0	0	0	0	0	1	0	0	0	0	0	0	0
26.	130	0	0	0	0	0	0	1	0	0	0	0	0	1	0
27.	132	0	0	0	0	0	0	1	0	0	0	0	1	0	0
28.	136	0	0	0	0	0	0	1	0	0	0	1	0	0	0
29.	144	0	0	0	0	0	0	1	0	0	1	0	0	0	0
30.	146	0	0	0	0	0	0	1	0	0	1	0	0	1	0
31.	148	0	0	0	0	0	0	1	0	0	1	0	1	0	0
32.	152	0	0	0	0	0	0	1	0	0	1	1	0	0	0
33.	160	0	0	0	0	0	0	1	0	1	0	0	0	0	0
34.	162	0	0	0	0	0	0	1	0	1	0	0	0	1	0
35.	164	0	0	0	0	0	0	1	0	0	0	0	1	0	0

Tabla C.1 Salida del programa 1: Combinaciones 1-35.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
36.	168	0	0	0	0	0	0	1	0	0	0	1	0	0	0
37.	192	0	0	0	0	0	0	1	1	0	0	0	0	0	0
38.	194	0	0	0	0	0	0	1	1	0	0	0	0	1	0
39.	196	0	0	0	0	0	0	1	1	0	0	0	1	0	0
40.	200	0	0	0	0	0	0	1	1	0	0	1	0	0	0
41.	256	0	0	0	0	0	1	0	0	0	0	0	0	0	0
42.	258	0	0	0	0	0	1	0	0	0	0	0	0	1	0
43.	260	0	0	0	0	0	1	0	0	0	0	0	1	0	0
44.	264	0	0	0	0	0	1	0	0	0	0	1	0	0	0
45.	272	0	0	0	0	0	1	0	0	0	1	0	0	0	0
46.	274	0	0	0	0	0	1	0	0	0	1	0	0	1	0
47.	276	0	0	0	0	0	1	0	0	0	1	0	1	0	0
48.	280	0	0	0	0	0	1	0	0	0	1	1	0	0	0
49.	288	0	0	0	0	0	1	0	0	1	0	0	0	0	0
50.	290	0	0	0	0	0	1	0	0	1	0	0	0	1	0
51.	292	0	0	0	0	0	1	0	0	1	0	0	1	0	0
52.	296	0	0	0	0	0	1	0	0	1	0	1	0	0	0
53.	320	0	0	0	0	0	1	0	1	0	0	0	0	0	0
54.	322	0	0	0	0	0	1	0	1	0	0	0	0	1	0
55.	324	0	0	0	0	0	1	0	1	0	0	0	1	0	0
56.	328	0	0	0	0	0	1	0	1	0	0	1	0	0	0
57.	512	0	0	0	0	1	0	0	0	0	0	0	0	0	0
58.	513	0	0	0	0	1	0	0	0	0	0	0	0	0	1
59.	514	0	0	0	0	1	0	0	0	0	0	0	0	1	0
60.	515	0	0	0	0	1	0	0	0	0	0	0	0	1	1
61.	516	0	0	0	0	1	0	0	0	0	0	0	1	0	0
62.	517	0	0	0	0	1	0	0	0	0	0	0	1	0	1
63.	520	0	0	0	0	1	0	0	0	0	0	1	0	0	0
64.	521	0	0	0	0	1	0	0	0	0	0	1	0	0	1
65.	528	0	0	0	0	1	0	0	0	0	1	0	0	0	0
66.	530	0	0	0	0	1	0	0	0	0	1	0	0	1	0
67.	532	0	0	0	0	1	0	0	0	0	1	0	1	0	0
68.	536	0	0	0	0	1	0	0	0	0	1	1	0	0	0
69.	544	0	0	0	0	1	0	0	0	1	0	0	0	0	0
70.	546	0	0	0	0	1	0	0	0	1	0	0	0	1	0

Tabla C.2 Salida del programa 1: Combinaciones 36-70.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
71.	548	0	0	0	0	1	0	0	0	1	0	0	1	0	0
72.	552	0	0	0	0	1	0	0	0	1	0	1	0	0	0
73.	576	0	0	0	0	1	0	0	1	0	0	0	0	0	0
74.	577	0	0	0	0	1	0	0	1	0	0	0	0	0	1
75.	578	0	0	0	0	1	0	0	1	0	0	0	0	1	0
76.	579	0	0	0	0	1	0	0	1	0	0	0	0	1	1
77.	580	0	0	0	0	1	0	0	1	0	0	0	1	0	0
78.	581	0	0	0	0	1	0	0	1	0	0	0	1	0	1
79.	584	0	0	0	0	1	0	0	1	0	0	1	0	0	0
80.	585	0	0	0	0	1	0	0	1	0	0	1	0	0	1
81.	1024	0	0	0	1	0	0	0	0	0	0	0	0	0	0
82.	1025	0	0	0	1	0	0	0	0	0	0	0	0	0	1
83.	1028	0	0	0	1	0	0	0	0	0	0	0	1	0	0
84.	1029	0	0	0	1	0	0	0	0	0	0	0	1	0	1
85.	1056	0	0	0	1	0	0	0	0	1	0	0	0	0	0
86.	1060	0	0	0	1	0	0	0	0	1	0	0	1	0	0
87.	1280	0	0	0	1	0	1	0	0	0	0	0	0	0	0
88.	1284	0	0	0	1	0	1	0	0	0	0	0	1	0	0
89.	1312	0	0	0	1	0	1	0	0	1	0	0	0	0	0
90.	1316	0	0	0	1	0	1	0	0	1	0	0	1	0	0
91.	2048	0	0	1	0	0	0	0	0	0	0	0	0	0	0
92.	2049	0	0	1	0	0	0	0	0	0	0	0	0	0	1
93.	2052	0	0	1	0	0	0	0	0	0	0	0	1	0	0
94.	2053	0	0	1	0	0	0	0	0	0	0	0	1	0	1
95.	2064	0	0	1	0	0	0	0	0	0	1	0	0	0	0
96.	2068	0	0	1	0	0	0	0	0	0	1	0	1	0	0
97.	2080	0	0	1	0	0	0	0	0	1	0	0	0	0	0
98.	2084	0	0	1	0	0	0	0	0	1	0	0	1	0	0
99.	2112	0	0	1	0	0	0	0	1	0	0	0	0	0	0
100.	2113	0	0	1	0	0	0	0	1	0	0	0	0	0	1
101.	2116	0	0	1	0	0	0	0	1	0	0	0	1	0	0
102.	2117	0	0	1	0	0	0	0	0	0	0	0	1	0	1
103.	2176	0	0	1	0	0	0	1	0	0	0	0	0	0	0
104.	2180	0	0	1	0	0	0	1	0	0	0	0	1	0	0
105.	2192	0	0	1	0	0	0	1	0	0	1	0	0	0	0

Tabla C.3 Salida del programa 1: Combinaciones 71-105.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
106.	2196	0	0	1	0	0	0	1	0	0	1	0	1	0	0
107.	2208	0	0	1	0	0	0	1	0	1	0	0	0	0	0
108.	2212	0	0	1	0	0	0	1	0	1	0	0	1	0	0
109.	2240	0	0	1	0	0	0	1	1	0	0	0	0	0	0
110.	2244	0	0	1	0	0	0	1	1	0	0	0	1	0	0
111.	2304	0	0	1	0	0	1	0	0	0	0	0	0	0	0
112.	2308	0	0	1	0	0	1	0	0	0	0	0	1	0	0
113.	2320	0	0	1	0	0	1	0	0	0	1	0	0	0	0
114.	2324	0	0	1	0	0	1	0	0	0	1	0	1	0	0
115.	2336	0	0	1	0	0	1	0	0	1	0	0	0	0	0
116.	2340	0	0	1	0	0	1	0	0	1	0	0	1	0	0
117.	2368	0	0	1	0	0	1	0	1	0	0	0	0	0	0
118.	2372	0	0	1	0	0	1	0	1	0	0	0	1	0	0
119.	2560	0	0	1	0	1	0	0	0	0	0	0	0	0	0
120.	2561	0	0	1	0	1	0	0	0	0	0	0	0	0	1
121.	2564	0	0	1	0	1	0	0	0	0	0	0	1	0	0
122.	2565	0	0	1	0	1	0	0	0	0	0	0	1	0	1
123.	2576	0	0	1	0	1	0	0	0	0	1	0	0	0	0
124.	2580	0	0	1	0	1	0	0	0	0	1	0	1	0	0
125.	2592	0	0	1	0	1	0	0	0	1	0	0	0	0	0
126.	2596	0	0	1	0	1	0	0	0	1	0	0	1	0	0
127.	2624	0	0	1	0	1	0	0	1	0	0	0	0	0	0
128.	2625	0	0	1	0	1	0	0	1	0	0	0	0	0	1
129.	2628	0	0	1	0	1	0	0	1	0	0	0	1	0	0
130.	2629	0	0	1	0	1	0	0	1	0	0	0	1	0	1
131.	3072	0	0	1	1	0	0	0	0	0	0	0	0	0	0
132.	3073	0	0	1	1	0	0	0	0	0	0	0	0	0	1
133.	3076	0	0	1	1	0	0	0	0	0	0	0	1	0	0
134.	3077	0	0	1	1	0	0	0	0	0	0	0	1	0	1
135.	3104	0	0	1	1	0	0	0	0	1	0	0	0	0	0
136.	3108	0	0	1	1	0	0	0	0	1	0	0	1	0	0
137.	3328	0	0	1	1	0	1	0	0	0	0	0	0	0	0
138.	3332	0	0	1	1	0	1	0	0	0	0	0	1	0	0
139.	3360	0	0	1	1	0	1	0	0	1	0	0	0	0	0
140.	3364	0	0	1	1	0	1	0	0	1	0	0	1	0	0

Tabla C.4 Salida del programa 1: Combinaciones 106-140.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
141.	4096	0	1	0	0	0	0	0	0	0	0	0	0	0	0
142.	4097	0	1	0	0	0	0	0	0	0	0	0	0	0	1
143.	4100	0	1	0	0	0	0	0	0	0	0	0	1	0	0
144.	4101	0	1	0	0	0	0	0	0	0	0	0	1	0	1
145.	4112	0	1	0	0	0	0	0	0	0	1	0	0	0	0
146.	4116	0	1	0	0	0	0	0	0	0	1	0	1	0	0
147.	4128	0	1	0	0	0	0	0	0	1	0	0	0	0	0
148.	4132	0	1	0	0	0	0	0	0	1	0	0	1	0	0
149.	4160	0	1	0	0	0	0	0	1	0	0	0	0	0	0
150.	4161	0	1	0	0	0	0	0	1	0	0	0	0	0	1
151.	4164	0	1	0	0	0	0	0	1	0	0	0	1	0	0
152.	4165	0	1	0	0	0	0	0	1	0	0	0	1	0	1
153.	4224	0	1	0	0	0	0	1	0	0	0	0	0	0	0
154.	4228	0	1	0	0	0	0	1	0	0	0	0	1	0	0
155.	4240	0	1	0	0	0	0	1	0	0	1	0	0	0	0
156.	4244	0	1	0	0	0	0	1	0	0	1	0	1	0	0
157.	4256	0	1	0	0	0	0	1	0	1	0	0	0	0	0
158.	4260	0	1	0	0	0	0	1	0	1	0	0	1	0	0
159.	4288	0	1	0	0	0	0	1	1	0	0	0	0	0	0
160.	4292	0	1	0	0	0	0	1	1	0	0	0	1	0	0
161.	4352	0	1	0	0	0	1	0	0	0	0	0	0	0	0
162.	4356	0	1	0	0	0	1	0	0	0	0	0	1	0	0
163.	4368	0	1	0	0	0	1	0	0	0	1	0	0	0	0
164.	4372	0	1	0	0	0	1	0	0	0	1	0	1	0	0
165.	4384	0	1	0	0	0	1	0	0	1	0	0	0	0	0
166.	4388	0	1	0	0	0	1	0	0	1	0	0	1	0	0
167.	4416	0	1	0	0	0	1	0	1	0	0	0	0	0	0
168.	4420	0	1	0	0	0	1	0	1	0	0	0	1	0	0
169.	4608	0	1	0	0	1	0	0	0	0	0	0	0	0	0
170.	4609	0	1	0	0	1	0	0	0	0	0	0	0	0	1
171.	4612	0	1	0	0	1	0	0	0	0	0	0	1	0	0
172.	4613	0	1	0	0	1	0	0	0	0	0	0	1	0	1
173.	4624	0	1	0	0	1	0	0	0	0	1	0	0	0	0
174.	4628	0	1	0	0	1	0	0	0	0	1	0	1	0	0
175.	4640	0	1	0	0	1	0	0	0	1	0	0	0	0	0

Tabla C.5 Salida del programa 1: Combinaciones 141-175.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
176.	4644	0	1	0	0	1	0	0	0	1	0	0	1	0	0
177.	4672	0	1	0	0	1	0	0	1	0	0	0	0	0	0
178.	4673	0	1	0	0	1	0	0	1	0	0	0	0	0	1
179.	4676	0	1	0	0	1	0	0	1	0	0	0	1	0	0
180.	4677	0	1	0	0	1	0	0	1	0	0	0	1	0	1
181.	5120	0	1	0	1	0	0	0	0	0	0	0	0	0	0
182.	5121	0	1	0	1	0	0	0	0	0	0	0	0	0	1
183.	5124	0	1	0	1	0	0	0	0	0	0	0	1	0	0
184.	5125	0	1	0	1	0	0	0	0	0	0	0	1	0	1
185.	5152	0	1	0	1	0	0	0	0	1	0	0	0	0	0
186.	5156	0	1	0	1	0	0	0	0	1	0	0	1	0	0
187.	5376	0	1	0	1	0	1	0	0	0	0	0	0	0	0
188.	5380	0	1	0	1	0	1	0	0	0	0	0	1	0	0
189.	5408	0	1	0	1	0	1	0	0	1	0	0	0	0	0
190.	5412	0	1	0	1	0	1	0	0	1	0	0	1	0	0
191.	8192	1	0	0	0	0	0	0	0	0	0	0	0	0	0
192.	8193	1	0	0	0	0	0	0	0	0	0	0	0	0	1
193.	8196	1	0	0	0	0	0	0	0	0	0	0	1	0	0
194.	8197	1	0	0	0	0	0	0	0	0	0	0	1	0	1
195.	8208	1	0	0	0	0	0	0	0	0	1	0	0	0	0
196.	8212	1	0	0	0	0	0	0	0	0	1	0	1	0	0
197.	8224	1	0	0	0	0	0	0	0	1	0	0	0	0	0
198.	8228	1	0	0	0	0	0	0	0	1	0	0	1	0	0
199.	8256	1	0	0	0	0	0	0	1	0	0	0	0	0	0
200.	8257	1	0	0	0	0	0	0	1	0	0	0	0	0	1
201.	8260	1	0	0	0	0	0	0	1	0	0	0	1	0	0
202.	8261	1	0	0	0	0	0	0	1	0	0	0	1	0	1
203.	8320	1	0	0	0	0	0	1	0	0	0	0	0	0	0
204.	8324	1	0	0	0	0	0	1	0	0	0	0	1	0	0
205.	8336	1	0	0	0	0	0	1	0	0	1	0	0	0	0
206.	8340	1	0	0	0	0	0	1	0	0	1	0	1	0	0
207.	8352	1	0	0	0	0	0	1	0	1	0	0	0	0	0
208.	8356	1	0	0	0	0	0	1	0	1	0	0	1	0	0
209.	8384	1	0	0	0	0	0	1	1	0	0	0	0	0	0
210.	8388	1	0	0	0	0	0	1	1	0	0	0	1	0	0

Tabla C.6 Salida del programa 1: Combinaciones 176-210.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
211.	8448	1	0	0	0	0	1	0	0	0	0	0	0	0	0
212.	8452	1	0	0	0	0	1	0	0	0	0	0	1	0	0
213.	8464	1	0	0	0	0	1	0	0	0	1	0	0	0	0
214.	8468	1	0	0	0	0	1	0	0	0	1	0	1	0	0
215.	8480	1	0	0	0	0	1	0	0	1	0	0	0	0	0
216.	8484	1	0	0	0	0	1	0	0	1	0	0	1	0	0
217.	8512	1	0	0	0	0	1	0	1	0	0	0	0	0	0
218.	8516	1	0	0	0	0	1	0	1	0	0	0	1	0	0
219.	8704	1	0	0	0	1	0	0	0	0	0	0	0	0	0
220.	8705	1	0	0	0	1	0	0	0	0	0	0	0	0	1
221.	8708	1	0	0	0	1	0	0	0	0	0	0	1	0	0
222.	8709	1	0	0	0	1	0	0	0	0	0	0	1	0	1
223.	8720	1	0	0	0	1	0	0	0	0	1	0	0	0	0
224.	8724	1	0	0	0	1	0	0	0	0	1	0	1	0	0
225.	8736	1	0	0	0	1	0	0	0	1	0	0	0	0	0
226.	8740	1	0	0	0	1	0	0	0	1	0	0	1	0	0
227.	8768	1	0	0	0	1	0	0	1	0	0	0	0	0	0
228.	8769	1	0	0	0	1	0	0	1	0	0	0	0	0	1
229.	8772	1	0	0	0	1	0	0	1	0	0	0	1	0	0
230.	8773	1	0	0	0	1	0	0	1	0	0	0	1	0	1
231.	9216	1	0	0	1	0	0	0	0	0	0	0	0	0	0
232.	9217	1	0	0	1	0	0	0	0	0	0	0	0	0	1
233.	9220	1	0	0	1	0	0	0	0	0	0	0	1	0	0
234.	9221	1	0	0	1	0	0	0	0	0	0	0	1	0	1
235.	9248	1	0	0	1	0	0	0	0	1	0	0	0	0	0
236.	9252	1	0	0	1	0	0	0	0	1	0	0	1	0	0
237.	9472	1	0	0	1	0	1	0	0	0	0	0	0	0	0
238.	9476	1	0	0	1	0	1	0	0	0	0	0	1	0	0
239.	9504	1	0	0	1	0	1	0	0	1	0	0	0	0	0
240.	9508	1	0	0	1	0	1	0	0	1	0	0	1	0	0
241.	12288	1	1	0	0	0	0	0	0	0	0	0	0	0	0
242.	12289	1	1	0	0	0	0	0	0	0	0	0	0	0	1
243.	12292	1	1	0	0	0	0	0	0	0	0	0	1	0	0
244.	12293	1	1	0	0	0	0	0	0	0	0	0	1	0	1
245.	12304	1	1	0	0	0	0	0	0	0	1	0	0	0	0

Tabla C.7 Salida del programa 1: Combinaciones 211-245.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
246.	12308	1	1	0	0	0	0	0	0	0	1	0	1	0	0
247.	12320	1	1	0	0	0	0	0	0	1	0	0	0	0	0
248.	12324	1	1	0	0	0	0	0	0	1	0	0	1	0	0
249.	12352	1	1	0	0	0	0	0	1	0	0	0	0	0	0
250.	12353	1	1	0	0	0	0	0	1	0	0	0	0	0	1
251.	12356	1	1	0	0	0	0	0	1	0	0	0	1	0	0
252.	12357	1	1	0	0	0	0	0	1	0	0	0	1	0	1
253.	12416	1	1	0	0	0	0	1	0	0	0	0	0	0	0
254.	12420	1	1	0	0	0	0	1	0	0	0	0	1	0	0
255.	12432	1	1	0	0	0	0	1	0	0	1	0	0	0	0
256.	12436	1	1	0	0	0	0	1	0	0	1	0	1	0	0
257.	12448	1	1	0	0	0	0	1	0	1	0	0	0	0	0
258.	12452	1	1	0	0	0	0	1	0	1	0	0	1	0	0
259.	12480	1	1	0	0	0	0	1	1	0	0	0	0	0	0
260.	12484	1	1	0	0	0	0	1	1	0	0	0	1	0	0
261.	12544	1	1	0	0	1	0	0	0	0	0	0	0	0	0
262.	12548	1	1	0	0	0	1	0	0	0	0	0	1	0	0
263.	12560	1	1	0	0	0	1	0	0	0	1	0	0	0	0
264.	12564	1	1	0	0	0	1	0	0	0	1	0	1	0	0
265.	12576	1	1	0	0	0	1	0	0	1	0	0	0	0	0
266.	12580	1	1	0	0	0	1	0	0	1	0	0	1	0	0
267.	12608	1	1	0	0	0	1	0	1	0	0	0	0	0	0
268.	12612	1	1	0	0	0	1	0	1	0	0	0	1	0	0
269.	12800	1	1	0	0	1	0	0	0	0	0	0	0	0	0
270.	12801	1	1	0	0	1	0	0	0	0	0	0	0	0	1
271.	12804	1	1	0	0	1	0	0	0	0	0	0	1	0	0
272.	12805	1	1	0	0	1	0	0	0	0	0	0	1	0	1
273.	12816	1	1	0	0	1	0	0	0	0	1	0	0	0	0
274.	12820	1	1	0	0	1	0	0	0	0	1	0	1	0	0
275.	12832	1	1	0	0	1	0	0	0	1	0	0	0	0	0
276.	12836	1	1	0	0	1	0	0	0	1	0	0	1	0	0
277.	12864	1	1	0	0	1	0	0	1	0	0	0	0	0	0
278.	12865	1	1	0	0	1	0	0	1	0	0	0	0	0	1
279.	12868	1	1	0	0	1	0	0	1	0	0	0	1	0	0
280.	12869	1	1	0	0	1	0	0	1	0	0	0	1	0	1

Tabla C.8 Salida del programa 1: Combinaciones 246-280.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
281.	13312	1	1	0	1	0	0	0	0	0	0	0	0	0	0
282.	13313	1	1	0	1	0	0	0	0	0	0	0	0	0	1
283.	13316	1	1	0	1	0	0	0	0	0	0	0	1	0	0
284.	13317	1	1	0	1	0	0	0	0	0	0	0	1	0	1
285.	13344	1	1	0	1	0	0	0	0	1	0	0	0	0	0
286.	13348	1	1	0	1	0	0	0	0	1	0	0	1	0	0
287.	13568	1	1	0	1	0	1	0	0	0	0	0	0	0	0
288.	13572	1	1	0	1	0	1	0	0	0	0	0	1	0	0
289.	13600	1	1	0	1	0	1	0	0	1	0	0	0	0	0
290.	13604	1	1	0	1	0	1	0	0	1	0	0	1	0	0

Tabla C.9 Salida del programa 1: Combinaciones 281-290.

