Escuela Politécnica Superior (Jaén)
Ingeniería en Informática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior (Jaén)

Proyecto Fin de Carrera

IMPLEMENTACIÓN DEL MÉTODO TOPSIS Y NUEVOS OPERADORES DE AGREGACIÓN PARA LA HERRAMIENTA "FLINSTONES"

Alumno: Francisco Javier Pulgar Rubio

Tutor: Dra. D.ª Macarena Espinilla Estévez
Dpto: Informática

Septiembre 2013

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Dra. Prof. Dª. Macarena Espinilla Estévez, tutora del Proyecto Fin de Carrera titulado: Implementación del método Topsis y nuevos Operadores de Agregación para la herramienta "Flinstones", que presenta D. Francisco Javier Pulgar Rubio, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, Septiembre 2013

El alumno: La tutora:

D. Francisco Javier Pulgar Rubio Dª. Macarena Espinilla Estévez

Tabla de contenido

Capítulo 1: Introducción 9
1.1 Introducción al proyecto 11
1.2 Propósito y objetivos 14
1.3 Resultados esperados 15
1.4 Plan de proyecto 16
1.5 Estructura de la memoria 20
Capítulo 2: Teoría de decisión 23
2.1 Introducción a la teoría de decisión 25
2.2 Problemas de toma de decisión 26
2.3 Clasificación de los problemas de Toma de Decisión 32
2.3.1 Dominios de expresión 35
2.3.2 Estructuras de representación de preferencias 40
2.4 Sistemas de Soporte a la Decisión 41
2.4.1 Definición y características 41
2.4.2 Componentes funcionales que integran un DSS 44
2.4.3 Uso de un Sistema de Soporte a la Decisión 45
2.5 Aplicación Informática de Apoyo a la Decisión. Flinstones 46
2.5.1 Características de Flinstones 46
2.5.2 Arquitectura de Flinstones 48
2.5.3 Estructura de la interfaz 50
2.5.3.1 Marco de evaluación 50
2.5.3.2 Recogida de información 51
2.5.3.3 Valoración de alternativas 52
2.5.3.4 Análisis 53
2.5.4 Diagrama de clases inicial 55
2.5.4.1 Clases relacionadas con los operadores de agregación 55
2.5.4.2 Pasos para crear un nuevo operador de agregación 57
2.5.4.3 Clases relacionadas con los métodos de resolución 58
2.5.4.4 Pasos para crear un nuevo método de decisión 60
Capítulo 3: Operadores de Agregación y Método de Decisión Topsis 63
3.1 Operadores de agregación 65
3.1.1 Importancia de la agregación 65
3.1.2 Operadores de agregación implementados 67
3.1.2.1 Media geométrica 67
3.1.2.2 Media geométrica pesada 67
3.1.2.3 Media harmónica 68
3.1.2.4 Media harmónica pesada 68
3.1.2.5 Power Median 68
3.1.2.6 Power Median pesada 68
3.1.2.7 Bonferroni means 69

3.1.2.8 Heronian Means ... 69
3.1.2.9 Media cuadrática ... 69
3.2 Método de decisión Topsis ... 71
3.2.1 Ejemplo del método de decisión Topsis ... 73
Capítulo 4: Análisis del sistema ... 77
4.1 Especificación de requerimientos ... 80
4.1.1 Requerimientos funcionales ... 80
4.1.2 Requerimientos no funcionales ... 83
4.2 Análisis del sistema ... 85
4.2.1 Perfiles de usuario ... 85
4.2.2 Casos de uso ... 87
4.2.3 Escenarios ... 106
4.2.4 Modelo del dominio ... 110
Capítulo 5: Diseño del software ... 115
5.1 Diagrama de clases de diseño ... 118
5.2 Diagramas de interacción (secuencia) ... 125
5.3 Diseño de la interfaz ... 128
5.3.1 Estilo ... 130
5.3.2 Metáforas ... 132
5.3.3 Diseño de las pantallas de la aplicación ... 135
5.3.3.1 Pantallas método Topsis ... 136
5.3.3.2 Cuadros de diálogo para los operadores de agregación. ... 142
5.3.4 Storyboards ... 146
5.3.5 Notificaciones y mensajes de error ... 150
5.3.5.1 Notificaciones. ... 150
5.3.5.2 Mensajes de error. ... 152
Capítulo 6: Implementación, Validación y Pruebas del software ... 155
6.1 Implementación ... 157
6.1.1 Lenguajes de programación ... 157
6.1.2 Herramientas de desarrollo ... 159
6.2 Validación y Pruebas del Software ... 159
6.2.1 Casos de test ... 160
6.2.2 Resultados de las pruebas de validación ... 165
Capítulo 7: Conclusiones y Trabajos Futuros ... 167
7.1 Conclusiones ... 169
7.2 Trabajos Futuros ... 171
Capítulo 8: Referencias Bibliográficas ... 173

Apéndice A: Manual de Usuario ... 179
Apéndice B: Manual de Instalación y Desarrollador ... 208
B.1 Manual de instalación ... 210
B.2 Manual de desarrollador ... 212

Capítulo 1:

Introducción

1.1 Introducción al proyecto

El problema de la toma de decisiones es un proceso habitual en la vida diaria de cualquier persona, es decir, es un proceso que llevamos a cabo constantemente para seleccionar la mejor alternativa entre un conjunto.

Para tomar una buena decisión es necesario seguir un proceso adecuado que necesita tiempo y planificación. El propósito principal consiste en seleccionar entre varias alternativas la/s más adecuada/s según los objetivos que se persiguen.

El primer paso a la hora de tomar una decisión consiste en fijar el objetivo del problema que se desea resolver y establecer su estructura donde se encuentran las alternativas planteadas y las características propias de cada una. Una vez hecho esto, es necesario recoger toda la información que posteriormente es agregada para poder seleccionar la alternativa que mejor se ajusta al objetivo concreto.

La toma de decisión es un proceso complejo, por ello es habitual realizar una división del mismo en distintas fases. Existen muchas versiones de esta división, pero en todas ellas podemos identificar fundamentalmente las siguientes etapas: [2]

- Identificar y analizar el problema.
- Identificar los criterios de decisión y ponderarlos.
- Definir la prioridad para atender el problema.
- Recoger información sobre el problema.
- Valorar las distintas opciones disponibles.
- Elección de la mejor opción según la evaluación.
- Aplicación de la decisión.
- Evaluación de los resultados obtenidos.

Como podemos ver, el proceso de toma de decisiones es un proceso complejo que requiere la realización de una serie de etapas orientadas a obtener la mejor alternativa entre las distintas opciones existentes.

Así mismo, la toma de decisiones es un área que se aplica a distintas disciplinas, como por ejemplo, tecnología, ingeniería, etc. [1,2] El hecho de que exista una gama tan amplia de campos de aplicación tiene como consecuencia la existencia de diferentes modelos de toma de decisión, que han dado lugar a la teoría de decisión.

Para ayudar a resolver los problemas de toma de decisión surgen herramientas como los Sistemas de Soporte a la Decisión (DSS) o aplicaciones informáticas de apoyo a la decisión. El objetivo fundamental de estas herramientas es ejercer de apoyo a las personas encargadas del proceso de toma de decisiones, este apoyo implica multitud de tareas como son: definición de la estructura del problema de decisión, recogida de información, evaluación y comparación de las alternativas, así como, cualquier otra tarea que constituya una ayuda para que el usuario pueda tomar la mejor decisión.

En este contexto se encuadra la herramienta "Flinstones" que se corresponde con una aplicación informática que desarrolla diferentes métodos de resolución de problemas de toma de decisión bajo incertidumbre y un conjunto de operadores para agregar la información recogida en proceso de resolución.

El objetivo de este proyecto es incorporar un nuevo método de decisión a la herramienta Flinstones y ampliar el conjunto operadores de agregación de los que dispone. Así, se diseñará y desarrollará el método de decisión Topsis [3] (*The Technique for Order of Preference by Similarity to Ideal Solution*), el cual está basado en que la alternativa escogida debe ser aquella con la distancia más corta a la solución ideal positiva y la distancia más lejana a la solución ideal negativa. Además, como se ha comentado, se incluirá nuevos operadores de agregación que podrán ser seleccionados por los distintos métodos de resolución que dispone la herramienta. Es de destacar que cada operador tendrá unas características propias que lo harán más adecuado para la resolución de determinados problemas, siendo responsabilidad del usuario el hecho de analizar las características del problema que desea solucionar y optar por el operador de agregación que más se adecúe a sus necesidades.

De esta forma, con la propuesta de este proyecto fin de carrera se complementa el DSS "Flinstones" con un nuevo método de decisión y nuevos operadores de agregación.

Desde un punto de vista personal he optado por realizar este proyecto debido a la gran difusión que en la actualidad tienen los sistemas de apoyo a la toma de decisiones. Además, el hecho de trabajar con una herramienta ya desarrollada supone que sea necesario un análisis de dicha herramienta, así como, adaptar de forma adecuada los operadores y el método para que todo funcione correctamente.

Por último, indicar que la metodología de programación utilizada para el desarrollo de la aplicación será JAVA, ya que es una metodología muy potente y estoy más familiarizado que con otras metodologías existentes. Además, existen multitud de manuales y referencias sobre dicha metodología, entre todas ellas destaca el manual "Thinking in Java" [4] que resulta de gran ayuda a la hora de programar utilizando el lenguaje Java.

1.2 Propósito y objetivos

Una vez vista la introducción en la que se ha explicado la motivación que ha llevado a la realización del proyecto, así como, una primera visión de las técnicas que se van a utilizar, se procede a concretar el propósito y los objetivos del proyecto realizado.

El propósito del proyecto consiste en desarrollar el método de decisión TOPSIS y nuevos operadores de agregación para la herramienta de resolución de problemas de toma de decisión en grupo multicriterio "Flinstones".

El desarrollo del método de decisión y de los operadores de agregación supondrá un aumento en la funcionalidad de la herramienta, ya que aportaran nuevos elementos para la resolución de problemas de decisión por medio de la herramienta "Flinstones".

Los objetivos que se tienen en este proyecto son los siguientes:

- Búsqueda de bibliografía sobre sistemas de apoyo a la toma de decisiones.
- Búsqueda de bibliografía sobre los operadores de agregación y sobre el método que se va a desarrollar, así como, sobre la herramienta para la cual se van a llevar a cabo.
- Desarrollar los distintos operadores de agregación. Estos operadores serán desarrollados para dos tipos distintos de información: numérica y lingüística, en concreto para 2-tuplas lingüísticas.
- Diseñar y desarrollar el método Topsis, para ello será necesario modificar la interfaz para adaptarla al método seleccionado.
- Testear la aplicación para ver que se cumplen los objetivos establecidos y no existen errores en la misma.
- Completar la memoria que documente el proyecto realizado.

1.3 Resultados esperados

A medida que se progrese en la elaboración del proyecto, es decir, a medida que se vayan cumpliendo los objetivos anteriormente definidos, se irá elaborando una serie de resultados relacionados con el proyecto realizado. A continuación se exponen los principales resultados que se esperan conseguir con la realización de este proyecto:

- Herramienta "Flinstones" ampliada con los nuevos elementos desarrollados. La herramienta mantendrá la funcionalidad anterior y añadirá los siguientes elementos:
    - Desarrollo del nuevo método de decisión Topsis.
    - Nuevos operadores de agregación implementados.

  Este es el elemento principal del proyecto y en torno al que girará el mismo, el resto de resultados se relaciona en mayor o menor medida con este.

- Informe que documente la información sobre los distintos operadores de agregación implementados durante el desarrollo del proyecto. Dicho informe irá en la memoria final del proyecto.

- Informe que documente la información sobre el método de decisión "Topsis" que se implementará durante el desarrollo del proyecto. Este informe incluirá la explicación del método, así como, las principales fases en las que se descompone.

- Manual de usuario donde se especifique de forma detallada cómo se maneja el software creado, así como, las distintas formas de uso que se puede tener del mismo. Dicho manual se realizará debido a que la aplicación estará orientada a usuarios de distinta formación, y debe proporcionar una guía para ayudar a los usuarios a manejar la aplicación y orientarle sobre las principales funcionalidades de la misma.

- Memoria que documente todo lo realizado a lo largo del proyecto. Esta memoria incluirá tanto los informes sobre los operadores de agregación y método anteriormente nombrados, como el manual de usuario creado. Así mismo, incluirá información necesaria para contextualizar de manera adecuada el proyecto desarrollado e información sobre el proceso de desarrollo llevado a cabo en el mismo.

1.4 Plan de proyecto

La planificación del proyecto se estructura en una serie de pasos que deben seguirse para tener claro que debemos hacer durante el transcurso del propio proyecto. Como apunte, decir que la fecha de inicio del mismo fue el día 25 de Febrero de 2013, y no se tendrán en cuenta los fines de semana.

En primer lugar, se debe llevar a cabo una división del trabajo. Esta división parte de los objetivos principales del proyecto descritos anteriormente los cuales se irán dividiendo a su vez en otros de menor importancia y más concretos. Finalmente, obtendremos tareas específicas, detalladas y bien separadas (se indicará la duración de las mismas al lado).

- Búsqueda de bibliografía sobre sistemas de apoyo a la toma de decisiones.
    - Búsqueda de bibliografía sobre sistemas de apoyo a la toma de decisiones (1 semana).
    - Revisión y análisis de la bibliografía sobre sistemas de apoyo a la toma de decisiones (1 semana).

- Búsqueda de bibliografía sobre los operadores de agregación y sobre el método que se va a desarrollar, así como, sobre la herramienta para la cual se van a llevar a cabo.
    - Búsqueda de bibliografía sobre los operadores de agregación y el método desarrollado (1 semana).
    - Revisión y análisis de la bibliografía sobre los operadores de agregación y el método desarrollado (1 semana).
    - Análisis de la herramienta utilizada "Flinstones" (2 semanas).

- Desarrollar los distintos operadores de agregación. Estos operadores serán desarrollados para tres tipos distintos de datos: numérico y variables lingüísticas (2 tuplas).
    - Análisis de los distintos operadores de agregación que se van a desarrollar (1 semana).
    - Desarrollo de los operadores para conjuntos numéricos (1 semana).
    - Desarrollo de los operadores para conjuntos cuyo tipo de dato es variables lingüísticas (1 semana).

- Diseñar y desarrollar el método Topsis, para ello será necesario modificar la interfaz para adaptarla al método seleccionado.
    - Análisis del método que se debe desarrollar (2 semanas).
    - Diseño y desarrollo del método Topsis (4 semanas).

- Testear la aplicación para ver que se cumplen los objetivos establecidos y no existen errores en la misma.
    - Diseño de las pruebas que se realizarán sobre el software que se desea analizar (1 semana).
    - Prueba del software a partir de las pruebas anteriormente realizadas (1 semana).
    - Análisis de los resultados obtenidos (1 semana).

- Finalmente se debe tener en cuenta que se debe completar la memoria del proyecto (4 semanas).
- La realización del proyecto supone un esfuerzo total de 22 semanas.

A continuación, a partir del conjunto de actividades anteriormente definidos, se realizará la red de actividades correspondiente al proyecto a realizar (se ha supuesto que el proyecto se comenzará el día 25 de Febrero de 2013):

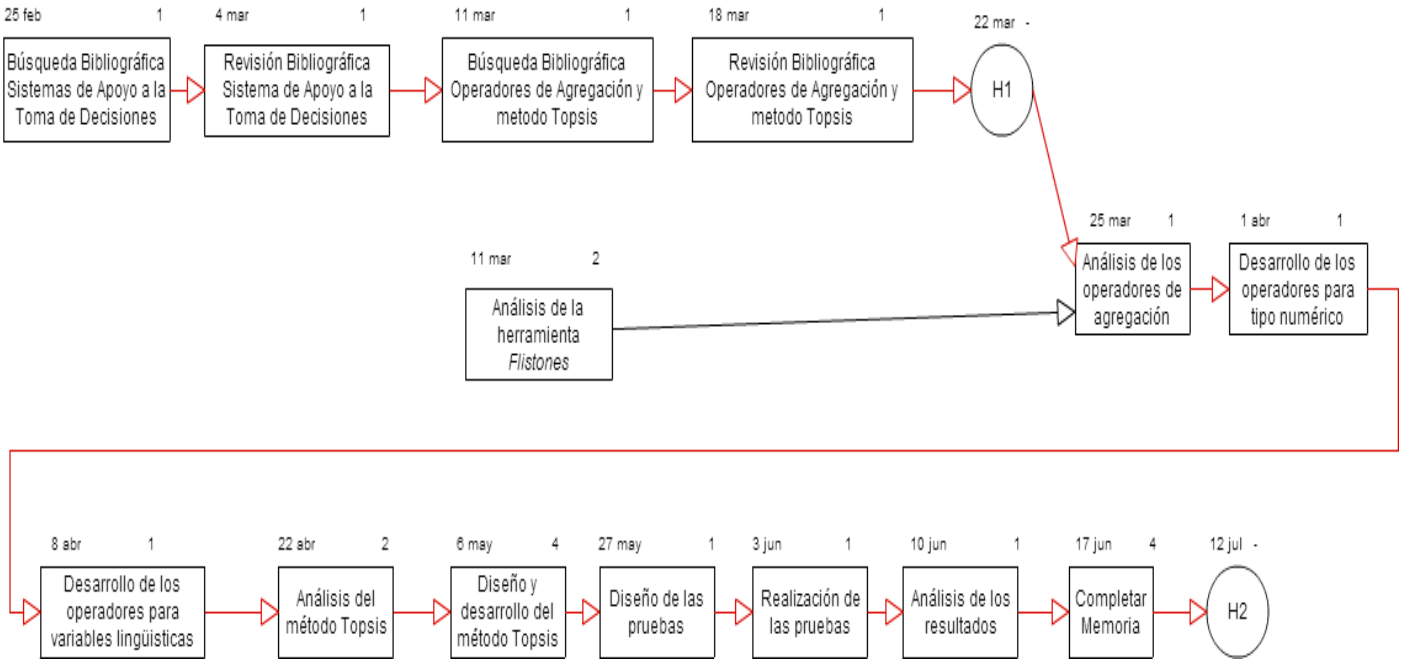


Ilustración 1.1 Red de Actividades.

Como se puede ver en la red de actividades anterior se ha detallado el camino crítico de color rojo y se han incluido dos hitos los cuales se detallan a continuación:

- Primer hito: Ocurre tras realizar la revisión y análisis de la distinta bibliografía encontrada.
- Segundo hito: Ocurre al finalizar el proceso, una vez se ha completado la memoria del mismo.

Finalmente, mostramos el diagrama de Gantt correspondiente a la planificación del proyecto que se va a llevar a cabo:

a) En primer lugar mostramos la tabla de las actividades así como su duración y las actividades que preceden a cada una:

Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
Búsqueda de Bibliografía (SATD)	1 sem	lun 25/02/13	vie 01/03/13	
Revisión y Análisis Bibliografía (SATD)	1 sem	lun 04/03/13	vie 08/03/13	1
Búsqueda de Bibliografía (Operadores de agregación y método Topsis)	1 sem	lun 11/03/13	vie 15/03/13	2
Revisión y Análisis Bibliografía (Operadores de Agregación y método Topsis)	1 sem	lun 18/03/13	vie 22/03/13	3
H1	0 días	vie 22/03/13	vie 22/03/13	4
Análisis de la herramienta "Flinstones"	2 sem.	lun 11/03/13	vie 22/03/13	
Análisis de los operadores de agregación	1 sem	lun 25/03/13	vie 29/03/13	5;6
Desarrollo de los operadores para tipo númerico	1 sem	lun 01/04/13	vie 05/04/13	7
Desarrollo de los operadores para valiables lingüisticas	1 sem	lun 08/04/13	vie 12/04/13	8
Análisis del método Topsis	2 sem.	lun 15/04/13	vie 26/04/13	9
Diseño y desarrollo del método Topsis	4 sem.	lun 29/04/13	vie 24/05/13	10
Diseño de pruebas	1 sem	lun 27/05/13	vie 31/05/13	11
Realización de pruebas de usabilidad	1 sem	lun 03/06/13	vie 07/06/13	12
Análisis de las pruebas realizadas	1 sem	lun 10/06/13	vie 14/06/13	13
Completar la memoria	4 sem.	lun 17/06/13	vie 12/07/13	14
H2	0 días	vie 12/07/13	vie 12/07/13	15

Tabla 1.1 Actividades del proyecto.

b) En segundo lugar mostramos una imagen correspondiente al diagrama de Gantt generado:

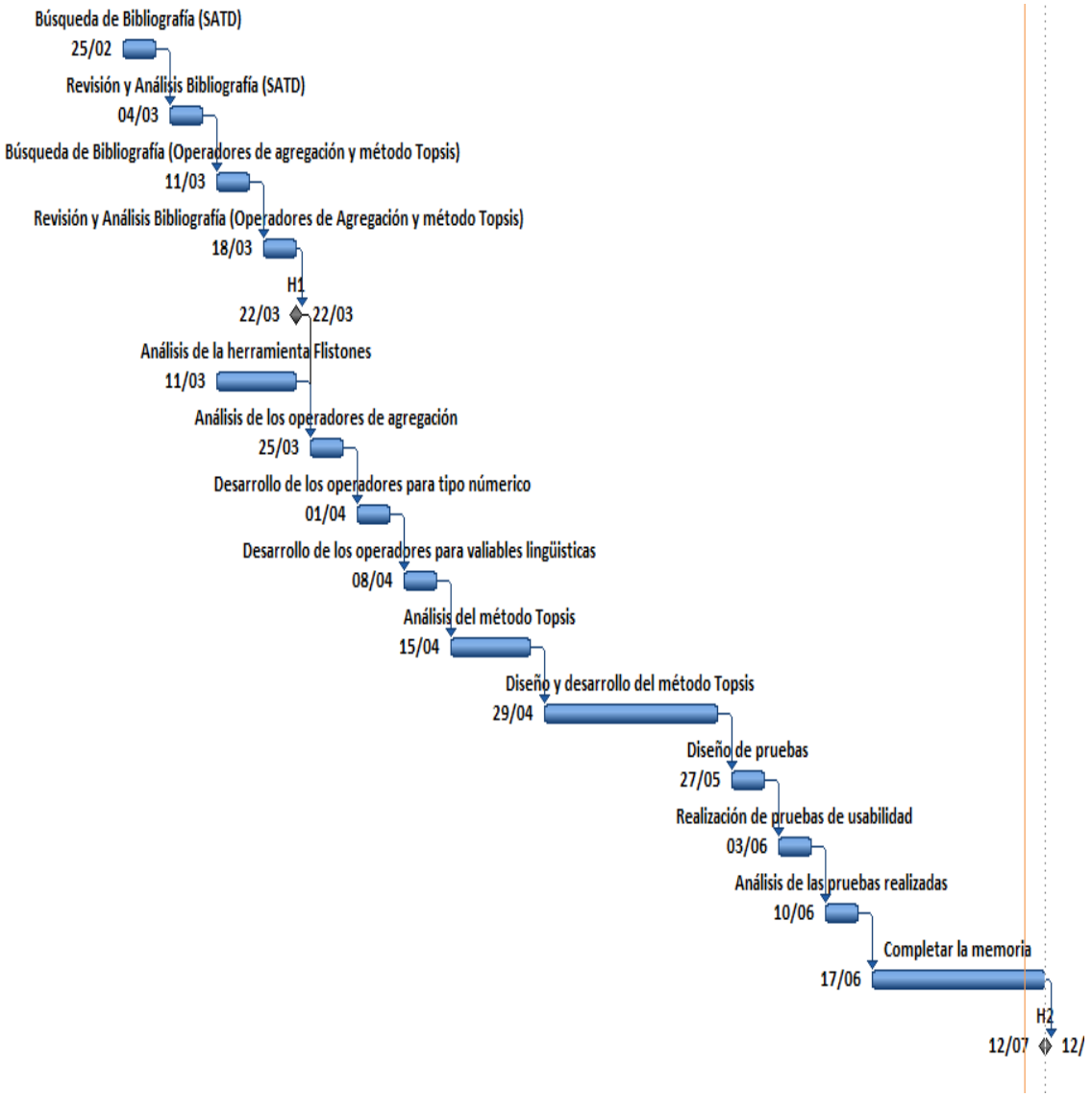


Ilustración 1.2 Diagrama de Gantt.