1.3 Programa 2: Comprueba que las Combinaciones Válidas cumplen los Axiomas

```
/* Este programa valcumple.c comprueba que las combinaciones no elimina-
das con «redun.c» cumplen los axiomas de consistencia y, por tanto, son
consistentes y válidas */

#include <stdio.h>
#define casos 16384 /* número de casos posibles */
#define res 14      /* número de restricciones */
#define comvalida 290 /* número de combinaciones válidas */
#define revalida 16  /* número de axiomas de consistencia */

int main (){

unsigned short int Rs[res][casos]; /* matriz con los casos posibles */
unsigned short int valido[casos];  /* matriz con los casos válidos */
unsigned short int cumple[revalida][casos]; /* matriz que indica si cum-
ple el axioma*/
int f,c,i,numc;
unsigned short int p0, p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11, p12,
p13;
```

```
/* se genera las combinaciones posibles */
c = 0;
for (p0=0; p0<=1; p0++){
  for (p1=0; p1<=1; p1++){
    for (p2=0; p2<=1; p2++){
       for (p3=0; p3<=1; p3++){
          for (p4=0; p4<=1; p4++){
             for (p5=0; p5<=1; p5++){
                for (p6=0; p6<=1; p6++){
                   for (p7=0; p7<=1; p7++){
                      for (p8=0; p8<=1; p8++){
                         for (p9=0; p9<=1; p9++){
                            for (p10=0; p10<=1; p10++){
                               for (p11=0; p11<=1; p11++){
                                  for (p12=0; p12<=1; p12++){
                                     for (p13=0; p13<=1; p13++){

         Rs[0][c] = p0; Rs[1][c] = p1;
         Rs[2][c] = p2; Rs[3][c] = p3;
         Rs[4][c] = p4; Rs[5][c] = p5;
         Rs[6][c] = p6; Rs[7][c] = p7;
         Rs[8][c] = p8; Rs[9][c] = p9;
         Rs[10][c] = p10; Rs[11][c] = p11;
         Rs[12][c] = p12; Rs[13][c] = p13;

                  c++;

      }}}}}}}}}}}}}}

/* se lista las combinaciones posibles */

      c=0;
      while (c <casos){
        for(i=0; i<res; i++){
          printf(«%i «, Rs[i][c]);
          }
        puts(«»);
        c++;}

/* se inicializa a 1 la matriz con los casos no-redundantes, en princi-
pio, válidos */
     for(i=0;i<casos;i++){
        valido[i]=1;}
```

```
/* se elimina aquellos casos no válidos */
     for (i=0; i<casos; i++){

        if ((Rs[2][i])&&((Rs[0][i])||(Rs[1][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[4][i])&&((Rs[3][i])||(Rs[5][i])||(Rs[6][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[5][i])&&((Rs[4][i])||(Rs[6][i])||(Rs[13][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if
((Rs[6][i])&&((Rs[3][i])||(Rs[4][i])||(Rs[5][i])||(Rs[13][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[7][i])&&((Rs[3][i])||(Rs[8][i])||(Rs[9][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[8][i])&&((Rs[7][i])||(Rs[9][i])||(Rs[13][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if
((Rs[9][i]&&((Rs[3][i])||(Rs[7][i])||(Rs[8][i])||(Rs[13][i]))))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if
((Rs[10][i])&&((Rs[0][i])||(Rs[1][i])||(Rs[2][i])||(Rs[3][i])||(Rs[11][i])||(Rs[12][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if ((Rs[11][i])&&((Rs[10][i])||(Rs[12][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}
        if
((Rs[12][i])&&((Rs[0][i])||(Rs[1][i])||(Rs[2][i])||(Rs[3][i])||(Rs[10][i])||(Rs[11][i])))
         {if (valido[i]){valido[i]= 0;numc++;}}

     }
/* se lista todos los casos que son a priori válidos */
    for (i=0; i<casos;i++){
       if (valido[i]) {
         printf(«columna %i   «, i);
         for(c=0; c<res;c++){
            printf(«%i «, Rs[c][i]);
            }
         puts(«»);
       }
     }
  printf(«combinaciones quitadas %i «, numc);
  puts(«»);

/* se inicializa a 0 la matriz que indica si la combinación cumple algún
axioma de consistencia */

  for(f=0;f<casos;f++){
     for(c=0;c<revalida;c++){
        cumple[c][f]=0;}}
```

```
/* se determina si la combinación cumple algún axioma de consistencia */
  for (i=0; i<casos;i++){
       if (valido[i]) {
          if ((Rs[0][i]) && (Rs[1][i])){
             cumple[0][i]=1;}
          if (((Rs[0][i])||(Rs[1][i])||(Rs[2][i])) && (Rs[3][i])){
             cumple[1][i]=1;}
          if (((Rs[0][i])||(Rs[1][i])||(Rs[2][i])) &&
((Rs[4][i])||(Rs[5][i])||(Rs[6][i]))){
             cumple[2][i]=1;}
          if (((Rs[0][i])||(Rs[1][i])||(Rs[2][i])) &&
((Rs[7][i])||(Rs[8][i])||(Rs[9][i]))){
             cumple[3][i]=1;}
          if (((Rs[0][i])||(Rs[1][i])||(Rs[2][i])) && (Rs[11][i])){
             cumple[4][i]=1;}
          if (((Rs[0][i])||(Rs[1][i])||(Rs[2][i])) && (Rs[13][i])){
             cumple[5][i]=1;}
          if ((Rs[3][i]) && (Rs[5][i])){
             cumple[6][i]=1;}
          if ((Rs[3][i]) && (Rs[8][i])){
             cumple[7][i]=1;}
          if ((Rs[3][i]) && (Rs[11][i])){
             cumple[8][i]=1;}
          if ((Rs[3][i]) && (Rs[13][i])){
             cumple[9][i]=1;}
          if (((Rs[4][i])||(Rs[5][i])||(Rs[6][i])) &&
((Rs[7][i])||(Rs[8][i])||(Rs[9][i]))){
             cumple[10][i]=1;}
          if (((Rs[4][i])||(Rs[5][i])||(Rs[6][i])) ||
((Rs[10][i])||(Rs[11][i])||(Rs[12][i]))){
             cumple[11][i]=1;}
          if ((Rs[4][i]) && (Rs[13][i])){
             cumple[12][i]=1;}
          if (((Rs[7][i])||(Rs[8][i])||(Rs[9][i])) &&
((Rs[10][i])||(Rs[11][i])||(Rs[12][i]))){
             cumple[13][i]=1;}
          if ((Rs[7][i]) && (Rs[13][i])){
             cumple[14][i]=1;}
          if (((Rs[10][i])||(Rs[11][i])||(Rs[12][i])) && (Rs[13][i])){
             cumple[15][i]=1;}

/* se lista la matriz con las combinaciones válidas, indicando si cumple
algún axioma de consistencia*/

          printf(«cumple %i   «, i);
          for(c=0; c<revalida;c++){
             printf(«%i «, cumple[c][i]);
             }
          puts(«»);
       }
     }
  }
```

1.4 Resultados 2: Comprueba que las Combinaciones Válidas cumplen los Axiomas

Las tablas C10-C17 muestran la salida que se obtiene al ejecutar el programa 2. En este caso, las columnas de la tabla hacen referencia al número de combinación y a los axiomas. Para cada una de las combinaciones anteriormente calculada se indica el subconjunto de axiomas de consistencia que cumple. El 1 se interpreta como verdadero y el 0 como falso. Aquellas combinaciones que tienen toda la fila a 0 son aquellas que tan sólo se les ha asociado una única Restricción.

Comb	Ax1	Ax2	Ax3	Ax4	Ax5	Ax6	Ax7	Ax8	Ax9	Ax10	Ax11	Ax12	Ax13	Ax14	Ax15	Ax16
1.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
4.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
5.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
6.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
7.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
8.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
9.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
11.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
12.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
13.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
15.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
16.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
17.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
19.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
20.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1
21.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
22.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1
23.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
24.	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1
25.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
26.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
27.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
28.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
29.	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
30.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
31.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
32.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
33.	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
34.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
35.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
36.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
37.	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
38.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
39.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
40.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0

Tabla C.10 Salida del programa 2: Combinaciones 1-40.

Comb	Ax1	Ax2	Ax3	Ax4	Ax5	Ax6	Ax7	Ax8	Ax9	Ax10	Ax11	Ax12	Ax13	Ax14	Ax15	Ax16
41.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
42.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
43.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
44.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
45.	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
46.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
47.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
48.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
49.	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
50.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
51.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
52.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
53.	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
54.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
55.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
56.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
57.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
58.	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
59.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
60.	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1
61.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
62.	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1
63.	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
64.	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1
65.	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
66.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
67.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
68.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
69.	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
70.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
71.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
72.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
73.	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
74.	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0
75.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
76.	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
77.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
78.	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
79.	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
80.	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Tabla C.11 Salida del programa 2: Combinaciones 41-80.

Comb	Ax1	Ax2	Ax3	Ax4	Ax5	Ax6	Ax7	Ax8	Ax9	Ax10	Ax11	Ax12	Ax13	Ax14	Ax15	Ax16
81.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
82.	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
83.	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
84.	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	1
85.	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
86.	0	0	0	0	0	0	0	1	1	0	0	1	0	1	0	0
87.	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0
88.	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0
89.	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	0
90.	0	0	0	0	0	0	1	1	1	0	1	1	0	1	0	0
91.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
92.	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
93.	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
94.	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	1
95.	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
96.	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
97.	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
98.	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
99.	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
100.	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0
101.	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
102.	0	0	0	1	1	1	0	0	0	0	0	1	0	1	1	1
103.	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
104.	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
105.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
106.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
107.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
108.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
109.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
110.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
111.	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
112.	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
113.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
114.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
115.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
116.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
117.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
118.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
119.	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
120.	0	0	1	0	0	1	0	0	0	0	0	1	1	0	0	0

Tabla C.12 Salida del programa 2: Combinaciones 81-120.

Comb	Ax1	Ax2	Ax3	Ax4	Ax5	Ax6	Ax7	Ax8	Ax9	Ax10	Ax11	Ax12	Ax13	Ax14	Ax15	Ax16
121.	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
122.	0	0	1	0	1	1	0	0	0	0	0	1	1	0	0	1
123.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
124.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
125.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
126.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
127.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
128.	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	0
129.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
130.	0	0	1	1	1	1	0	0	0	0	1	1	1	1	1	1
131.	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
132.	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0
133.	0	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0
134.	0	1	0	0	1	1	0	0	1	1	0	1	0	0	0	1
135.	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0
136.	0	1	0	1	1	0	0	1	1	0	0	1	0	1	0	0
137.	0	1	1	0	0	0	1	0	0	0	0	1	0	0	0	0
138.	0	1	1	0	1	0	1	0	1	0	0	1	0	0	0	0
139.	0	1	1	1	0	0	1	1	0	0	1	1	0	0	0	0
140.	0	1	1	1	1	0	1	1	1	0	1	1	0	1	0	0
141.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
142.	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
143.	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
144.	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	1
145.	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
146.	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
147.	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
148.	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
149.	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
150.	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0
151.	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
152.	0	0	0	1	1	1	0	0	0	0	0	1	0	1	1	1
153.	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
154.	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
155.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
156.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
157.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
158.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
159.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
160.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0

Tabla C.13 Salida del programa 2: Combinaciones 121-160.

Comb	Ax1	Ax2	Ax3	Ax4	Ax5	Ax6	Ax7	Ax8	Ax9	Ax10	Ax11	Ax12	Ax13	Ax14	Ax15	Ax16
161.	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
162.	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
163.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
164.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
165.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
166.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
167.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
168.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
169.	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
170.	0	0	1	0	0	1	0	0	0	0	0	1	1	0	0	0
171.	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
172.	0	0	1	0	1	1	0	0	0	0	0	1	1	0	0	1
173.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
174.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
175.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
176.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
177.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
178.	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	0
179.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
180.	0	0	1	1	1	1	0	0	0	0	1	1	1	1	1	1
181.	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
182.	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0
183.	0	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0
184.	0	1	0	0	1	1	0	0	1	1	0	1	0	0	0	1
185.	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0
186.	0	1	0	1	1	0	0	1	1	0	0	1	0	1	0	0
187.	0	1	1	0	0	0	1	0	0	0	0	1	0	0	0	0
188.	0	1	1	0	1	0	1	0	1	0	0	1	0	0	0	0
189.	0	1	1	1	0	0	1	1	0	0	1	1	0	0	0	0
190.	0	1	1	1	1	0	1	1	1	0	1	1	0	1	0	0
191.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
192.	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
193.	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
194.	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	1
195.	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
196.	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
197.	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
198.	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
199.	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
200.	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0

Tabla C.14 Salida del programa 2: Combinaciones 161-200.

Comb	Ax1	Ax2	Ax3	Ax4	Ax5	Ax6	Ax7	Ax8	Ax9	Ax10	Ax11	Ax12	Ax13	Ax14	Ax15	Ax16
201.	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
202.	0	0	0	1	1	1	0	0	0	0	0	1	0	1	1	1
203.	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
204.	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
205.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
206.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
207.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
208.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
209.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
210.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
211.	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
212.	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
213.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
214.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
215.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
216.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
217.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
218.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
219.	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
220.	0	0	1	0	0	1	0	0	0	0	0	1	1	0	0	0
221.	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
222.	0	0	1	0	1	1	0	0	0	0	0	1	1	0	0	1
223.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
224.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
225.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
226.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
227.	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
228.	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	0
229.	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
230.	0	0	1	1	1	1	0	0	0	0	1	1	1	1	1	1
231.	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
232.	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0
233.	0	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0
234.	0	1	0	0	1	1	0	0	1	1	0	1	0	0	0	1
235.	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0
236.	0	1	0	1	1	0	0	1	1	0	0	1	0	1	0	0
237.	0	1	1	0	0	0	1	0	0	0	0	1	0	0	0	0
238.	0	1	1	0	1	0	1	0	1	0	0	1	0	0	0	0
239.	0	1	1	1	0	0	1	1	0	0	1	1	0	0	0	0
240.	0	1	1	1	1	0	1	1	1	0	1	1	0	1	0	0

Tabla C.15 Salida del programa 2: Combinaciones 201-240.

Comb	Ax1	Ax2	Ax3	Ax4	Ax5	Ax6	Ax7	Ax8	Ax9	Ax10	Ax11	Ax12	Ax13	Ax14	Ax15	Ax16
241.	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
242.	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
243.	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
244.	1	0	0	0	1	1	0	0	0	0	0	1	0	0	0	1
245.	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
246.	1	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
247.	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
248.	1	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
249.	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
250.	1	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0
251.	1	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
252.	1	0	0	1	1	1	0	0	0	0	0	1	0	1	1	1
253.	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
254.	1	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
255.	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
256.	1	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
257.	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
258.	1	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
259.	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
260.	1	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
261.	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
262.	1	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
263.	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
264.	1	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
265.	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
266.	1	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
267.	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
268.	1	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
269.	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
270.	1	0	1	0	0	1	0	0	0	0	0	1	1	0	0	0
271.	1	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0
272.	1	0	1	0	1	1	0	0	0	0	0	1	1	0	0	1
273.	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
274.	1	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
275.	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
276.	1	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
277.	1	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
278.	1	0	1	1	0	1	0	0	0	0	1	1	1	0	1	0
279.	1	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0
280.	1	0	1	1	1	1	0	0	0	0	1	1	1	1	1	1

Tabla C.16 Salida del programa 2: Combinaciones 241-280.

Comb	Ax1	Ax2	Ax3	Ax4	Ax5	Ax6	Ax7	Ax8	Ax9	Ax10	Ax11	Ax12	Ax13	Ax14	Ax15	Ax16
281.	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
282.	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0
283.	1	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0
284.	1	1	0	0	1	1	0	0	1	1	0	1	0	0	0	1
285.	1	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0
286.	1	1	0	1	1	0	0	1	1	0	0	1	0	1	0	0
287.	1	1	1	0	0	0	1	0	0	0	0	1	0	0	0	0
288.	1	1	1	0	1	0	1	0	1	0	0	1	0	0	0	0
289.	1	1	1	1	0	0	1	1	0	0	1	1	0	0	0	0
290.	1	1	1	1	1	0	1	1	1	0	1	1	0	1	0	0

Tabla C.17 Salida del programa 2: Combinaciones 281-290.

1.5 Programa 3: Genera a partir de los Axiomas las Combinaciones Válidas

```
Programa Lógico en Prolog
Intérprete: JIP- Jave Internet Prolog version 2.2.3
by Ugo Chirico – www.geocities.com/jiprolog/jiprolog
```

Axiomas

```
aciclico(ac1).
aciclico(ac2).
aciclico(ac3).
anti(at).
salida(s1).
salida(s2).
salida(s3).
llegada(ll1).
llegada(ll2).
llegada(ll3).
transmision(tr1).
transmision(tr2).
transmision(tr3).
asociacion(as).
```

```
posible2(ac1, ac2).
posible2(A,B):- aciclico(A), anti(B).
posible2(A,B):- aciclico(A), salida(B).
posible2(A,B):- aciclico(A), llegada(B).
posible2(A,tr2):- aciclico(A).
posible2(A,as):- aciclico(A).
posible2(at,s2).
posible2(at,ll2).
posible2(at,tr2).
posible2(at,as).
posible2(A,B):- salida(A), llegada(B).
posible2(A,B):- salida(A), transmision(B).
posible2(s1,as).
posible2(A,B):- llegada(A), transmision(B).
posible2(ll1,as).
posible2(A,as):- transmision(A).

posible3(A, B, C):- posible2(A, B), posible2(A, C), posible2(B, C).

posible4(A,B,C,D):- posible3(A,B,C), posible3(A,B,D), posible3(A,C,D),
posible3(B,C,D).

posible5(A,B,C,D,E):- posible4(A,B,C,D), posible4(A,B,C,E),
posible4(A,B,D,E), posible4(A,C,D,E), posible4(B,C,D,E).

posible6(A,B,C,D,E,F):- posible5(A,B,C,D,E), posible5(A,B,C,D,F),
posible5(A,B,C,E,F), posible5(A,B,D,E,F), posible5(A,C,D,E,F),
posible5(B,C,D,E,F).
```

1.6 Resultados 3: Genera a partir de los Axiomas las Combinaciones Válidas

Los resultados del programa 3 corresponden a las salidas que genera el programa lógico cuando se le hacen las siguientes consultas:

posible2(A, B).

posible3(A, B, C).

posible4(A, B, C, D).

posible5(A, B, C, D, E).

posible6(A, B, C, D, E, F).

Las tablas C18-20 muestran las salidas para la consultas posible2(A, B), posible3(A, B, C), posible4(A, B, C, D) respectivamente, mientras que la tabla C21 contiene la salida de la ejecución de posible5(A, B, C, D, E) y posible6(A, B, C, D, E, F).

Salida	JIP:- posible2(A, B).	Salida	JIP:- posible2(A, B).
1.	posible2(ac1 ac2).	35.	posible2(s1 ll3).
2.	posible2(ac1 at).	36.	posible2(s2 ll1).
3.	posible2(ac2 at).	37.	posible2(s2 ll2).
4.	posible2(ac3 at).	38.	posible2(s2 ll3).
5.	posible2(ac1 s1).	39.	posible2(s3 ll1).
6.	posible2(ac1 s2).	40.	posible2(s3 ll2).
7.	posible2(ac1 s3).	41.	posible2(s3 ll3).
8.	posible2(ac2 s1).	42.	posible2(s1 tr1).
9.	posible2(ac2 s2).	43.	posible2(s1 tr2).
10.	posible2(ac2 s3).	44.	posible2(s1 tr3).
11.	posible2(ac3 s1).	45.	posible2(s2 tr1).
12.	posible2(ac3 s2).	46.	posible2(s2 tr2).
13.	posible2(ac3 s3).	47.	posible2(s2 tr3).
14.	posible2(ac1 ll1).	48.	posible2(s3 tr1).
15.	posible2(ac1 ll2).	49.	posible2(s3 tr2).
16.	posible2(ac1 ll3).	50.	posible2(s3 tr3).
17.	posible2(ac2 ll1).	51.	posible2(s1 as).
18.	posible2(ac2 ll2).	52.	posible2(ll1 tr1).
19.	posible2(ac2 ll3).	53.	posible2(ll1 tr2).
20.	posible2(ac3 ll1).	54.	posible2(ll1 tr3).
21.	posible2(ac3 ll2).	55.	posible2(ll2 tr1).
22.	posible2(ac3 ll3).	56.	posible2(ll2 tr2).
23.	posible2(ac1 tr2).	57.	posible2(ll2 tr3).
24.	posible2(ac2 tr2).	58.	posible2(ll3 tr1).
25.	posible2(ac3 tr2).	59.	posible2(ll3 tr2).
26.	posible2(ac1 as).	60.	posible2(ll3 tr3).
27.	posible2(ac2 as).	61.	posible2(ll1 as).
28.	posible2(ac3 as).	62.	posible2(tr1 as).
29.	posible2(at s2).	63.	posible2(tr2 as).
30.	posible2(at ll2).	64.	posible2(tr3 as).
31.	posible2(at tr2).		
32.	posible2(at as).		
33.	posible2(s1 ll1).		
34.	posible2(s1 ll2).		

Tabla C.18 Salida del programa 3: Consulta posible2(A, B).

Salida	JIP:- posible3(A B C).	Salida	JIP:- posible3(A B C).	Salida	JIP:- posible3(A B C).
1.	posible3(ac1 ac2 at).	41.	posible3(ac2 s2 ll2).	81.	posible3(s1 ll1 tr2).
2.	posible3(ac1 ac2 s1).	42.	posible3(ac2 s2 ll3).	82.	posible3(s1 ll1 tr3).
3.	posible3(ac1 ac2 s2).	43.	posible3(ac2 s2 tr2).	83.	posible3(s1 ll1 as).
4.	posible3(ac1 ac2 s3).	44.	posible3(ac2 s3 ll1).	84.	posible3(s1 ll2 tr1).
5.	posible3(ac1 ac2 ll1).	45.	posible3(ac2 s3 ll2).	85.	posible3(s1 ll2 tr2).
6.	posible3(ac1 ac2 ll2).	46.	posible3(ac2 s3 ll3).	86.	posible3(s1 ll2 tr3).
7.	posible3(ac1 ac2 ll3).	47.	posible3(ac2 s3 tr2).	87.	posible3(s1 ll3 tr1).
8.	posible3(ac1 ac2 tr2).	48.	posible3(ac3 s1 ll1).	88.	posible3(s1 ll3 tr2).
9.	posible3(ac1 ac2 as).	49.	posible3(ac3 s1 ll2).	89.	posible3(s1 ll3 tr3).
10.	posible3(ac1 at s2).	50.	posible3(ac3 s1 ll3).	90.	posible3(s2 ll1 tr1).
11.	posible3(ac1 at ll2).	51.	posible3(ac3 s1 tr2).	91.	posible3(s2 ll1 tr2).
12.	posible3(ac1 at tr2).	52.	posible3(ac3 s1 as).	92.	posible3(s2 ll1 tr3).
13.	posible3(ac1 at as).	53.	posible3(ac3 s2 ll1).	93.	posible3(s2 ll2 tr1).
14.	posible3(ac2 at s2).	54.	posible3(ac3 s2 ll2).	94.	posible3(s2 ll2 tr2).
15.	posible3(ac2 at ll2).	55.	posible3(ac3 s2 ll3).	95.	posible3(s2 ll2 tr3).
16.	posible3(ac2 at tr2).	56.	posible3(ac3 s2 tr2).	96.	posible3(s2 ll3 tr1).
17.	posible3(ac2 at as).	57.	posible3(ac3 s3 ll1).	97.	posible3(s2 ll3 tr2).
18.	posible3(ac3 at s2).	58.	posible3(ac3 s3 ll2).	98.	posible3(s2 ll3 tr3).
19.	posible3(ac3 at ll2).	59.	posible3(ac3 s3 ll3).	99.	posible3(s3 ll1 tr1).
20.	posible3(ac3 at tr2).	60.	posible3(ac3 s3 tr2).	100.	posible3(s3 ll1 tr2).
21.	posible3(ac3 at as).	61.	posible3(ac1 ll1 tr2).	101.	posible3(s3 ll1 tr3).
22.	posible3(ac1 s1 ll1).	62.	posible3(ac1 ll1 as).	102.	posible3(s3 ll2 tr1).
23.	posible3(ac1 s1 ll2).	63.	posible3(ac1 ll2 tr2).	103.	posible3(s3 ll2 tr2).
24.	posible3(ac1 s1 ll3).	64.	posible3(ac1 ll3 tr2).	104.	posible3(s3 ll2 tr3).
25.	posible3(ac1 s1 tr2).	65.	posible3(ac2 ll1 tr2).	105.	posible3(s3 ll3 tr1).
26.	posible3(ac1 s1 as).	66.	posible3(ac2 ll1 as).	106.	posible3(s3 ll3 tr2).
27.	posible3(ac1 s2 ll1).	67.	posible3(ac2 ll2 tr2).	107.	posible3(s3 ll3 tr3).
28.	posible3(ac1 s2 ll2).	68.	posible3(ac2 ll3 tr2).	108.	posible3(s1 tr1 as).
29.	posible3(ac1 s2 ll3).	69.	posible3(ac3 ll1 tr2).	109.	posible3(s1 tr2 as).
30.	posible3(ac1 s2 tr2).	70.	posible3(ac3 ll1 as).	110.	posible3(s1 tr3 as).
31.	posible3(ac1 s3 ll1).	71.	posible3(ac3 ll2 tr2).	111.	posible3(ll1 tr1 as).
32.	posible3(ac1 s3 ll2).	72.	posible3(ac3 ll3 tr2).	112.	posible3(ll1 tr2 as).
33.	posible3(ac1 s3 ll3).	73.	posible3(ac1 tr2 as).	113.	posible3(ll1 tr3 as).
34.	posible3(ac1 s3 tr2).	74.	posible3(ac2 tr2 as).		
35.	posible3(ac2 s1 ll1).	75.	posible3(ac3 tr2 as).		
36.	posible3(ac2 s1 ll2).	76.	posible3(at s2 ll2).		
37.	posible3(ac2 s1 ll3).	77.	posible3(at s2 tr2).		
38.	posible3(ac2 s1 tr2).	78.	posible3(at ll2 tr2).		
39.	posible3(ac2 s1 as).	79.	posible3(at tr2 as).		
40.	posible3(ac2 s2 ll1).	80.	posible3(s1 ll1 tr1).		

Tabla C.19 Salida del programa 3: Consulta posible3(A, B, C).

Salida	JIP:- posible4(A B C D).	Salida	JIP:- posible4(A B C D).
1.	posible4(ac1 ac2 at s2).	41.	posible4(ac1 s2 ll2 tr2).
2.	posible4(ac1 ac2 at ll2).	42.	posible4(ac1 s2 ll3 tr2).
3.	posible4(ac1 ac2 at tr2).	43.	posible4(ac1 s3 ll1 tr2).
4.	posible4(ac1 ac2 at as).	44.	posible4(ac1 s3 ll2 tr2).
5.	posible4(ac1 ac2 s1 ll1).	45.	posible4(ac1 s3 ll3 tr2).
6.	posible4(ac1 ac2 s1 ll2).	46.	posible4(ac2 s1 ll1 tr2).
7.	posible4(ac1 ac2 s1 ll3).	47.	posible4(ac2 s1 ll1 as).
8.	posible4(ac1 ac2 s1 tr2).	48.	posible4(ac2 s1 ll2 tr2).
9.	posible4(ac1 ac2 s1 as).	49.	posible4(ac2 s1 ll3 tr2).
10.	posible4(ac1 ac2 s2 ll1).	50.	posible4(ac2 s1 tr2 as).
11.	posible4(ac1 ac2 s2 ll2).	51.	posible4(ac2 s2 ll1 tr2).
12.	posible4(ac1 ac2 s2 ll3).	52.	posible4(ac2 s2 ll2 tr2).
13.	posible4(ac1 ac2 s2 tr2).	53.	posible4(ac2 s2 ll3 tr2).
14.	posible4(ac1 ac2 s3 ll1).	54.	posible4(ac2 s3 ll1 tr2).
15.	posible4(ac1 ac2 s3 ll2).	55.	posible4(ac2 s3 ll2 tr2).
16.	posible4(ac1 ac2 s3 ll3).	56.	posible4(ac2 s3 ll3 tr2).
17.	posible4(ac1 ac2 s3 tr2).	57.	posible4(ac3 s1 ll1 tr2).
18.	posible4(ac1 ac2 ll1 tr2).	58.	posible4(ac3 s1 ll1 as).
19.	posible4(ac1 ac2 ll1 as).	59.	posible4(ac3 s1 ll2 tr2).
20.	posible4(ac1 ac2 ll2 tr2).	60.	posible4(ac3 s1 ll3 tr2).
21.	posible4(ac1 ac2 ll3 tr2).	61.	posible4(ac3 s1 tr2 as).
22.	posible4(ac1 ac2 tr2 as).	62.	posible4(ac3 s2 ll1 tr2).
23.	posible4(ac1 at s2 ll2).	63.	posible4(ac3 s2 ll2 tr2).
24.	posible4(ac1 at s2 tr2).	64.	posible4(ac3 s2 ll3 tr2).
25.	posible4(ac1 at ll2 tr2).	65.	posible4(ac3 s3 ll1 tr2).
26.	posible4(ac1 at tr2 as).	66.	posible4(ac3 s3 ll2 tr2).
27.	posible4(ac2 at s2 ll2).	67.	posible4(ac3 s3 ll3 tr2).
28.	posible4(ac2 at s2 tr2).	68.	posible4(ac1 ll1 tr2 as).
29.	posible4(ac2 at ll2 tr2).	69.	posible4(ac2 ll1 tr2 as).
30.	posible4(ac2 at tr2 as).	70.	posible4(ac3 ll1 tr2 as).
31.	posible4(ac3 at s2 ll2).	71.	posible4(at s2 ll2 tr2).
32.	posible4(ac3 at s2 tr2).	72.	posible4(s1 ll1 tr1 as).
33.	posible4(ac3 at ll2 tr2).	73.	posible4(s1 ll1 tr2 as).
34.	posible4(ac3 at tr2 as).	74.	posible4(s1 ll1 tr3 as).
35.	posible4(ac1 s1 ll1 tr2).		
36.	posible4(ac1 s1 ll1 as).		
37.	posible4(ac1 s1 ll2 tr2).		
38.	posible4(ac1 s1 ll3 tr2).		
39.	posible4(ac1 s1 tr2 as).		
40.	posible4(ac1 s2 ll1 tr2).		

Tabla C.20 Salida del programa 3: Consulta posible4(A, B, C, D).

Salida	JIP:- posible5(A B C D E).	Salida	JIP:- posible6(A B C D E F).
1.	posible5(ac1 ac2 at s2 ll2).	1.	posible6(ac1 ac2 at s2 ll2 tr2).
2.	posible5(ac1 ac2 at s2 tr2).	2.	posible6(ac1 ac2 s1 ll1 tr2 as).
3.	posible5(ac1 ac2 at ll2 tr2).		
4.	posible5(ac1 ac2 at tr2 as).		
5.	posible5(ac1 ac2 s1 ll1 tr2).		
6.	posible5(ac1 ac2 s1 ll1 as).		
7.	posible5(ac1 ac2 s1 ll2 tr2).		
8.	posible5(ac1 ac2 s1 ll3 tr2).		
9.	posible5(ac1 ac2 s1 tr2 as).		
10.	posible5(ac1 ac2 s2 ll1 tr2).		
11.	posible5(ac1 ac2 s2 ll2 tr2).		
12.	posible5(ac1 ac2 s2 ll3 tr2).		
13.	posible5(ac1 ac2 s3 ll1 tr2).		
14.	posible5(ac1 ac2 s3 ll2 tr2).		
15.	posible5(ac1 ac2 s3 ll3 tr2).		
16.	posible5(ac1 ac2 ll1 tr2 as).		
17.	posible5(ac1 at s2 ll2 tr2).		
18.	posible5(ac2 at s2 ll2 tr2).		
19.	posible5(ac3 at s2 ll2 tr2).		
20.	posible5(ac1 s1 ll1 tr2 as).		
21.	posible5(ac2 s1 ll1 tr2 as).		
22.	posible5(ac3 s1 ll1 tr2 as).		

Tabla C.21 Salida del programa 3: Consulta posible5(A, B, C, D, E) y posible6(A, B, C, D, E, F).

1.7 Comparación entre los Resultados del Programa 1 y 3

Finalmente, para facilitar la comparación entre las salidas de los programas 1 y 3 se han ordenado las combinaciones del primer programa por el número de Restricciones que admite. Como puede observarse, los resultados coinciden: las combinaciones calculadas por el programa 1 son las que genera el programa lógico 3.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Combinaciones con 1 Restricción															
1.	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2.	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0
3.	4	0	0	0	0	0	0	0	0	0	0	0	1	0	0
4.	8	0	0	0	0	0	0	0	0	0	0	1	0	0	0
5.	16	0	0	0	0	0	0	0	0	0	1	0	0	0	0
6.	32	0	0	0	0	0	0	0	0	1	0	0	0	0	0
7.	64	0	0	0	0	0	0	0	1	0	0	0	0	0	0
8.	128	0	0	0	0	0	0	1	0	0	0	0	0	0	0
9.	256	0	0	0	0	0	1	0	0	0	0	0	0	0	0
10.	512	0	0	0	0	1	0	0	0	0	0	0	0	0	0
11.	1024	0	0	0	1	0	0	0	0	0	0	0	0	0	0
12.	2048	0	0	1	0	0	0	0	0	0	0	0	0	0	0
13.	4096	0	1	0	0	0	0	0	0	0	0	0	0	0	0
14.	8192	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Tabla C.22 Comparación: Combinaciones con 1 Restricción.

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Combinaciones con 2 Restricciones															
1.	3	0	0	0	0	0	0	0	0	0	0	0	0	1	1
2.	5	0	0	0	0	0	0	0	0	0	0	0	1	0	1
3.	9	0	0	0	0	0	0	0	0	0	0	1	0	0	1
4.	18	0	0	0	0	0	0	0	0	0	1	0	0	1	0
5.	20	0	0	0	0	0	0	0	0	0	1	0	1	0	0
6.	24	0	0	0	0	0	0	0	0	0	1	1	0	0	0
7.	34	0	0	0	0	0	0	0	0	1	0	0	0	1	0
8.	36	0	0	0	0	0	0	0	0	1	0	0	1	0	0
9.	40	0	0	0	0	0	0	0	0	1	0	1	0	0	0
10.	65	0	0	0	0	0	0	0	1	0	0	0	0	0	1
11.	66	0	0	0	0	0	0	0	1	0	0	0	0	1	0
12.	68	0	0	0	0	0	0	0	1	0	0	0	1	0	0
13.	72	0	0	0	0	0	0	0	1	0	0	1	0	0	0
14.	130	0	0	0	0	0	0	1	0	0	0	0	0	1	0
15.	132	0	0	0	0	0	0	1	0	0	0	0	1	0	0
16.	136	0	0	0	0	0	0	1	0	0	0	1	0	0	0
17.	144	0	0	0	0	0	0	1	0	0	1	0	0	0	0
18.	160	0	0	0	0	0	0	1	0	1	0	0	0	0	0
19.	192	0	0	0	0	0	0	1	1	0	0	0	0	0	0
20.	258	0	0	0	0	0	1	0	0	0	0	0	0	1	0

Tabla C.23 Comparación: Combinaciones con 2 Restricciones (1-20).

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Combinaciones con 2 Restricciones															
21.	260	0	0	0	0	0	1	0	0	0	0	0	1	0	0
22.	264	0	0	0	0	0	1	0	0	0	0	1	0	0	0
23.	272	0	0	0	0	0	1	0	0	0	1	0	0	0	0
24.	288	0	0	0	0	0	1	0	0	1	0	0	0	0	0
25.	320	0	0	0	0	0	1	0	1	0	0	0	0	0	0
26.	513	0	0	0	0	1	0	0	0	0	0	0	0	0	1
27.	514	0	0	0	0	1	0	0	0	0	0	0	0	1	0
28.	516	0	0	0	0	1	0	0	0	0	0	0	1	0	0
29.	520	0	0	0	0	1	0	0	0	0	0	1	0	0	0
30.	528	0	0	0	0	1	0	0	0	0	1	0	0	0	0
31.	544	0	0	0	0	1	0	0	0	1	0	0	0	0	0
32.	576	0	0	0	0	1	0	0	1	0	0	0	0	0	0
33.	1025	0	0	0	1	0	0	0	0	0	0	0	0	0	1
34.	1028	0	0	0	1	0	0	0	0	0	0	0	1	0	0
35.	1056	0	0	0	1	0	0	0	0	1	0	0	0	0	0
36.	1280	0	0	0	1	0	1	0	0	0	0	0	0	0	0
37.	2049	0	0	1	0	0	0	0	0	0	0	0	0	0	1
38.	2052	0	0	1	0	0	0	0	0	0	0	0	1	0	0
39.	2064	0	0	1	0	0	0	0	0	0	1	0	0	0	0
40.	2080	0	0	1	0	0	0	0	0	1	0	0	0	0	0
41.	2112	0	0	1	0	0	0	0	1	0	0	0	0	0	0
42.	2176	0	0	1	0	0	0	1	0	0	0	0	0	0	0
43.	2304	0	0	1	0	0	1	0	0	0	0	0	0	0	0
44.	2560	0	0	1	0	1	0	0	0	0	0	0	0	0	0
45.	3072	0	0	1	1	0	0	0	0	0	0	0	0	0	0
46.	4097	0	1	0	0	0	0	0	0	0	0	0	0	0	1
47.	4100	0	1	0	0	0	0	0	0	0	0	0	1	0	0
48.	4112	0	1	0	0	0	0	0	0	0	1	0	0	0	0
49.	4128	0	1	0	0	0	0	0	0	1	0	0	0	0	0
50.	4160	0	1	0	0	0	0	0	1	0	0	0	0	0	0
51.	4224	0	1	0	0	0	0	1	0	0	0	0	0	0	0
52.	4352	0	1	0	0	0	1	0	0	0	0	0	0	0	0
53.	4608	0	1	0	0	1	0	0	0	0	0	0	0	0	0
54.	5120	0	1	0	1	0	0	0	0	0	0	0	0	0	0
55.	8193	1	0	0	0	0	0	0	0	0	0	0	0	0	1
56.	8196	1	0	0	0	0	0	0	0	0	0	0	1	0	0
57.	8208	1	0	0	0	0	0	0	0	0	1	0	0	0	0
58.	8224	1	0	0	0	0	0	0	0	1	0	0	0	0	0
59.	8256	1	0	0	0	0	0	0	1	0	0	0	0	0	0
60.	8320	1	0	0	0	0	0	1	0	0	0	0	0	0	0
61.	8448	1	0	0	0	0	1	0	0	0	0	0	0	0	0
62.	8704	1	0	0	0	1	0	0	0	0	0	0	0	0	0
63.	9216	1	0	0	1	0	0	0	0	0	0	0	0	0	0
64.	12288	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Tabla C.24 Comparación: Combinaciones con 2 Restricciones (21-64).

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Combinaciones con 3 Restricciones															
1.	67	0	0	0	0	0	0	0	1	0	0	0	0	1	1
2.	69	0	0	0	0	0	0	0	1	0	0	0	1	0	1
3.	73	0	0	0	0	0	0	0	1	0	0	1	0	0	1
4.	146	0	0	0	0	0	0	1	0	0	1	0	0	1	0
5.	148	0	0	0	0	0	0	1	0	0	1	0	1	0	0
6.	152	0	0	0	0	0	0	1	0	0	1	1	0	0	0
7.	162	0	0	0	0	0	0	1	0	1	0	0	0	1	0
8.	164	0	0	0	0	0	0	1	0	1	0	0	1	0	0
9.	168	0	0	0	0	0	0	1	0	1	0	1	0	0	0
10.	194	0	0	0	0	0	0	1	1	0	0	0	0	1	0
11.	196	0	0	0	0	0	0	1	1	0	0	0	1	0	0
12.	200	0	0	0	0	0	0	1	1	0	0	1	0	0	0
13.	274	0	0	0	0	0	1	0	0	0	1	0	0	1	0
14.	276	0	0	0	0	0	1	0	0	0	1	0	1	0	0
15.	280	0	0	0	0	0	1	0	0	0	1	1	0	0	0
16.	290	0	0	0	0	0	1	0	0	1	0	0	0	1	0
17.	292	0	0	0	0	0	1	0	0	1	0	0	1	0	0
18.	296	0	0	0	0	0	1	0	0	1	0	1	0	0	0
19.	322	0	0	0	0	0	1	0	1	0	0	0	0	1	0
20.	324	0	0	0	0	0	1	0	1	0	0	0	1	0	0
21.	328	0	0	0	0	0	1	0	1	0	0	1	0	0	0
22.	515	0	0	0	0	1	0	0	0	0	0	0	0	1	1
23.	517	0	0	0	0	1	0	0	0	0	0	0	1	0	1
24.	521	0	0	0	0	1	0	0	0	0	0	1	0	0	1
25.	530	0	0	0	0	1	0	0	0	0	1	0	0	1	0
26.	532	0	0	0	0	1	0	0	0	0	1	0	1	0	0
27.	536	0	0	0	0	1	0	0	0	0	1	1	0	0	0
28.	546	0	0	0	0	1	0	0	0	1	0	0	0	1	0
29.	548	0	0	0	0	1	0	0	0	1	0	0	1	0	0
30.	552	0	0	0	0	1	0	0	0	1	0	1	0	0	0
31.	577	0	0	0	0	1	0	0	1	0	0	0	0	0	1
32.	578	0	0	0	0	1	0	0	1	0	0	0	0	1	0
33.	580	0	0	0	0	1	0	0	1	0	0	0	1	0	0
34.	584	0	0	0	0	1	0	0	1	0	0	1	0	0	0
35.	1029	0	0	0	1	0	0	0	0	0	0	0	1	0	1
36.	1060	0	0	0	1	0	0	0	0	1	0	0	1	0	0
37.	1284	0	0	0	1	0	1	0	0	0	0	0	1	0	0
38.	1312	0	0	0	1	0	1	0	0	1	0	0	0	0	0
39.	2053	0	0	1	0	0	0	0	0	0	0	0	1	0	1
40.	2068	0	0	1	0	0	0	0	0	0	1	0	1	0	0

Tabla C.25 Comparación: Combinaciones con 3 Restricciones (1-40).