Se debe tener en cuenta que la planificación desarrollada puede sufrir modificaciones a lo largo del proyecto, ya que dicha planificación se ha realizado sin tener en cuenta los problemas que puedan surgir en la realización del mismo.

1.5 Estructura de la memoria

A continuación, se presenta una breve descripción de los diferentes capítulos en los que se estructura la memoria del proyecto realizado y los contenidos de los mismos.

Este capítulo, como se ha visto, establece una visión global del proyecto realizado, así como, lo que ha llevado a la realización del mismo, también se establecen el propósito y los objetivos que se llevarán a cabo, los resultados que se obtendrán y la planificación temporal del proyecto realizado.

En el segundo capítulo, se abordará el tema de la teoría de decisión. En primer lugar, ofrecerá una introducción donde se expondrán los problemas de toma de decisión realizando una clasificación de los mismo atendiendo a diversos criterios. A continuación, se realizará una definición de sistema soporte a la decisión indicando los objetivos de estos tipos de sistemas, así como, las fases en las que se estructura su funcionamiento. Finalmente, se analizará brevemente la herramienta "Flinstones" que se trata de la aplicación que será ampliada en el desarrollo de este proyecto, así como, las clases de la aplicación que deben ser modificadas para añadir nuevos métodos de decisión y operadores de agregación.

En el tercer capítulo se realizara un análisis de los operadores de agregación que se implementarán en el transcurso del proyecto, así como, el método "Topsis" que también será implementado, tanto los operadores como el método ampliarán la funcionalidad del a herramienta "Flisntones" anteriormente mencionada.

El cuarto capítulo supone uno de los ejes principales de la memoria realizada, al estar dedicado al proceso completo de desarrollo del proyecto. Como proyecto software que es, se hará en primer lugar un fugaz repaso a las diferentes etapas de la Ingeniería del Software, estas etapas se aplicarán en el desarrollo del sistema implementado. Así, se definen los requerimientos funcionales y no funcionales del sistema y se realizará la etapa de análisis, incluyendo el modelo de casos de uso.

El quinto capítulo tiene también una gran importancia pues está dedicado a la etapa de diseño del software, en el se incluirán el diagrama de clases de diseño ampliado desde la versión original de la aplicación y el diagrama de interacción. Así mismo, se incluirá también el diseño de la interfaz implementada.

El capítulo sexto de la memoria está dedicado a las tecnologías de implementación que se han utilizado durante el proyecto desarrollado. En dicho capítulo se realizará un análisis de la arquitectura de la aplicación, así como, de los lenguajes y herramientas de programación utilizados. Por otra parte, en este mismo capítulo se establecen las pruebas que se van a realizar sobre el software implementado para comprobar que cumple de manera adecuada con los requisitos funcionales previamente establecidos y que no existen fallos en dicho software.

El capítulo séptimo de la memoria está dedicado a las conclusiones generales que se han alcanzado tras desarrollar el proyecto, mientras que el capítulo octavo está dedicado a las referencias bibliográficas empleadas tanto para el proyecto como para el desarrollo de la memoria.

La sección final de la memoria contiene una serie de apéndices en el que se pueden encontrar los manuales de usuario para ayudar al manejo inicial de la aplicación.

Capítulo 2:

Teoría de Decisión

La teoría de decisión es el eje principal en torno al que gira el desarrollo de este proyecto, por tanto, se dedicará este segundo capítulo a dar una visión general de los problemas de toma de decisión y, posteriormente, se realizará un análisis de los sistemas de soporte a la decisión, así como, de la herramienta "Flinstones", a la cual se introducirán los nuevos operadores de agregación y el método Topsis. Finalmente, se explicarán tanto los operadores de agregación como el método que serán implementados.

2.1 Introducción a la teoría de decisión

El problema de la toma de decisiones es un proceso habitual en la vida diaria de cualquier persona, es decir, es un proceso que llevamos a cabo constantemente para seleccionar la mejor alternativa. El primer paso a la hora de tomar una decisión consiste en fijar el objetivo que se desea conseguir, una vez hecho esto, es necesario recoger toda la información que influya de alguna forma en el objetivo que se desea cumplir, así como, las características propias de cada una de las alternativas planteadas para, de esta manera, poder seleccionar la alternativa que mejor se ajusta a lo objetivos concretos de cada usuario.

La Toma de Decisiones es un área que se aplica a distintas disciplinas, como por ejemplo, tecnología, ingeniería, etc. El hecho de que exista una gama tan amplia de campos de aplicación tiene como consecuencia la existencia de diferentes modelos de toma de decisión, que han dado lugar a la teoría de decisión.

2.2 Problemas de toma de decisión

Los problemas de Toma de Decisión, como se ha comentado anteriormente, intentan seleccionar la alternativa que mejor se adapta a un usuario concreto y en un contexto determinado. Para ello, es necesario tener en cuenta una serie de elementos básicos que influyen en este tipo de problemas:

1- Conjunto de alternativas entre las que se desea elegir.
2- Conjunto de criterios diferentes que influyen en la resolución del problema planteado.
3- Conjunto de evaluaciones, cada una de ellas estará asociada a un par formado por una alternativa y un criterio del problema.
4- Función que determina las preferencias del experto sobre los posibles resultados obtenidos.

Además de los elementos que forman parte de un problema de Toma de Decisión se deben tener en cuenta los siguientes factores que influyen en este tipo de problemas:

a) Información: Es fundamental recoger información a la hora de solucionar cualquier tipo de problema. Dicha información guiará el proceso de solución para obtener la alternativa que mejor se adapte a las características del problema que se está solucionando.
b) Conocimientos: Se trata de los conocimientos de las personas que serán las encargadas de tomar la decisión. Estos conocimientos pueden ser de circunstancias relacionadas con el problema o de situaciones similares.
c) Experiencia: Se corresponde con la experiencia que el experto adquiere al solucionar diferentes problemas a los que se enfrenta a lo largo del tiempo. Dicha experiencia puede ser de mucha utilidad para solucionar problemas parecidos en el futuro.
d) Análisis: Se trata de analizar los diferentes elementos que influyen en el problema para obtener toda la información posible de ellos, con el objetivo de obtener la mejor solución para resolver el problema planteado.
e) Juicio: Es necesario saber elegir entre las diferentes alternativas presentadas para obtener la mejor solución posible, para ello, habrá que evaluar los diferentes aspectos que influyen en el problema.

Una vez expuestos los elementos básicos, así como, los factores que influyen en un problema de Toma de Decisión, se va a enumerar las etapas que forman parte del proceso de toma de decisión:

1- Identificar y analizar el problema.
2- Identificar los criterios de decisión y ponderarlos.
3- Recoger la información.
4- Valorar las distintas alternativas.
5- Selección de la mejor opción.
6- Aplicación de la decisión.
7- Evaluación de los resultados obtenidos.

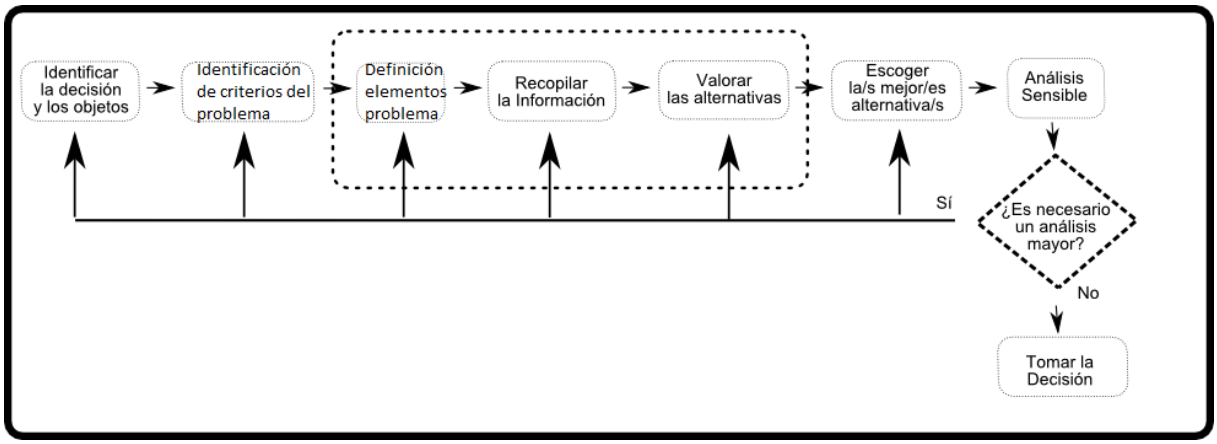


Ilustración 2.1Proceso de toma de decisión.

1- <u>Identificar y analizar el problema.</u>

En esta etapa se analiza el problema y se reconoce que es necesario tomar una decisión para poder solucionarlo. En la identificación del problema es imprescindible tener una visión clara y objetiva, además de saber escuchar las opiniones de los demás para poder formular una solución colectiva. Esta primera etapa es muy importante, pues constituye el punto de partida de toda decisión.

2- Identificar los criterios de decisión y ponderarlos.

Consiste en identificar aquellos elementos que influyen en la decisión que se va a tomar, es decir, aquellos aspectos de los cuales depende la decisión que se va a alcanzar.

En ocasiones es necesario realizar una ponderación de estos criterios, es decir, asignar un valor relativo a la importancia de dicho criterio en la decisión que se tome. Esto se hará cuando sea necesario acentuar la importancia de determinados criterios con respecto a otros.

Además es de destacar que las distintas alternativas son caracterizadas por el conjunto de criterios que pueden clasificarse en dos tipos:

a) Criterios cuantitativos: Son criterios que se pueden medir en términos numéricos, por ejemplo, el tiempo, los costes asociados, medidas o pesos, etc.
b) Criterios cualitativos: Estos criterios son difíciles de medir de forma numérica,, ya que se basan en conceptos vagos, imprecisos y/o subjetivos, por lo que se suelen medir a través de términos lingüísticos. Ejemplos de estos factores son calidad o servicio.

Muchas veces, la identificación de los criterios se hace de forma inconsciente, ya que las decisiones se alcanzan sin explicitarlos pero teniéndolos en cuenta debido a la experiencia personal de las personas dedicadas a tomar decisiones. Aun así, siempre es importante explicitarlos para evitar que se manipulen los criterios para favorecer una u otra opción por encima de las demás.

3- Recoger la información del problema.

En esta etapa se debe recolectar la información del problema, es decir, se debe analizar los diferentes elementos que influyen en la resolución del problema que se está llevando a cabo para extraer la máxima cantidad de información que permita obtener una solución lo más precisa posible.

4- Valoración de las distintas alternativas.

En esta etapa se computará para cada una de las alternativas un valor que resume la información recogida sobre ella y determina lo adecuadas que serán cada una de ellas para solucionar el problema. Dicho valor resume la información recogida en el anterior paso, es decir, mira sus ventajas y desventajas con respecto a los criterios de decisión.

En determinados problemas, se deben considerar también las consecuencias de su aplicación, ya que la decisión a tomar puede influir en el porvenir y se debe determinar si esta será una ayuda o un obstáculo en el futuro.

5- Selección de la mejor opción.

En este paso se escoge la opción que mejores resultados aportaría al problema tratado según la evaluación previamente asignada. Existen diferentes técnicas que permiten valorar los resultados de las evaluaciones atendiendo a múltiples criterios.

A la hora de seleccionar la mejor opción se pueden emplear diferentes alternativas según la precisión o rapidez del resultado que se desea aportar. En este sentido podemos optar por:

a) Maximizar: En este caso, se opta por la mejor decisión posible entre todas las alternativas aportadas.
b) Satisfacer: Otra opción sería elegir la primera opción que sea mínimamente aceptable, siempre que satisfaga la meta u objetivo planteado.
c) Optimizar: En este último caso, se seleccionaría la opción que mejor equilibrio aportara entre los diferentes objetivos que se persiguen.

6- Aplicación de la decisión.

El proceso no acaba una vez se toma la mejor decisión para solucionar el problema, ya que dicha decisión debe ser aplicada, es decir, puesta en práctica. La puesta en marcha de la solución

puede ser llevada a cabo por las mismas personas que han llegado a ella o, como en otros casos, ambas actividades son llevadas por grupos de personas distintos, ya que en muchas ocasiones las decisiones viene de niveles superiores en la jerarquía de la organización. Sea de un modo u otro, siempre debe existir la comprensión total de la decisión tomada, así como, las razones que han llevado a esa conclusión para que la implementación sea lo más exitosa posible. Por ello, siempre es recomendable que, aunque los responsables que toman la decisión sean distintos que los que la pongan en práctica, se impliquen a todos los integrantes desde el inicio del proceso.

En el proceso de aplicación de la decisión tomada es probable que surjan ciertos problemas, por tanto, las personas implicadas en la puesta en marcha deben estar preparados para dichos inconvenientes, reconocer cuando se produzcan, intentar predecir cuándo ocurrirán y solucionarlos de la forma más eficaz posible.

7- Evaluación de los resultados obtenidos.

Una vez se ha realizado la aplicación de la decisión que se ha alcanzado en las etapas previas, se debe realizar una evaluación de los resultados que se han obtenido. Para ello, es necesario recopilar todos los datos que aportan información acerca de cómo funciona la decisión que se ha tomado, es decir, un proceso de retroalimentación. Tendremos fundamentalmente dos tipos de retroalimentación:

a) Retroalimentación positiva: Este tipo indica que la decisión alcanzada aporta buenos resultados, por tanto, se puede continuar sin problemas e incluso aplicar la misma solución cuando surjan en el futuro problemas similares al planteado.
b) Retroalimentación negativa: Este tipo de retroalimentación puede deberse a que no ha pasado tiempo suficiente desde que se ha implantado la solución o no se han asignado los recursos necesarios para que aporte buenos resultados. También podría deberse a que a pesar de asignar tiempo y recursos adecuados la solución que se dio no fue la correcta, por lo que se debería volver a redefinir el problema y obtener una mejor solución. Este segundo caso es más peligroso y más difícil de solucionar, ya que se debe comenzar todo el proceso desde el principio, aunque se dispondrá de nueva información adicional que puede ser de utilidad.

En este proceso de Toma de Decisión influyen una serie de procesos cognitivos que se deben tener en cuenta a la hora de analizar este tipo de problemas. A continuación se enumeran estos procesos:

1. Observación: Estudiar los distintos objetos o elementos que intervienen en el problema de la forma más precisa posible.
2. Comparación: Fijar la atención en dos o más objetos para descubrir sus relaciones o estimar sus semejanzas o diferencias.
3. Codificación: Transformar mediante las reglas de un código la formulación de un determinado mensaje.
4. Organización: Evaluar las distintas opciones, para determinar las fases necesarias y el curso de acción más responsable.
5. Clasificación: Ordenar atendiendo a algún tipo de criterio un conjunto de objetos en clases o categorías.
6. Resolución: Implementación de las decisiones tomadas, en esta parte se demuestran los resultados obtenidos.
7. Evaluación: Análisis de los resultados obtenidos con el proceso de toma de decisiones alcanzando las soluciones más adecuadas.
8. Retroalimentación: Una vez evaluados los resultados obtenidos, se deben analizar y determinar las conclusiones alcanzadas y optar por la mejor decisión de cara al futuro.

2.3 Clasificación de los problemas de Toma de Decisión

Los problemas de Toma de Decisión se clasifican atendiendo a diferentes criterios. En este caso se van a dividir según el tipo de problema que se plantea, es decir, según el número de expertos y criterios que intervienes; y según el ambiente en el que se toma la decisión.

a. Según el número de expertos

Los problemas de toma de decisión en los que sólo participa un experto se conocen como "Toma de Decisiones Individual", mientas que en los casos en los que participan varios expertos hablamos de "Toma de Decisiones Multiexperto" o en "grupo".

En los casos en los que sólo tenemos un experto, este será el encargado de valorar las distintas alternativas de acuerdo a sus preferencias. En este caso, se presenta el inconveniente que la solución obtenida proviene de la información de un único experto, por lo que no tendremos diferentes opiniones para valorar las diferentes alternativas propuestas.

En los casos donde tenemos múltiples expertos, se obtendrá una conjunto de valoraciones de cada uno de ellos para las alternativas que se van a evaluar. Una vez se obtengan todos los resultados, se deberán integrar para formar un resultado global, para ello se empleará el proceso de agregación. Este tipo de problemas tiene las ventajas de que la información suministrada es más completa, incrementa la aceptación de la solución y reduce los problemas de comunicación. En cambio, requiere mucho más tiempo y puede llevar a algunos miembros del grupo a eximirse de la responsabilidad asociada a la tarea.

b. Según el número de criterios.

Según el número de criterios que influyen en el problema de toma de decisión podemos hablar de: "optimización simple" cuando las alternativas se optimizan atendiendo a un único criterio o "optimización múltiple" cuando se realiza atendiendo a varios criterios, en este último caso hablamos de Toma de Decisiones Multicriterio.

En los problemas en los que sólo disponemos de un criterio se realizará la evaluación de las distintas alternativas atendiendo a este único criterio, lo que implica que la decisión final está ligada a dicho criterio únicamente.

En los problemas de decisión multicriterio, las alternativas se evalúan de acuerdo a un conjunto de criterios. Cada uno de los criterios establece un orden entre las alternativas, por tanto, es necesario realizar una agregación de los distintos criterios para obtener un orden global. En dicha agregación, se puede asignar distinta importancia a los criterios que intervienen mediante diferentes tipos de ponderaciones. Es similar al caso de problemas de decisión multiexperto, pero mientras que en este caso las valoraciones representan la importancia de los diferentes criterios, en el caso anterior, las valoraciones representan las preferencias de distintas personas.

c. Ambiente.

El ambiente, la situación o el contexto en los que se toman las decisiones tienen una gran influencia en el desarrollo del proceso, así como, la obtención de la solución final. Este aspecto se puede clasificar según el conocimiento y control que se tenga sobre las variables que intervienen o influyen en el problema, ya que dichas variables influirán en la solución que se obtendrá. De esta forma podemos hablar de:

- Ambiente de certeza:

  Cuando hablamos de ambiente de certeza o preciso nos referimos a un contexto en el que se tiene conocimiento total del problema y las opciones de solución que se tienen van a causar siempre resultados conocidos e invariables. Una vez se tome la decisión sólo se debe pensar en la opción que aporte un mayor beneficio.

  La información que se maneja en este tipo de problemas es completa, es decir, se conoce el problema, las posibles soluciones, pero no los resultados que dichas soluciones pueden arrojar.

  En este contexto es adecuado hablar de variables de tipo numérico para las evaluaciones aportadas por los distintos expertos para los diferentes criterios. Este tipo de valores ofrecen

una información precisa y exacta acerca de la opinión que tiene un determinado experto de un criterio concreto, además permite la realización de operaciones aritméticas entre los distintos valores sin necesidad de realizar ningún tipo de transformación.

- Ambiente de incertidumbre:

Un problema de decisión está definido en un ambiente de incertidumbre cuando la información disponible sobre las distintas alternativas puede ser incompleta, vaga o imprecisa, lo que implica que la utilidad asignada a cada alternativa tenga que ser valorada de forma cualitativa. Esta incertidumbre surge a raíz del intento de modelar la imprecisión propia del comportamiento humano o la inherente a ciertos fenómenos que por su naturaleza son inciertos (imprecisos).

En este contexto es adecuado hablar de variables lingüísticas para las evaluaciones de las diferentes alternativas aportadas por los distintos expertos para el conjunto de criterios. Una variable lingüística, como su nombre lo sugiere, es una variable cuyos valores son palabras o sentencias del lenguaje natural o sintético. Por ejemplo, la velocidad de un coche sería una variable lingüística, siendo sus posibles valores, por ejemplo, "alta", "media" y "baja".