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Combinaciones con 3 Restricciones															
41.	2084	0	0	1	0	0	0	0	0	1	0	0	1	0	0
42.	2113	0	0	1	0	0	0	0	1	0	0	0	0	0	1
43.	2116	0	0	1	0	0	0	0	1	0	0	0	1	0	0
44.	2180	0	0	1	0	0	0	1	0	0	0	0	1	0	0
45.	2192	0	0	1	0	0	0	1	0	0	1	0	0	0	0
46.	2208	0	0	1	0	0	0	1	0	1	0	0	0	0	0
47.	2240	0	0	1	0	0	0	1	1	0	0	0	0	0	0
48.	2308	0	0	1	0	0	1	0	0	0	0	0	1	0	0
49.	2320	0	0	1	0	0	1	0	0	0	1	0	0	0	0
50.	2336	0	0	1	0	0	1	0	0	1	0	0	0	0	0
51.	2368	0	0	1	0	0	1	0	1	0	0	0	0	0	0
52.	2561	0	0	1	0	1	0	0	0	0	0	0	0	0	1
53.	2564	0	0	1	0	1	0	0	0	0	0	0	1	0	0
54.	2576	0	0	1	0	1	0	0	0	0	1	0	0	0	0
55.	2592	0	0	1	0	1	0	0	0	1	0	0	0	0	0
56.	2624	0	0	1	0	1	0	0	1	0	0	0	0	0	0
57.	3073	0	0	1	1	0	0	0	0	0	0	0	0	0	1
58.	3076	0	0	1	1	0	0	0	0	0	0	0	1	0	0
59.	3104	0	0	1	1	0	0	0	0	1	0	0	0	0	0
60.	3328	0	0	1	1	0	1	0	0	0	0	0	0	0	0
61.	4101	0	1	0	0	0	0	0	0	0	0	0	1	0	1
62.	4116	0	1	0	0	0	0	0	0	0	1	0	1	0	0
63.	4132	0	1	0	0	0	0	0	0	1	0	0	1	0	0
64.	4161	0	1	0	0	0	0	0	1	0	0	0	0	0	1
65.	4164	0	1	0	0	0	0	0	1	0	0	0	1	0	0
66.	4228	0	1	0	0	0	0	1	0	0	0	0	1	0	0
67.	4240	0	1	0	0	0	0	1	0	0	1	0	0	0	0
68.	4256	0	1	0	0	0	0	1	0	1	0	0	0	0	0
69.	4288	0	1	0	0	0	0	1	1	0	0	0	0	0	0
70.	4356	0	1	0	0	0	1	0	0	0	0	0	1	0	0
71.	4368	0	1	0	0	0	1	0	0	0	1	0	0	0	0
72.	4384	0	1	0	0	0	1	0	0	1	0	0	0	0	0
73.	4416	0	1	0	0	0	1	0	1	0	0	0	0	0	0
74.	4609	0	1	0	0	1	0	0	0	0	0	0	0	0	1
75.	4612	0	1	0	0	1	0	0	0	0	0	0	1	0	0
76.	4624	0	1	0	0	1	0	0	0	0	1	0	0	0	0
77.	4640	0	1	0	0	1	0	0	0	1	0	0	0	0	0
78.	4672	0	1	0	0	1	0	0	1	0	0	0	0	0	0
79.	5121	0	1	0	1	0	0	0	0	0	0	0	0	0	1
80.	5124	0	1	0	1	0	0	0	0	0	0	0	1	0	0

Tabla C.26 Comparación: Combinaciones con 3 Restricciones (41-80).

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Combinaciones con 3 Restricciones															
81.	5152	0	1	0	1	0	0	0	0	1	0	0	0	0	0
82.	5376	0	1	0	1	0	1	0	0	0	0	0	0	0	0
83.	8197	1	0	0	0	0	0	0	0	0	0	0	1	0	1
84.	8212	1	0	0	0	0	0	0	0	0	1	0	1	0	0
85.	8228	1	0	0	0	0	0	0	0	1	0	0	1	0	0
86.	8257	1	0	0	0	0	0	0	1	0	0	0	0	0	1
87.	8260	1	0	0	0	0	0	0	1	0	0	0	1	0	0
88.	8324	1	0	0	0	0	0	1	0	0	0	0	1	0	0
89.	8336	1	0	0	0	0	0	1	0	0	1	0	0	0	0
90.	8352	1	0	0	0	0	0	1	0	1	0	0	0	0	0
91.	8384	1	0	0	0	0	0	1	1	0	0	0	0	0	0
92.	8452	1	0	0	0	0	1	0	0	0	0	0	1	0	0
93.	8464	1	0	0	0	0	1	0	0	0	1	0	0	0	0
94.	8480	1	0	0	0	0	1	0	0	1	0	0	0	0	0
95.	8512	1	0	0	0	0	1	0	1	0	0	0	0	0	0
96.	8705	1	0	0	0	1	0	0	0	0	0	0	0	0	1
97.	8708	1	0	0	0	1	0	0	0	0	0	0	1	0	0
98.	8720	1	0	0	0	1	0	0	0	0	1	0	0	0	0
99.	8736	1	0	0	0	1	0	0	0	1	0	0	0	0	0
100.	8768	1	0	0	0	1	0	0	1	0	0	0	0	0	0
101.	9217	1	0	0	1	0	0	0	0	0	0	0	0	0	1
102.	9220	1	0	0	1	0	0	0	0	0	0	0	1	0	0
103.	9248	1	0	0	1	0	0	0	0	1	0	0	0	0	0
104.	9472	1	0	0	1	0	1	0	0	0	0	0	0	0	0
105.	12289	1	1	0	0	0	0	0	0	0	0	0	0	0	1
106.	12292	1	1	0	0	0	0	0	0	0	0	0	1	0	0
107.	12304	1	1	0	0	0	0	0	0	0	1	0	0	0	0
108.	12320	1	1	0	0	0	0	0	0	1	0	0	0	0	0
109.	12352	1	1	0	0	0	0	0	1	0	0	0	0	0	0
110.	12416	1	1	0	0	0	0	1	0	0	0	0	0	0	0
111.	12544	1	1	0	0	0	1	0	0	0	0	0	0	0	0
112.	12800	1	1	0	0	1	0	0	0	0	0	0	0	0	0
113.	13312	1	1	0	1	0	0	0	0	0	0	0	0	0	0

Tabla C.27 Comparación: Combinaciones con 3 Restricciones (81-113).

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Combinaciones con 4 Restricciones															
1.	579	0	0	0	0	1	0	0	1	0	0	0	0	1	1
2.	581	0	0	0	0	1	0	0	1	0	0	0	1	0	1
3.	585	0	0	0	0	1	0	0	1	0	0	1	0	0	1
4.	1316	0	0	0	1	0	1	0	0	1	0	0	1	0	0
5.	2117	0	0	1	0	0	0	0	1	0	0	0	1	0	1
6.	2196	0	0	1	0	0	0	1	0	0	1	0	1	0	0
7.	2212	0	0	1	0	0	0	1	0	1	0	0	1	0	0
8.	2244	0	0	1	0	0	0	1	1	0	0	0	1	0	0
9.	2324	0	0	1	0	0	1	0	0	0	1	0	1	0	0
10.	2340	0	0	1	0	0	1	0	0	1	0	0	1	0	0
11.	2372	0	0	1	0	0	1	0	1	0	0	0	1	0	0
12.	2565	0	0	1	0	1	0	0	0	0	0	0	1	0	1
13.	2580	0	0	1	0	1	0	0	0	0	1	0	1	0	0
14.	2596	0	0	1	0	1	0	0	0	1	0	0	1	0	0
15.	2625	0	0	1	0	1	0	0	1	0	0	0	0	0	1
16.	2628	0	0	1	0	1	0	0	1	0	0	0	1	0	0
17.	3077	0	0	1	1	0	0	0	0	0	0	0	1	0	1
18.	3108	0	0	1	1	0	0	0	0	1	0	0	1	0	0
19.	3332	0	0	1	1	0	1	0	0	0	0	0	1	0	0
20.	3360	0	0	1	1	0	1	0	0	1	0	0	0	0	0
21.	4165	0	1	0	0	0	0	0	1	0	0	0	1	0	1
22.	4244	0	1	0	0	0	0	1	0	0	1	0	1	0	0
23.	4260	0	1	0	0	0	0	1	0	1	0	0	1	0	0
24.	4292	0	1	0	0	0	0	1	1	0	0	0	1	0	0
25.	4372	0	1	0	0	0	1	0	0	0	1	0	1	0	0
26.	4388	0	1	0	0	0	1	0	0	1	0	0	1	0	0
27.	4420	0	1	0	0	0	1	0	1	0	0	0	1	0	0
28.	4613	0	1	0	0	1	0	0	0	0	0	0	1	0	1
29.	4628	0	1	0	0	1	0	0	0	0	1	0	1	0	0
30.	4644	0	1	0	0	1	0	0	0	1	0	0	1	0	0
31.	4673	0	1	0	0	1	0	0	1	0	0	0	0	0	1
32.	4676	0	1	0	0	1	0	0	1	0	0	0	1	0	0
33.	5125	0	1	0	1	0	0	0	0	0	0	0	1	0	1
34.	5156	0	1	0	1	0	0	0	0	1	0	0	1	0	0
35.	5380	0	1	0	1	0	1	0	0	0	0	0	1	0	0
36.	5408	0	1	0	1	0	1	0	0	1	0	0	0	0	0
37.	8261	1	0	0	0	0	0	0	1	0	0	0	1	0	1
38.	8340	1	0	0	0	0	0	1	0	0	1	0	1	0	0
39.	8356	1	0	0	0	0	0	1	0	1	0	0	1	0	0
40.	8388	1	0	0	0	0	0	1	1	0	0	0	1	0	0

Tabla C.28 Comparación: Combinaciones con 4 Restricciones (1-40).

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Combinaciones con 4 Restricciones															
41.	8468	1	0	0	0	0	1	0	0	0	1	0	1	0	0
42.	8484	1	0	0	0	0	1	0	0	1	0	0	1	0	0
43.	8516	1	0	0	0	0	1	0	1	0	0	0	1	0	0
44.	8709	1	0	0	0	1	0	0	0	0	0	0	1	0	1
45.	8724	1	0	0	0	1	0	0	0	0	1	0	1	0	0
46.	8740	1	0	0	0	1	0	0	0	1	0	0	1	0	0
47.	8769	1	0	0	0	1	0	0	1	0	0	0	0	0	1
48.	8772	1	0	0	0	1	0	0	1	0	0	0	1	0	0
49.	9221	1	0	0	1	0	0	0	0	0	0	0	1	0	1
50.	9252	1	0	0	1	0	0	0	0	1	0	0	1	0	0
51.	9476	1	0	0	1	0	1	0	0	0	0	0	1	0	0
52.	9504	1	0	0	1	0	1	0	0	1	0	0	0	0	0
53.	12293	1	1	0	0	0	0	0	0	0	0	0	1	0	1
54.	12308	1	1	0	0	0	0	0	0	0	1	0	1	0	0
55.	12324	1	1	0	0	0	0	0	0	1	0	0	1	0	0
56.	12353	1	1	0	0	0	0	0	1	0	0	0	0	0	1
57.	12356	1	1	0	0	0	0	0	1	0	0	0	1	0	0
58.	12420	1	1	0	0	0	0	1	0	0	0	0	1	0	0
59.	12432	1	1	0	0	0	0	1	0	0	1	0	0	0	0
60.	12448	1	1	0	0	0	0	1	0	1	0	0	0	0	0
61.	12480	1	1	0	0	0	0	1	1	0	0	0	0	0	0
62.	12548	1	1	0	0	0	1	0	0	0	0	0	1	0	0
63.	12560	1	1	0	0	0	1	0	0	0	1	0	0	0	0
64.	12576	1	1	0	0	0	1	0	0	1	0	0	0	0	0
65.	12608	1	1	0	0	0	1	0	1	0	0	0	0	0	0
66.	12801	1	1	0	0	1	0	0	0	0	0	0	0	0	1
67.	12804	1	1	0	0	1	0	0	0	0	0	0	1	0	0
68.	12816	1	1	0	0	1	0	0	0	0	1	0	0	0	0
69.	12832	1	1	0	0	1	0	0	0	1	0	0	0	0	0
70.	12864	1	1	0	0	1	0	0	1	0	0	0	0	0	0
71.	13313	1	1	0	1	0	0	0	0	0	0	0	0	0	1
72.	13316	1	1	0	1	0	0	0	0	0	0	0	1	0	0
73.	13344	1	1	0	1	0	0	0	0	1	0	0	0	0	0
74.	13568	1	1	0	1	0	1	0	0	0	0	0	0	0	0

Tabla C.29 Comparación: Combinaciones con 4 Restricciones (41-74).

Combinación válida		Ac			At	S			Ll			Tr			As
nº comb	comb dec	Ac1	Ac2	Ac3	At	S1	S2	S3	Ll1	Ll2	Ll3	Tr1	Tr2	Tr3	As3
Combinaciones con 5 Restricciones															
1.	2629	0	0	1	0	1	0	0	1	0	0	0	1	0	1
2.	3364	0	0	1	1	0	1	0	0	1	0	0	1	0	0
3.	4677	0	1	0	0	1	0	0	1	0	0	0	1	0	1
4.	5412	0	1	0	1	0	1	0	0	1	0	0	1	0	0
5.	8773	1	0	0	0	1	0	0	1	0	0	0	1	0	1
6.	9508	1	0	0	1	0	1	0	0	1	0	0	1	0	0
7.	12357	1	1	0	0	0	0	0	1	0	0	0	1	0	1
8.	12436	1	1	0	0	0	0	1	0	0	1	0	1	0	0
9.	12452	1	1	0	0	0	0	1	0	1	0	0	1	0	0
10.	12484	1	1	0	0	0	0	1	1	0	0	0	1	0	0
11.	12564	1	1	0	0	0	1	0	0	0	1	0	1	0	0
12.	12580	1	1	0	0	0	1	0	0	1	0	0	1	0	0
13.	12612	1	1	0	0	0	1	0	1	0	0	0	1	0	0
14.	12805	1	1	0	0	1	0	0	0	0	0	0	1	0	1
15.	12820	1	1	0	0	1	0	0	0	0	1	0	1	0	0
16.	12836	1	1	0	0	1	0	0	0	1	0	0	1	0	0
17.	12865	1	1	0	0	1	0	0	1	0	0	0	0	0	1
18.	12868	1	1	0	0	1	0	0	1	0	0	0	1	0	0
19.	13317	1	1	0	1	0	0	0	0	0	0	0	1	0	1
20.	13348	1	1	0	1	0	0	0	0	1	0	0	1	0	0
21.	13572	1	1	0	1	0	1	0	0	0	0	0	1	0	0
22.	13600	1	1	0	1	0	1	0	0	1	0	0	0	0	0
Combinaciones con 6 Restricciones															
1.	12869	1	1	0	0	1	0	0	1	0	0	0	1	0	1
2.	13604	1	1	0	1	0	1	0	0	1	0	0	1	0	0

Tabla C.30 Comparación: Combinaciones con 5 y 6 Restricciones.

Apéndice D

Restricciones Asociativas

1 Introducción

Las Restricciones relacionadas con las relaciones conceptuales han sido expresadas mediante la Teoría de Grafos. Éstas han sido reducidas a un total de dieciséis posibles caracterizaciones, pudiendo el autor asociar un subconjunto de las mismas a cada relación conceptual.

Estas Restricciones de Autor no tienen en cuenta las conceptos que relacionan. Se podría ofrecer un lenguaje basado en lógica temporal parecido a los empleados en las Restricciones de Navegación. Con este lenguaje se proporcionaría un mecanismo capaz de expresar cualquier tipo de limitación relacionada con las asociaciones. En este caso, el autor podría indicar qué conceptos o conjuntos de conceptos pueden asociarse, y mediante qué relaciones conceptuales.

1.1 Un Lenguaje para las Restricciones Asociativas RT_A

Se propone en este apartado una primera aproximación a este posible lenguaje. De forma general, las cláusulas del mismo serían modeladas mediante la relación entre dos acciones, de forma que la acción —o el conjunto de las acciones del cuerpo— es una precondición de la acción de la cabeza que se expresa con la regla:

$$p(x) \leftarrow \text{previo } q(y)$$

En nuestro caso, las acciones del cuerpo estarán relacionadas con los conceptos, las relaciones y/o las asociaciones previamente establecidos. Estas acciones se convierten en una condición fundamental para poder llegar a admitir la asociación que aparece en la cabeza. Cada una de las Restricciones de Asociación se interpreta como: "*se puede realizar la asociación de la cabeza si se cumple la condición impuesta en el cuerpo de la regla temporal*".

Por tanto, en este lenguaje únicamente se utilizará un predicado: ***asociar***(c_o, r_c, c_d). Este predicado podrá estar instanciado o parcialmente instanciado. El predicado ***asociar*** de la cabeza únicamente se puede aplicar a tripletas de la forma *<nombre-concepto, nombre-relación, nombre-concepto>*, *<nombre-concepto, nombre-relación,* ***x****>* ó *<****x****, nombre-relación, nombre-concepto>*. La primera tripleta se interpreta como "*tal concepto se puede relacionar con tal concepto mediante tal relación*". La segunda tripleta significa que "*de tal concepto puede partir tal relación conceptual, sin importar a qué concepto llegue*". La última tripleta quiere decir que "*de tal concepto puede recibir tal relación conceptual, sin importar el concepto que lo alcance*"

Sin embargo, en el cuerpo de la cláusula pueden aparecer las variantes de la tripleta enunciada que se indican en la tabla D.1, en donde ***x*** es un comodín que puede ser sustituido por cualquier concepto origen, concepto destino y relación conceptual, en función del lugar de la tripleta en el que aparezca.

asociar(c_o, ϰ, ϰ)	Existe un concepto c_o que es un predecesor o alcanza al concepto origen de la asociación de la cabeza
asociar(ϰ, ϰ, c_d)	Existe un concepto c_d que es un predecesor o alcanza al concepto destino de la asociación de la cabeza
asociar(ϰ, r_c, ϰ)	Existe una relación conceptual r_c que precede o alcanza al concepto origen de la asociación de la cabeza
asociar(c_o, r_c, ϰ)	Existe un concepto c_o del que parte la relación conceptual r_c que es un predecesor o alcanza al concepto origen de la asociación de la cabeza
asociar(ϰ, r_c, c_d)	Existe un concepto c_d al llega la relación conceptual r_c que es un predecesor o alcanza al concepto origen de la asociación de la cabeza
asociar(c_o, r_c, c_d)	Existe una asociación <c_o, r_c, c_d> que precede o alcanza al concepto origen de la asociación de la cabeza

Tabla D.1 Predicados del cuerpo de las cláusulas.

Dado que el predicado siempre es el mismo se puede prescindir de indicarlo y considerar que, por defecto, están implícitos en la Restricción de Asociación.

En la cabeza de la regla temporal siempre aparecerá una única asociación que será la acción que se puede llegar a realizar. El desencadenar esa acción dependerá de las condiciones impuestas en el cuerpo que estará formado por predicados ***asociar*** sobre los seis tipos de tripletas de la tabla D.1 precedidos de los operadores temporales **desde** y **antes** unidos mediante operadores lógicos **and**, **or** y **not**:

$$\langle c_o, r_c, \text{ϰ} \rangle \leftarrow \textit{fórmula-temporal}$$

$$\langle \text{ϰ}, r_c, c_d \rangle \leftarrow \textit{fórmula-temporal}$$

$$\langle c_o, r_c, c_d \rangle \leftarrow \textit{fórmula-temporal}$$

El autor podrá escribir tantas Restricciones Asociativas como considere oportunas siempre y cuando este conjunto de cláusulas sean consistentes o carezcan de toda contradicción. Esto es, el lenguaje debería estar acompañado de un conjunto de Meta-Restricciones que garanticen, en todo momento, la integridad del programa lógico.

2 Las Restricciones Relacionales expresadas en el Lenguaje de las Asociaciones

Las Restricciones sobre las Relaciones conceptuales se pueden reescribir fácilmente mediante el lenguaje que se ha esbozado. A continuación se expresan las dieciséis Restricciones propuestas en el apartado 2 del capítulo 5 —catalogadas como Acíclicas, Antitransitiva, Salida, Llegada, Transmisión y Asociación— con lógica temporal de predicados.

Restricciones Acíclicas

Ac1 Antirreflexiva. El concepto origen y destino de una relación no puede coincidir.

$$\forall < c_1, r_c, c_2 > \in A_c \Rightarrow c_1 \neq c_2 \text{ ó } \Gamma^{+1}(c_1, r_c) \neq c_1.$$

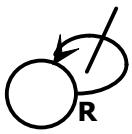
Ac1	Acíclicas
	$<c_o, \text{x}, \text{x}> \leftarrow <c_o, \text{x}, c_o>$

Tabla D.2 Restricción Ac1 expresada en el lenguaje de las Asociaciones.

Ac2 Antisimétrica. Si un concepto es el origen de una relación, el concepto destino de esa relación no puede ser el concepto origen del anterior.

$$\forall < c_1, r_c, c_2 > \in A_c \Rightarrow < c_2, r_c, c_1 > \notin A_c \text{ ó } \Gamma^{+1}(c_2, r_c) \neq c_1.$$

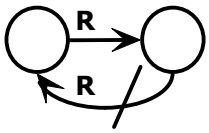
Ac2	Acíclicas
	$<c_o, r_c, c_d> \leftarrow$ **not** $<c_d, r_c, c_o>$

Tabla D.3 Restricción Ac2 expresada en el lenguaje de las Asociaciones.

Ac3 Aciclicidad. No puede existir una cadena cerrada de relaciones entre los conceptos de la EC_m. Esto es, para todo camino formado por relaciones con el mismo nombre el concepto origen de la primera relación del camino no puede coincidir con el concepto destino de la última relación.

$$\forall \mu = < c_1, r_{ci}, c_2 >, < c_2, r_{ci}, c_3 >, \ldots, < c_{n-1}, r_{ci}, c_n > \in EC_m, \text{ tal que } r_{ci} \in R_c \Rightarrow c_1 \neq c_n$$
$$\text{ó } c_1 \not\subset \Gamma^{+}(c_1, r_{ci}).$$

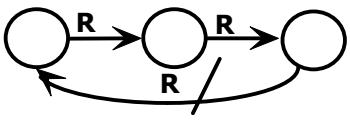
Ac3	Acíclicas
	$<c_o, r_c, c_d> \leftarrow$ **not** $<c_d, r_c, \text{x}>$

Tabla D.4 Restricción Ac3 expresada en el lenguaje de las Asociaciones.

Restricción Antitransitiva

At Antitransitiva. Si un concepto c_1 es el origen de una relación r_{ci}, y el destino de ésta c_2 es el concepto origen de la misma relación para un tercer concepto c_3 entonces el primer concepto c_1 no puede ser el concepto origen de este tercer concepto c_3, con la misma relación. En general, para cualquier número de conceptos se puede enunciar como: para todo camino formado por una determinada relación r_{ci} entre el concepto origen de la primera relación del camino y el concepto destino de la última relación no puede existir la misma relación.

$$\forall\ \mu = < c_1, r_{ci}, c_2 >, < c_2, r_{ci}, c_3 >, \ldots, < c_{n-1}, r_{ci}, c_n > \in EC_m \Rightarrow < c_1, r_{ci}, c_n > \notin A_c$$
$$\text{ó } \Gamma^{+1}(c_1, r_{ci}) \neq c_1.$$

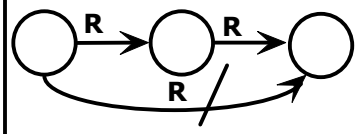
At	Antitransitiva
	$<c_o, r_c, c_d>$ ← **not** $<\nexists, r_c, c_o>$

Tabla D.5 Restricción At expresada en el lenguaje de las Asociaciones.

Restricciones de Salida

S1 Un concepto no puede ser origen de más de una asociación conceptual en las que participe la misma relación conceptual. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de salida*.

$$\forall c_i \in C \text{ y } r_c \in R_c \Rightarrow |\Gamma^{+}(c_i, r_c)| \leq 1.$$

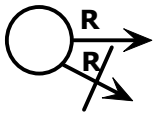
S1	Salida
	$<c_o, r_c, \nexists>$ ← **not** $<c_o, r_c, \nexists>$

Tabla D.6 Restricción S1 expresada en el lenguaje de las Asociaciones.

S2 Si un concepto es origen en una asociación conceptual en la que participe la relación conceptual r_c, ese concepto no puede ser origen en asociaciones en las que intervengan el resto de las relaciones. Por tanto, *la relación conceptual presenta una Incompatibilidad con las otras relaciones.*

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^{+}(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^{+}(c_i, r_q)| = 0.$$

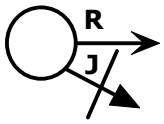
S2	Salida
	$<c_o, r_c, \aleph> \leftarrow \forall\, r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<c_o, r_{ci}, \aleph>$ esto es equivalente a decir $<c_o, r_c, \aleph> \leftarrow$ **not** ($<c_o, r_{c1}, \aleph>$ **and** $<c_o, r_{c2}, \aleph>$ **and** ... **and** $<c_o, r_{ci}, \aleph>$)

Tabla D.7 Restricción S2 expresada en el lenguaje de las Asociaciones.

S3 Si un concepto es origen en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen en ninguna otra asociación. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\, r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^+(c_i, r)| = 0.$$

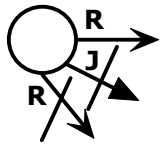
S3	Salida
	$<c_o, r_c, \aleph> \leftarrow \forall\, r_c \in R_c$ **not** $<c_o, r_c, \aleph>$ esto es equivalente a decir $<c_o, r_c, \aleph> \leftarrow$ **not** ($<c_o, r_{c1}, \aleph>$ **and** $<c_o, r_{c2}, \aleph>$ **and** ... **and** $<c_o, r_{cn}, \aleph>$)

Tabla D.8 Restricción S3 expresada en el lenguaje de las Asociaciones.

Restricciones de Llegada

LI1 Un concepto no puede ser destino de más de una asociación conceptual en las que participe la misma relación conceptual. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de llegada*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 2.$$

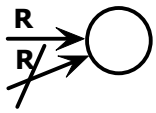
LI1	Llegada
	$<\aleph, r_c, c_d> \leftarrow$ **not** $<\aleph, r_c, c_d>$

Tabla D.9 Restricción LI1 expresada en el lenguaje de las Asociaciones.

LI2 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, ese concepto no puede ser destino en asociaciones en las que intervengan el resto de las relaciones. Por tanto, la relación conceptual presenta una *Incompatibilidad con las otras relaciones*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\, r_q \in R_c - \{r_c\},\ |\Gamma^-(c_i, r_q)| = 0.$$

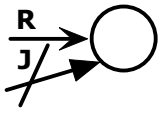
Ll2	Llegada
	$<\mathfrak{x}, r_c, c_d> \leftarrow \forall\, r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<\mathfrak{x}, r_{ci}, c_d>$ esto es equivalente a decir $<\mathfrak{x}, r_c, c_d> \leftarrow$ **not** $(<\mathfrak{x}, r_{c1}, c_d>$ **and** $<\mathfrak{x}, r_{c2}, c_d>$ **and** ... **and** $<\mathfrak{x}, r_{ci}, c_d>)$

Tabla D.10 Restricción Ll2 expresada en el lenguaje de las Asociaciones.

Ll3 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser destino en ninguna otra asociación. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall\, r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^-(c_i, r)| = 0.$$

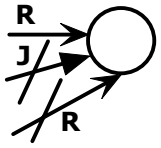
Ll3	Llegada
	$<\mathfrak{x}, r_c, c_d> \leftarrow \forall\, r_c \in R_c$ **not** $<\mathfrak{x}, r_c, c_d>$ esto es equivalente a decir $<\mathfrak{x}, r_c, c_d> \leftarrow$ **not** $(<\mathfrak{x}, r_{c1}, c_d>$ **and** $<\mathfrak{x}, r_{c2}, c_d>$ **and** ... **and** $<\mathfrak{x}, r_{cn}, c_d>)$

Tabla D.11 Restricción Ll3 expresada en el lenguaje de las Asociaciones.

Restricciones de Transmisión

Tr1 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen en otra asociación conceptual en la que intervenga la misma relación, aunque puede ser origen de asociaciones en cuyas tripletas aparezcan otras relaciones conceptuales. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de transmisión*.

$$\forall\, c_i \in C,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_c)| = 0.$$

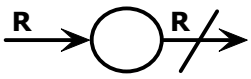
Tr1	Transmisión
	$<c_o, r_c, \mathfrak{x}> \leftarrow$ **not** $<\mathfrak{x}, r_c, c_o>$

Tabla D.12 Restricción Tr1 expresada en el lenguaje de las Asociaciones.

Tr2 Si un concepto es el destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen de asociaciones en cuyas tripletas aparezcan las otras relaciones conceptuales. Por tanto, la relación conceptual presenta *Incompatibilidad con las otras relaciones*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^+(c_i, r_q)| = 0.$$

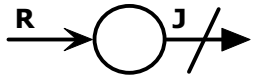
Tr2	Transmisión
	$<c_o, r_c, \mathfrak{x}> \leftarrow \forall\ r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<\mathfrak{x}, r_{ci}, c_o>$ esto es equivalente a decir $<c_o, r_c, \mathfrak{x}> \leftarrow$ **not** ($<\mathfrak{x}, r_{c1}, c_o>$ **and** $<\mathfrak{x}, r_{c2}, c_o>$ **and** ... **and** $<\mathfrak{x}, r_{ci}, c_o>$)

Tabla D.13 Restricción Tr2 expresada en el lenguaje de las Asociaciones.

Tr3 Si un concepto es el destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen en ninguna otra asociación conceptual. Se trata de un concepto hoja o *dead-end*. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^+(c_i, r)| = 0.$$

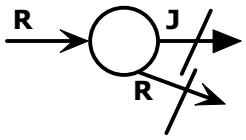
Tr3	Transmisión
	$<c_o, r_c, \mathfrak{x}> \leftarrow \forall\ r_c \in R_c$ **not** $<\mathfrak{x}, r_c, c_o>$ esto es equivalente a decir $<c_o, r_c, \mathfrak{x}> \leftarrow$ **not** ($<\mathfrak{x}, r_{c1}, c_o>$ **and** $<\mathfrak{x}, r_{c2}, c_o>$ **and** ... **and** $<\mathfrak{x}, r_{cn}, c_o>$)

Tabla D.14 Restricción Tr3 expresada en el lenguaje de las Asociaciones.

Restricciones de Asociación

As1 Si un concepto origen y un concepto destino están asociados mediante la relación conceptual r_c, no podrá existir otra asociación conceptual con los mismos elementos. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de asociación*. En realidad esta restricción tiene que verificarla toda asociación, porque no tendría sentido que existiesen elementos repetidos en un conjunto. A esta restricción la llamamos *universal* para todas las relaciones, y será la cuarta invariante que tiene que verificar toda relación conceptual.

$$\forall\ <c_o, r_c, c_d> \in A_c,\ \nexists\ <c_o, s_c, c_d> \in A_c, \text{ tal que } r_c.\text{nombre} = s_c.\text{nombre}.$$

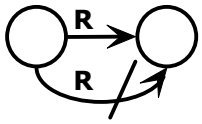
As1	Asociación
	$<c_o, r_c, c_d> \leftarrow$ **not** $<c_o, r_c, c_d>$

Tabla D.15 Restricción As1 expresada en el lenguaje de las Asociaciones.

As2 Si entre un concepto origen y un concepto destino existe una asociación en la que participe la relación conceptual r_c, entonces no podrá existir ninguna asociación entre ambos conceptos en la que intervengan las otras relaciones conceptuales. Por tanto, la relación conceptual presenta una *Incompatibilidad con las otras relaciones*.

$$\forall < c_o, r_c, c_d> \in A_c \Rightarrow \ \forall\ r_q \in R_c - \{r_c\}, <c_o, r_q, c_d> \notin A_c.$$

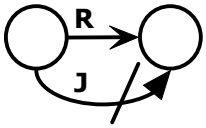
As2	Asociación
	$<c_o, r_c, c_d> \leftarrow \forall\ r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<c_o, r_{ci}, c_d>$ esto es equivalente a decir $<c_o, r_c, c_d> \leftarrow$ **not** ($<c_o, r_{c1}, c_d>$ **and** $<c_o, r_{c2}, c_d>$ **and** ... **and** $<c_o, r_{ci}, c_d>$)

Tabla D.16 Restricción As2 expresada en el lenguaje de las Asociaciones.

As3 Si entre un concepto origen y un concepto destino existe una asociación conceptual en la que participe la relación conceptual r_c, entonces no podrá existir ningún otra asociación entre ambos. *Incompatibilidad total*.

$$\forall < c_o, r_c, c_d> \in A_c \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c\ , <c_o, r, c_d> \notin A_c.$$

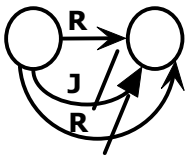
As3	Asociación
	$<c_o, r_c, c_d> \leftarrow \forall\ r_c \in R_c$ **not** $<c_o, r_c, c_d>$ esto es equivalente a decir $<c_o, r_c, c_d> \leftarrow$ **not** ($<c_o, r_{c1}, c_d>$ **and** $<c_o, r_{c2}, c_d>$ **and** ... **and** $<c_o, r_{cn}, c_d>$)

Tabla D.17 Restricción As3 expresada en el lenguaje de las Asociaciones.

Apéndice D

Restricciones Asociativas

1 Introducción

Las Restricciones relacionadas con las relaciones conceptuales han sido expresadas mediante la Teoría de Grafos. Éstas han sido reducidas a un total de dieciséis posibles caracterizaciones, pudiendo el autor asociar un subconjunto de las mismas a cada relación conceptual.

Estas Restricciones de Autor no tienen en cuenta las conceptos que relacionan. Se podría ofrecer un lenguaje basado en lógica temporal parecido a los empleados en las Restricciones de Navegación. Con este lenguaje se proporcionaría un mecanismo capaz de expresar cualquier tipo de limitación relacionada con las asociaciones. En este caso, el autor podría indicar qué conceptos o conjuntos de conceptos pueden asociarse, y mediante qué relaciones conceptuales.

1.1 Un Lenguaje para las Restricciones Asociativas RT_A

Se propone en este apartado una primera aproximación a este posible lenguaje. De forma general, las cláusulas del mismo serían modeladas mediante la relación entre dos acciones, de forma que la acción —o el conjunto de las acciones del cuerpo— es una precondición de la acción de la cabeza que se expresa con la regla:

$$p(x) \leftarrow \text{previo } q(y)$$

En nuestro caso, las acciones del cuerpo estarán relacionadas con los conceptos, las relaciones y/o las asociaciones previamente establecidos. Estas acciones se convierten en una condición fundamental para poder llegar a admitir la asociación que aparece en la cabeza. Cada una de las Restricciones de Asociación se interpreta como: "*se puede realizar la asociación de la cabeza si se cumple la condición impuesta en el cuerpo de la regla temporal*".

Por tanto, en este lenguaje únicamente se utilizará un predicado: ***asociar***(c_o, r_c, c_d). Este predicado podrá estar instanciado o parcialmente instanciado. El predicado ***asociar*** de la cabeza únicamente se puede aplicar a tripletas de la forma *<nombre-concepto, nombre-relación, nombre-concepto>*, *<nombre-concepto, nombre-relación,* ***x****>* ó *<****x****, nombre-relación, nombre-concepto>*. La primera tripleta se interpreta como "*tal concepto se puede relacionar con tal concepto mediante tal relación*". La segunda tripleta significa que "*de tal concepto puede partir tal relación conceptual, sin importar a qué concepto llegue*". La última tripleta quiere decir que "*de tal concepto puede recibir tal relación conceptual, sin importar el concepto que lo alcance*"

Sin embargo, en el cuerpo de la cláusula pueden aparecer las variantes de la tripleta enunciada que se indican en la tabla D.1, en donde ***x*** es un comodín que puede ser sustituido por cualquier concepto origen, concepto destino y relación conceptual, en función del lugar de la tripleta en el que aparezca.

asociar(c_o, ϰ, ϰ)	Existe un concepto c_o que es un predecesor o alcanza al concepto origen de la asociación de la cabeza
asociar(ϰ, ϰ, c_d)	Existe un concepto c_d que es un predecesor o alcanza al concepto destino de la asociación de la cabeza
asociar(ϰ, r_c, ϰ)	Existe una relación conceptual r_c que precede o alcanza al concepto origen de la asociación de la cabeza
asociar(c_o, r_c, ϰ)	Existe un concepto c_o del que parte la relación conceptual r_c que es un predecesor o alcanza al concepto origen de la asociación de la cabeza
asociar(ϰ, r_c, c_d)	Existe un concepto c_d al llega la relación conceptual r_c que es un predecesor o alcanza al concepto origen de la asociación de la cabeza
asociar(c_o, r_c, c_d)	Existe una asociación <c_o, r_c, c_d> que precede o alcanza al concepto origen de la asociación de la cabeza

Tabla D.1 Predicados del cuerpo de las cláusulas.

Dado que el predicado siempre es el mismo se puede prescindir de indicarlo y considerar que, por defecto, están implícitos en la Restricción de Asociación.

En la cabeza de la regla temporal siempre aparecerá una única asociación que será la acción que se puede llegar a realizar. El desencadenar esa acción dependerá de las condiciones impuestas en el cuerpo que estará formado por predicados ***asociar*** sobre los seis tipos de tripletas de la tabla D.1 precedidos de los operadores temporales **desde** y **antes** unidos mediante operadores lógicos **and**, **or** y **not**:

$$\langle c_o, r_c, \text{ϰ} \rangle \leftarrow \textit{fórmula-temporal}$$

$$\langle \text{ϰ}, r_c, c_d \rangle \leftarrow \textit{fórmula-temporal}$$

$$\langle c_o, r_c, c_d \rangle \leftarrow \textit{fórmula-temporal}$$

El autor podrá escribir tantas Restricciones Asociativas como considere oportunas siempre y cuando este conjunto de cláusulas sean consistentes o carezcan de toda contradicción. Esto es, el lenguaje debería estar acompañado de un conjunto de Meta-Restricciones que garanticen, en todo momento, la integridad del programa lógico.

2 Las Restricciones Relacionales expresadas en el Lenguaje de las Asociaciones

Las Restricciones sobre las Relaciones conceptuales se pueden reescribir fácilmente mediante el lenguaje que se ha esbozado. A continuación se expresan las dieciséis Restricciones propuestas en el apartado 2 del capítulo 5 —catalogadas como Acíclicas, Antitransitiva, Salida, Llegada, Transmisión y Asociación— con lógica temporal de predicados.

Restricciones Acíclicas

Ac1 Antirreflexiva. El concepto origen y destino de una relación no puede coincidir.

$$\forall < c_1, r_c, c_2 > \in A_c \Rightarrow c_1 \neq c_2 \text{ ó } \Gamma^{+1}(c_1, r_c) \neq c_1.$$

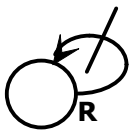
Ac1	Acíclicas
	$<c_o, \text{x}, \text{x}> \leftarrow <c_o, \text{x}, c_o>$

Tabla D.2 Restricción Ac1 expresada en el lenguaje de las Asociaciones.

Ac2 Antisimétrica. Si un concepto es el origen de una relación, el concepto destino de esa relación no puede ser el concepto origen del anterior.

$$\forall < c_1, r_c, c_2 > \in A_c \Rightarrow < c_2, r_c, c_1 > \notin A_c \text{ ó } \Gamma^{+1}(c_2, r_c) \neq c_1.$$

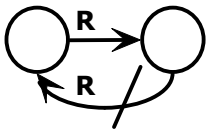
Ac2	Acíclicas
	$<c_o, r_c, c_d> \leftarrow$ **not** $<c_d, r_c, c_o>$

Tabla D.3 Restricción Ac2 expresada en el lenguaje de las Asociaciones.

Ac3 Aciclicidad. No puede existir una cadena cerrada de relaciones entre los conceptos de la EC_m. Esto es, para todo camino formado por relaciones con el mismo nombre el concepto origen de la primera relación del camino no puede coincidir con el concepto destino de la última relación.

$$\forall \mu = < c_1, r_{ci}, c_2 >, < c_2, r_{ci}, c_3 >, \ldots, < c_{n-1}, r_{ci}, c_n > \in EC_m, \text{ tal que } r_{ci} \in R_c \Rightarrow c_1 \neq c_n$$
$$\text{ó } c_1 \not\subset \Gamma^{+}(c_1, r_{ci}).$$

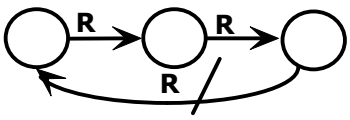
Ac3	Acíclicas
	$<c_o, r_c, c_d> \leftarrow$ **not** $<c_d, r_c, \text{x}>$

Tabla D.4 Restricción Ac3 expresada en el lenguaje de las Asociaciones.

Restricción Antitransitiva

At Antitransitiva. Si un concepto c_1 es el origen de una relación r_{ci}, y el destino de ésta c_2 es el concepto origen de la misma relación para un tercer concepto c_3 entonces el primer concepto c_1 no puede ser el concepto origen de este tercer concepto c_3, con la misma relación. En general, para cualquier número de conceptos se puede enunciar como: para todo camino formado por una determinada relación r_{ci} entre el concepto origen de la primera relación del camino y el concepto destino de la última relación no puede existir la misma relación.

$$\forall\ \mu = < c_1, r_{ci}, c_2 >, < c_2, r_{ci}, c_3 >, \ldots, < c_{n-1}, r_{ci}, c_n > \in EC_m \Rightarrow < c_1, r_{ci}, c_n > \notin A_c$$
$$\text{ó } \Gamma^{+1}(c_1, r_{ci}) \neq c_1.$$

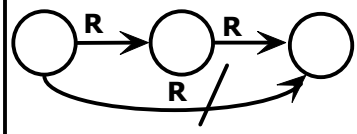
At	Antitransitiva
	$<c_o, r_c, c_d>$ ← **not** $<\nexists, r_c, c_o>$

Tabla D.5 Restricción At expresada en el lenguaje de las Asociaciones.

Restricciones de Salida

S1 Un concepto no puede ser origen de más de una asociación conceptual en las que participe la misma relación conceptual. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de salida*.

$$\forall c_i \in C \text{ y } r_c \in R_c \Rightarrow |\Gamma^{+}(c_i, r_c)| \leq 1.$$

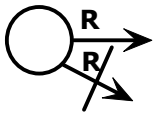
S1	Salida
	$<c_o, r_c, \nexists>$ ← **not** $<c_o, r_c, \nexists>$

Tabla D.6 Restricción S1 expresada en el lenguaje de las Asociaciones.

S2 Si un concepto es origen en una asociación conceptual en la que participe la relación conceptual r_c, ese concepto no puede ser origen en asociaciones en las que intervengan el resto de las relaciones. Por tanto, *la relación conceptual presenta una Incompatibilidad con las otras relaciones.*

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^{+}(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^{+}(c_i, r_q)| = 0.$$

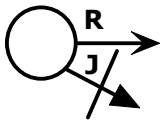
S2	Salida
	$<c_o, r_c, \aleph> \leftarrow \forall\, r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<c_o, r_{ci}, \aleph>$ esto es equivalente a decir $<c_o, r_c, \aleph> \leftarrow$ **not** ($<c_o, r_{c1}, \aleph>$ **and** $<c_o, r_{c2}, \aleph>$ **and** ... **and** $<c_o, r_{ci}, \aleph>$)

Tabla D.7 Restricción S2 expresada en el lenguaje de las Asociaciones.

S3 Si un concepto es origen en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen en ninguna otra asociación. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\, r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^+(c_i, r)| = 0.$$

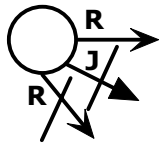
S3	Salida
	$<c_o, r_c, \aleph> \leftarrow \forall\, r_c \in R_c$ **not** $<c_o, r_c, \aleph>$ esto es equivalente a decir $<c_o, r_c, \aleph> \leftarrow$ **not** ($<c_o, r_{c1}, \aleph>$ **and** $<c_o, r_{c2}, \aleph>$ **and** ... **and** $<c_o, r_{cn}, \aleph>$)

Tabla D.8 Restricción S3 expresada en el lenguaje de las Asociaciones.

Restricciones de Llegada

LI1 Un concepto no puede ser destino de más de una asociación conceptual en las que participe la misma relación conceptual. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de llegada*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 2.$$

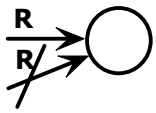
LI1	Llegada
	$<\aleph, r_c, c_d> \leftarrow$ **not** $<\aleph, r_c, c_d>$

Tabla D.9 Restricción LI1 expresada en el lenguaje de las Asociaciones.

LI2 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, ese concepto no puede ser destino en asociaciones en las que intervengan el resto de las relaciones. Por tanto, la relación conceptual presenta una *Incompatibilidad con las otras relaciones*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\, r_q \in R_c - \{r_c\},\ |\Gamma^-(c_i, r_q)| = 0.$$

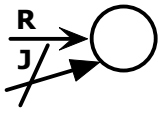
Ll2	Llegada
	$<\mathfrak{x}, r_c, c_d> \leftarrow \forall\, r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<\mathfrak{x}, r_{ci}, c_d>$ esto es equivalente a decir $<\mathfrak{x}, r_c, c_d> \leftarrow$ **not** $(<\mathfrak{x}, r_{c1}, c_d>$ **and** $<\mathfrak{x}, r_{c2}, c_d>$ **and** … **and** $<\mathfrak{x}, r_{ci}, c_d>)$

Tabla D.10 Restricción Ll2 expresada en el lenguaje de las Asociaciones.