Entre los distintos enfoques y herramientas para superar las limitaciones del tratamiento de esta incertidumbre se encuentran la lógica difusa y el enfoque lingüístico difuso, ya que el uso de información lingüística mejora la fiabilidad y flexibilidad de los modelos clásicos de decisión.

2.3.1 Dominios de expresión

Anteriormente, se han explicado los diferentes tipos de ambientes que se pueden dar en los problemas de toma de decisión y se ha establecido su relación con los diferentes dominios de expresión. A continuación se exponen con más detalle dichos dominios de expresión.

Dentro del contexto de los problemas de decisión se entiende por dominio de expresión al conjunto de valores utilizado por los diferentes expertos para realizar las distintas valoraciones, es decir, para expresar sus preferencias. La elección del dominio adecuado al problema que se desea resolver puede deberse a varios motivos: [20]

1- *Naturaleza cuantitativa o cualitativa de la información:* La naturaleza del atributo puede condicionar el dominio utilizado. De esta forma, dominios numéricos se adecuan más a atributos cuantitativos, mientras que para atributos cualitativos es más recomendable usar dominios lingüísticos.

2- *Pertenencia de los expertos a diferentes áreas de conocimiento:* Los expertos tenderán a utilizar el dominio de información que más se adapte a su área de trabajo, ya que será el que maneje mejor y con el que estará más familiarizado.

3- *Expertos con diferente grado de conocimiento sobre el problema:* La experiencia de los expertos en resolución de problemas similares puede llevar a elegir determinados dominios que le hayan aportado mejores soluciones en el pasado o descartar otros que le hubiesen proporcionado malos resultados.

La elección del domino de expresión adecuado es muy importante y puede ser un factor diferencial a la hora de obtener buenos resultados en la resolución de los problemas de toma de decisión. Los dominios de expresión que se utilizan de forma más habitual son los siguientes:

1- *Dominio Numérico* [21].
2- *Dominio Lingüístico* [22].
3- *Dominio Intervalar* [23].

Tanto los nuevos operadores de agregación como el método "Topsis" implementados en el desarrollo de la práctica están desarrollados para trabajar con dominios de expresión numéricos y lingüísticos, en concreto la variante denominada 2-tuplas. Por ello, se explican con más detalle dichos dominios.

- Dominio numérico: El dominio de expresión de tipo numérico aporta un ambiente preciso, es decir, la información proporcionada es más exacta y aporta un conocimiento más completo del problema que se intenta solucionar. Por todo ello, este tipo de dominios de expresión se adapta más a atributos cuantitativos que aportan valores exactos y precisos.

- Dominio lingüístico: La utilización de un dominio de expresión de tipo lingüístico viene motivada por la presencia de un ambiente de incertidumbre, ya que la información proporcionada no es exacta, siendo el conocimiento del problema difuso e incierto. Como se ha comentado, el enfoque lingüístico difuso, que tiene como base teórica la Teoría de Conjuntos Difusos, se ha mostrado como una técnica eficaz en este contexto. Para ello, representa los aspectos cualitativos como variables lingüísticas, cuyo dominio de expresión son conjuntos de palabras o términos lingüísticos.

  Una variable lingüística se caracteriza por un valor sintáctico o etiqueta y por un valor semántico o significado. La etiqueta es una palabra o frase perteneciente a un conjunto de términos lingüísticos y el significado de dicha etiqueta viene dado por un subconjunto difuso en un universo de discurso.

  Este tipo de dominios de expresión se adapta sobre todo a atributos cualitativos, cuyos valores se asocian con un conjunto difuso y no con un valor exacto como es el caso de los valores numéricos. Este tipo de dominios produce cierta pérdida de información, lo que provoca una falta de precisión en los resultados aportados. Para reducir dicha pérdida de información y aumentar la precisión, se emplea el modelo de representación y computación basado en 2-tuplas lingüísticas, el cual se describe a continuación.

- El modelo de representación basado en 2-tuplas: Permite realizar cómputos con palabras de forma sencilla y sin pérdida de información. A continuación, se presenta el modelo de representación lingüística basado en 2-tuplas: [24]

Considerando que S = $\{s_0,....,s_g\}$ es un conjunto de términos lingüísticos con cardinalidad impar, donde el término intermedio representa una valoración de aproximadamente 0.5 y con el resto de términos distribuidos simétricamente alrededor de ese punto intermedio; asumiendo que en la escala sobre la que están distribuidos todos los términos hay definida una relación de orden total. En este contexto, se puede obtener un valor β ε [0,g] que mediante una función de aproximación exprese el resultado obtenido como un valor de S.

Definición 2.1: Sea β el resultado de una agregación de los índices de un conjunto de etiquetas valoradas sobre un conjunto de términos lingüísticos S, es decir, el resultado de una operación de agregación simbólica, β ε [0,g]. Dados i = round (β) y α = β – i , dos valores, tales que, i ε [0,g] y α ε [-0.5, 0.5) entonces α es lo que se denomina *Traslación Simbólica,* que expresa la diferencia entre la información expresada por β y la etiqueta lingüística más cercana del conjunto S.

De esta forma, el enfoque lingüístico basado en 2-tuplas representa la información lingüística por medio de 2-tuplas:

(s_i, α_i), s_i ε S y α_i ε [-0.5,0.5)

Donde, s_i representa la etiqueta lingüística y α_i es un valor numérico que expresa la traslación simbólica de β al índice de la etiqueta más cercana, i, en el conjunto de términos lingüísticos S.

Este modelo define un conjunto de funciones que permiten realizar la transformación entre valores numéricos y 2-tuplas.

Definición 2.2: Sea S_i ε S un término lingüístico, su representación mediante una 2-tupla se obtiene mediante la función Θ:

$$\theta : [0, g] \longrightarrow \mathcal{S} \times [-0.5, 0.5)$$

$$\theta(s_i) = (s_i, 0) / s_i \in \mathcal{S}$$

Ecuación 2.1. Representación 2-tuplas

Definición 2.3: Siendo S = {s_0,…,s_g} un conjunto de términos lingüísticos y β ε [0,g] un valor que representa el resultado de una operación de agregación simbólica, la 2-tupla que expresa la información equivalente a β se obtiene mediante la función:

$$\Delta : [0,g] \longrightarrow S \times [-0.5, 0.5)$$

$$\Delta(\beta) = (s_i, \alpha), \ with \begin{cases} s_i & i = round(\beta) \\ \alpha = \beta - i & \alpha \in [-0.5, 0.5) \end{cases}$$

Ecuación 2.2. Transformación valor numérico a 2-tuplas.

Donde round() es un operador típico de redondeo, s_i es la etiqueta cuyo índice es el más cercano a β y α es el valor de la traslación simbólica.

Definición 2.4: Siendo S = {s0,…,sg} un conjunto de términos lingüísticos y (s_i, α) una 2-tupla. Se define la función Δ^{-1}, tal que aplicada sobre una 2-tupla (s_i, α) devuelve su valor numérico β ε [0,g].

$$\Delta^{-1} : S \times [-0.5, 0.5) \longrightarrow [0, g]$$

$$\Delta^{-1}(s_i, \alpha) = i + \alpha = \beta$$

Ecuación 2.3. Transformación 2-tuplas a valor numérico.

Una vez definidas las funciones para transformar valores numéricos a 2-tuplas y viceversa, se pasa a presentar el modelo computacional que permite operar sobre la representación lingüística de las 2-tuplas, basándonos en los operadores de comparación, negación y agregación de 2-tuplas:

1- Operador de comparación de 2-tuplas: La comparación de información lingüística representada por medio de 2-tuplas se realiza de acuerdo a un orden lexicográfico normal. Considerando dos 2-tuplas (s_k, α_1) y (s_1, α_2) que representan cálculos de información:

- Si k < 1 entonces (s_k, α_1) es menor que (s_1, α_2).
- Si k = 1 entonces:

- Si $\alpha_1 = \alpha_2$ entonces (s_k, α_1) y (s_1, α_2) representan la misma información.
- Si $\alpha_1 < \alpha_2$ entonces (s_k, α_1) es menor que (s_1, α_2).
- Si $\alpha_1 > \alpha_2$ entonces (s_k, α_1) es mayor que (s_1, α_2).

2- Operador de negación de 2-tuplas: El operador de negación sobre una 2-tupla se define como:

$$Neg((s_i, \alpha)) = \Delta(g - (\Delta^{-1}(s_i, \alpha))).$$

Ecuación 2.4.Operador de negación.

Siendo g+1 la cardinalidad del conjunto de etiquetas S.

3- Operador de agregación de 2-tuplas: La agregación de información consiste en obtener un valor que resuma un conjunto de valores, por lo que el resultado de agregar un conjunto de 2-tuplas debe ser una 2-tupla. Existen multitud de operadores de agregación que permiten combinar información de acuerdo a diferentes criterios, cualquiera de ellos puede ser extendido a 2-tuplas empleando las funciones para transformar valores numéricos en 2-tuplas y viceversa sin pérdida de información. En el apartado 3.1 relativo a los operadores de agregación se pueden ver ejemplos de algunos de estos operadores.

2.3.2 Estructuras de representación de preferencias

Una vez explicados los dominios de expresión que se han tenido en cuenta en el desarrollo del proyecto, se va a revisar brevemente las estructuras más comunes utilizadas en los problemas de decisión para representar las preferencias de los distintos expertos:

1- *Órdenes de preferencia:* Este formato de representación de preferencias establece un ranking u orden de alternativas que representa la idoneidad de cada una de ellas como solución al problema de decisión según el punto de vista de cada experto, es decir, cada experto proporciona un vector de alternativas ordenándolas de mejor a peor.

2- *Vectores de preferencia:* Este forma ha sido ampliamente utilizado. En este caso, las preferencias de un experto sobre un conjunto de alternativas se describe como un conjunto de valores de utilidad dentro de un dominio D. Este formato de representación es más fino que el anterior, pero es necesario que cada experto valore de manera global cada alternativa con respecto a las demás.

   La aplicación "Flinstones" en la cual se centra la realización de este proyecto emplea este tipo de estructura para expresar las preferencias de cada experto, ya que es la estructura más adecuada para almacenar la información proporcionada por cada experto.

3- *Relaciones de preferencia:* Este formato de representación establece relaciones binarias entre las distintas alternativas de la forma: $x_1 \, R \, x_2 \leftrightarrow$ *"x_1 no es peor que x_2".*

   Está definición considera una relación binaria como una relación de preferencia débil e implica que dicha relación es reflexiva, dicha relación refleja en cierto sentido el grado de preferencia de una alternativa sobre otra cualquiera.

2.4 Sistemas de Soporte a la Decisión

Una vez dada una visión general de la Teoría de Decisión, así como, de un análisis detallado de los problemas de decisión y los distintos tipos que existen, el siguiente apartado se centrará en explicar en qué consisten los Sistemas de Soporte a la Decisión, ya que en la realización del proyecto consiste en aumentar la funcionalidad de una herramienta de este tipo.

2.4.1 Definición y características

El concepto de Sistema de Soporte a la Decisión (DSS) es muy amplio, debido a que existen diferentes enfoques para la toma de decisiones y debido a la extensa gama de áreas en las que se aplica. En general, podemos decir que un DSS es un sistema informático utilizado para ayudar en el proceso de toma de decisiones, siendo la decisión una elección entre alternativas basándose en los valores de dichas alternativas. El apoyo implica ayudar a las personas encargadas de la tarea de tomar la decisión a reunir la información necesaria, generar las alternativas posibles y tomar la decisión más adecuada, además, de apoyo a la hora de estimar, evaluar y comparar las distintas alternativas. [5,6]

En un sentido más amplio, se define este tipo de sistemas como un conjunto de programas y herramientas que permiten obtener de manera oportuna la información que se requiere para el proceso de toma de decisiones que se desarrolla en un ambiente de incertidumbre. Ayudan a la toma de decisiones al combinar datos, modelos analíticos sofisticas y software amigable en un solo sistema poderoso que puede dar soporte a la toma de decisiones semiestructuradas como no estructuradas.

La finalidad del Sistema de Soporte a la Decisión es apoyar a la toma de decisiones mediante la generación y evaluación sistemática de diferentes alternativas o escenarios de decisión, todo esto utilizando modelos y herramientas computacionales. Un DSS no soluciona los problemas por sí mismo, sino que sirve de apoyo al proceso de toma de decisión, la responsabilidad de tomar la decisión corresponde a los administradores del sistema.

Los DSS son herramientas software poderosas que permiten a personas de diferentes niveles de la empresa, agrupar los datos necesarios y procesarlos de diferentes formas para simular una situación de la realidad. Esto permite a dichas personas tomar decisiones más acertadas debido a que disponen de una información mucho más amplia y precisa, además de disponer de escenarios más variados.

Una estrategia simple para desarrollar, implementar y poner en marcha un Sistema de Soporte a la Decisión estaría formada por los siguientes pasos:

- Definir los objetivos que se persiguen con la puesta en marcha del DSS.
- Indicar las principales variables que afectan al objetivo previamente definido.
- Definir las fuentes desde donde se van a obtener los datos de las variables de entrada.
- Definir el modelo o los modelos utilizados para procesar las variables de entrada.
- Definir los análisis en los que se modificarán las variables de entrada para ver cómo se comporta o cambia el objetivo para alcanzar la mejor solución.

Algunas áreas de aplicación típicas de los DSS son: la gestión y planificación de negocios, atención sanitaria, militar, y cualquier área de gestión en el que sea necesario tomar decisiones complejas entre un variado abanico de posibilidades. Se usan principalmente para decisiones estratégicas y tácticas en la gestión de nivel superior, donde la frecuencia de la toma de decisiones es baja, pero las consecuencias de las mismas pueden ser muy altas.

Una vez vista la definición de Sistemas de Apoyo a la Decisión (DSS) se van exponer las principales características de este tipo de sistemas, aunque alguna de estas características se pueden extraer de la definición previamente analizada. Las principales características de los DSS son [5,6]:

a) Interactividad: Se trata de un sistema computacional que ofrece la posibilidad de interactuar de forma amigable y con respuestas en tiempo real con el usuario del sistema o encargado de tomar las decisiones.
b) Tipo de decisiones: Apoya el proceso de toma de decisiones tanto estructuras como no estructuradas.

c) Frecuencia de uso: Este tipo de sistemas son utilizados de forma frecuente por parte de la administración media y alta para el desempeño de su función.
d) Variedad de usuarios: Los usuarios que emplean este tipo de sistemas son muy variados y pertenecientes a distintas áreas, como por ejemplo, ventas, producción o recursos humanos.
e) Flexibilidad: El sistema se puede acoplar a una variedad determinada de estilos administrativos.
f) Desarrollo: Permite que los usuarios desarrollen de manera directa nuevos modelos de decisión.
g) Interacción ambiental: Permite la posibilidad de interactuar tanto con información interna como con información externa.
h) Comunicación Inter-Organizacional: Facilita la comunicación de información entre los diferentes niveles de la organización.
i) Acceso a base de datos: Permite acceder a información de las bases de datos corporativas.
j) Simplicidad: Simple y fácil de aprende y utilizar por parte de los usuarios finales.
k) Adaptabilidad: Permite adecuar el modelo a cambios en el entorno o en el contexto del problema que se desea solucionar.

2.4.2 Componentes funcionales que integran un DSS

Una de las principales características que ofrecen los DSS es la facilidad que un usuario pueda desarrollar sus propios modelos de decisión. Estos modelos son construidos con ayuda de herramientas tanto software como hardware. Las herramientas hardware están constituidas por todos los elementos de hardware que forman porte del DSS. [5,6]

Las herramientas software son aquellas que permiten al usuario generar sus propias aplicaciones, manipular su información e interactuar con el DSS, dentro de este tipo podemos distinguir:

- Modelo: Se trata de una representación abstracta en donde se ilustran los componentes y relaciones de un determinado fenómeno.
- Base de datos: Se corresponde con una colección de datos actuales o históricos de un número de aplicaciones o grupos, organizada para un acceso fácil a partir de un conjunto de aplicaciones. Además, los DSS permiten manejar y almacenar este tipo de información.
- Sistema de software: Permite una interacción fácil entre los usuarios del sistema, la base de datos del DSS y la base de modelos disponible.
- Interfaz con el usuario: Se trata de la parte fundamental del DSS ya que permite al usuario explorar la información e interactuar con el sistema de forma directa, introduciendo los valores del problema que desea solucionar y obteniendo los resultados que el sistema proporciona atendiendo a dichos valores.

2.4.3 Uso de un Sistema de Soporte a la Decisión

En el uso de un Sistema de Soporte a la Decisión interviene un proceso iterativo. En este proceso se pueden considerar fundamentalmente cinco etapas:

1- Se examina el problema y se elabora una formulación que permita el estudio del problema.
2- Se identifican las principales variables y parámetros que intervienen en el mismo, fundamentales para llegar a la conclusión.
3- Se construye un modelo interrelacionado los parámetros y las variables según el contexto del problema.
4- Identificar y proporcionar los datos para las variables y realizar los cálculos necesarios para obtener los resultados.
5- En último lugar es necesario realizar una etapa de afinación del problema, en esta etapa se analizan los resultados y se ajustan las variables, eliminándolas o agregando nuevas, para ajustar el resultado obtenido al contexto real del problema.

2.5 Aplicación Informática de Apoyo a la Decisión. Flinstones

Una vez se ha realizado una introducción a la Teoría de Decisión en la que se ha expuesto las características de los problemas de decisión, así como, los tipos que nos podemos encontrar y se ha introducido el concepto de Sistema de Soporte a la Decisión, se pasa a presentar la aplicación informática de apoyo a la decisión denominada "Flinstones". Dicha aplicación será ampliada en la realización del proyecto introduciendo nuevos operadores de agregación, así como, un nuevo método de decisión denominado "Topsis".

2.5.1 Características de Flinstones

La aplicación "Flinstones" [7] es una herramienta de apoyo a la toma de decisión que surgió motivada por la necesidad de resolver problemas de Toma de Decisión en ambientes de incertidumbre, utilizando métodos basados en el modelo lingüístico 2-tupla. Para la resolución de problemas de Toma de Decisión a través de la aplicación son necesarios una serie de pasos que conducen a obtener la alternativa que se adapta mejor a las necesidades del usuario y al contexto del problema. Estos pasos son:

1- Marco del problema: En el que se definen todas las características y la estructura del problema.
2- Recogida de información: Donde se recoge la información de los expertos considerando en todo momento el marco de evaluación.
3- Valoración de las alternativas: Se obtiene una valoración para cada una de las alternativas, para ello será necesario establecer el método de resolución y los operadores de agregación que se van a utilizar.
4- Análisis de los resultados: Finalmente, una vez obtenidas las valoraciones de las distintas alternativas se deben analizar los resultados y volver a redefinir, si es necesario, los parámetros del problema para obtener unos resultados más adecuados.

En la siguiente imagen se puede observar de forma más esquemática en qué parte del proceso de Toma de Decisión visto en el apartado 2.2 se centra la aplicación "Flinstones" para apoyar a los usuarios a resolver este tipo de problemas.

Ilustración 2.2 Etapas de resolución del problema.

El esquema de resolución de problemas anteriormente indicado es el que sigue la aplicación a la hora de resolver los problemas de Toma de Decisión. Por ello, la aplicación estructura tanto el código como la interfaz para dejar claro los distintos pasos que se deben llevar a cabo y facilitar así la labor del usuario. En los apartados 2.5.2 y 2.5.3 se explicará de forma más detallada la arquitectura y la estructura de la interfaz respectivamente.

Una vez visto el esquema de resolución seguido por la aplicación para resolver los problemas de toma de decisión, se explicarán las características de los problemas de Toma de Decisión para los que la aplicación "Flinstones" aporta soluciones. Las principales características son:

- Problemas de decisión multicriterio: La aplicación permite resolución de problemas que requieren múltiples criterios para obtener la solución.
- Problemas de decisión multiexperto: La aplicación permite resolución de problemas en los que múltiples expertos aportan valoraciones acerca de los múltiples criterios que se deben tener en cuenta.
- Representación de las preferencias: Se establece un orden de preferencia de las distintas alternativas empleando vectores de utilidad y relaciones de preferencia.
- Resultados expresados de forma lingüística: Estos resultados facilitan la interpretación de los resultados por parte del usuario del sistema.
- Diferentes dominios de expresión: La aplicación permite la elección de diferentes dominios de expresión: valores numéricos, variables lingüísticas, 2-tuplas, etc.
- Métodos de resolución: La aplicación dispone de una serie de métodos de resolución basados en el modelo de 2-tuplas lingüísticos. Dichos modelos permiten resolver el problema de acuerdo a los dominios de expresión definidos en el marco de evaluación del problema.
- Operadores de agregación: La aplicación dispone de diversos operadores de agregación disponibles para los distintos dominios de expresión.
- Análisis de los resultados: Cada método ofrece sus resultados de manera clara y concisa, permitiendo realizar análisis de los mismos que facilitan la labor de los usuarios.

2.5.2 Arquitectura de Flinstones

La aplicación "Flinstones" se ha desarrollado empleando la plataforma Rich Client Platform (RCP), que proporciona un marco para construir aplicaciones de escritorio que integren una gran funcionalidad. La principal utilidad que aporta RCP es que permite desarrollar rápidamente aplicaciones profesionales manteniendo el aspecto y el comportamiento en múltiples plataformas. Por otra parte, los componentes que forman parte del RCP son de alta calidad, se mantiene de forma activa y son de código abierto. [7]

En "Flinstones", se han definido diferentes módulos que permiten separar los diferentes componentes de la aplicación, de esta forma existirá una separación entre la lógica principal del DSS, el conjunto de elementos de representación de la interfaz, los métodos de resolución incluidos y los operadores de agregación implementados para resolver los problemas de toma de decisiones. Esto permite actualizar la aplicación si sólo se realizan cambios en un determinado módulo sin necesidad de realizar modificaciones en módulos independientes.