Ll3 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser destino en ninguna otra asociación. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall\, r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^-(c_i, r)| = 0.$$

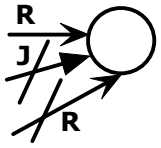
Ll3	Llegada
	$<\mathfrak{x}, r_c, c_d> \leftarrow \forall\, r_c \in R_c$ **not** $<\mathfrak{x}, r_c, c_d>$ esto es equivalente a decir $<\mathfrak{x}, r_c, c_d> \leftarrow$ **not** $(<\mathfrak{x}, r_{c1}, c_d>$ **and** $<\mathfrak{x}, r_{c2}, c_d>$ **and** … **and** $<\mathfrak{x}, r_{cn}, c_d>)$

Tabla D.11 Restricción Ll3 expresada en el lenguaje de las Asociaciones.

Restricciones de Transmisión

Tr1 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen en otra asociación conceptual en la que intervenga la misma relación, aunque puede ser origen de asociaciones en cuyas tripletas aparezcan otras relaciones conceptuales. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de transmisión*.

$$\forall\, c_i \in C,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_c)| = 0.$$

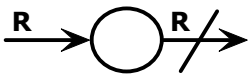
Tr1	Transmisión
	$<c_o, r_c, \mathfrak{x}> \leftarrow$ **not** $<\mathfrak{x}, r_c, c_o>$

Tabla D.12 Restricción Tr1 expresada en el lenguaje de las Asociaciones.

Tr2 Si un concepto es el destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen de asociaciones en cuyas tripletas aparezcan las otras relaciones conceptuales. Por tanto, la relación conceptual presenta *Incompatibilidad con las otras relaciones*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^+(c_i, r_q)| = 0.$$

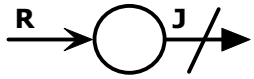
Tr2	Transmisión
	$<c_o, r_c, \mathfrak{x}> \leftarrow \forall\ r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<\mathfrak{x}, r_{ci}, c_o>$ esto es equivalente a decir $<c_o, r_c, \mathfrak{x}> \leftarrow$ **not** ($<\mathfrak{x}, r_{c1}, c_o>$ **and** $<\mathfrak{x}, r_{c2}, c_o>$ **and** ... **and** $<\mathfrak{x}, r_{ci}, c_o>$)

Tabla D.13 Restricción Tr2 expresada en el lenguaje de las Asociaciones.

Tr3 Si un concepto es el destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen en ninguna otra asociación conceptual. Se trata de un concepto hoja o *dead-end*. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^+(c_i, r)| = 0.$$

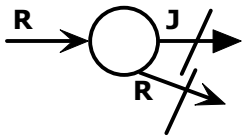
Tr3	Transmisión
	$<c_o, r_c, \mathfrak{x}> \leftarrow \forall\ r_c \in R_c$ **not** $<\mathfrak{x}, r_c, c_o>$ esto es equivalente a decir $<c_o, r_c, \mathfrak{x}> \leftarrow$ **not** ($<\mathfrak{x}, r_{c1}, c_o>$ **and** $<\mathfrak{x}, r_{c2}, c_o>$ **and** ... **and** $<\mathfrak{x}, r_{cn}, c_o>$)

Tabla D.14 Restricción Tr3 expresada en el lenguaje de las Asociaciones.

Restricciones de Asociación

As1 Si un concepto origen y un concepto destino están asociados mediante la relación conceptual r_c, no podrá existir otra asociación conceptual con los mismos elementos. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de asociación*. En realidad esta restricción tiene que verificarla toda asociación, porque no tendría sentido que existiesen elementos repetidos en un conjunto. A esta restricción la llamamos *universal* para todas las relaciones, y será la cuarta invariante que tiene que verificar toda relación conceptual.

$$\forall\ <c_o, r_c, c_d> \in A_c,\ \nexists\ <c_o, s_c, c_d> \in A_c, \text{ tal que } r_c.\text{nombre} = s_c.\text{nombre}.$$

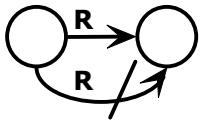
As1	Asociación
	$<c_o, r_c, c_d> \leftarrow$ **not** $<c_o, r_c, c_d>$

Tabla D.15 Restricción As1 expresada en el lenguaje de las Asociaciones.

As2 Si entre un concepto origen y un concepto destino existe una asociación en la que participe la relación conceptual r_c, entonces no podrá existir ninguna asociación entre ambos conceptos en la que intervengan las otras relaciones conceptuales. Por tanto, la relación conceptual presenta una *Incompatibilidad con las otras relaciones*.

$$\forall < c_o, r_c, c_d> \in A_c \Rightarrow \ \forall\ r_q \in R_c - \{r_c\}, <c_o, r_q, c_d> \notin A_c.$$

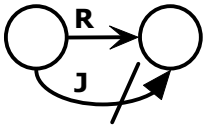
As2	Asociación
	$<c_o, r_c, c_d> \leftarrow \forall\ r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<c_o, r_{ci}, c_d>$ esto es equivalente a decir $<c_o, r_c, c_d> \leftarrow$ **not** ($<c_o, r_{c1}, c_d>$ **and** $<c_o, r_{c2}, c_d>$ **and** ... **and** $<c_o, r_{ci}, c_d>$)

Tabla D.16 Restricción As2 expresada en el lenguaje de las Asociaciones.

As3 Si entre un concepto origen y un concepto destino existe una asociación conceptual en la que participe la relación conceptual r_c, entonces no podrá existir ningún otra asociación entre ambos. *Incompatibilidad total*.

$$\forall < c_o, r_c, c_d> \in A_c \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c\ , <c_o, r, c_d> \notin A_c.$$

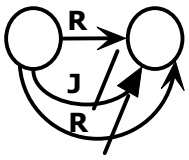
As3	Asociación
	$<c_o, r_c, c_d> \leftarrow \forall\ r_c \in R_c$ **not** $<c_o, r_c, c_d>$ esto es equivalente a decir $<c_o, r_c, c_d> \leftarrow$ **not** ($<c_o, r_{c1}, c_d>$ **and** $<c_o, r_{c2}, c_d>$ **and** ... **and** $<c_o, r_{cn}, c_d>$)

Tabla D.17 Restricción As3 expresada en el lenguaje de las Asociaciones.

Apéndice D

Restricciones Asociativas

1 Introducción

Las Restricciones relacionadas con las relaciones conceptuales han sido expresadas mediante la Teoría de Grafos. Éstas han sido reducidas a un total de dieciséis posibles caracterizaciones, pudiendo el autor asociar un subconjunto de las mismas a cada relación conceptual.

Estas Restricciones de Autor no tienen en cuenta las conceptos que relacionan. Se podría ofrecer un lenguaje basado en lógica temporal parecido a los empleados en las Restricciones de Navegación. Con este lenguaje se proporcionaría un mecanismo capaz de expresar cualquier tipo de limitación relacionada con las asociaciones. En este caso, el autor podría indicar qué conceptos o conjuntos de conceptos pueden asociarse, y mediante qué relaciones conceptuales.

1.1 Un Lenguaje para las Restricciones Asociativas RT_A

Se propone en este apartado una primera aproximación a este posible lenguaje. De forma general, las cláusulas del mismo serían modeladas mediante la relación entre dos acciones, de forma que la acción —o el conjunto de las acciones del cuerpo— es una precondición de la acción de la cabeza que se expresa con la regla:

$$p(x) \leftarrow \text{previo } q(y)$$

En nuestro caso, las acciones del cuerpo estarán relacionadas con los conceptos, las relaciones y/o las asociaciones previamente establecidos. Estas acciones se convierten en una condición fundamental para poder llegar a admitir la asociación que aparece en la cabeza. Cada una de las Restricciones de Asociación se interpreta como: "*se puede realizar la asociación de la cabeza si se cumple la condición impuesta en el cuerpo de la regla temporal*".

Por tanto, en este lenguaje únicamente se utilizará un predicado: ***asociar***(c_o, r_c, c_d). Este predicado podrá estar instanciado o parcialmente instanciado. El predicado ***asociar*** de la cabeza únicamente se puede aplicar a tripletas de la forma *<nombre-concepto, nombre-relación, nombre-concepto>*, *<nombre-concepto, nombre-relación,* ***x****>* ó *<****x****, nombre-relación, nombre-concepto>*. La primera tripleta se interpreta como "*tal concepto se puede relacionar con tal concepto mediante tal relación*". La segunda tripleta significa que "*de tal concepto puede partir tal relación conceptual, sin importar a qué concepto llegue*". La última tripleta quiere decir que "*de tal concepto puede recibir tal relación conceptual, sin importar el concepto que lo alcance*"

Sin embargo, en el cuerpo de la cláusula pueden aparecer las variantes de la tripleta enunciada que se indican en la tabla D.1, en donde ***x*** es un comodín que puede ser sustituido por cualquier concepto origen, concepto destino y relación conceptual, en función del lugar de la tripleta en el que aparezca.

asociar(c_o, ϰ, ϰ)	Existe un concepto c_o que es un predecesor o alcanza al concepto origen de la asociación de la cabeza
asociar(ϰ, ϰ, c_d)	Existe un concepto c_d que es un predecesor o alcanza al concepto destino de la asociación de la cabeza
asociar(ϰ, r_c, ϰ)	Existe una relación conceptual r_c que precede o alcanza al concepto origen de la asociación de la cabeza
asociar(c_o, r_c, ϰ)	Existe un concepto c_o del que parte la relación conceptual r_c que es un predecesor o alcanza al concepto origen de la asociación de la cabeza
asociar(ϰ, r_c, c_d)	Existe un concepto c_d al llega la relación conceptual r_c que es un predecesor o alcanza al concepto origen de la asociación de la cabeza
asociar(c_o, r_c, c_d)	Existe una asociación <c_o, r_c, c_d> que precede o alcanza al concepto origen de la asociación de la cabeza

Tabla D.1 Predicados del cuerpo de las cláusulas.

Dado que el predicado siempre es el mismo se puede prescindir de indicarlo y considerar que, por defecto, están implícitos en la Restricción de Asociación.

En la cabeza de la regla temporal siempre aparecerá una única asociación que será la acción que se puede llegar a realizar. El desencadenar esa acción dependerá de las condiciones impuestas en el cuerpo que estará formado por predicados ***asociar*** sobre los seis tipos de tripletas de la tabla D.1 precedidos de los operadores temporales **desde** y **antes** unidos mediante operadores lógicos **and**, **or** y **not**:

$$\langle c_o, r_c, \text{ϰ} \rangle \leftarrow \textit{fórmula-temporal}$$

$$\langle \text{ϰ}, r_c, c_d \rangle \leftarrow \textit{fórmula-temporal}$$

$$\langle c_o, r_c, c_d \rangle \leftarrow \textit{fórmula-temporal}$$

El autor podrá escribir tantas Restricciones Asociativas como considere oportunas siempre y cuando este conjunto de cláusulas sean consistentes o carezcan de toda contradicción. Esto es, el lenguaje debería estar acompañado de un conjunto de Meta-Restricciones que garanticen, en todo momento, la integridad del programa lógico.

2 Las Restricciones Relacionales expresadas en el Lenguaje de las Asociaciones

Las Restricciones sobre las Relaciones conceptuales se pueden reescribir fácilmente mediante el lenguaje que se ha esbozado. A continuación se expresan las dieciséis Restricciones propuestas en el apartado 2 del capítulo 5 —catalogadas como Acíclicas, Antitransitiva, Salida, Llegada, Transmisión y Asociación— con lógica temporal de predicados.

Restricciones Acíclicas

Ac1 Antirreflexiva. El concepto origen y destino de una relación no puede coincidir.

$$\forall < c_1, r_c, c_2 > \in A_c \Rightarrow c_1 \neq c_2 \text{ ó } \Gamma^{+1}(c_1, r_c) \neq c_1.$$

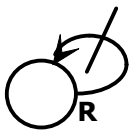
Ac1	Acíclicas
	$<c_o, \text{x}, \text{x}> \leftarrow <c_o, \text{x}, c_o>$

Tabla D.2 Restricción Ac1 expresada en el lenguaje de las Asociaciones.

Ac2 Antisimétrica. Si un concepto es el origen de una relación, el concepto destino de esa relación no puede ser el concepto origen del anterior.

$$\forall < c_1, r_c, c_2 > \in A_c \Rightarrow < c_2, r_c, c_1 > \notin A_c \text{ ó } \Gamma^{+1}(c_2, r_c) \neq c_1.$$

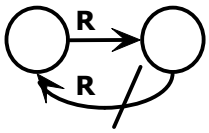
Ac2	Acíclicas
	$<c_o, r_c, c_d> \leftarrow$ **not** $<c_d, r_c, c_o>$

Tabla D.3 Restricción Ac2 expresada en el lenguaje de las Asociaciones.

Ac3 Aciclicidad. No puede existir una cadena cerrada de relaciones entre los conceptos de la EC_m. Esto es, para todo camino formado por relaciones con el mismo nombre el concepto origen de la primera relación del camino no puede coincidir con el concepto destino de la última relación.

$$\forall \mu = < c_1, r_{ci}, c_2 >, < c_2, r_{ci}, c_3 >, \ldots, < c_{n-1}, r_{ci}, c_n > \in EC_m, \text{ tal que } r_{ci} \in R_c \Rightarrow c_1 \neq c_n$$
$$\text{ó } c_1 \not\subset \Gamma^{+}(c_1, r_{ci}).$$

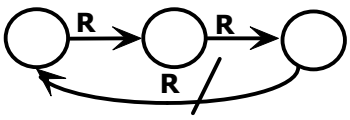
Ac3	Acíclicas
	$<c_o, r_c, c_d> \leftarrow$ **not** $<c_d, r_c, \text{x}>$

Tabla D.4 Restricción Ac3 expresada en el lenguaje de las Asociaciones.

Restricción Antitransitiva

At Antitransitiva. Si un concepto c_1 es el origen de una relación r_{ci}, y el destino de ésta c_2 es el concepto origen de la misma relación para un tercer concepto c_3 entonces el primer concepto c_1 no puede ser el concepto origen de este tercer concepto c_3, con la misma relación. En general, para cualquier número de conceptos se puede enunciar como: para todo camino formado por una determinada relación r_{ci} entre el concepto origen de la primera relación del camino y el concepto destino de la última relación no puede existir la misma relación.

$$\forall\ \mu = < c_1, r_{ci}, c_2 >, < c_2, r_{ci}, c_3 >, \ldots, < c_{n-1}, r_{ci}, c_n > \in EC_m \Rightarrow < c_1, r_{ci}, c_n > \notin A_c$$
$$\text{ó } \Gamma^{+1}(c_1, r_{ci}) \neq c_1.$$

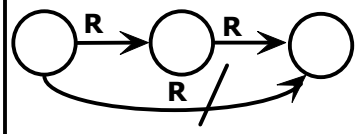
At	Antitransitiva
	$<c_o, r_c, c_d>$ ← **not** $<\nexists, r_c, c_o>$

Tabla D.5 Restricción At expresada en el lenguaje de las Asociaciones.

Restricciones de Salida

S1 Un concepto no puede ser origen de más de una asociación conceptual en las que participe la misma relación conceptual. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de salida*.

$$\forall c_i \in C \text{ y } r_c \in R_c \Rightarrow |\Gamma^{+}(c_i, r_c)| \leq 1.$$

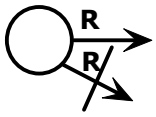
S1	Salida
	$<c_o, r_c, \nexists>$ ← **not** $<c_o, r_c, \nexists>$

Tabla D.6 Restricción S1 expresada en el lenguaje de las Asociaciones.

S2 Si un concepto es origen en una asociación conceptual en la que participe la relación conceptual r_c, ese concepto no puede ser origen en asociaciones en las que intervengan el resto de las relaciones. Por tanto, *la relación conceptual presenta una Incompatibilidad con las otras relaciones.*

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^{+}(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^{+}(c_i, r_q)| = 0.$$

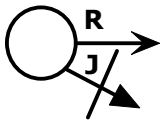
S2	Salida
	$<c_o, r_c, \aleph> \leftarrow \forall\, r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<c_o, r_{ci}, \aleph>$ esto es equivalente a decir $<c_o, r_c, \aleph> \leftarrow$ **not** ($<c_o, r_{c1}, \aleph>$ **and** $<c_o, r_{c2}, \aleph>$ **and** ... **and** $<c_o, r_{ci}, \aleph>$)

Tabla D.7 Restricción S2 expresada en el lenguaje de las Asociaciones.

S3 Si un concepto es origen en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen en ninguna otra asociación. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^+(c_i, r_c)| \neq 0 \Rightarrow \forall\, r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^+(c_i, r)| = 0.$$

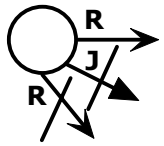
S3	Salida
	$<c_o, r_c, \aleph> \leftarrow \forall\, r_c \in R_c$ **not** $<c_o, r_c, \aleph>$ esto es equivalente a decir $<c_o, r_c, \aleph> \leftarrow$ **not** ($<c_o, r_{c1}, \aleph>$ **and** $<c_o, r_{c2}, \aleph>$ **and** ... **and** $<c_o, r_{cn}, \aleph>$)

Tabla D.8 Restricción S3 expresada en el lenguaje de las Asociaciones.

Restricciones de Llegada

LI1 Un concepto no puede ser destino de más de una asociación conceptual en las que participe la misma relación conceptual. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de llegada*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c \Rightarrow |\Gamma^-(c_i, r_c)| \leq 2.$$

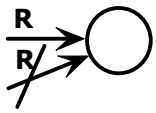
LI1	Llegada
	$<\aleph, r_c, c_d> \leftarrow$ **not** $<\aleph, r_c, c_d>$

Tabla D.9 Restricción LI1 expresada en el lenguaje de las Asociaciones.

LI2 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, ese concepto no puede ser destino en asociaciones en las que intervengan el resto de las relaciones. Por tanto, la relación conceptual presenta una *Incompatibilidad con las otras relaciones*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\, r_q \in R_c - \{r_c\},\ |\Gamma^-(c_i, r_q)| = 0.$$

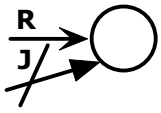
Ll2	Llegada
	$<\mathfrak{x}, r_c, c_d> \leftarrow \forall\, r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<\mathfrak{x}, r_{ci}, c_d>$ esto es equivalente a decir $<\mathfrak{x}, r_c, c_d> \leftarrow$ **not** $(<\mathfrak{x}, r_{c1}, c_d>$ **and** $<\mathfrak{x}, r_{c2}, c_d>$ **and** ... **and** $<\mathfrak{x}, r_{ci}, c_d>)$

Tabla D.10 Restricción Ll2 expresada en el lenguaje de las Asociaciones.

Ll3 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser destino en ninguna otra asociación. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\, c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow \forall\, r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^-(c_i, r)| = 0.$$

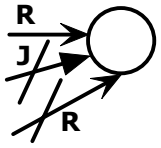
Ll3	Llegada
	$<\mathfrak{x}, r_c, c_d> \leftarrow \forall\, r_c \in R_c$ **not** $<\mathfrak{x}, r_c, c_d>$ esto es equivalente a decir $<\mathfrak{x}, r_c, c_d> \leftarrow$ **not** $(<\mathfrak{x}, r_{c1}, c_d>$ **and** $<\mathfrak{x}, r_{c2}, c_d>$ **and** ... **and** $<\mathfrak{x}, r_{cn}, c_d>)$

Tabla D.11 Restricción Ll3 expresada en el lenguaje de las Asociaciones.

Restricciones de Transmisión

Tr1 Si un concepto es destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen en otra asociación conceptual en la que intervenga la misma relación, aunque puede ser origen de asociaciones en cuyas tripletas aparezcan otras relaciones conceptuales. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de transmisión*.

$$\forall\, c_i \in C,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow |\Gamma^+(c_i, r_c)| = 0.$$

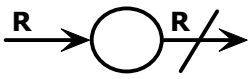
Tr1	Transmisión
	$<c_o, r_c, \mathfrak{x}> \leftarrow$ **not** $<\mathfrak{x}, r_c, c_o>$

Tabla D.12 Restricción Tr1 expresada en el lenguaje de las Asociaciones.

Tr2 Si un concepto es el destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen de asociaciones en cuyas tripletas aparezcan las otras relaciones conceptuales. Por tanto, la relación conceptual presenta *Incompatibilidad con las otras relaciones*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_c)| \neq 0 \Rightarrow\ \forall\ r_q \in R_c - \{r_c\},\ |\Gamma^+(c_i, r_q)| = 0.$$

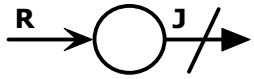
Tr2	Transmisión
	$<c_o, r_c, \mathfrak{x}> \leftarrow \forall\ r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<\mathfrak{x}, r_{ci}, c_o>$ esto es equivalente a decir $<c_o, r_c, \mathfrak{x}> \leftarrow$ **not** ($<\mathfrak{x}, r_{c1}, c_o>$ **and** $<\mathfrak{x}, r_{c2}, c_o>$ **and** ... **and** $<\mathfrak{x}, r_{ci}, c_o>$)

Tabla D.13 Restricción Tr2 expresada en el lenguaje de las Asociaciones.

Tr3 Si un concepto es el destino en una asociación conceptual en la que participe la relación conceptual r_c, entonces no puede ser origen en ninguna otra asociación conceptual. Se trata de un concepto hoja o *dead-end*. En consecuencia, presenta una *Incompatibilidad total*.

$$\forall\ c_i \in C \text{ y } r_c \in R_c,\ |\Gamma^-(c_i, r_i)| \neq 0 \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c,\ |\Gamma^+(c_i, r)| = 0.$$

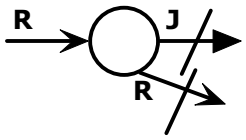
Tr3	Transmisión
	$<c_o, r_c, \mathfrak{x}> \leftarrow \forall\ r_c \in R_c$ **not** $<\mathfrak{x}, r_c, c_o>$ esto es equivalente a decir $<c_o, r_c, \mathfrak{x}> \leftarrow$ **not** ($<\mathfrak{x}, r_{c1}, c_o>$ **and** $<\mathfrak{x}, r_{c2}, c_o>$ **and** ... **and** $<\mathfrak{x}, r_{cn}, c_o>$)

Tabla D.14 Restricción Tr3 expresada en el lenguaje de las Asociaciones.