Los principales módulos básicos que forman parte de la aplicación son los siguientes:

- Bibliotecas o librerías: Estas librerías ofrecen las estructuras y procedimientos con el objetivo de apoyar y facilitar la resolución de problemas de decisión. Estas librerías incluyen elementos tales como: los expertos, los criterios, las alternativas y los dominios de expresión.
- Interfaz gráfica de usuario: La interfaz de usuario (GUI) permite a los usuarios interactuar con el sistema de soporte a la decisión.
- Método de resolución: Estos métodos desarrollan las diferentes metodologías basadas en distintos tipos de datos (numéricos, lingüísticos, etc.), con el fin de resolver los problemas de toma de decisiones.
- Operadores de agregación: Estos operadores implementan el conjunto de operadores de agregación que se han utilizado para agregar la distinta información involucrada en el problema de decisión.

Debido a la forma en la que se ha desarrollado la aplicación "Flinstones" y las herramientas que se han utilizando en su desarrollo, este DSS ofrece una serie de ventajas:

En primer lugar, al emplear una PCR basada en Java en su desarrollo este software se puede utilizar en cualquier máquina que disponga de una máquina virtual java. De esta forma los usuarios pueden utilizar la aplicación en sus máquinas independientemente del sistema operativo.

En segundo lugar, "Flinstones" se prepara para incluir a nuevos operadores de agregación así como nuevas metodologías de resolución de forma rápida y simple. Esto hace que se reduzca la programación de la tarea, ya que ofrece un marco que incluye una estructura completa, simplificando la integración de nuevos operadores de agregación y nuevos métodos.

En este contexto, la aplicación "Flinstones" permite resolver problemas de toma de decisiones y analizar los resultados obtenidos de acuerdo con los operadores de agregación y el método de resolución seleccionado para resolver el problema.

2.5.3 Estructura de la interfaz

Anteriormente se ha dicho que tanto la arquitectura de la aplicación como la estructura de la interfaz se organizan teniendo en cuenta las distintas fases que forman parte del proceso de toma de decisión. En el apartado anterior se ha tratado la arquitectura de la aplicación desde el punto de vista de los desarrolladores, a continuación se mostrará la estructura que presenta la interfaz de la aplicación a los distintos usuarios de la misma.

La aplicación se divide fundamentalmente en tres bloques funcionales independientes que se corresponden con las distintas fases de resolución de los problemas de toma de decisión: marco de evaluación, recogida de información, valoración de alternativas y análisis. En los siguientes apartados se explicarán cada uno de ellos. [7]

2.5.3.1 Marco de evaluación

Se corresponde con la primera ventana que se muestra cuando se pone en marcha la aplicación, se trata de establecer el marco de evaluación del problema. En esta ventana, se define la estructura del problema que incluirá todos los elementos involucrados en el mismo:

- Expertos: La aplicación permite la definición de múltiples expertos, grupos de expertos o sub-grupos de los mismos.
- Criterios: La aplicación permite la definición de los criterios involucrados en la definición del problema, así como, árboles de criterios de hasta tres niveles. Además, permite definir para cada criterio si es de coste o beneficio.
- Alternativas: Permite la introducción de las distintas alternativas propuestas por cada uno de los expertos.
- Dominios de expresión: La aplicación permite el uso de diferentes dominios de expresión que deben ser asignados a las distintas alternativas. Los dominios permitidos son: numérico, intervalar y lingüístico.
- Asignación de dominios de expresión: La aplicación habilita una zona para la asignación de los distintos dominios de expresión a cada experto, criterio y alternativa.

En la siguiente imagen se puede apreciar una captura de pantalla de esta pantalla de la aplicación:

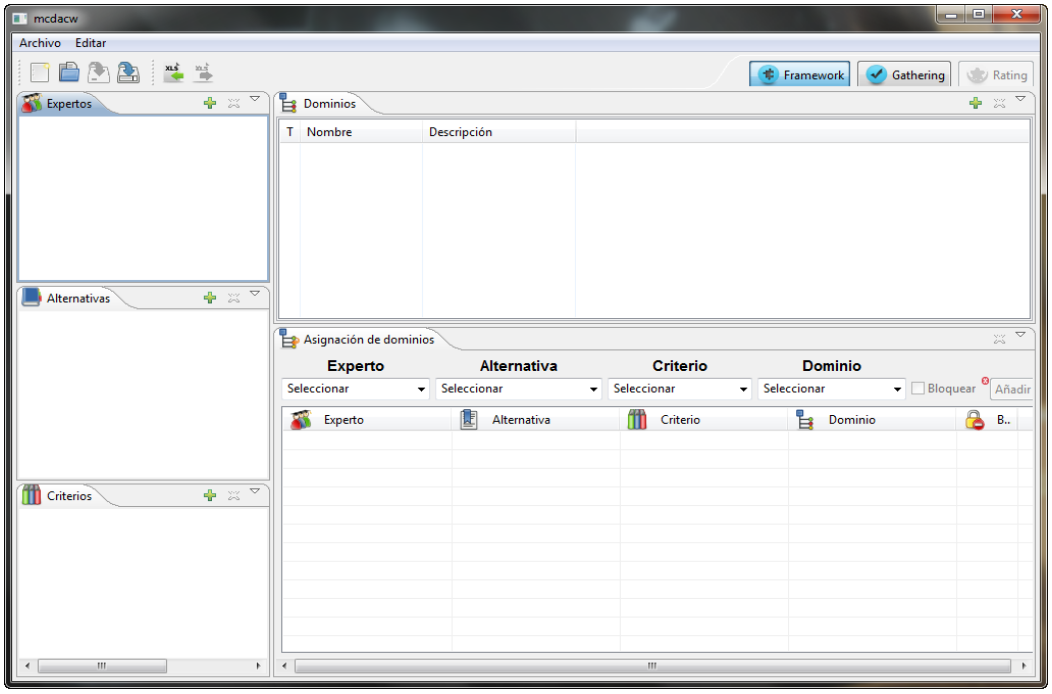


Ilustración 2.3 Pantalla Marco de Evalaución de la aplicación.

Como se puede ver en la imagen anterior la pantalla se divide en una serie de zonas correspondientes a los elementos previamente nombrados. A la izquierda de la pantalla se ven tres zonas correspondientes a los expertos, las alternativas y los criterios, en cada una de esas zonas se pueden definir nuevos elementos del problema. En el centro tenemos 2 nuevas zonas, la parte superior presenta los distintos dominios de expresión y permite añadir nuevos dominios al problema, por su parte la zona inferior se emplea para asignar los dominios de expresión a los distintos elementos del problema.

2.5.3.2 Recogida de información

Una vez el problema ha sido correctamente definido, se pasa a la recogida de información en la segunda ventana de la aplicación. Las distintas valoraciones son recogidas directamente a través del software que permite realizar filtrado por criterios, expertos o alternativas para facilitar la recolección de información.

En la siguiente imagen se puede apreciar una captura de esta pantalla de la aplicación:

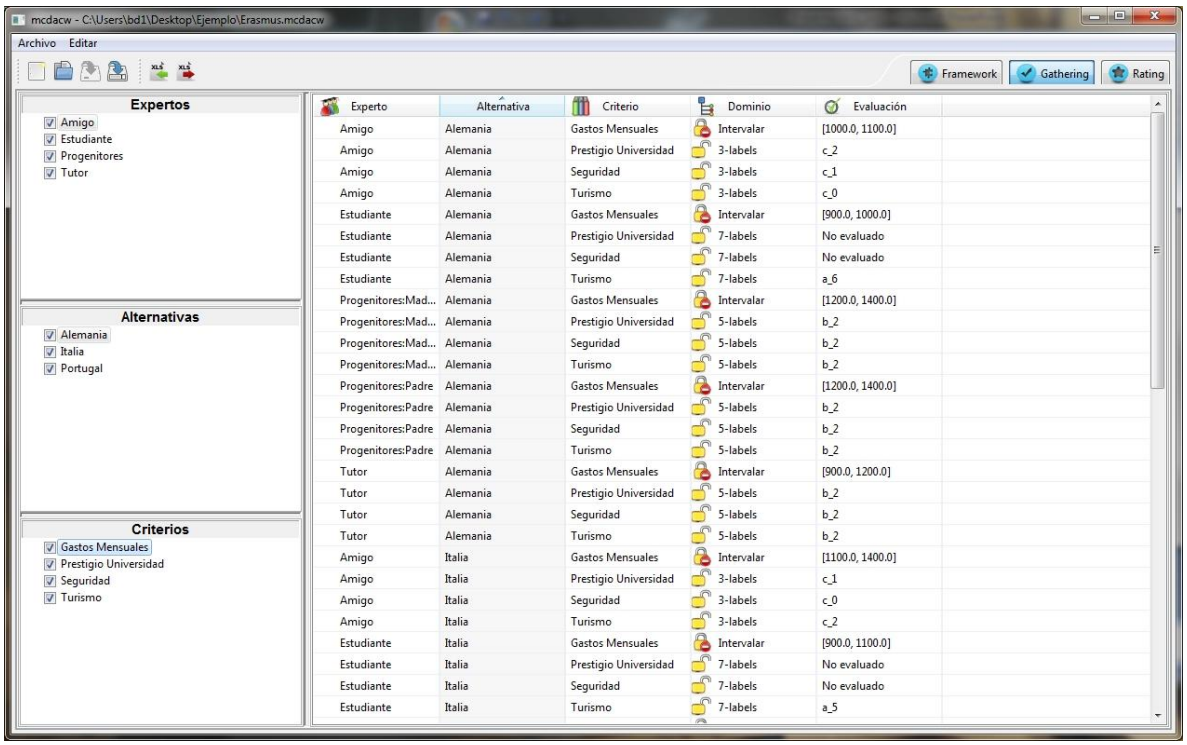


Ilustración 2.4 Pantalla Recogida de Información de la aplicación.

En la imagen anterior se puede ver la estructura de la ventana para la recogida de información. La zona izquierda muestra los distintos elementos del problema y permite seleccionarlos o eliminarlos para filtrar la información y facilitar la introducción de los elementos. En la zona central, se pueden realizar las distintas valoraciones de los elementos del problema o modificar las valoraciones anteriores.

2.5.3.3 Valoración de alternativas

La tercera ventana en la que se estructura la parte principal de la aplicación se corresponde con la parte de valoración de las diferentes alternativas. En esta pantalla se realiza la selección de la metodología de resolución del problema que se debe poner en práctica, además de mostrar información acerca de dicha metodología y mostrar las condiciones que debe de cumplir el marco de evaluación para escoger dicha metodología.

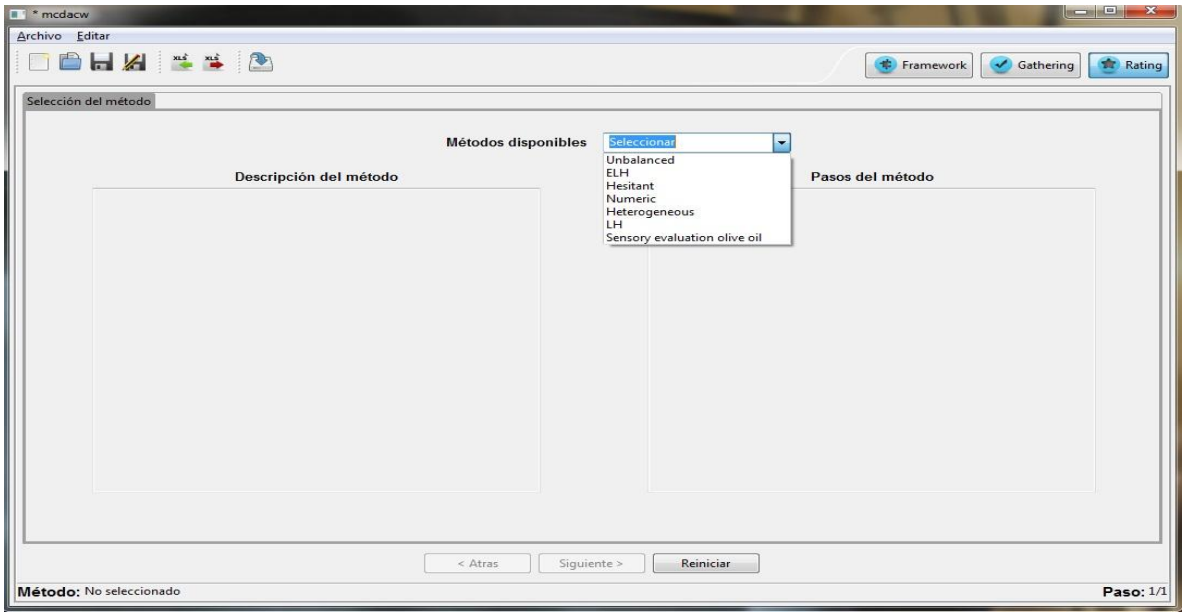


Ilustración 2.5 Pantalla Valoración de Alternativas de la aplicación.

En la imagen anterior se puede ver la estructura de la ventana correspondiente a la valoración de alternativas. En ella sólo se muestra una pestaña para seleccionar el método que se debe aplicar e información acerca del método que se ha seleccionado.

Una vez seleccionado el método, se añadirán una serie de pestañas de acuerdo a los pasos propios de cada método. Estas pestañas son distintas para cada método y se adaptan a los pasos en los que este estructurado internamente cada uno de ellos.

Fundamentalmente, cada método dispone de pestañas que permiten seleccionar los operadores de agregación involucrados en la resolución del problema, así como, la visualización de los resultados obtenidos tras la aplicación del método tanto resultados parciales como resultados finales.

2.5.3.4 Análisis

Algunos métodos disponen de pantallas para realizar un análisis detallado de los resultados, pudiendo eliminar de forma sencilla diferentes elementos del problema para ver como se modifican los resultados del mismo. Además, disponen de gráficas y tablas que facilitan mucho la lectura de los resultados obtenidos.

En la siguiente imagen se puede apreciar una captura de la pantalla de análisis habitualmente empleada:

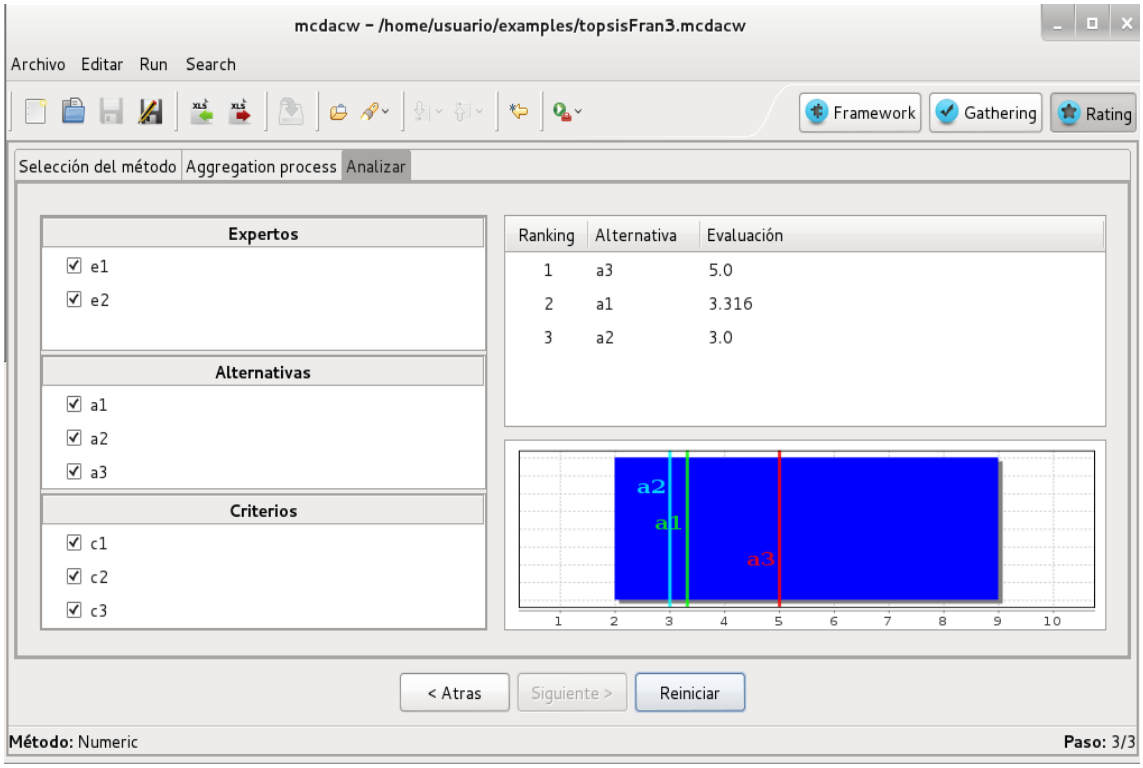


Ilustración 2.6 Pantalla Análisis de la aplicación.

En la imagen anterior se puede ver la estructura de la ventana de análisis. A la izquierda de la pantalla se muestran todos los elementos del problema, de forma que se pueden eliminar algunos de ellos y los resultados del problema se recalcularían. A la derecha tenemos dos partes, la parte superior ofrece los resultados obtenidos con el método y la parte inferior una representación gráfica de los mismos. Tanto los resultados como la gráfica se refrescarían si alguno de los elementos se elimina o se añade en la parte izquierda.

2.5.4 Diagrama de clases inicial

Una vez introducida la aplicación "Flinstones" que será ampliada mediante nuevos operadores y el método Topsis en la realización de este proyecto, se explicará con más detalle la estructura seguida para el desarrollo de la misma. Para ello, se utilizará el diagrama de clases del que se partió antes de comenzar la realización del proyecto.

Este diagrama de clases es muy amplio debido a la complejidad de la aplicación y a la gran funcionalidad que aporta. En este caso, se expondrán las partes del diagrama de clases correspondientes a las clases que son necesarias modificar o tener en cuenta a la hora de ampliar la aplicación con los nuevos operadores y el nuevo método. A continuación, se expone en primer lugar el conjunto de clases que intervienen en la realización de nuevos operadores y, posteriormente, las clases que intervienen en la realización de un nuevo método, así como, los pasos necesarios para incluirlos.

2.5.4.1 Clases relacionadas con los operadores de agregación

En este primer apartado, se verán las principales clases que intervienen o que deben ser tenidas en cuenta cuando se añade un nuevo operador de agregación al sistema. A continuación se muestran los diagramas de clases correspondientes a dichas partes del sistema:

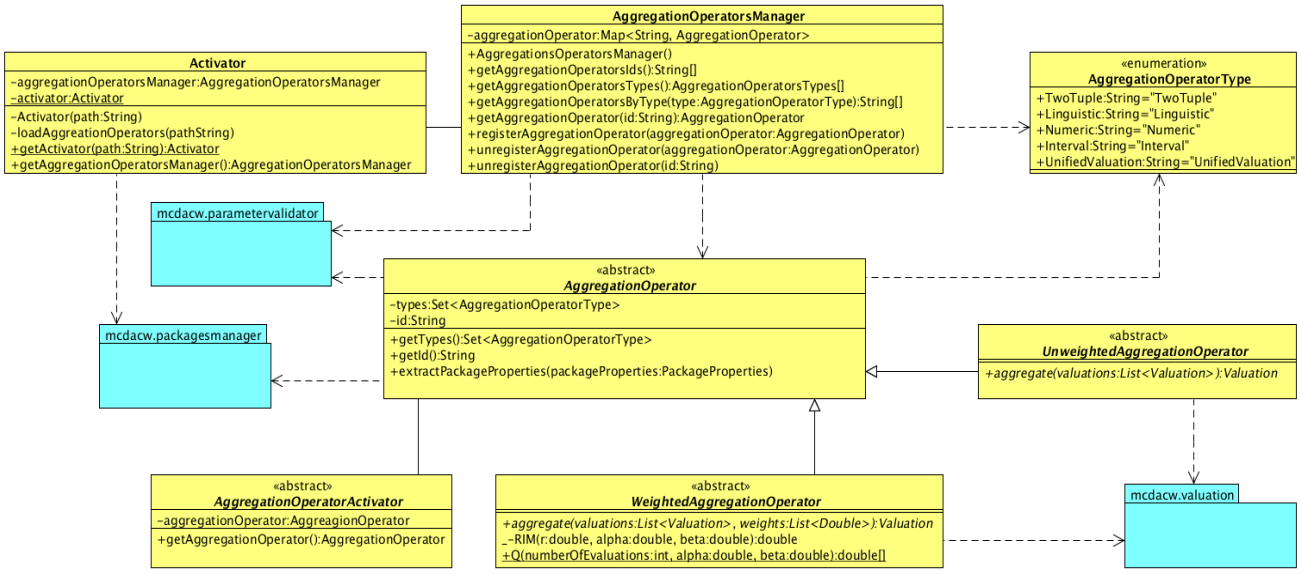


Ilustración 2.7 Diagrama de clases Operador de Agregación

En esta imagen se pueden ver las clases que deben ser tenidas en cuenta cuando se crea un nuevo operador de agregación en el sistema. Se puede ver que existen fundamentalmente dos tipos de operadores: operadores que requieren pesos y operadores sin pesos. Los operadores de agregación implementados deberán extender de uno de estos dos tipos, además deberán de implementar una serie de funciones para que la aplicación pueda manejar de forma correcta dichos operadores.

También se pueden ver los principales tipos de datos con los que pueden operar los distintos operadores de agregación, estos pueden ser fundamentalmente: numéricos, lingüísticos, 2-tuplas, intervalar y unificado.