Restricciones de Asociación

As1 Si un concepto origen y un concepto destino están asociados mediante la relación conceptual r_c, no podrá existir otra asociación conceptual con los mismos elementos. Esto quiere decir que la relación conceptual presenta una *Incompatibilidad consigo misma de asociación*. En realidad esta restricción tiene que verificarla toda asociación, porque no tendría sentido que existiesen elementos repetidos en un conjunto. A esta restricción la llamamos *universal* para todas las relaciones, y será la cuarta invariante que tiene que verificar toda relación conceptual.

$$\forall\ <c_o, r_c, c_d> \in A_c,\ \nexists\ <c_o, s_c, c_d> \in A_c, \text{ tal que } r_c.\text{nombre} = s_c.\text{nombre}.$$

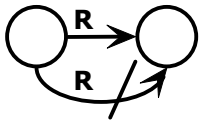
As1	Asociación
	$<c_o, r_c, c_d> \leftarrow$ **not** $<c_o, r_c, c_d>$

Tabla D.15 Restricción As1 expresada en el lenguaje de las Asociaciones.

As2 Si entre un concepto origen y un concepto destino existe una asociación en la que participe la relación conceptual r_c, entonces no podrá existir ninguna asociación entre ambos conceptos en la que intervengan las otras relaciones conceptuales. Por tanto, la relación conceptual presenta una *Incompatibilidad con las otras relaciones*.

$$\forall < c_o, r_c, c_d> \in A_c \Rightarrow \ \forall\ r_q \in R_c - \{r_c\}, <c_o, r_q, c_d> \notin A_c.$$

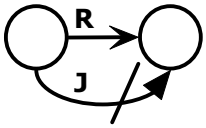
As2	Asociación
	$<c_o, r_c, c_d> \leftarrow \forall\ r_{ci} \in R_c$ con $r_{ci} \neq r_c$ **not** $<c_o, r_{ci}, c_d>$ esto es equivalente a decir $<c_o, r_c, c_d> \leftarrow$ **not** ($<c_o, r_{c1}, c_d>$ **and** $<c_o, r_{c2}, c_d>$ **and** ... **and** $<c_o, r_{ci}, c_d>$)

Tabla D.16 Restricción As2 expresada en el lenguaje de las Asociaciones.

As3 Si entre un concepto origen y un concepto destino existe una asociación conceptual en la que participe la relación conceptual r_c, entonces no podrá existir ningún otra asociación entre ambos. *Incompatibilidad total*.

$$\forall < c_o, r_c, c_d> \in A_c \Rightarrow \forall\ r \in R_c, \text{ con } r \neq r_c\ , <c_o, r, c_d> \notin A_c.$$

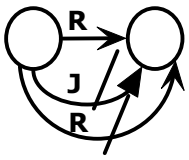
As3	Asociación
	$<c_o, r_c, c_d> \leftarrow \forall\ r_c \in R_c$ **not** $<c_o, r_c, c_d>$ esto es equivalente a decir $<c_o, r_c, c_d> \leftarrow$ **not** ($<c_o, r_{c1}, c_d>$ **and** $<c_o, r_{c2}, c_d>$ **and** ... **and** $<c_o, r_{cn}, c_d>$)

Tabla D.17 Restricción As3 expresada en el lenguaje de las Asociaciones.

Bibliografía

«Cuanto más se lee, menos se imita.»

Jules Renard.
Escritor francés (1864-1910).

«Que otros se jacten de las páginas que han escrito; a mí me enorgullecen las que he leído.»

Jorge Luis Borges.
Escritor argentino (1899-1986).

«No deseo conversar con un hombre que haya escrito más de lo que ha leído.»

Samuel Johnson.
Ensayista y lexicógrafo británico (1709-1784).

«Quien lee mucho sabe mucho, pero quien observa sabe todavía más.»

Alejandro Dumas, hijo.
Novelista y dramaturgo francés (1824-1895).

[Aalst, 1993] Aalst, W. van der, De Bra, P., Houben, G.J., Kornatzky, Y., Browsing Semantics in the Tower Model, Computing Science Note 93-47, Eindhoven Univ. of Technology, december 1993. (Also presented at the CSN-94 conference, Utrecht, 1994.)

[Afrati, 1990] Afrati, R.; Koutras, C. D. A Hypertext Model Supporting Query Mechanisms. Proceedings of the ECHT'90 European Conference on Hypertext, 52-66, 1990.

[Agresti, 1986] Agresti, W. W. Tutorial: New Paradigms for Software Development. Los Angeles. CA. IEEE Computer Society Press, 1986.

[AH, 1998] Brusilovsky, P.; Kobsa, A.; Vassileva, J. (editors). Adaptive Hypertext and Hypermedia, Kluwer Academic Publishers, Dordrecht, Febrery 1998, ISBN 0-7923-4843-5.

[AH, 2000] International Conference on Adaptive Hypermedia and Adaptive Web-based Systems, August 2000, ITC/IRST Trento, Italy, 2000 [On-Line] http://AH2000.itc.it

[Akscyn, 1988] Akscyn, R. M.; McCraken D. L.; Yoder, E. A. KMS: A Distributed Hypermedia System for Managing Knowledge in Organizations. CACM, 1988.

[Alberts, 1993] Alberts, L. K. YMIR: an Ontology for Engineering Design. University of Twente, Ph. Thesis, 1993.

[Amano, 2000] Amano, M; Watanabe, T. An Approach for constructing Component-base Software Systems with Dynamic Adaptability using LEAD++. Principles of Software Evolution. ISPSE 2000. IEEE Computer Society, 118-127, 2000

[Aoyama, 2000] Aoyama, M. Evolutionary Patterns of Design and Design Patterns. Principles of Software Evolution. ISPSE 2000. IEEE Computer Society, 110-117, 2000.

[Apple, 1989] Apple Computer. HyperCard Stack Design Guidelines. Addison-Wesley, 1989.

[Ashman, 1996] Ashman, H.L.; Balasubramanian, V.; Bieber, M.; Oinas-Kukkonen, H. 1996. Proceedings of the Second International Workshop on Incorporating Hypertext Functionality Into Software Systems (HTF II), Hypertext'96 Conference. [On-Line] http://space.njit.edu:5080/HTFII/Proceedings.html

[Baader, 1990] Baader, F. A Formal Definition for Expressive Power of Knowledge Representation Languages. Proceedings ECAI-90, Stockholm, 53-58, 1990.

[Baader, 1991] Baader, F., Hollunder, B. KRIS: Knowledge Representation and Inference System. ACM SIGART Bulleting (3), 8-14, 1991.

[Balasubramanian, 1993] Balasubramanian, V. Hypermedia Issues and Applications : A State-of-the-Art Review, www, Diciembre 1993. [On-Line] http://www.isg.sfu.ca/~duchier/misc/hypertext_review/.

[Banerjee, 1987] Banerjee, J., Kim, W., Kim, H.K., Korth, H.F. Semantics and implementation of schema evolution in object-oriented databases, In Proc. Of ACM-SIGMOD International Conference on Management of Data, San Francisco CA, May 1987.

[Barrow, 1993] Barrow, J. D. Pi in the Sky. Penguin Books, 1993. Publicado por primera vez por Oxford University Press, 1992.

[Belady, 1976] Belady, L. A.; Lehman, M. M. A Model of Large Program Development. IBM Syst. J. (15) 3, 225-252, 1976.

[Bench-Capon, 1989] Bench-Capon, T.; Dunne, P. Some Computational Properties of a Model for Electronic Documents. Electronic Pub. Orig. Diseem. Des. 2, 4, 231-256, 1989.

[Berners-Lee, 1992] Berners-Lee, T.; Cailliau R.; Groff, J.; Pollerman, B. World-Wide-Web: The Information Universe. Electronic Networking: Research, Applications and Policy, Vol 2 (1), Meckler Publishing, 1992.

[Berners-Lee, 1994] Berners-Lee, T.; Cailliau R.; Luotonen, A.; Nielsen, H.; Secret, A. The World-Wide-Web. Communication of the ACM, Vol 37 (8), 1994.

[Bertalanffy, 1968] Bertalanffy, L. Von. General System Theory. New York. George Braziller, Inc, 1968. (Trad. Fr. Théorie générale des sytèmes. Paris. Dunod. 1973).

[Bertalanffy, 1975] Bertalanffy, L. Von. Perspective on General System Theory. New York. George Braziller, Inc, 1975. (Trad. Es. Perspectivas en la Teoría General de Sistemas. Madrid. Alianza Universidad. 1979).

[Berzins, 1993] Berzins , V., Luqi, Yehudai, A:. Using Transformations in Specification-Based Prototyping, IEEE Trans.S.E. Vol. 19.5, 436-452, 1993.

[Bieber, 1991] Bieber, M. Issues in Modeling a "Dynamic" Hypertext Interface for Non-Hypertext Systems. Hypertext'91 Proceedings ACM, December 1991, 203-217, 1991.

[Bieber, 1992] Bieber, Michael. Automating Hypermedia for Decision Support. Hypermedia, 4(2), 83-110, 1992.

[Bieber, 1993] Bieber, M. Providing Information Systems with Full Hypermedia Functionality, Proceedings of the 26 Hawaii International Conference on System Sciences, 1993.

[Bieber, 1995] Bieber, M; Isakowitz, T. Designing Hypermedia Applications. Comm. of the ACM, Vol. 38, Nº8, August 1995, pp26-29.

[Bieber, 1997a] Bieber, M.; Vitali, F. Toward Support for Hypermedia on the World Wide Web. IEEE Computer, January 1997.

[Bieber, 1997b] Bieber, M.; Vitali, F.; Ashman, H.; Balasubramanian, V.; Oinas-Kukkonen, H. Fourth Generation Hypermedia: Some Missing Links for the Word Wide Web, 1997. [On-Line] http://www.cs.unibo.it/~fabio/bio/papers/1997/IJHCS97/issuev57.html.

[Boehm, 1986] Boehm, B.W. A Spiral Model of Software Development and Enhancement, ACM SIGSOFT S.E.NOTES. AUGUST Vol. 11, 14-24, 1986.

[Bolter, 1989] Bolter, J. D. Beyond Word Processing: The Computer as a New Writing Space. Language and Communication, 9 (2/3), 129-142, 1989.

[Booch, 1999] Booch, G.; Rumbaugh, J.; Jocobson, I. The Unified Modeling Language. Reference Manual. Addison-Wesley, 1999.

[Booch, 2000] Booch, G.; Rumbaugh, J.; Jocobson, I. El Lenguaje Unificado de Modelado. Addison-Wesley, 2000.

[Borgida, 1989] Borgida, A., Brachman, R. J., McGuiness, D. L., Resnick. CLASSIC: a Structural Data Model for Objects. Proceedings ACM SIGMOD International Conference on Management of Data, 59-67. 1989.

[Borgida, 1992] Borgida, A., Devanbu, P. Knowledge Base Magnagement Systems usings Description Logics and their Role in Software Information Systems. Information Processing 92 (vol. 3) 171-181, 1992. Elsevier Science Publishers.

[Botafogo, 1992] Botafogo, R.A., Rivlin, E., Shneiderman, B., Structural Analysis of Hypertexts: Identifying Hierarchies and Useful Metrics, ACM Transactions on Information Systems, 10:2, April 1992, 142-180, 1992.

[Brachman, 1985a] Brachman, R., Schmolze, J.G. An owerview of the KL-ONE Knowledge Representation System. Cognitive Science, 9(2), 171-216, 1985.

[Brachman, 1985b] Brachman, R.; McGuinness, D. L.; Patel-Schneider, P. F.; Resnick, L. A.; Borgida, A. When and How to use a KL-ONE like Language. Principles of Semantic Networks, 401-456, J. Sowa ed., 1985.

[Brachman, 1990] Brachman, R.; Schmolze, J. G. An Overview of the KL-ONE Knowledge Representations Systems. Cognitive Science, 8 (2), 171-216, 1990.

[Bray, 1998] Bray, T.; Paoli, J.; Sperberg-McQueen, C. M. Extensible Markup Language (XML) 1.0, Cambridge, Massachusets: W3C. [On-Line] http://www.w3.org/TR/REC-xml, Febrery 1998.

[Brown, 1991] Brown, P. J. Hypertext: Dreams and Reality. Hypermedia/Hypertext and Object Oriented Databases. Ed. Chapman&Hall, 1991.

[Brusilovsky, 1996] Brusilovsky, P. Methods and Techniques of Adaptive Hypermedia. User Modelling and User-Adapted Interaction 6, 87-129, 1996.

[Bush, 1945] Bush, V. As We May Think, The Atlantic Monthly, July 1945, 176, 101-108. [On-Line] http://www.isg.sfu.ca/~duchier/misc/vbush/.

[Bylander, 1988] Bylander, T.; Chandrasekaran, b. Generic Tasks in Knowledge-Based Reasoning: The Right Level of Abstraction for Knowledge Acquisition. In B. R. Gaines and J. H. Boose Eds., Knowledge Acquisition for Knowledge Based Systems. Academic Press, London, 1988.

[Campbell, 1988] Campbell, B., Goodman, J.M. HAM: A General Purpose Hypertext Abstract Machine, CACM 31:7, July 1988, 856-861.

[Candeira, 2001] Candeira, J. La Web como Memoria Organizada: el Hipocampo Colectivo de la Red. Revista de Occidente, nº 239, marzo 2001.

[Carr, 1998] Carr, L.; Hall, W. The Implications of XML for Open Hypermedia. Proceedings of 4th Workshop on Open Hypermedia Systems, 22-26, Hypertext'98 (Pittsburgh, USA), June 1998.

[Carre, 1979] Carre, B. Graphs and Networks. Clarendon Press, Oxford, 1979.

[Casais, 1990] Casais, E. Managing Class Evolution in Object-Oriented Systems. In Object Management. Tsichritzis, D. GENEVE. Centre Universitaire d'Informatique. 133-195, 1990.

[Christofides, 1975] Christofides, N. Graph Theory: An Algorithmic Approach. Academic Press Inc. London, 1975, ISBN: 0-12-174350-0.

[Chua, 1991] Chua, T. S.; Lai, E. P. M. Supporting Composition in a Hypermedia Environment. Hypermedia (3) 3, 1991, 207-238.

[Codd, 1981] Relational Database: A Practical Foundation for Productivity, E.F. Codd, 1981 Turing Award Lecture. ACM Turing Award Lectures The First Twenty Years 1966-1985 ACM Press, Anthology Series ISBN : 0-201-54885-2.

[Collins, 1992] Collins W. Collins Cobuild English Language Dictionary. HarperCollins Publishers, 1992.

[Conklin, 1987] Conklin, J. Hypertext: An Introduction and Survey. IEEE Computer, 20(9), 17-41, 1987.

[Conklin, 1989] Conklin, E. J.; Begeman, M. L. gIBIS: A Tool for All Reasons. Journal of the American Society for Information Science 40(3), 200-213, 1989.

[De Bra, 1992] De Bra, P., Houben, G.J., Kornatzky, Y. An Extensible Data Model for Hyperdocuments, 4th ACM Conference on Hypertext, Milan, December 1992, pp 222-231.

[De Bra, 1999] De Bra, P., Brusilovsky, P. Adaptive Hypermedia: From Systems to Framework, ACM Computing Surveys, 31(4), December 1999, art. 12.

[De Giacomo, 1994] De Giacomo, G.; Lenzerini, M. Boosting the Correspondence between Description Logics and Prepositional Dynamic Logics. Proceeding of AAAI-94, 205-212, 1994.

[De Rose, 1999] De Rose, S. J. XML Linking, ACM Computing Surveys 31 (4), December, 1999. [On-Line] http://www.cs.brown.edu/memex/ACM_HypertextTestbed/papers/47.html

[De Vocht, 1994] De Vocht, J. Experiments for the Characterisation of Hypertext Structures, Masters Thesis, Eindhoven University of Technology, 1994.

[De Young, 1990] De Young, L. Linking considered harmful. Proceedings of ECHT 90, (Versailles, France, Nov), 238-249, 1990.

[Devanbu, 1994] Devanbu, P., Jones, M. The Use of Description Logics in KBSE Systems. Proceedings of 17th International on Conference on Software Engineering, Sorrento, Italy, 1994.

[Engelbart, 1963] Engelbart, D. C. A Conceptual Framework for the Augmentation of Man's Intellect. Vistas in Information Handling, Spartan Books, 1963.

[Engelbart, 1990] Engelbart, D. Knowledge Domain Interoperability and an Open Hyperdocument System. Proceedings of the Conference onComputer-Supported Cooperative Work, ACM Press, New York, 143-156, 1990.

[Fiderio, 1988] Fiderio, J. A. Grand Vision. Byte, 13 (10), 237-244, October, 1988.

[Fileding, 1997] Fielding, R.; Berners-Lee, T.; Gettys, J.; Mogul, J. C.; Nielsen, H. F. HTTP/1.1 Hypertext Transfer Protocol. RFC 2068, UCIrvine, 1997.

[Furnas, 1986] Furnas G. W. Generalized Fisheye Views. CHI' 86 Conference, Boston, 16-32, Abril 1986.

[Furuta, 1990] Furuta R., Stotts, P.D. 1990. The Trellis Hypertext Reference Model, NIST Hypertext Standardization Workshop, February 1990, pp 83-93.

[Gangemi, 1995] Gangemi, A.; Steve, G.; Giacomelli, F. 1995. ONIONS: an ontological methodology for taxonomic knowledge integration. Working notes of the workshop Ontological Engineering. ECAI'96, 29-40.

[García-Cabrera, 1995] García-Cabrera, L.; García-Cabrera, R.; Rivero-Cejudo, M.L.. Reflexiones sobre el uso del Computador en la Educación. Novática, 48-51, Sep-Nov1995.

[García-Cabrera, 1996] García-Cabrera, L. ; Rivero-Cejudo, M.L.. Educación e Hipermedia. II Jornadas de Informátia, Almuñecar (Granada), 349-358, Julio 1996.

[García-Cabrera, 1997] García-Cabrera, L. ; Parets-Llorca, J. La Integración de Hipermedia en Sistemas de Información: (Una) Cuestión de Semántica. III Jornadas de Informátia, El Puerto de Santa María (Cádiz), 245-254 Julio 1997.

[García-Cabrera, 1999] García-Cabrera, L.; Parets-Llorca, J. A Semantic-Dynamic Model for Evolutionary Hypermedia Systems. Dpt. Lenguajes y Sistemas Informáticos. Working report. Un Modelo Semántico-Dinámico para Sistemas Hipermedia L. García, J. Parets DOCUMENTO DE TRABAJO LSI-99-1.

[García-Cabrera, 2000] García-Cabrera, L., Parets-Llorca, J.: A Cognitive Model for Adaptive Hypermedia Systems. The 1st International Conference on Web Information Systems Engineering, Workshop on WWW Semantincs. IEEE Press. Hong-Kong, China, June (2000) 29-33.

[García-Cabrera, 2001a] García-Cabrera, L., Rodríguez-Fórtiz, M. José, Parets-Llorca, J.: Formal Foundations for the Evolution of Hypermedia Systems. 5th European Conference on Software Maintenance and Reengineering, Workshop on FFSE. IEEE Press. Lisbon, Portugal, March (2001) 5-12.

[García-Cabrera, 2001b] García-Cabrera, L., Rodríguez-Fórtiz, M. José, Parets-Llorca, J.: . Toward a Formalisation of Evolutionary Hypermedia Systems based on System Theory, Computer Aided Systems Theory, EUROCAST'2001. In Proc. of EuroCAST'2001 (Eight International Conference on Computer Aided Systems Theory), LNCS 2178. Springer-Verlag, 2001.

[Garg, 1988] Garg, P. K. Abstraction Mechanisms in Hypertext. Communication ACM, (31) 7, 1988, 862-870.

[Garzotto, 1993] Garzotto, F; Paolini, P.; Schwabe, D. HDM-A model for the design of hypertext applications. ACM Trans. Info. Syst. Vol. 11, Nº1, Enero 1993, pp1-26.

[Garzotto, 1995] Garzotto, Mainetti, L. Paolini, P. Hypermedia Design Analysis, and Evaluation Issues. Commun. ACM. Vol. 38, Nº8, Agosto 1995,pp75-85.

[Golberg, 1983a] Golberg, A. Smalltalk-80: The Interactive Programming Environment. Addison-Wesley Publishers, Menio Park, 1983.

[Golberg, 1983b] Golberg, A; Robson D. Smalltalk-80: The Language and its Implementation. Addison-Wesley Publishers, Menio Park, 1983.

[Goldberg, 1995] Golberg, A.; Rubin, K. Succeding with Objects: Decision Framework for Project Management. Addison-Wesley, 1995.

[Goldfarb, 1990] Goldfarb, Charles F. The SGML Handbook. Edited and with a foreword by Yuri Rubinsky. Oxford: Oxford University Press, 1990.

[Goldfarb, 2000] Goldfarb, Charles F. The XML handbook. — 2nd ed. — Upper Saddle River (New Jersey) : Prentice Hall, cop. 2000.

[Gruber, 1995] Gruber, T. Toward Principles for the Design of Ontologies Used for Knowledge Sharing, International Journal of Human and Computer Studies, 43 (5/6), 907-928, 1995.

[Guarino, 1995] Guarino, N.; Giaretta, P. Ontologies and Knowledge Bases: Towards a Terminological Clarification. N. Mards (Ed.) Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing. IOS Press, Amsterdam, 25-32, 1995.

[Guinan, 1992] Guinan, C,; Smeaton, A. F. Information Retrieval from Hypertext using Dynamically Planned Guided Tours. 4th ACM Conference on Hypertext, Milan, 122-130, December, 1992.

[Gulbransen, 1998] Gulbransen, D. HTML dinámico: edición especial. — Madrid [etc.]: Prentice Hall Iberia, 1998

[Halasz, 1988a] Halasz, F. NoteCards: A Multimedia Idea Processing Environment. Interactive Multimedia, Microsoft Press, 1988.

[Halasz, 1988b] Halasz, F. Reflections on NoteCard: Seven Issues for the Next Generation of Hypermedia Systmes. Communications de ACM, July 1988.

[Halasz, 1990] Halasz, F., Schwartz, M. The Dexter Hypertext Reference Model, NIST Hypertext Standardization Workshop, February 1990, pp 95-133, 1990.

[Hammond, 1987] Hammond, N., Allinson, L., The Travel Metaphor as Design Principle and Training Aid for Navigating around Complex Systems, 3rd Conference of the British Computer Society Human-Computer Interaction Specialist Group, 1987, 75-90.

[Harold, 1998] Harold, E. R. XML: Extensible Markup Language, Inc. IDG Books Worldwide, cop, 1998.

[Heckel, 2001] Heckel, R; Engels, G: Graph Transformation as a Meta Language for Dynamic Modelling and Model Evolution. Formal Foundations for the Evolution of Hypermedia Systems. 5th European Conference on Software Maintenance and Reengineering, Workshop on FFSE. IEEE Press. Lisbon, Portugal, March 2001, 42-47.

[Henderson-Sellers, 1990] Henderson-Sellers, B.; Edwards, J.L. The Object-Oriented Systems Life Cycle, CACM 33.9. p.143-159, 1990.

[Hendriks, 1999] Hendriks, P.H.J., Vriens, D.J. Knowledge-Based Systems and Knowledge management: Friends or Foes?. Information and Management, February 1999, Vol. 35, Nº2, pp 113-123, 1999.

[Hiltz, 1996] Hiltz, S. R., Duffner, D., Fjermestad, J., Kim, Y., Ocker, R., Rana, A., & Turoff, M. Distributed Group Support Systems: Theory Development and Experimentation. In Olson, B. M., Smith, J. B., & Malone, T. (eds), Coordination Theory and Collaboration Technology, Ed: Hillsdale, NJ, 1996.

[Hurtado-Torres, 2001] Hurtado-Torres, M. V., M. José, Parets-Llorca, J. Evolutionary Information and Decision Support Systems: An Integration based on Ontologies, Computer Aided Systems Theory, In Proc. of EuroCAST'2001 (Eight International Conference on Computer Aided Systems Theory), LNCS 2178. Springer-Verlag, 2001.

[Hypertext, 1998] Second Workshop on Adaptive Hypertext and Hypermedia at Ninth ACM International Hypertext Conference, Hypertext'98. [On-Line] http://wwwis.win.tue.nl/ah98/.

[Ince, 1990] Ince, D. Set Piece. In System and Software Requirements Engineering, R. H. Thayer and M. Dorfman Eds. IEEE Computer Press, 1990, 370-372.

[Isakowitz, 1995] Isakowitz, T.; Stohr, E.A. y Balasubramanian, P. RMM: A methodology for structured hypermedia design. Commun. ACM. Vol. 38, Nº8, Agosto 1995, 34-43.

[Jameson, 1998] Jameson, A. [On-Line] http://www.dfki.de/etai/bookreviews/jameson/jameson.html.

[Kaplan, 1994] Kaplan, N. Where no Mind Has Gone Before: Ontological Design for Virtual Spaces. ECHT 94 Proceeding, 206-216.

[Kibby, 1990] Kibby, M., Mayes, T., & Anderson, T. Learning about learning from hypertext. In Jonassen, D. H. and Mandel, H. (Eds.). Designing hypermedia for learning. Berlin, London: Springer-Verlag. 227-250, 1990

[Koo, 1989] Koo, R. A Model for Electronic Documents. ACM SIGOIS Bull. (10) 1, 1989, 23-33.

[Kozaczynsky, 1992] Kozaczynsky, W., Ning, J., Engberts, A. Program Concept Recognition and Transformation, IEEE Trans.S.E. 18,12. p.1065-1075, 1992.

[Landow, 1991] Landow, G. P.; Delany, P. Hypertext, Hypermedia and Literary Studies: The State of the Art. Hypermedia and Literary Studies, MIT Press, Cambridge, MA, 3-50, 1991.

[Lange, 1990] Lange, D.B. A Formal Model of Hypertext, NIST Hypertext Standardization Workshop, February 1990, 145-166.

[Lange, 1996] Lange, B. An Object-Oriented Design Approach for Developing Hypermedia Information Systems. Journal of Organization Computing and Electronic Commerce, Vol 6, nº3, 1996.

[Le Moigne, 1977, 1983, 1990] Le Moigne, J. L. La Théorie du Systéme Général. Théorie de la Modélisation. Paris. Presses Universitaires de France, 1977.

[Le Moigne, 1990] Le Moigne, J-L. La modélisation du système génèral. Théorie de la modélisation. Paris. Presses Universitaires de France. p 62, 1990.

[Lehman, 2000] Lehman, M; Ramil, J. Towards a Theory of Software Evolution- And its Practical Impact. Principles of Software Evolution. ISPSE 2000. IEEE Computer Society 2000, 2-13.

[Marshall, 1994] Marshall C.C., Shipman III F.M., Coombs J. H. 1994. VIKI: Spatial Hypertext Supporting Emergent Structure. Xerox Palo Alto Research Center. ECHT 94 Proceeding, 83-93, 1994.

[Marshall, 1995] Marshall, C.C. & Shipman, F. Spatial Hypertext: Designing for Change. Communications of the ACM, 38(8), 88-97, 1995.

[Mayes, 1994] Mayes, M. A method for evaluating the efficiency of presenting information in a hypermedia environment. Computer in Education. 18 (1), 179-182, 1994.

[McDavis, 1991] McDaid, J. Breaking Frames: Hyper-Mass Media. Hypertext/Hypermedia Handbook. New York, Intertext Publications/Mc Graw-Hill, 445-458, 1991.

[McGregor, 1988] McGregor, R. M. A Deductive Pattern Matcher. Proceedings of AAAI-88, 1988.

[Mens, 2001] Mens, T. Transformational Software Evolution by Assertions. Formal Foundations for the Evolution of Hypermedia Systems. 5th European Conference on Software Maintenance and Reengineering, Workshop on FFSE. IEEE Press. Lisbon, Portugal, March 2001, 67-74.

[Meyrowitz, 1986] Meyrowitz, N. Intermedia: The Architecture and Construction of an Object-Oriented Hypermedia System and Application Framework. OOPLA'86, 1986.

[Meyrowitz, 1991] Meyrowitz, N. Hypertext-does it reduce cholesterol, too? From Memex to Hypertext (Kahn et al.,ed.) Academic Press, Boston, 1991.

[Morin, 1977] Morin, E. La Methòde. Tome I: La Nature de la Nature. Paris. Ed. Du Seuil. 1977. (trad. Es. El Método I: La Naturaleza de la Naturaleza. Madrid. Ed. Cátedra. 1986).

[Mosterín, 2000] Mosterín, J. Los Lógicos. Espasa Forum. Madrid, 2000.

[Murugesan, 1999] Murugesan, S.; Deshpande, Y.; Hansen, S.; Ginige, A. Web Engineering: A New Discipline for Development of Web-based Systems.First ICSE Workshop on Web Engineering, International Conference on Software Engineering, 16 -17 May 1999, Los Angeles, USA.

[Nanard, 1991] Nanard, J., Nanard, M. Using Structured Typed to Incorporate Knowledge in Hypertext. Proceeding of the Hypertext'91, (San Antonio, TX, ACM Press), 329-343, 1991.

[Nanard, 1995] Nanard, J., Nanard, M. Hypertext Design Environments and the Hypertext Design Process. Communications of the ACM, 38(8), 49-56, 1995.

[Nebel, 1990] Nebel, B. Reasoning and Revision in Hybrid Representation Systems. Lecture Notes in Artificial Intelligence, Springer-Verlag, 1990.

[Nebel, 1991] Nebe, B. Terminological Cycles: Semantics and Computational Properties. Principles of Semantics Networks, 331-361. Morgan Kaufmann, Los Altos, 1991.

[Nelson, 1967] Nelson, T. H. Getting it Out of Our System. In Schechter, G. Ed., Information Retrieval: A Critical Review, Thompson Books, Washington D. C., 1967.

[Nelson, 1987] Nelson, T. H. All for One and One for All. Hypertext'87, Univesity of North Carolina, Chapel Hill, NC, v-vii, 1987.

[Nielsen, 1990] Nielsen, J. Hypertext and Hypermedia. Academic Press. On Navigation in Textual Virtual Environments and Hypertext - page 155, 1990.

[Nielsen, 1991] Nielsen, J. Panel Discussion- The Nielsen Ratings: Hypertext Reviews, Proceedings of Hypertext'91. 1991.

[Niertrasz, 2000] Niertrasz, O; Achermann , F Supporting Compositional Styles for Software Evolution. Principles of Software Evolution. ISPSE 2000. IEEE Computer Society (2000) 14-22.

[Noll, 1996] Noll, J. & Scacchi, W. Repository Support for the Virtual Software Enterprise. In (Ashman et al., 1996).

[Nürnberg, 1997] Nürnberg, P.J., Leggett, J.J., Schneider, E.R. As We Should Have Thought. Proceedings of Hypertext 97, (Southampton, UK, ACM Press) 96-101, 1997.

[Paderewski, 1999] Paderewski, P; Parets-Llorca, J.; Anaya, A; Rodriguez, M.J; Sanchez, G; Torres, J; Hurtado, M.V. A software development tool for evolutionary prototyping of information systems". IEEE CSCC'99. Computers and Computational Engineering in Control. Electric and Computer. Engineering Series. World Scientific and Engineering Society Press, 347-352, 1999.

[Parets-Llorca, 1993] Parets, J; Anaya, A; Rodríguez, M.J.; Paderewski, P. A Representation of Software Systems Evolution Based on the Theory of the General System. Computer Aided Systems Theory, EUROCAST'93. Lecture Notes in Computer Science vol.763. Springer-Verlag. 96-109, 1993.

[Parets-Llorca, 1995a] Parets-Llorca, J. Reflexiones sobre el Proceso de Concepción de Sistemas Complejos. MEDES: un Método de Especificación, Desarrollo y Evolución de Sistemas Software. Tesis Doctoral, 1995.

[Parets-Llorca, 1995b] Parets, J.; Torres, J.C. A Language for Describing Complex-Evolutive Software Systems. In Pichler,F.; Moreno Díaz,R.; Albrecht,R. Computer Aided Systems Theory - EUROCAST'95. Lecture Notes in Computer Science. 1030. Berlin. Springer-Verlag. 1996

[Parets-Llorca, 1996a] Parets, J.; Torres, J.C. Software Maintenance versus Software Evolution: an Approach to Software Systems Evolution. IEEE Symposium and Workshop on Engineering of Computer Based-Systems (ECBS'96). Friedrichshafen, Germany, March 1996, 134-141.

[Parets-Llorca, 1996b] Parets-Llorca, J. The Evolution of Software Systems: a Framework for Modelling Organizational Evolution. Computational Engineering in Systems Applications, 846-851, 1996.

[Parets-Llorca, 1999] Parets, J.; Anaya, a., Hurtado, M.V.; Paderwski, P.; Rodríguez, M.J.; Sanchez, G.; Torres, J. A Software Development Tool for Evolutionary Prototyping of Information Systems. In Computer and Computational Engineering in Control. Electric and Computer Engineering. World Scientific and Engineering Society Press, 347-352, 1999.

[Parunak, 1991] Parunak, H. V. Don't Link me in: Set based Hypermedia for Taxonomic Reasoning. Proceedings of ACM Hypertext' 91, 1991, 233-242.

[Peltason, 1989] Peltason, C., Schmiedel, A., Kindermann, C., Quantz, J. The BACK System Revisited. Dept. of Computer Science, Technical University of Berling, Technical Report KIT-75, 1989.

[Peterson, 1981] Peterson, J. L. Petri Net Theory and the Modeling of Systems. Prentice-Hall. Englewood Cliffs, N. J., 1981.

[Rao, 1990] Rao, U., Turoff, M. Hypertext Functionality: A Theoretical Framework, International Journal of Human-Computer Interaction, Volume 4, Number 2, 333-358, 1990.

[REAL, 1992] Real Academia Española. Vigésima Primera Edición. Espasa Calpe, S. A. Madrid, pp 980, 1992.

[Rivlin, 1994] Ehud Rivlin, Rodrigo A. Botafogo, and Ben Shneiderman. Navigating in Hyperspace: Designing a Structure-based Toolbox. In Communications of the ACM (CACM), 37(2), 87-96, February 1994.

[Rizk, 1992] Rizk, A.; Sauter, L. MultiCard: An Open Hypermedia System, 4th ACM Conference on Hypertext, Proceedings of ECHT'92, ACM Press Milan, 4-10, December, 1992.

[Rodriguez-Fórtiz, 1997] Rodriguez-Fortiz, M.J; Anaya, A.; Parets-Llorca, J. Representation and Management of Memory and Decision in Evolving Software Systems. Computer Aided Systems Theory, EUROCAST'97, 29-34, 1997.

[Rodriguez-Fórtiz, 2000a] Rodriguez-Fortiz, M.J. Software Evolution: A Formalisation Based in Predicate Temporal Logic and Coloured Petri Nets. (in Spanish). Ph Thesis. October 2000.

[Rodríguez-Fórtiz, 2000b] Rodríguez, M.J.; Parets, J.; Paderewski, P.; Anaya, A.; Hurtado, M.V. HEDES: A System Theory based Tool to Support Evolutionary Software Systems. Lectures Notes in Computer Science. Vol. 1798, 450-464, Springer-Verlag, 2000.

[Rodriguez-Fórtiz, 2000c] Rodriguez-Fortiz, M.J, Parets Llorca, J. Using Predicate Temporal Logic and Coloured Petri Nets to specifying integrity restrictions in the structural evolution of temporal active systems. ISPSE 2000. International Symposium on Principles of Software Evolution. Kanazawa, Japan, November 2000, 81-85.

[Rodriguez-Fórtiz, 2001] Rodríguez-Fortiz, M.J.; Paderewski-Rodríguez, P.; García-Cabrera, L.; Parets Llorca, J. Evolutionary Modelling of Software Systems: Its Applications to Agent-Based and Hypermedia Systems. ISPSE 2001. International Symposium on Principles of Software Evolution. Vienna, Austria, September 2001, 54-61.

[Rumbaugh, 1991] Rumbaugh, M.; Blaha, W.; Premerlani, F.; Eddy, W; Lorensen. Object Oriented Modelling and Design. Prentice Hall, 1991.

[Said, 2001] Said, J; Steegmans, E. Transformations of Binary relations into Associations and Nested Classes. Formal Foundations for the Evolution of Hypermedia Systems. 5th European Conference on Software Maintenance and Reengineering, Workshop on FFSE. IEEE Press. Lisbon, Portugal, March 2001), 75-82.

[Salampasis, 1997] Salampasis, M. Hypermedia. [On-Line] http://www.it.teithe.gr/~cs1msa/hypermedia/hyp0.htm.

[Scacchi, 1989] Scacchi, W. On the Power of Domain-Specific Hypertext Environments. Journal of the American Society for Information Science, 40(3), 183-191, 1989.

[Shackelford, 1993] Shackelford, D.E., Smith, J.B. & Smith, F.D. The Architecture and Implementation of a Distributed Hypermedia Storage System. Hypertext '93 Proceedings, ACM Press, New York, 1-13, 1993.

[Schnase, 1993] Schnase, J. L., Leggett, J.J, Hicks, D.L., Szabo, R. L. Semantic Data Modelling of Hypermedia Associations. ACM Transactions on Information Systems, Vol. 11, Nº1, January 1993, 27-50.

[Schwabe. 1996] Schwabe, D., Rossi, G., Barbosa, S. Systematic Hypermedia Application Design with OOHDM. Communications of ACM, marzo 1996.

[Shipman, 1997] Shipman III, F.M., McCall, R. J. Integrating Different Perspectives on Design Rationale: Supporting the Emergence of Design Rationale form Design Communication. Artificial Intelligence in Engineering Design, Analysis, and Manufacturing (AIEDAM), 11, 2 (April, 1997), 141-154.

[Shneiderman, 1987] Shneiderman, B. User Interface Design for the Hiperties Electronic Encyclopedia. First ACM Conference on Hypertext, Chapel Hill, Hypertext'87, 189-194, November 1987.

[Shneiderman, 1989a] Shneiderman, B., & Kearsley, G. Hypertext Hands-On!: An Introduction to a New Way of Organizing and Accessing Information. 1st. ed. Addison-Wesley Publishing Company, 1989.

[Shneiderman, 1989b] Shneiderman, B. (1989). Reflections on authoring, editing, and managing hypertext. In E. Barrett (Ed.), The society of text: Hypertext, hypermedia, and the social construction of information, 115-131. Cambridge, MA: The MIT Press.

[SIGLINK, 1994] SIGLINK Worshop Report on Incorporating Hypertext Functionality into Software Systems ECHT'94. Edinburgh, Scotland, Septiembre 18, 1994.

[Silva, 1985] Siva, M. Las Redes de Petri en la Informática y la Automática. Editorial A.C. Madrid, 1985.

[Sowa, 1991] Sowa, J. F., De. Principles of Semantic Networks: Explorations in the Representation of Knowledge. The Morgan Kaufmann Series in Representation and Reasoning. San Mateo, California, Morgan Kaufmann, 1991.

[Spivey, 1989] Spivey, J. M. The Z Notation-A Reference Manual. Prentice-Hall, 1989.

[Stotts, 1989] Stotts, P.D.; Furuta, R. Programmable Browsing Semantics in Trellis. Proceedings Hypertext'89, 1989, 27-42.

[Stotts, 1991] Stotts, J.B. ; Furuta, R. Dynamic Adaptation of Hypertext Structure, Proceedings of Hypertext'91, ACM Press, 1991.

[Stotts, 1998] Stotts, P.D., Furuta, R., Ruiz-Cabarrus, C., Hyperdocuments as Automata: Verification of Trace-Based Browsing Properties by Model Checking, ACM TOIS, Vol. 16, Nº1, January 1998, 1-30.

[Tanik, 1996] Tanik, M., Yeh, R., Bieber, M., Kurfess, F., Liu, Q., Mchugh, J., Rana, A., Rossak, W.& Ng, P. Issues and Architecture for Electronic Enterprise Engineering (EEE). Proceedings of the Second World Conference on Integrated Design and Process Technology, Society for Design and Process Science, Austin, Volume 2, 57-62, 1996.

[Tardieu, 1992] Tardieu, H. Issues for Dynamic Modelling through Recent Development in European Methods. Dynamic of Infomation Systems, II. SOL. H. G., CROSSLIN, R. L. Eds. Amsterdam, North-Holland, 3-23, 1992.

[Thayse, 1989] Thayse, A.; Gochet, E.; Grégorie, P,; Cribomont, G.; Hulin, A.; Pirotte, D.; Roelants, D.; Snyers, M.; Vauclair, P.; Wolper. From Modal Logic to Deductive Databases. Introducing a Logic Based Approach to Artificial Intelligence. John Wiley & Sons, 1989.

[Tochtermann, 1995] Tochtermann, K.; Dittrich, G. Towards a Family of Formal Models for Hypermedia. In Proceedings of HIM"95, 1995, 77-91.

[Turoff, 1991] Turoff M., Rao U., Hiltz S.R. Collaborative Hypertext in Computer-Mediated Communications. Proceedings of the 24th Hawaii International Conference on System Sciences, IEEE Computer Society Press, Vol. IV, Hawaii, January 8-11, 1991, 357-366.

[UM,1994] Workshop Adaptive Hypertext and Hypermedia at 4th International Conference on User Modeling, UM'94. [On-Line] http://wwwis.win.tue.nl/ah94/.

[UM, 1997] Workshop Adaptive Systems and User Modeling on the World Wide Web at 6th International Conference on User Modeling, UM'97. [On-Line] http://zeus.gmd.de/UM97/index.html.

[UM, 1999] Second Workshop on Adaptive Systems and User Modeling on WWW at 8th International Word Wide Web Conference and 7-th International Conference on User Modeling, UM'99. [On-Line] http://wwwis.win.tue.nl/asum99/.

[Usandizaga, 1998] Usandizaga-Lombana, I. MADAME: un Marco de Trabajo para el Desarrollo de Aplicaciones y Sistemas Hipermedia. San Sebastián, Enero, 1998. Tesis Doctoral.

[Uschold, 1996] Uschold, M. 1996. Ontologies: Principles, Methods and Applications. Knowledge Engineering Review, vol. 11, n. 2, June 1996.

[Van Heijst, 1997] Van Heijst, g.; Schrreiber, A. T.; Wielinga, B. J. Using Explicit Ontologies in KBS Development. International Journal of Human and Computer Studies, 45, 183-292, 1997.

[W3C] [On-Line] http://www.w3.org/.

[Wang, 1995] Wang, W. Semantic Net based Hypertext for Authoring and Reuse. Ph. D. Dissertation. Department of Computer Science, Liverpool University, 1995.

[Wang, 1998] Wang, W., Rada, R. Structured Hypertext with Domain Semantics, ACM Trans. Information Systems, 16(4), October 1998.

[Wayner, 1996] Wayner, P. Dentro del NetWork Computer. Byte, 90-96, noviembre, 1996.

[Wermelinger, 2001] Wermelinger, M; Lopes, A; Fiadeiro, J. L. A Graph Transformation Approach to Architectural Run-Time Reconfiguration. Formal Foundations for the Evolution of Hypermedia Systems. 5th European Conference on Software Maintenance and Reengineering, Workshop on FFSE. IEEE Press. Lisbon, Portugal, March 2001, 59-66.

[Widom, 1996] Widom, J.; Ceri, S. Active Database Systems. Triggers and Rules for Advanced Database Processing. The Morgan Kaufmann series in Data Management Systems. Morgan Kauffmann, 1996.

[Winograd, 1997] Winograd, T. The Design of Interaction. Beyond Calculation. The Next Fifty Years of Computing. Spring-Verlag, New York, 149-265, 1997.

[Woods, 1991] Woods, W. A., Schmolze, J.G. The KL-ONE Family. Computers and Mathematics with Applications, special issue on knowledge representation, Fritz Lehman, editor, 23(2-5), 133-177, 1991.

[Wu, 2001] Wu, H., De Kort, E., De Bra, P. Design Issues for General-Purpose Adaptive Hypermedia Systems. Proceedings of the ACM Conference on Hypertext and Hypermedia, pp. 141-150, Aarhus, Denmark, August 2001.

[Yankelovich, 1988] Yankelovich N.; Haan, B. J.; Meyrowitz, M. K.; Drucker, S. M. Intermedia: The Concept and the Construction of a Seamless Information Environment. IEEE Journal, Janury, 1988.

[Ziegler, 1997] Ziegler, J. Interactive Techniques. VII Human-Computer Interaction. The Computer Science and Engineering HandBook. CRC Press, 1531-1550, 1997.