Otro elemento fundamental es el activador que deben incluir todos los operadores de agregación, este elemento se puede ver de forma más detallada en la siguiente imagen:

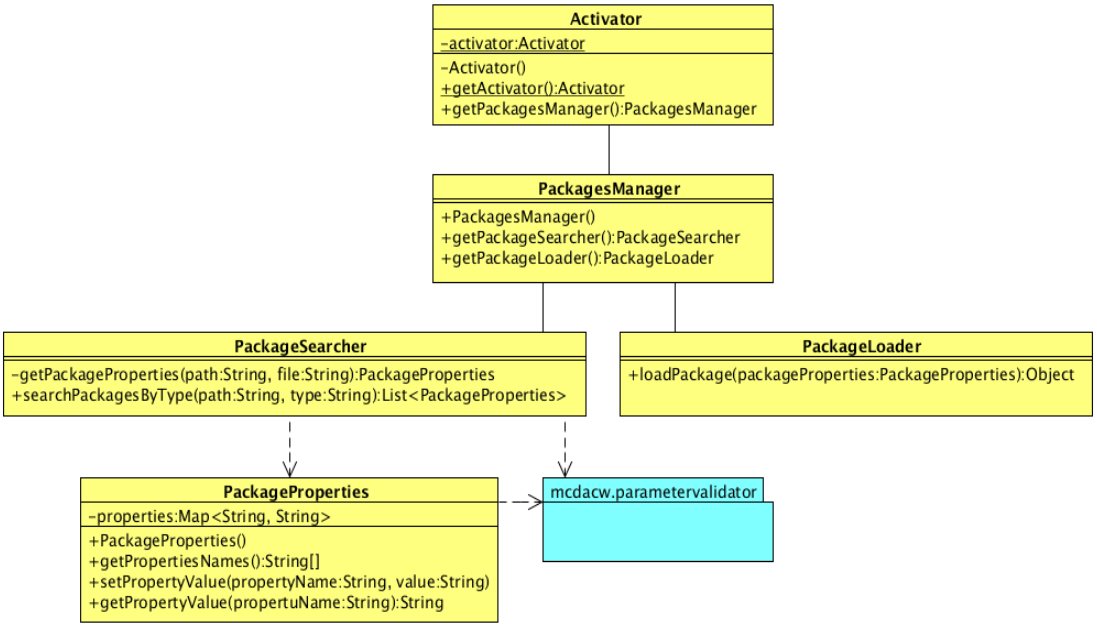


Ilustración 2.8 Diagrama de clases Activator

En esta imagen se pueden ver las clases que intervienen en el activador necesario para que la aplicación detecte de forma correcta los operadores de agregación implementados y los incluya en la aplicación de forma transparente. Es fundamental que cada operador de agregación incorpore su propio activador para que sea reconocido por la aplicación, en caso contrario, no podrá ser utilizado por los usuarios del sistema.

Estas son las principales clases que influyen en la creación de un nuevo operador de agregación, como se puede ver los nuevos operadores deben extender de determinadas clases abstractas que obligan a la implementación de múltiples funciones para que la aplicación funcione de forma correcta y todos los operadores sigan una estructura similar.

2.5.4.2 Pasos para crear un nuevo operador de agregación

Una vez vistas las principales clases involucradas en la creación de un nuevo operador de agregación, se expondrán los pasos necesarios para incluir de forma correcta un nuevo operador en nuestro sistema:

1- Se debe crear un nuevo paquete dentro de la carpeta correspondiente a los operadores de agregación de forma similar al resto de operadores de agregación incluidos. Este paquete debe incluir fundamentalmente:

   a. Un paquete del mismo nombre del operador que incluya: la clase activador que se encargará de incorporar dicho paquete a la aplicación y otra clase del mismo nombre del operador que extenderá del tipo de operador apropiado y se encargará de realizar llamar a las funciones de evaluación adecuadas según el tipo de dato.
   b. Un paquete adicional denominado "Valuation" que contendrá tantas clases como tipos de datos permita el operador. Estas clases tendrán el nombre del tipo de dato y ofrecerán la funcionalidad apropiada en cada caso.
   c. Un fichero de meta-información en el que se indicará los tipos de datos que maneja el operador, el nombre que le queremos dar y la versión del mismo.

2- Una vez creado el paquete anterior con todos sus elementos, se pasa a la implementación de la funcionalidad, dicha implementación será distinta para cada tipo de dato habilitado por el operador de agregación.

3- Una vez implementado el operador de agregación se exportará el mismo en forma de plugin para la aplicación y se incluirá en la carpeta correspondiente a los distintos plugins de los operadores de agregación.

Nota: El proceso anteriormente descrito es común tanto para operadores que requieran pesos como operadores que no los requieran, pero en el caso de operadores pesados el proceso debe incluir unos pasos adicionales.

En estos casos, se debe modificar ligeramente los pasos de los distintos métodos de resolución para que la aplicación solicite los distintos pesos cuando sea necesario, así mismo, ocurre cuando los operadores de agregación requieren algún tipo de parámetro adicional para su funcionamiento. Estas modificaciones se deben realizar siempre que la aplicación requiera al usuario la selección de un operador de agregación, ya que en ese momento será cuando se deben solicitar los pesos o parámetros necesarios.

En el desarrollo del proyecto, se han implementado algunos operadores de agregación pesados o que requieren parámetros, por tanto, ha sido necesario diseñar algunas pantallas para solicitarlos al usuario. Estos diseños se verán en el capítulo 5 relativo al diseño de software.

2.5.4.3 Clases relacionadas con los métodos de resolución

En este apartado, se verán las principales clases que intervienen o que deben ser tenidas en cuenta cuando se añade un nuevo método de resolución al sistema. A continuación se muestran los diagramas de clases correspondientes a dichas partes del sistema:

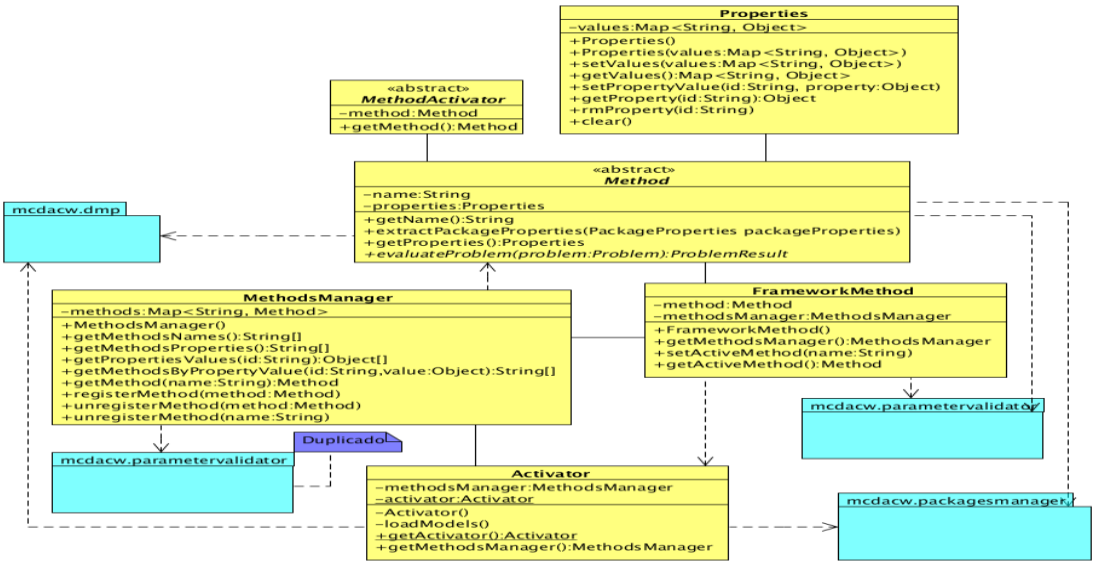


Ilustración 2.9 Diagrama de clases Método de Resolución

En esta imagen podemos ver las clases que deben ser tenidas en cuenta cuando se crea un nuevo método de resolución en el sistema. Cualquier nuevo método creado debe extender de esta clase abstracta para que todos los métodos implementados en la aplicación sigan una mista estructura y su funcionamiento sea correcto.

Al igual que en el caso de los operadores de agregación destaca la presencia del activador que deben incluir todos los métodos para que la aplicación detecte de forma correcta los operadores de agregación implementados y los incluya en la aplicación de forma transparente.

En el caso de los métodos otra clase importante que se debe tener es la que se corresponde con los pasos de cada método. Cada método incorporará una serie de pasos que le aportarán la funcionalidad necesaria para obtener los resultados necesarios y suministrarlos al usuario de forma correcta. La clase relativa al paso define la interfaz de la aplicación para dicho paso y realiza las llamadas a las funciones del método para la realización de los cálculos necesarios.

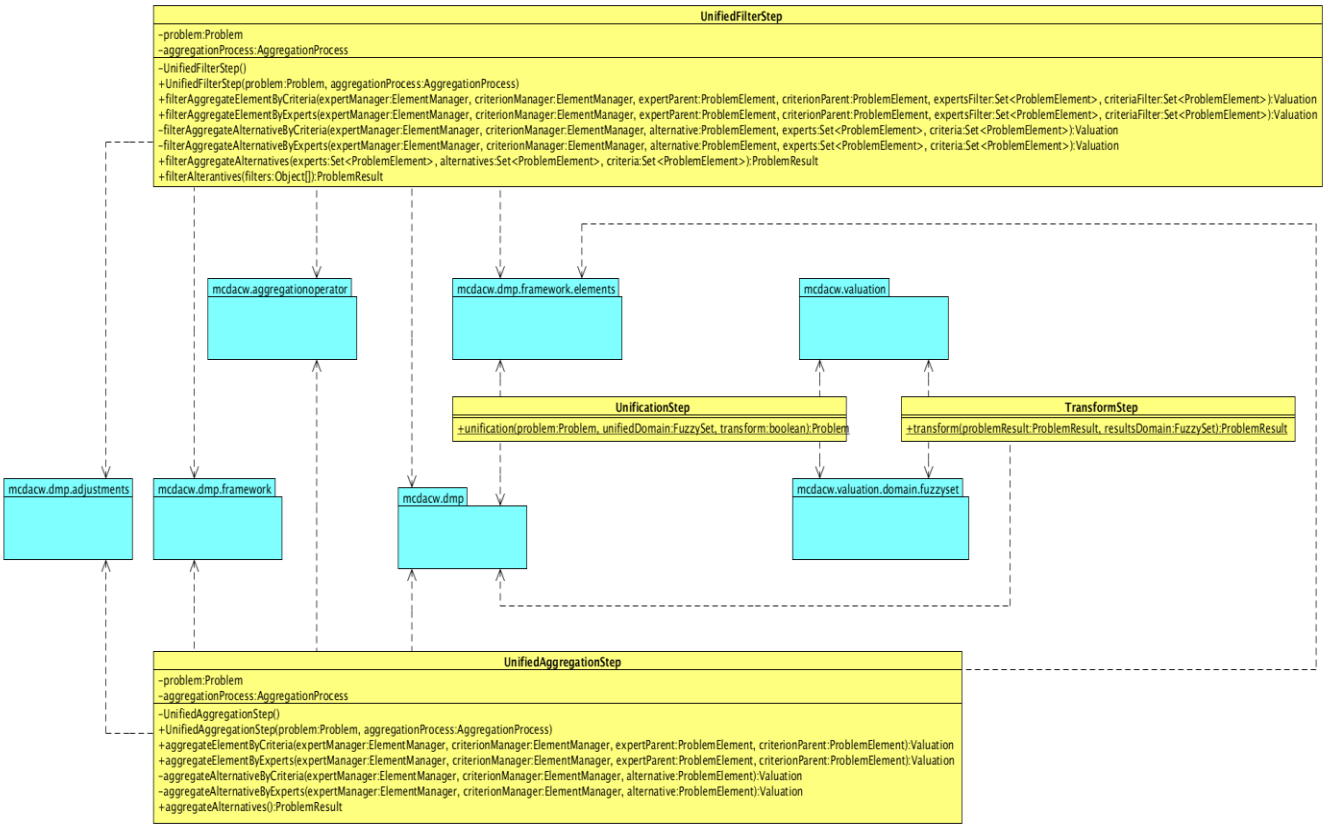


Ilustración 2.10 Diagrama de clases pasos Método Resolución

En esta imagen podemos ver los distintos pasos que forman parte de los métodos incluidos en la aplicación antes del desarrollo del proyecto. Como se ha dicho anteriormente, estos métodos definen la interfaz de la aplicación y las llamadas a las funciones del método parar realizar los cálculos apropiados siempre que sea necesario. A la hora de definir un nuevo método se deben incluir los pasos necesarios o reutilizar algunos de los que ya existan siempre que sea posible.

Estas son las principales clases que influyen en la creación de un nuevo método resolución. A continuación, se exponen los pasos que se deben seguir para crear un nuevo método e incorporarlo al sistema.

2.5.4.4 Pasos para crear un nuevo método de decisión

Una vez vistas las principales clases involucradas en la creación de un nuevo método de resolución, se expondrán los pasos necesarios para incluir de forma correcta un nuevo método en nuestro sistema:

1- Se debe crear un nuevo paquete dentro de la carpeta correspondiente a los métodos de resolución. Este paquete debe incluir fundamentalmente:
   a. Un paquete del mismo nombre del método que incluya: la clase activador que se encargará de incorporar dicho paquete a la aplicación y otra clase del mismo nombre del método que extenderá de la clase abstracta vista anteriormente en el diagrama de clases correspondiente a los métodos de resolución.
   b. Un fichero de meta-información en el que se indicará los tipos de datos que maneja el método, el nombre que le queremos dar y la versión del mismo.

2- Una vez creado el paquete anterior con todos sus elementos, se pasa a la implementación de la funcionalidad del método, dicha implementación será distinta para cada tipo de dato habilitado por el método e incorporará los cálculos necesarios para obtener los resultados adecuados.

3- Una vez implementado el método de resolución se exportará el mismo en forma de plugin para la aplicación y se incluirá en la carpeta correspondiente a los distintos plugins del resto de métodos.

Como se ha dicho anteriormente, esto es solo una parte de la incorporación de un nuevo método a la aplicación. Para completarlo se deben crear los pasos correspondientes que definirán el comportamiento de la interfaz cuando se ejecute el método. Para relacionarlo con el método simplemente se le indicará en el constructor del mismo los pasos que debe seguir y se suministrarán las funciones necesarias para realizar los cálculos de dicho método.

En caso de que la implementación del método requiera la creación de nuevos pasos en lugar de la reutilización de algunos ya creados, se deben realizar lo siguiente:

1- Se crea un nuevo paquete dentro de la carpeta correspondiente a los distintos pasos implementados con el nombre del nuevo paso que contendrá una clase con el mismo nombre.

2- Se incorporará el contenido de dicha clase que contendrá la definición de la interfaz correspondiente a dicho paso, así como, las invocaciones al método siempre que sea necesario.

3- Además es necesario realizar algunas modificaciones en otras clases para el correcto funcionamiento de los métodos.
    a. La clase abstracta correspondiente a los métodos debe añadir el identificador de los nuevos pasos creados.
    b. En el paso ya creado correspondiente a la selección del método se deben añadir los nuevos pasos cuando sea necesario.

Estos son los pasos a seguir para incorporar tanto los nuevos operadores de agregación como los nuevos métodos de resolución. Existen otros elementos que la aplicación "Flinstones" incorpora, pero los que se han visto anteriormente son los que tienen influencia en la realización del proyecto, además de ser dos de los elementos más importantes a tener en cuenta en los problemas de Toma de Decisión.

Capítulo 3:

Operadores de Agregación y Método de Decisión Topsis

3.1 Operadores de agregación

Una vez explicada la estructura interna de la aplicación y las clases que se deben tener en cuenta para añadir nuevos operadores de agregación y nuevos métodos de resolución, se va a exponer los distintos operadores de agregación que se implementarán en el desarrollo del proyecto.

Entendemos por agregación la operación consistente en transformar un conjunto de elementos amplio en un único elemento representativo del mismo. En los problemas de decisión podemos hablar de agregación por expertos, en las que las preferencias individuales que los expertos proporcionan sobre las distintas alternativas se agregan en una única preferencia global o de agregación por criterios, en las que se agregan las valoraciones para un mismo criterio dando una valoración global para dicho criterio.

Antes de pasar a exponer cada uno de los operadores de forma individual se va a explicar la importancia de la agregación en los problemas de Toma de Decisión.

3.1.1 Importancia de la agregación

La agregación ha sido un problema ampliamente estudiado cuando se trata de dar solución a los distintos problemas de toma de decisión. La agregación de información de manera eficiente, flexible e intentando disminuir al máximo la pérdida de información sensible se ha convertido en una tarea fundamental en problemas de decisión multi-criterio, ya que precisan procesar una cantidad de información muy elevada cuya calidad y precisión es variada. [8, 9, 10]

Por otra parte, la toma de decisiones en distintos ámbitos suele tener un componente difuso que aporta flexibilidad. A la hora de elegir el operador de agregación más adecuado se pueden considerar múltiples criterios, algunos de ellos son: [11]

1- Fuerza axiomática: Un operador se considera más adecuado que otro si está menos limitado por los axiomas que debe satisfacer.

2- Ajuste empírico: Los operadores deben ajustarse a la realidad y al contexto del problema lo que hace más adecuado determinados operadores según el problema que se intente solucionar.

3- Adaptabilidad: Los operadores deben poder adaptarse al contexto cambiante en el que se encuentran, principalmente mediante la parametrización.

4- Eficiencia numérica: El esfuerzo computacional de cálculo necesario es importante a la hora de resolver problemas grandes y puede ser un factor clave para elegir un operador u otro.

5- Compensación y rango de compensación: A medida que mayor sea el grado en que se contrarrestan las distintas funciones de pertenencia de los conjuntos difusos agregados, el operador de agregación se adaptará mejor y ofrecerá mejores soluciones.

6- Comportamiento agregado: El número de conjuntos agregados puede influir en los grados de pertenencia resultantes, por tanto, este hecho debe ser tenido en cuenta cuando se elige el operador.

7- Nivel de escala requerido de las funciones de pertenencia: Determinados operadores pueden requerir diferentes niveles de escala para ser admisibles. En igualdad de condiciones se prefiere el que aporte un nivel de escala más bajo.

Como se ha dicho anteriormente el concepto de agregación tiene una influencia muy importante en la resolución de problemas de toma de decisiones, por ello, existe múltiples tipos de operadores diferentes. A continuación, se exponen los nuevos operadores de agregación incorporados a la aplicación "Flinstones" en este proyecto.

3.1.2 Operadores de agregación implementados

Anteriormente, se ha definido el concepto de agregación y se ha argumentado la importancia que tiene la misma en la resolución de problemas de toma de decisión. Existen multitud de operadores de agregación, en este apartado se definirán aquellos que se van a implementar en el desarrollo de este proyecto para incluirlos en la aplicación Flisntones. Estos son:

- *Media geométrica*
- *Media geométrica pesada*
- *Media harmónica*
- *Media harmónica pesada*
- *Power Median*
- *Power Media Pesada*
- *Bonferroni Means*
- *Heronian mean*
- *Media cuadrática*

3.1.2.1 Media geométrica

Definición 3.1. [12]. Sea x = {x_1,…,x_n} un conjunto de valores numéricos para una variable x. La media geométrica viene definida por la siguiente función:

$$G(x) = \sqrt[n]{x_1 x_2 \dots x_n} = \left(\prod_{i=1}^{n} x_i\right)^{1/n}$$

Ecuación 3.1. Media Geométrica

3.1.2.2 Media geométrica pesada

Definición 3.2. [12]. Sea x = {x_1,…,x_n} un conjunto de valores numéricos para una variable x y w={w_1,…,w_n} el vector de pesos para cada uno de esos valores numéricos. La media geométrica pesada viene definida por la siguiente función:

$$G_w(x) = \prod_{i=1}^{n} x_i^{w_i}$$

Ecuación 3.2. Media Geométrica Pesada

3.1.2.3 Media harmónica

Definición 3.3. [12]. Sea x = {x_1,…,x_n} un conjunto de valores numéricos para una variable x. La media harmónica viene definida por la siguiente función:

$$H(x) = n\left(\sum_{i=1}^{n}\frac{1}{x_i}\right)^{-1}$$

Ecuación 3.3. Media Harmónica

3.1.2.4 Media harmónica pesada

Definición 3.4. [12]. Sea x = {x_1,…,x_n} un conjunto de valores numéricos para una variable x y w={w_1,…,w_n} el vector de pesos para cada uno de esos valores numéricos. La media harmónica pesada viene definida por la siguiente función:

$$H_w(x) = \left(\sum_{i=1}^{n}\frac{w_i}{x_i}\right)^{-1}$$

Ecuación 3.4. Media Harmónica Pesada

3.1.2.5 Power Median

Definición 3.5. [12]. Sea x = {x_1,…,x_n} un conjunto de valores numéricos para una variable x con r perteneciente a los números reales. La power median viene definida por la siguiente función:

$$M_{[r]}(x) = \left(\frac{1}{n}\sum_{i=1}^{n}{x_i}^r\right)^{1/r}$$

Ecuación 3.5. Power Median

3.1.2.6 Power Median pesada

Definición 3.6. [12]. Sea x = {x_1,…,x_n} un conjunto de valores numéricos para una variable x, w={w_1,…,w_n} el vector de pesos para cada uno de esos valores numéricos x y con r perteneciente a los números reales. La power median pesada viene definida por la siguiente función:

$$M_{w,[r]}(x) = \left(\sum_{i=1}^{n}w_i{x_i}^r\right)^{1/r}$$

Ecuación 3.6. Power Median Pesada

3.1.2.7 Bonferroni means

Definición 3.7. [12]. Sea x = {x_1,…,x_n} un conjunto de valores numéricos para una variable x y los parámetros p y q mayores o iguales que 0. La bonferroni means viene definida por la siguiente función:

$$B^{p,q}(x) = \left(\frac{1}{n(n-1)} \sum_{i,j=1, i \neq j}^{n} x_i^{\,p} x_j^{\,q} \right)^{1/(p+q)}$$

Ecuación 3.7. Bonferroni Means

3.1.2.8 Heronian Means

Definición 3.8. [12]. Sea x = {x_1,…,x_n} un conjunto de valores numéricos para una variable x. La heronian means viene definida por la siguiente función:

$$HR(x) = \frac{2}{n(n+1)} \sum_{i=1}^{n} \sum_{j=i}^{n} \sqrt{x_i x_j}$$

Ecuación 3.8. Heronian Means

3.1.2.9 Media cuadrática

Definición 3.9. [12]. Sea x = {x_1,…,x_n} un conjunto de valores numéricos para una variable x. La media cuadrática viene definida por la siguiente función:

$$MC(x) = \sqrt{\sum_{i=1}^{n} \frac{x_i^{\,2}}{n}}$$

Ecuación 3.9. Media Cuadrática

Nota: Todos los operadores de agregación se han definido para valores de tipo numérico, se debe tener en cuenta que todos ellos extenderán también a 2-tuplas. Para realizar los cálculos con 2-tuplas se deben considerar las dos siguientes transformaciones:

Definición 3.10. [13]. Siendo S = $\{s_0,s_1,\ldots,s_t\}$ un conjunto de términos lingüísticos y el valor β entre 0 y t el resultado de la operación de agregación, el valor 2-tuplas que expresa la misma información es obtenido a partir de la función:

$$\Delta: [0,t] \rightarrow S \times [-0.5,0.5)$$
$$\Delta(\beta) = \begin{cases} s_i, i = round(\beta) \\ \alpha = \beta - i, \alpha \in [-0.5,0.5) \end{cases}$$

Ecuación 3.10. Transformación numérica a 2-tuplas

Donde round es la operación habitual de redondeo, s_i se corresponde con la etiqueta más cercana al valor β y α es el valor correspondiente a la traslación simbólica.

Definición 3.11. [13]. Siendo S = $\{s_0,s_1,\ldots,s_t\}$ un conjunto de términos lingüísticos y el valor (si,α_i) el valor 2-tuplas existe siempre una función inversa que a partir de dicho valor 2-tuplas obtiene un valor numérico equivalente:

$$\Delta^{-1}: S \times [-0.5,0.5) \rightarrow [0,t]$$
$$\Delta^{-1}(s_i,\alpha) = i + \alpha = \beta$$

Ecuación 3.11. Transformación 2-tuplas a numérica

Para ambas definiciones anteriores (10 y 11) es necesario indicar que la transformación de un término lingüístico a 2-tuplas consiste en añadir el valor 0 como translación simbólica.

Para realizar los cálculos con 2- tuplas, en primer lugar las valoraciones en 2-tuplas se transforman a valor numérico mediante la primera transformación (Ecuación 3.10), se realiza la agregación y se vuelve a transformar el valor numérico resultante a 2-tuplas mediante la segunda transformación (Ecuación 3.11).

3.2 Método de decisión Topsis

Una vez explicada la estructura interna de la aplicación y las clases que se deben tener en cuenta para añadir nuevos métodos de resolución, se va a definir el funcionamiento del método Topsis que será implementado y añadido a la aplicación "Flinstones".

Topsis [13] es un método para múltiples atributos que identifica soluciones a partir de un conjunto finito de alternativas. El principio básico es que la alternativa elegida debe tener la distancia más corta a la solución ideal y la distancia más lejana a la solución no ideal. El proceso seguido por el método de resolución Topsis puede ser expresado en una serie de pasos [14-17]:

Nota: Los siguientes pasos se realizan considerando que en el problema intervienen varios expertos, valorando n criterios y m alternativas.

Paso 1: A partir de las diferentes matrices de información ofrecida por los diferentes expertos se debe realizar un proceso de agregación y obtener una única matriz colectiva global. En dicho proceso de agregación intervendrá el operador de agregación seleccionado por el usuario.

Paso 2: Calcular la matriz de decisión normalizada. El valor normalizado b_{ij} es calculado como

$$b_{ij} = a_{ij} \Big/ \sqrt{\sum_{j=1}^{n} {a_{ij}}^2} \qquad i = 1,2,\dots,m\,, \quad j = 1,2,\dots n$$

Ecuación 3.12. Paso 2

Paso 3: Calcular la matriz de decisión normalizada pesada. El valor normalizado pesado c_{ij} es calculado como

$$c_{ij} = w_j b_{ij} \quad i = 1,2,\dots.m, \quad j = 1,2\dots..n$$

Ecuación 3.13. Paso 3

Donde w_j es el peso asignado al criterio j.

Paso 4: Determina la solución ideal positiva y la solución ideal negativa:

$$C^+ = \{c_1^+, \ldots., c_n^+\} = \{(\max(i)\, c_{ij} \mid j \in I), (\min\,(i) c_{ij} \mid j \in J)\}$$
$$C^- = \{c_1^-, \ldots., c_n^-\} = \{(\min(i)\, c_{ij} \mid j \in I), (\max\,(i) c_{ij} \mid j \in J)\}$$

Ecuación 3.14. Paso 4

Donde I está asociado con criterios de beneficio y J está asociado con criterios de coste.

Paso 5: Calcular las medidas de separación usando la distancia Euclidea. La solución de cada alternativa con la solución ideal viene dada por:

$$d_i^+ = \sqrt{\sum_{j=1}^{n} (c_{ij} - c_j^+)^2}\ , \quad i = 1, \ldots., m$$

Ecuación 3.15. Paso 5.1

De forma similar la distancia negativa viene dada por:

$$d_i^- = \sqrt{\sum_{j=1}^{n} (c_{ij} - c_j^-)^2}\ , \quad i = 1, \ldots., m$$

Ecuación 3.16. Paso 5.2

Paso 6: Calcular el coeficiente de cercanía a la solución ideal. El coeficiente de cercanía de la alternativa A_i con respecto a A^+ viene dado por:

$$R_i = d_i^- / (d_i^- + d_i^+)\ , \quad i = 1, \ldots., m$$

Ecuación 3.17. Paso 6

Paso 7: Ordenar los resultados por orden de preferencia. Para ordenar las alternativas usando este índice se deben ordenar los resultados de forma decreciente.

Nota: La definición del método Topsis se ha realizado considerando tipos numéricos, pero al igual que con los operadores de agregación, el método extenderá a 2-tuplas. Para ello, se tendrán que utilizar cuando sean necesarias las ecuaciones 3.10 y 3.11 para transformar valores 2-tuplas en valores numéricos y viceversa.

3.2.1 Ejemplo del método de decisión Topsis

En el apartado anterior se ha expuesto la definición del método Topsis paso a paso pero desde un punto de vista teórico. A continuación se muestra un ejemplo práctico de la aplicación de dicho método considerando un dominio de expresión de tipo 2-tupla lingüística.

Se supone que se está realizando una investigación para descubrir que alternativa es mejor para realizar una inversión de capital. Las 5 posibles alternativas son: A_1 una compañía de coches, A_2 una compañía de alimentos, A_3 una compañía de ordenadores, A_4 una compañía de armas y A_5 una compañía de televisión. La decisión debe ser tomada atendiendo a 4 criterios: G_1 es el análisis de riesgo, G_2 es el análisis de expansión, G_3 es el impacto socio-político, G_4 es el análisis del impacto ambiental.

Las valoraciones realizadas por los 3 expertos son las siguientes:

	G1	*G2*	*G3*	*G4*
A1	M	G	P	P
A2	P	VP	M	P
A3	G	M	G	EP
A4	VG	P	P	G
A5	EG	EP	VP	M

Tabla 3.1. Valoraciones Experto 1

	G1	*G2*	*G3*	*G4*
A1	P	M	VP	VP
A2	VP	EP	G	G
A3	M	G	P	EG
A4	EG	VP	VP	M
A5	P	VP	M	VP

Tabla 3.2. Valoraciones Experto 2

	G1	*G2*	*G3*	*G4*
A1	G	P	VP	VG
A2	VP	G	P	G
A3	VG	VP	G	P
A4	G	VG	EG	VP
A5	M	VP	M	G

Tabla 3.3. Valoraciones Experto 3

Nota: La resolución del método se debe realizar para 2-tuplas lingüísticas, por tanto, los pasos del método varían con respecto a los introducidos en el apartado anterior.

A continuación, pasamos a resolver el problema aplicando el método Topsis:

Paso 1: Transformar la matriz de decisión lingüística de cada experto en matriz de decisión 2-tuplas lingüística.

	G1	*G2*	*G3*	*G4*
A1	(M,0)	(G,0)	(P,0)	(P,0)
A2	(P,0)	(VP,0)	(M,0)	(P,0)
A3	(G,0)	(M,0)	(G,0)	(EP,0)
A4	(VG,0)	(P,0)	(P,0)	(G,0)
A5	(EG,0)	(EP,0)	(VP,0)	(M,0)

Tabla 3.3. Matriz 2-tuplas Experto 1

	G1	*G2*	*G3*	*G4*
A1	(P,0)	(M,0)	(VP,0)	(VP,0)
A2	(VP,0)	(EP,0)	(G,0)	(G,0)
A3	(M,0)	(G,0)	(P,0)	(EG,0)
A4	(EG,0)	(VP,0)	(VP,0)	(M,0)
A5	(P,0)	(VP,0)	(M,0)	(VP,0)

Tabla 3.4. Matriz 2-tuplas Experto 2

	G1	*G2*	*G3*	*G4*
A1	(G,0)	(P,0)	(VP,0)	(VG,0)
A2	(VP,0)	(G,0)	(P,0)	(G,0)
A3	(VG,0)	(VP,0)	(G,0)	(P,0)
A4	(G,0)	(VG,0)	(EG,0)	(VP,0)
A5	(M,0)	(VP,0)	(M,0)	(G,0)

Tabla 3.5. Matriz 2-tuplas Experto 3

Paso 2: Calcular la matriz de decisión colectiva, es decir, obtener una sola matriz a partir de las 3 anteriores realizando la agregación de expertos. Para realizar la agregación se va a utilizar el operador de agregación de "Media Aritmética".

	G1	*G2*	*G3*	*G4*
A1	(M,0)	(M,0)	(VP,0.33)	(M,-0.33)
A2	(VP,0.33)	(P,-0.33)	(M,0)	(M,0.33)
A3	(G,0)	(M,-0.33)	(M,0.33)	(M,-0.33)
A4	(VG,0)	(M,-0.33)	(M,0)	(M,-0.33)
A5	(G,-0.33)	(VP,-0.33)	(P,0.33)	(M,-0.33)

Tabla 3.6. Matriz Colectiva de Expertos

Paso 3: Determinar la solución ideal positiva y la solución ideal negativa a partir de la matriz colectiva obtenida en el paso anterior.

	C1	***C2***	***C3***	***C4***
Solución Ideal Positiva	(VG,0)	(M,0)	(M,0.33)	(M,0.33)
Solución Ideal Negativa	(VP,0.33)	(VP,-0.33)	(VP,0.33)	(M,-0.33)

Tabla 3.7. Solución Ideal Positiva y Negativa

Como se puede ver, para la solución ideal positiva se escoge el valor mayor entre las distintas alternativas para cada uno de los criterios puesto que todos los criterios son de beneficio, mientras que para la solución ideal negativa se escoge el menor valor.

Paso 4: Obtener la distancia de cada alternativa a la solución ideal positiva y negativa. En este caso, se utilizará un operador de distancias pesado, siendo w = (0.18, 0.32, 0.356, 0.144), los pesos para cada uno de los criterios.

	A1	***A2***	***A3***	***A4***	***A5***
Distancia a Solución Ideal Positiva	(VP,0.17)	(VP,0.21)	(EP,0.38)	(EP,0.32)	(VP,0.44)
Distancia a Solución Ideal Negativa	(VP,0.05)	(VP,0.01)	(P,-0.17)	(P,-0.11)	(VP,-0.22)

Tabla 3.8. Distancias de cada alternativa a solución ideal positiva y negativa

Para aclarar la forma de obtención de los valores anteriores, se va a indicar como ejemplo el cálculo de ambos valores para la alternativa 1 (A1). Se deben tener en cuenta los valores de la solución ideal y no ideal (tabla 3.7), así como, los valores correspondientes a la alternativa 1 de la matriz colectiva de expertos (tabla 3.6) y, al ser distancia pesada, los valores del vector de pesos w. A continuación se muestra el ejemplo de los cálculos:

$$Distancia\ Solución\ Ideal\ Positiva\ A1 = (VP,0.17) = \\ = \Delta(|\Delta^{-1}(M,0) - \Delta^{-1}(VG,0)| \times 0.18 + |\Delta^{-1}(M,0) - \Delta^{-1}(M,0)| \times 0.32 + |\Delta^{-1}(VP,0.33) - \Delta^{-1}(M,0.33)| \times 0.356 \\ + |\Delta^{-1}(M,-0.33) - \Delta^{-1}(M,0.33)| \times 0.144)$$

Ecuación 3.18 Cálculo de la distancia a Solución Ideal Positiva para la Alternativa A1

$$Distancia\ Solución\ Ideal\ Negativa\ A1 = (VP,0.05) = \\ = \Delta(|\Delta^{-1}(M,0) - \Delta^{-1}(VP,0.33)| \times 0.18 + |\Delta^{-1}(M,0) - \Delta^{-1}(VP,-0.33)| \times 0.32 + |\Delta^{-1}(VP,0.33) - \Delta^{-1}(VP,0.33)| \times 0.356 \\ + |\Delta^{-1}(M,-0.33) - \Delta^{-1}(M,-0.33)| \times 0.144)$$

Ecuación 3.19 Cálculo de la distancia a Solución Ideal Negativa para la Alternativa A1

Paso 5: Calcular el coeficiente de cercanía de cada alternativa con la solución ideal positiva.

	A1	***A2***	***A3***	***A4***	***A5***
Coeficiente de Cercanía	(EP,0.47)	(EP,0.45)	(VP,-0.17)	(VP,-0.14)	(EP,0.35)

Tabla 3.9. Coeficientes de cercanía para cada alternativa

Para aclarar la forma de obtención de los valores anteriores, se va a indicar como ejemplo el cálculo del valor para la alternativa 1 (A1). Se deben tener en cuenta los valores de distancia con la solución ideal positiva y negativa calculada en el paso anterior (tabla 3.8). A continuación se muestra el ejemplo de los cálculos:

$$Coeficiente\ de\ Cercanía\ A1 = (EP,0.47) = \Delta\left(\frac{\Delta^{-1}(VP,0.05)}{\Delta^{-1}(VP,0.17) + \Delta^{-1}(VP,0.05)}\right)$$

Tabla 3.20. Coeficientes de cercanía para la alternativa 1

Paso 6: Ordenar las alternativas según el coeficiente de cercanía anteriormente calculado, dicha ordenación debe realizarse de forma decreciente. De esta forma, las alternativas quedarían ordenadas de la siguiente forma: $A_4 > A_3 > A_1 > A_2 > A_5$.

Se puede concluir que la mejor alternativa para realizar la inversión de capital según los resultados obtenidos tras aplicar el método Topsis es la cuarta, la correspondiente a la compañía de armas.

Capítulo 4:

Análisis del sistema

Después de introducir el problema de toma de decisiones desde un punto de vista teórico y de exponer tanto los operadores de agregación como el método que se van a implementar para introducirlos en la aplicación "Flinstones", el siguiente paso consiste en describir el desarrollo de los mismos y su integración en el sistema manteniendo la estructura tanto interna como visual del mismo.

Para la realización de la aplicación, seguimos el ciclo de la Ingeniería del Software. La Ingeniería del Software se compone de las siguientes actividades [18]:

- **Especificación de requerimientos:** Se obtiene el propósito del sistema, así como las propiedades y restricciones impuestas sobre el mismo.
- **Análisis del sistema:** Se obtiene un modelo del sistema correcto, completo, consistente, claro y verificable.
- **Diseño del sistema:** Se establecen los objetivos del proyecto y las estrategias a seguir para conseguirlos.
- **Implementación:** Consiste en la traducción del modelo lógico del sistema a código fuente.
- **Pruebas:** Verificar y validar el sistema.

Debido a la entidad de cada una de las fases anteriores, se han dividido en capítulos. Así la fase de especificación de requerimientos y análisis del sistema se expone en este mismo capítulo (capítulo 4), la fase de diseño en el capítulo 5 y, finalmente, la fase de implementación y pruebas en el capítulo 6.

4.1 Especificación de requerimientos

La primera actividad a realizar en el proceso de Ingeniería del Software es determinar el propósito del proyecto, las propiedades que debe satisfacer y las restricciones impuestas. Esta primera actividad es de gran importancia, pues para que la aplicación software sea capaz de cumplir el propósito, debemos con anterioridad conocer tanto los requerimientos como las limitaciones impuestas.

Tras haber definido el propósito del proyecto en el capítulo 1, el siguiente paso consiste en especificar los requerimientos del mismo. Estos son un conjunto de propiedades o restricciones definidas de forma totalmente precisa, que dicho proyecto ha de satisfacer.

4.1.1 Requerimientos funcionales

Los requerimientos funcionales de un sistema software son aquellos requisitos referidos al estricto funcionamiento del sistema, es decir, a los requisitos que describen las funcionalidades que el sistema debe proporcionar a los usuarios para cumplir sus objetivos.

Los requerimientos funcionales del sistema serán siguientes:

- Creación de un nuevo problema: Actividades relacionadas con la creación de un nuevo problema y los elementos que contiene.
    - Creación de un nuevo experto (RF-01): Un usuario crea un nuevo experto asociado al problema que está abierto en ese momento o a un nuevo problema creado.
    - Creación de una nueva alternativa (RF-02): Un usuario crea una nueva alternativa asociada al problema que está abierto en ese momento o a un nuevo problema creado.
    - Creación de un nuevo criterio (RF-03): Un usuario crea un nuevo criterio asociado al problema que está abierto en ese momento o a un nuevo problema creado.
    - Creación de un nuevo dominio de expresión (RF-04): Un usuario crea un nuevo dominio de expresión que podrá ser asociado a cualquier elemento del problema que

está abierto en ese momento, de esta forma las valoraciones de ese elemento deberán ser dadas en dicho dominio.

  - Asignación de dominio de expresión (RF-05): Un usuario asocia un dominio de expresión previamente creado a cualquiera de los elementos que ya han sido creados en el sistema.

- Gestión del problema: Actividades relacionadas con el almacenamiento y recuperación de un problema.
  - Cargar problema guardado (RF-06): Un usuario carga un problema previamente guardado incorporando todos los elementos del mismo al sistema para su posterior análisis.
  - Guardar problema (RF-07): Un usuario guarda el problema abierto en ese momento en el sistema, para poder recuperar posteriormente todos los elementos que lo componen.

- Valoración de las distintas alternativas (RF-08): Un usuario realiza una valoración de una determinada alternativa asignada a un determinado experto y sobre un determinado criterio dentro del dominio de expresión que ese elemento tenga asignado.

- Selección del método de resolución (RF-09): Actividades relacionadas con la selección del método de resolución con el que se desea resolver el problema planteado previamente.

- Método Topsis: Actividades concretas que debe llevar a cabo el método Topsis.
  - Selección de operador de agregación para expertos (RF-10): El usuario podrá seleccionar el operador de agregación que desee para realizar la agregación de expertos al aplicar dicho método de resolución.
  - Selección del operador de distancia (RF-11): El usuario podrá seleccionar el operador de distancias que desee para obtener los resultados aplicando el método Topsis.

- Actividades relacionadas con la selección de los distintos operadores de agregación por parte del usuario, existen los siguientes tipos de operadores:
  - Selección de operador sin parámetros ni pesos (RF-12): El usuario selecciona un operador de agregación que no requiere parámetros ni pesos para obtener los resultados.

- Selección de operadores con parámetros (RF-13): El usuario selecciona un operador de agregación que requiere determinados parámetros para obtener los resultados.
- Selección de operadores con pesos (RF-14): El usuario selecciona un operador de agregación que requiere pesos asignados a los expertos para obtener los resultados.

Nota: Junto con los requerimientos funcionales del sistema se ha incluido un identificador, para facilitar tareas posteriores y poder referirse a ellos de manera más breve y cómoda.

4.1.2 Requerimientos no funcionales

Los requerimientos no funcionales de un sistema software son todos aquellos requisitos no referidos al estricto funcionamiento del sistema, sino a otros factores externos, son las restricciones impuestas por los requisitos funcionales del sistema. Son tan importantes como los requerimientos funcionales y, en muchas ocasiones, pueden incluso llegar a ser críticos para la aceptación del sistema. Normalmente, estos requerimientos especifican propiedades del sistema software en sí (velocidad, rendimiento,…) y de la interfaz de usuario, además de todas las restricciones impuestas por la organización (legalidad vigente, política empresarial,…).

Para el caso concreto del proyecto tratado no van a existir limitaciones organizacionales. Por tanto, se obtendrán y analizarán los requerimientos no funcionales a partir de las necesidades hardware y software de los equipos informáticos, para dar al usuario la funcionalidad requerida de forma eficiente, y de la interfaz gráfica entre el usuario y la aplicación. Además se debe tener en cuenta que el software creado deberá ser integrado en una aplicación ya existente, por tanto, este hecho conllevará una serie de requerimientos no funcionales adicionales.

Los requerimientos no funcionales son los siguientes:

- Tiempo de respuesta aceptable ante la interacción con cualquier usuario.

- La aplicación se podrá ejecutar en diferentes plataformas manteniendo una estructura visual y funcionalidad similar

- Requerimientos de la interfaz: Estos requerimientos están estrechamente ligados a la usabilidad y sus principios.
    - Facilidad de aprendizaje: Se refiere a aquellas características de la interfaz que permiten a los usuarios noveles comprender cómo usarla inicialmente y cómo obtener un máximo grado de productividad. Para cumplir con este requisito el sistema debe cumplir una serie de características como son: predecible, síntesis, familiaridad, generalización y consistencia.
    - Flexibilidad: Establece que debe haber varias formas en que el sistema y el usuario intercambian la información. Las características relacionadas con este requisito son:

control del diálogo por parte del usuario, migración de tareas, capacidad de sustitución, capacidad de adaptación.

- Robustez: Es el nivel de fiabilidad del sistema, o el grado en que el sistema es capaz de tolerar fallos durante la interacción. Las características que el sistema debe cumplir son: navegabilidad, uso de valores por defecto, persistencia, recuperación de información, tiempo de respuesta.

- Requerimientos relacionados con la necesidad de integrar el software creado en la aplicación "Flinstones":
  - El nuevo software creado deberá mantener la estructura visual seguida por el resto de la aplicación siempre que sea posible.
  - Los métodos de agregación creados deberán seguir la estructura interna similar a los que ya existen en el sistema, así mismo, deberán poder utilizarse en las mismas condiciones que los ya creados y de la misma forma.
  - El nuevo método de agregación creado deberá seguir la misma estructura que los ya existentes tanto interna como visual, además, su utilización deberá ser similar al resto de métodos para que los usuarios que ya hayan manejado otros métodos sepan la forma de utilizar el nuevo método introducido.

4.2 Análisis del sistema

Una vez conocidos los objetivos, requerimientos y limitaciones que tendrá el software que se va a desarrollar, es el momento de analizarlo y crear un modelo que sea correcto, completo, consistente, claro y verificable. Para ello, el siguiente paso será definir los diferentes perfiles de usuario, los casos de uso según los requisitos vistos en el apartado 4.1, y definir las narrativas de los mismos.

4.2.1 Perfiles de usuario

El primer paso a la hora de realizar el análisis de un sistema software consiste en determinar los posibles usuarios de la aplicación, para poder adaptar la funcionalidad a dichos usuarios. De esta forma, podremos obtener las características generales que nos permitan definir los requisitos de usabilidad que habrá que tener en cuenta en las diferentes etapas de desarrollo de software.

En este caso, se han establecido dos tipos fundamentales de usuarios distintos del sistema, dicha división se ha realizado fundamentalmente atendiendo al conocimiento que tienen dichos usuarios sobre la aplicación y el problema de toma de decisiones.

Experto: Se trata de un usuario experto en el problema de toma de decisiones, además estará familiarizado con la aplicación o con aplicaciones similares.

- Debe tener un conocimiento profundo del problema de toma de decisiones, para saber cómo debe proceder para obtener los resultados del problema planteado.

- Conocimientos medios sobre manejo de equipos informáticos, además debe tener cierto conocimiento sobre el manejo de la aplicación o sobre aplicaciones similares que le faciliten su labor.

- Normalmente, este tipo de usuarios dispone de un espacio de trabajo localizado y concreto.
- Se supone que este tipo de usuarios tienen un nivel cultural alto.

- Este tipo de usuarios debe tener conocimientos y capacidad para poder decidir cuáles serán los métodos u operadores más apropiados en cada caso.

Intermedio: Se trata de un usuario con conocimientos sobre el problema de toma de decisiones y sobre aplicaciones informáticas relacionadas con el mismo, pero dichos conocimientos no llegan al nivel de un usuario experto en el problema.

- Debe tener un buen conocimiento del problema que le permita saber qué pasos debe seguir para resolver un problema, así como, saber cuándo debe emplear cada elemento que interviene en el mismo.

- Conocimientos medios sobre manejo de equipos informáticos, además debe tener cierto conocimiento sobre el manejo de la aplicación o sobre aplicaciones similares que le faciliten su labor.

- Normalmente, este tipo de usuarios dispone de un espacio de trabajo localizado y concreto.

- El nivel cultural de este tipo de usuarios será medio-alto, ya que pueden ser usuarios involucrados en este tipo de problemas o usuarios dentro de un ámbito académico.

Como se puede ver al existir dos tipos de usuarios se deberá adaptar la aplicación para que ambos puedan manejarla de forma fácil y sacándole el máximo partido posible. Como principio básico, se debe intentar que la interacción con la aplicación sea lo más sencilla posible y que la información se ofrezca de una forma clara y concisa sin dar lugar a posibles errores de interpretación.

No se ha considerado necesario incluir un tipo de usuario novel o principiante debido a que las características de la aplicación, así como, el hecho de que su contexto de uso sea muy específico hace que sea difícil que se den usuarios sin conocimientos del problema utilicen la aplicación.

4.2.2 Casos de uso

Los casos de uso, al ser parte del análisis, nos ayudan a describir qué es lo que es sistema debe hacer. Los casos de uso definen qué hace el sistema desde el punto de vista del usuario. Es decir, describen un uso del sistema y cómo este interactúa con el usuario.

Los casos de uso son tareas con significado, coherentes y con cierta independencia, que los actores realizan de forma cotidiana al utilizar el sistema. En un mismo caso de uso podrían participar varios actores o tan solo uno. [18]

En definitiva, los casos de uso explican cómo se realiza una tarea forma precisa, y consta de los siguientes elementos:

- Nombre único e unívoco del caso de uso.
- Actores participantes.
- Condiciones de entrada.
- Flujo de eventos (narrativa).
- Condiciones de salida.
- Requerimientos especiales.

Por tanto, lo primero que debemos determinar son los actores que participan en cada uno de los casos de uso realizados.

Un actor modela una entidad externa que interacciona con el sistema, es decir, es un tipo de usuario del sistema. Al igual que ocurre con los casos de uso, los actores deben tener nombres únicos y descriptivos. En nuestro sistema tendremos únicamente el siguiente actor:

- Usuario: Se refiere a cada uno de los usuarios que utilizan los distintos servicios proporcionados por el sistema, en este tipo incluimos tanto los usuarios intermedios como los expertos comentados en el apartado anterior.

Una vez especificados los actores del sistema, se definirán los diferentes casos de uso. Para ello, un requisito esencial es que dichos casos de uso cubran los requerimientos funcionales anteriormente especificados.

El primer paso para realizar el modelo de casos de uso es establecer la funcionalidad básica del sistema. Para ello se emplea un diagrama frontera, es decir, un diagrama que describe completamente la funcionalidad del sistema (Ver Ilustración 4.1).

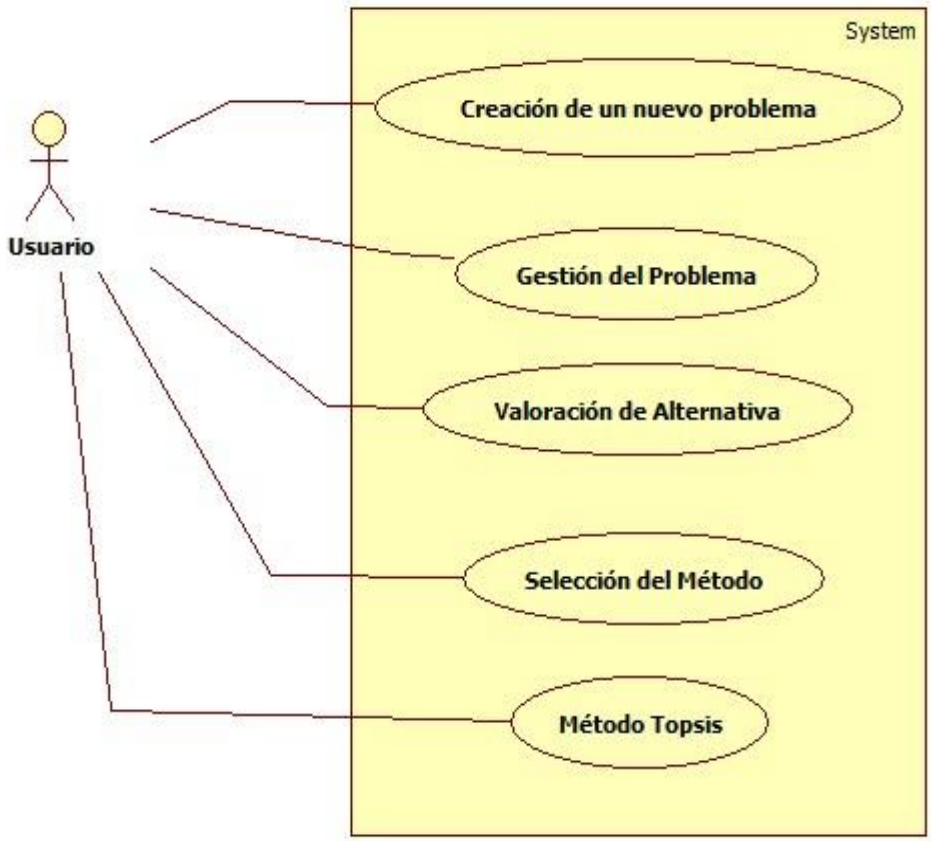


Ilustración 4.1 Diagrama frontera

Los casos de uso reflejados en un diagrama frontera pueden ser lo suficientemente precisos o, por el contrario, necesitar ser explicados con mayor detalle.

En este caso, existen varios casos de uso que deben ser tratados con mayor detalle. A continuación, en los siguientes apartados se verán con más detalle dichos casos de uso.

Diagrama de casos de uso "Creación del problema" (Ver Ilustración 4.2)

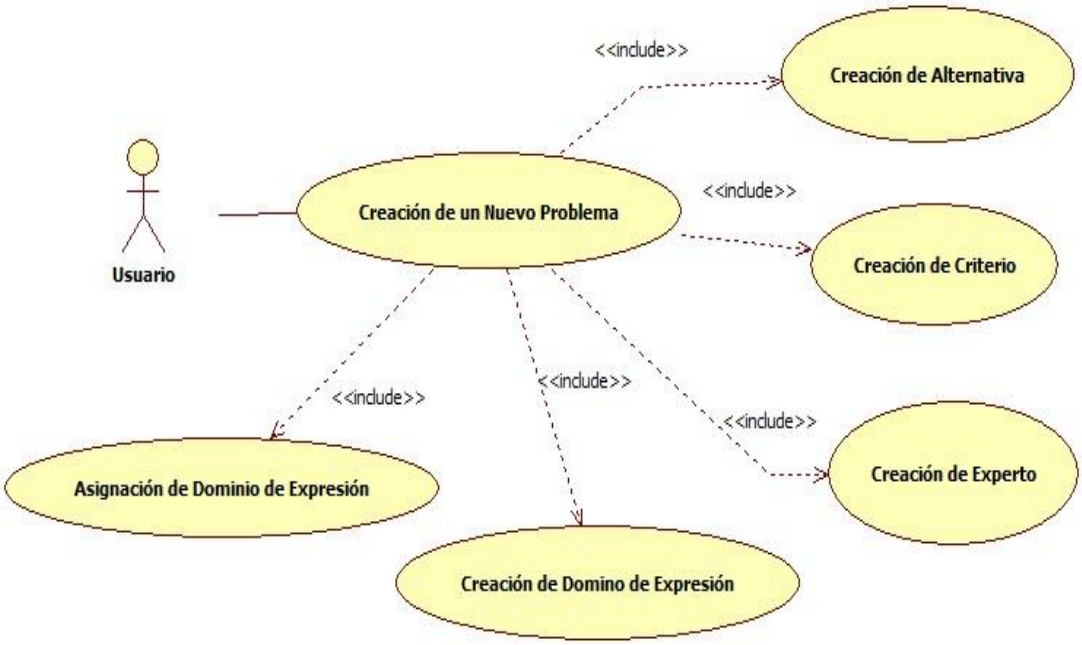


Ilustración 4.2: Diagrama de Casos de uso "Creación del problema"

Diagrama de casos de uso "Gestión del problema" (Ver Ilustración 4.3)

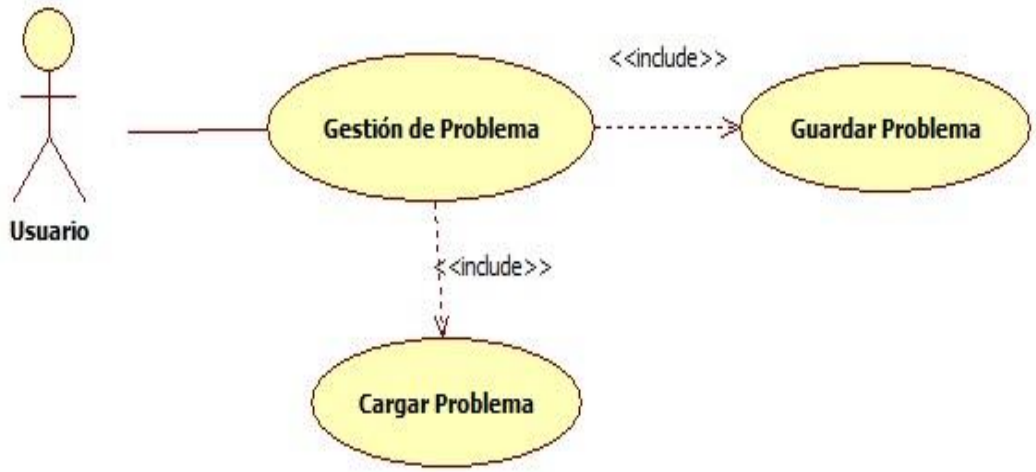


Ilustración 4.3: Diagrama de Casos de uso "Gestión del problema"

Diagrama de casos de uso "Resolución mediante Topsis" (Ver Ilustración 4.4)

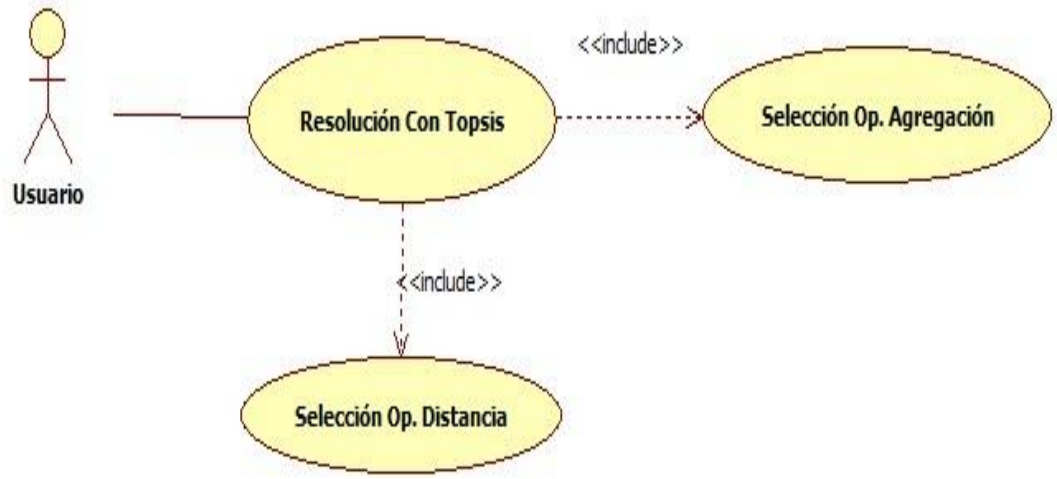


Ilustración 4.4: Diagrama de Casos de uso "Resolución mediante Topsis"

Nota: El siguiente caso de uso forma parte del anterior, pero se han separado para simplificar el diagrama previo.

Diagrama de casos de uso "Selección Operador de Agregación" (Ver Ilustración 4.5)

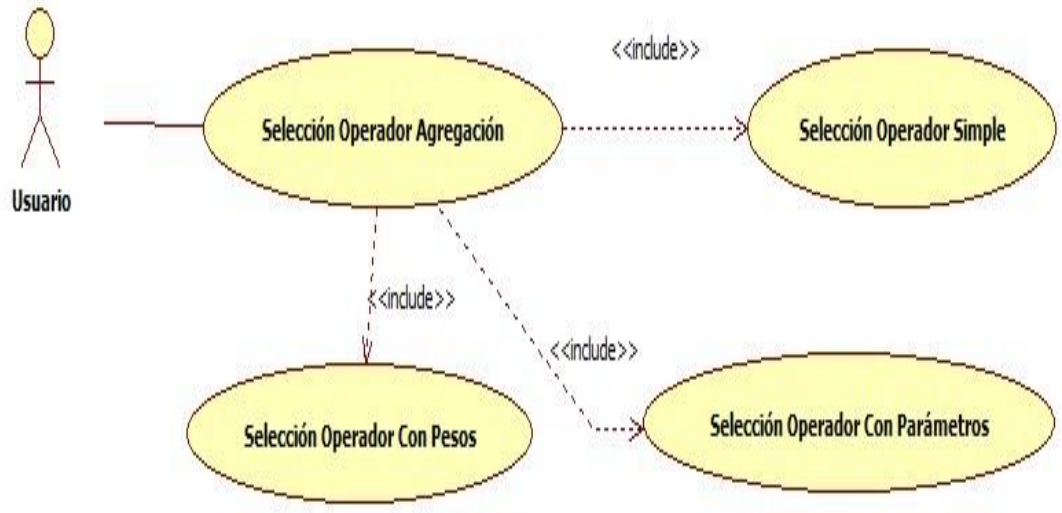


Ilustración 4.5: Diagrama de Casos de uso "Selección Operador de Agregación"

Como se puede ver en los diagramas anteriores tanto en el diagrama frontera como en los diagramas de casos de uso, existen multitud de casos de uso diferentes. Para especificarlos no basta con los diagramas anteriores, por ello, a continuación se describen de forma detallada cada uno de los casos de uso mostrados en las ilustraciones superiores.

- **Caso de Uso 1: Crear Problema**

Actores principales: Usuario

Condiciones de entrada: Ninguna.

Flujo de eventos:

1. El usuario solicita la creación de un nuevo problema.
2. Según el estado anterior del sistema:
   - 3.1 Si no existen datos de un problema previo o el problema mostrado en ese momento no ha sido modificado, el sistema sólo vuelve a la página principal.
   - 3.2 Si existen datos de un problema previo que han sido modificados:
     - 3.2.1 El sistema muestra un mensaje para guardar o descartar los cambios.
     - 3.2.2 El usuario solicita la opción que desee.
     - 3.2.3 El sistema realiza guarda los datos o los descarta y vuelve a la pantalla inicial de la aplicación.

Condiciones de salida: El sistema proporciona un problema nuevo donde el usuario podrá introducir sus elementos.

Excepciones: Ninguna.

- **Caso de Uso 2: Añadir Experto**

Actores principales: Usuario

Condiciones de entrada: El usuario se encuentra en la página inicial de la aplicación donde se pueden introducir nuevos elementos al problema.

Flujo de eventos:

1. El usuario solicita la opción de crear un nuevo experto.
2. El sistema muestra un formulario para que el usuario introduzca los datos del nuevo experto.
3. El usuario introduce los datos del experto y solicita la opción de añadirlo al problema.
4. El sistema almacena los nuevos datos introduciéndolos en el problema y muestra al usuario una confirmación de que el experto se ha añadido correctamente (E-1).

Condiciones de salida: El usuario ha introducido un nuevo experto en el sistema.

Excepciones:

E-1: Si el usuario introduce un nombre de experto igual a otro ya existente, el sistema informa del error al usuario.

- **Caso de Uso 3: Añadir Alternativa**

Actores principales: Usuario

Condiciones de entrada: El usuario se encuentra en la página inicial de la aplicación donde se pueden introducir nuevos elementos al problema.

Flujo de eventos:

1. El usuario solicita la opción de añadir una nueva alternativa.
2. El sistema muestra un formulario para que el usuario introduzca los datos de la nueva alternativa.
3. El usuario introduce los datos de la alternativa y solicita la opción de añadirla al problema.
4. El sistema almacena los nuevos datos introduciéndolos en el problema y muestra al usuario una confirmación de que la alternativa se ha añadido correctamente (E-1).

Condiciones de salida: El usuario ha introducido una nueva alternativa en el problema.

Excepciones:

E-1: Si el usuario introduce un nombre de alternativa igual a otro ya existente o vacío, el sistema informa del error al usuario y no permite introducir dicha alternativa en el problema.

- **Caso de Uso 4: Añadir Criterio**

Actores principales: Usuario

Condiciones de entrada: El usuario se encuentra en la página inicial de la aplicación donde se pueden introducir nuevos elementos al problema.

Flujo de eventos:

1. El usuario solicita la opción de añadir un nuevo criterio al problema.
2. El sistema muestra un formulario para que el usuario introduzca los datos del nuevo criterio.
3. El usuario introduce los datos del criterio y solicita la opción de añadirlo al problema.
4. El sistema almacena los nuevos datos introduciéndolos en el problema y muestra al usuario una confirmación de que el criterio se ha añadido correctamente (E-1).

Condiciones de salida: El usuario ha introducido un nuevo criterio en el problema.

Excepciones:

E-1: Si el usuario introduce un nombre de criterio igual a otro ya existente o vacío, el sistema informa del error al usuario y no permite introducir dicho criterio en el problema.

- **Caso de Uso 5: Crear Nuevo Dominio de Expresión**

Actores principales: Usuario

Condiciones de entrada: El usuario se encuentra en la página inicial de la aplicación donde se pueden introducir nuevos elementos al problema.

Flujo de eventos:

1. El usuario solicita la opción de añadir un nuevo dominio de expresión al problema.
2. El sistema muestra un formulario para añadir dicho dominio de expresión donde se indicarán cuales son los dominios que pueden ser seleccionados.
3. El usuario selecciona el dominio de expresión que desea entre las distintas opciones existentes y selecciona la opción de introducir valores.
4. El sistema muestra un formulario para introducir los valores del dominio de expresión según el tipo elegido.
5. El usuario introduce los nuevos valores y selecciona la opción de añadir dicho dominio al sistema.
6. El sistema almacena los nuevos datos introduciéndolos en el problema y muestra al usuario una confirmación de que el nuevo dominio de expresión se ha añadido correctamente (E-1).

Condiciones de salida: El usuario ha introducido un nuevo dominio de expresión en el problema.

Excepciones:

E-1: Si el usuario introduce un nombre de dominio igual a otro ya existente o vacío, el sistema informa del error al usuario y no permite introducir dicho dominio en el problema.

- **Caso de Uso 6: Asignación de Dominio de Expresión**

Actores principales: Usuario

Condiciones de entrada: El usuario se encuentra en la página inicial de la aplicación donde se pueden introducir nuevos elementos al problema.

Flujo de eventos:

1. El usuario solicita la opción de asignar un dominio de expresión a alguno de los elementos del sistema.
2. El sistema muestra un formulario seleccionar el dominio, así como, el elemento al que se desea asignar.
3. El usuario selecciona el elemento o elementos a los que desea asignar el dominio que también tendrá que seleccionar y selecciona la opción añadir asignación.
4. El sistema realiza la asignación correspondiente a los datos introducidos por el usuario y muestra al usuario una confirmación de que se ha realizado correctamente.

Condiciones de salida: El sistema realiza la asignación indicada por el usuario y la tendrá en cuenta para pasos posteriores.

Excepciones: Ninguna.

- **Caso de Uso 7: Cargar Problema**

Actores principales: Usuario

Condiciones de entrada: Existe un problema que ha sido previamente guardado en el sistema.

Flujo de eventos:

1. El usuario solicita la opción de cargar un problema guardado.
2. Según el estado anterior del sistema:
   2.1 Si no existen datos de un problema previo o el problema mostrado en ese momento no ha sido modificado, solicita al usuario la selección del archivo correspondiente con el problema previamente guardado.
   2.2 Si existen datos de un problema previo que han sido modificados:
      2.2.1 El sistema muestra un mensaje para guardar o descartar los cambios.
      2.2.2 El usuario solicita la opción que desee.
      2.2.3 El sistema guarda los datos o los descarta y solicita al usuario la selección del archivo correspondiente con el problema previamente guardado.
3. El usuario selecciona el archivo que contiene el problema que desea recuperar y selecciona la opción de confirmación.
4. El sistema limpia los datos anteriores si existen e introduce los elementos almacenados en el fichero indicado por el usuario.

Condiciones de salida: El usuario dispone de todos los elementos del problema contenido en el fichero seleccionado.

Excepciones: Ninguna.

- **Caso de Uso 8: Guardar Problema**

Actores principales: Usuario

Condiciones de entrada: El usuario ha realizado modificaciones en un problema o ha creado elementos en un problema nuevo.

Flujo de eventos:

1. El usuario solicita la opción de guardar problema.
2. Según el problema abierto en ese momento:
   2.1 Si el problema se corresponde con un archivo que ha sido cargado por el usuario previamente, el sistema guarda los datos directamente en el mismo archivo y muestra un mensaje de confirmación.
   2.2 Si el problema se corresponde con un problema nuevo:
      2.2.1 El sistema muestra un formulario donde se debe introducir los datos del archivo donde se desea guardar el problema.
      2.2.2 El usuario introduce los datos del problema y pulsa el botón aceptar.
      2.2.3 El sistema guarda los datos en el archivo con el nombre introducido por el usuario y muestra un mensaje de confirmación (E-1).

Condiciones de salida: El usuario ha guardado correctamente el problema que está actualmente en el sistema.

Excepciones:

E-1: Si el usuario introduce un nombre de archivo igual a otro existente en la misma carpeta, el sistema informará del error y no permitirá realizar el almacenamiento de la información.

- **Caso de Uso 9: Valorar Alternativa**

Actores principales: Usuario

Condiciones de entrada: El problema contiene elementos para poder realizar distintas valoraciones.

Flujo de eventos:

1. El usuario selecciona la pantalla para realizar las valoraciones de las distintas alternativas.
2. El sistema muestra la información de las distintas alternativas, así como, las valoraciones de las mismas si es que están valoradas.
3. El usuario selecciona la alternativa que desea valorar.
4. El sistema muestra un formulario para que el usuario introduzca la valoración de la alternativa seleccionada.
5. El usuario introduce la valoración de la alternativa seleccionada y confirma la acción.
6. El sistema asigna la valoración introducida a la alternativa seleccionada y muestra una confirmación de que la acción se ha realizado correctamente.

Condiciones de salida: El usuario ha realizado la valoración de una de las alternativas del problema.

Excepciones: Ninguna.

- **Caso de Uso 10: Selección Método de Resolución**

Actores principales: Usuario

Condiciones de entrada: El usuario ha realizado todas las valoraciones correspondientes a los elementos introducidos en el problema.

Flujo de eventos:

1. El usuario selecciona la pantalla para resolver el problema que está abierto en ese momento (E-1).
2. El sistema muestra al usuario un formulario para seleccionar el método de resolución que desee aplicar para resolver el problema.
3. El usuario selecciona el problema de resolución que desea aplicar para resolver el problema.
4. El sistema muestra la información sobre el método de resolución seleccionado y habilita las nuevas pantallas adaptadas a cada uno de los métodos diferentes que se pueden seleccionar (E-2).

Condiciones de salida: El usuario ha realizado la selección del método de resolución con el que resolver el problema y el sistema ha iniciado el proceso de resolución del mismo aplicando dicho método de resolución.

Excepciones:

E-1: Si el usuario selecciona la opción de resolver el problema y no ha realizado todas las valoraciones necesarias, el sistema informará que debe realizarlas y no habilitará la pantalla de selección del método.

E-2: Si las condiciones del método de resolución seleccionado no se adaptan con las características del problema, el sistema informará del error y no iniciará la resolución de dicho problema.

Nota: Los siguientes casos de uso 11 (12,13,14) y 15 especifican los pasos en los que se ha dividido el método Topsis para facilitar su resolución, sólo se especifica este método debido a que es el implementado en el desarrollo del proyecto.

- **Caso de Uso 11: Selección Método de Agregación para Expertos (Topsis)**

Actores principales: Usuario

Condiciones de entrada: El usuario ha seleccionado el método de resolución Topsis.

Flujo de eventos:

1. El usuario selecciona la pantalla de método Topsis correspondiente con el paso de agregación de expertos.
2. Según el número de expertos definidos en el problema:
   - 2.1 Si sólo existe un experto, el sistema muestra los resultados de este primer paso sin dar opción de seleccionar el operador de agregación ya que no es necesario (E-1).
   - 2.2 Si existen varios expertos:
     - 2.2.1 El sistema muestra al usuario el conjunto de operadores de agregación que puede utilizar para realizar la agregación por expertos para el problema que se está solucionando.
     - 2.2.2 El usuario selecciona el operador de agregación que desea aplicar para realizar la agregación.
     - 2.2.3 El sistema realiza la agregación por expertos y muestra al usuario los resultados de la misma (E-1).

Condiciones de salida: El usuario realiza la agregación por expertos del método Topsis seleccionando el operador de agregación apropiado.

Excepciones:

E-1: El usuario selecciona un operador de agregación y las valoraciones no cumplen los requisitos de dicho operador, el sistema informa del error y no realiza la agregación.

Nota: Para no complicar el caso de uso se han realizado los casos de uso 12, 13 y 14 según los distintos operadores que se pueden utilizar y sus características propias.

- **Caso de Uso 12: Selección Operador de Agregación sin parámetros ni pesos**

Actores principales: Usuario

Condiciones de entrada: El usuario se encuentra ante la situación de elegir un operador de agregación para expertos, existiendo más de un experto en el problema.

Flujo de eventos:

1. El usuario selecciona un operador de agregación para expertos que no requiere ni parámetros ni pesos para su funcionamiento.
2. El sistema realiza la agregación mediante el parámetro seleccionado y con las valoraciones del problema (E-1).

Condiciones de salida: El sistema realizar la agregación mediante el operador sin pesos ni parámetros seleccionado por el usuario.

Excepciones:

E-1: Si el usuario selecciona un operador de agregación y las valoraciones no cumplen los requisitos de dicho operador, el sistema informa del error y no realiza la agregación.

- **Caso de Uso 13: Selección Operador de Agregación con parámetros**

Actores principales: Usuario

Condiciones de entrada: El usuario se encuentra ante la situación de elegir un operador de agregación para expertos, existiendo más de un experto en el problema.

Flujo de eventos:

1. El usuario selecciona un operador de agregación para expertos que requiere parámetros para obtener la solución.
2. El sistema solicita al usuario mediante un formulario los parámetros necesarios para poder realizar la agregación.
3. El usuario introduce los valores de los parámetros y selecciona la opción para realizar la agregación.
4. El sistema valida los datos introducidos y realiza la agregación mostrando al usuario el resultado de la misma (E-1).

Condiciones de salida: El sistema realizar la agregación mediante el operador con parámetros seleccionado por el usuario.

Excepciones:

E-1: Si el usuario selecciona un operador de agregación y las valoraciones no cumplen los requisitos de dicho operador o los parámetros introducidos son erróneos, el sistema informa del error y no realiza la agregación.

- **Caso de Uso 14: Selección Operador de Agregación con pesos**

Actores principales: Usuario

Condiciones de entrada: El usuario se encuentra ante la situación de elegir un operador de agregación para expertos, existiendo más de un experto en el problema.

Flujo de eventos:

1. El usuario selecciona un operador de agregación para expertos que requiere pesos para obtener la solución.
2. El sistema solicita al usuario mediante un formulario los pesos necesarios para poder realizar la agregación.
3. El usuario introduce los distintos pesos y selecciona la opción para realizar la agregación.
4. El sistema valida los datos introducidos y realiza la agregación mostrando al usuario el resultado de la misma (E-1).

Condiciones de salida: El sistema realizar la agregación mediante el operador con pesos seleccionado por el usuario.

Excepciones:

E-1: Si el usuario selecciona un operador de agregación y las valoraciones no cumplen los requisitos de dicho operador o los pesos introducidos son erróneos, el sistema informa del error y no realiza la agregación.

- **Caso de Uso 15: Selección Operador Distancia (Topsis)**

Actores principales: Usuario

Condiciones de entrada: El usuario ha seleccionado el método de resolución Topsis y ha realizado la agregación de expertos propia de dicho método.

Flujo de eventos:

1. El usuario selecciona la pantalla de cálculo de distancias entre las distintas alternativas y la solución ideal.
2. El sistema solicita al usuario la elección del operador de distancias para el cálculo de las mismas.
3. El usuario selecciona el operador de distancias para realizar los cálculos.
4. Según el operador seleccionado:
   - 4.1 Si el operador de seleccionado no requiere pesos, el sistema realiza los cálculos y devuelve los resultados al usuario.
   - 4.2 Si el operador seleccionado requiere pesos:
     - 4.2.1 El sistema solicita al usuario mediante un formulario los pesos necesarios.
     - 4.2.2 El usuario introduce los pesos y selecciona la opción para confirmar la operación.
     - 4.2.3 El sistema realiza los cálculos y devuelve los resultados al usuario (E-1).

Condiciones de salida: El sistema aporta los cálculos obtenidos tras aplicar el método Topsis al conjunto de datos disponible en el problema.

Excepciones:

E-1: Si los pesos introducidos por el usuario son erróneos, el sistema informa del error y no realiza los cálculos.

4.2.3 Escenarios

En el apartado anterior se han definido los diferentes casos de uso, estos diagramas aportan una representación abstracta de las distintas funcionalidades que incluye el sistema de recomendación. Para realizar una definición más concreta y precisa de esta de dicha funcionalidad se utilizan los escenarios que muestran las posibles interacciones entre el sistema y los usuarios del mismo.

Los escenarios son interacciones ficticias con el sistema que describen posibles usos que los distintos usuarios puedan hacer del mismo, así como, la forma en la que interactuarán con la interfaz. Es una herramienta muy útil para los diseñadores y desarrolladores de software ya que les permiten anticiparse a posibles problemas, así como, detectar que estructura de la interfaz se adapta mejor a los diferentes usuarios. A pesar de ser ficticias, deben poseer todo tipo de detalles para facilitar a los diseñadores a centrarse en una situación concreta de uso y no manejar un concepto más abstracto [18].

Este hecho, además, ayuda a que los responsables del diseño y desarrollo del software conozcan mejor a los usuarios potenciales de dicho software, así como, para refinar el conjunto de actividades y los objetivos que persiguen con su uso.

Al igual que los casos de uso, los escenarios están compuestos de una serie de elementos básicos que lo definen:

- Un nombre único y unívoco.
- Una descripción.
- Los actores participantes en la acción descrita.
- El flujo de eventos que define a la acción.

Debido al hecho de ser algo ficticio, el número de escenarios puede ser inmenso. En este caso, se van a definir los escenarios que definen las actividades más importantes. Estos escenarios serán:

- **Escenario 1: Creación Del Problema.**

Nombre: Creación del problema

Descripción: Antonio Martín un usuario habitual de sistemas de soporte a la decisión desea crear un nuevo problema.

Actores principales: Antonio Martín

Flujo de eventos:

1. El usuario selecciona opción de nuevo problema.
2. El sistema muestra la página inicial.
3. El usuario selecciona la opción añadir criterios.
4. El sistema muestra el formulario con los datos del criterio que se desea añadir.
5. El usuario introduce los datos del criterio y pulsa aceptar.
6. El sistema actualiza la información del problema con los mismos datos.
7. El usuario repite los pasos para la creación del resto de criterios, alternativas y expertos, así como, para los dominios de expresión
8. El sistema actualiza la información del problema.
9. El usuario asigna a todos el mismo dominio de expresión, en este caso numérico, a todos los elementos del problema.
10. El sistema muestra la información del problema actualizada al usuario.

Escenario 4.1: Creación Del Problema

- **Escenario 2: Valoración Del Problema**

Nombre: Valoración del Problema

Descripción: Antonio Martín un usuario habitual de sistemas de soporte a la decisión desea realizar las valoraciones de acuerdo a los elementos del problema que ha creado previamente.

Actores principales: Antonio Martín

Flujo de eventos:

1. El usuario selecciona opción de valorar alternativas.
2. El sistema muestra la información relativa a las diferentes alternativas del sistema.
3. El usuario selecciona las alternativas que desea valorar.
4. El sistema muestra el formulario para que el usuario introduzca la valoración deseada.
5. El usuario introduce la valoración de la alternativa deseada y pulsa aceptar.
6. El sistema actualiza la información de las alternativas con los nuevos datos introducidos.
7. El usuario repite los pasos 3-6 para todas las alternativas que desea valorar.
8. El sistema actualiza la información de las distintas alternativas con las valoraciones realizadas.

Escenario 4.2: Valoración del Problema

- **Escenario 3: Resolución Mediante Topsis**

Nombre: Creación del problema

Descripción: Antonio Martín un usuario habitual de sistemas de soporte a la decisión desea crear un nuevo problema.

Actores principales: Antonio Martín

Flujo de eventos:

1. El usuario selecciona opción de resolver problema.
2. El sistema solicita la elección del método.
3. El usuario selecciona el método Topsis.
4. El sistema comprueba que los datos son correctos y muestra el primer paso del método.
5. El usuario selecciona un operador de agregación con pesos y pulsa aceptar.
6. El sistema solicita los pesos necesarios para realizar la agregación.
7. El usuario introduce los pesos y pulsa aceptar.
8. El sistema realiza la agregación por expertos, muestra los resultados al usuario y habilita la pantalla correspondiente al segundo paso del método.
9. El usuario selecciona el segundo paso del método.
10. El sistema solicita el operador de distancias que se desea utilizar.
11. El usuario selecciona el operador de distancias sin pesos.
12. El sistema realiza los cálculos y muestra los resultados al usuario.

Escenario 4.3: Resolución Mediante Topsis

4.2.4 Modelo del dominio

Para finalizar el análisis del sistema y a partir de la descripción de los requisitos del problema (apartado 4.1) y de los casos de uso descritos anteriormente (apartado 4.2.2), se realiza el modelo de dominio.

El modelo de dominio se utiliza con frecuencia como fuente de inspiración para el diseño posterior de los objetos software. El modelo de dominio muestra a los diseñadores y desarrolladores de software un conjunto de clases conceptuales significativas en un dominio concreto de problema [19].

Este diagrama se usa para ver cuáles son los objetos de interés dentro de nuestra aplicación y como se interrelacionan entre ellos. Hay que tener en cuenta que en el modelo del dominio no se representan componentes software, sino un "diccionario visual" de los objetos conceptuales de nuestro problema. Para crearlo, en primer lugar se deben identificar las clases conceptuales del dominio del problema y después se representaran esas clases en un diagrama UML, en el que también se añadirán asociaciones y atributos que ayuden a entender la solución aportada. [18-19]

Un modelo de dominio representa un conjunto de clases u objetos conceptuales dentro del contexto de un problema, pero para los que no se les define ninguna operación. Suelen mostrar los siguientes elementos:

- Objetos del dominio o clases conceptuales.
- Asociaciones entre las distintas clases conceptuales.
- Atributos de las clases conceptuales.

El modelo de dominio constituye una forma simple y concisa de representar la información relativa al problema que se intenta solucionar. Esto supone una ventaja ya que de una forma sencilla se pueden entender los diferentes elementos que forman parte del problema, así como, la forma en la que se relacionan entre ellos a partir del procesamiento visual del diagrama correspondiente.

A continuación se muestra el modelo de dominio de la aplicación desde un punto de vista conceptual, sin entrar en detalles de desarrollo e implementación:

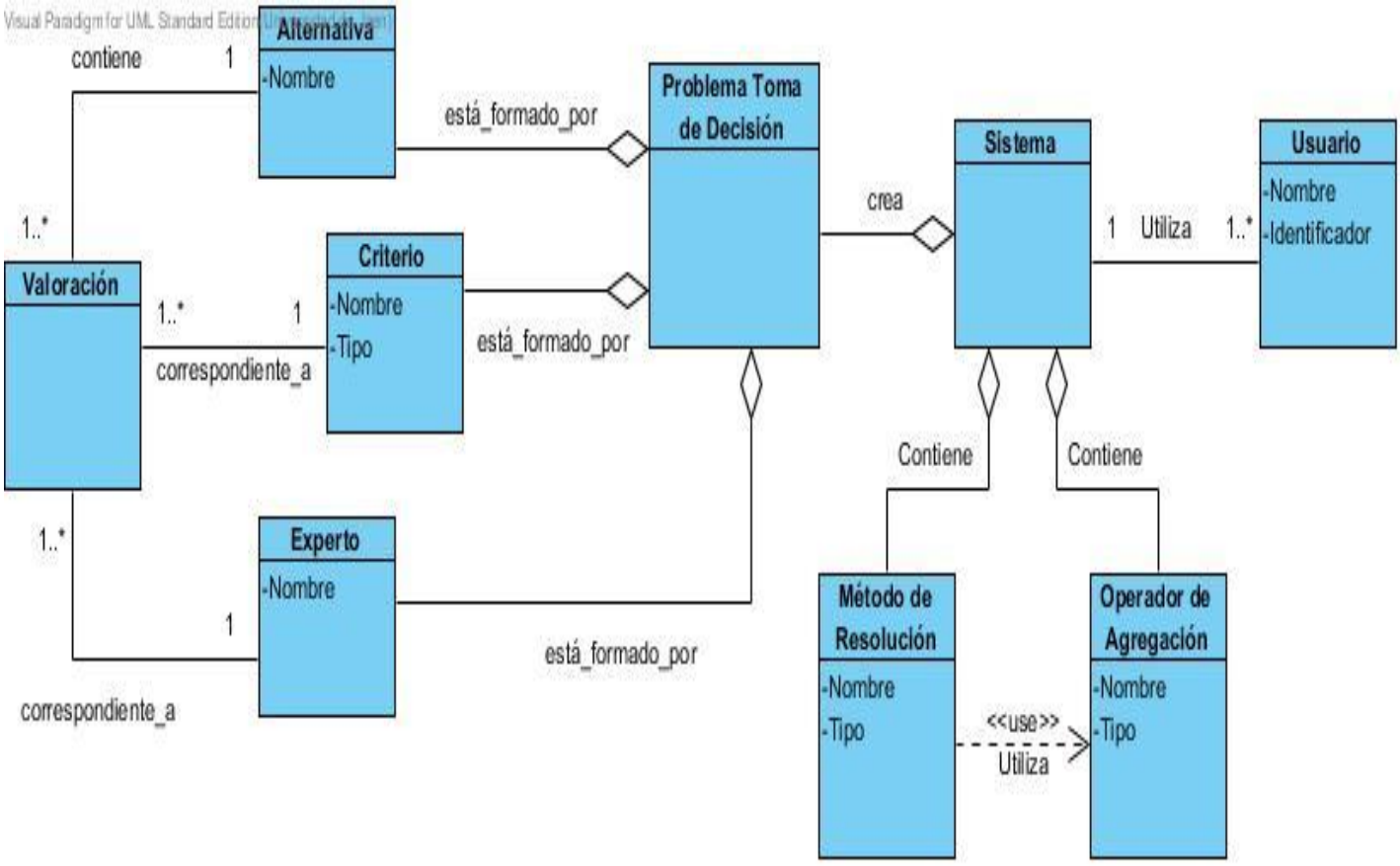


Ilustración 4.6 Modelo de dominio

A continuación, se describen las clases presentes en el modelo de dominio anterior para especificar lo que representan, así como, el significado de las relaciones que tiene cada uno de ellos:

- **Usuario**

Clase que representa a todos los usuarios que utilizan el sistema para resolver distintos problema de toma de decisión. Esta clase no tendrá ningún componente software asociado, ya que la aplicación no dispone de acceso diferenciado por usuarios ni guarda ningún tipo de información sobre los mismos. Aun así, se ha incluido en modelo de dominio, ya que se trata de un concepto importante a la hora de resolver un problema de toma de decisiones.

- **Sistema**

Esta clase representará a la aplicación en sí, por lo que a partir de ella se podrán crear los distintos elementos del problema, contendrá los diferentes métodos de resolución y operadores de agregación implementados y dará acceso a los usuarios para utilizar la funcionalidad suministrada por la aplicación. En este caso, si existirá un componente software encargado de gestionar la aplicación, pero sólo existirá una instancia de dicha clase.

- **Método de Resolución**

Clase que representa los diferentes métodos de resolución disponibles en el sistema. Dicho método de resolución será utilizado para resolver los distintos problemas de resolución, empleando para ello determinados operadores de agregación cuando sea necesario.

- **Operador de Agregación**

Clase que representa los diferentes operadores de agregación disponibles en el sistema. Los operadores de agregación serán utilizados por los diferentes métodos de resolución para resolver los problemas de toma de decisiones que se planteen, ya que no tiene sentidos utilizarlos por sí mismos, fuera del método de resolución concreto.

- **Problema de Toma de Decisión**

Clase que representa un problema de toma de decisión concreto creado por el usuario a través del sistema y que se solucionará empleando diferentes métodos de resolución. Estará compuesto por una serie de elementos básicos que forman parte de este tipo de problemas: criterios, expertos y alternativas.

- **Criterio**

Clase que representa un criterio concreto definido dentro de un problema de toma de decisiones. Este elemento no tiene sentido por sí mismo, asociando su existencia siempre a un problema de toma de decisión concreto.

- **Alternativa**

Clase que representa una alternativa concreta definida dentro de un problema de toma de decisiones. Este elemento no tiene sentido por sí mismo, asociando su existencia siempre a un problema de toma de decisión concreto.

- **Experto**

Clase que representa un experto concreto definido dentro de un problema de toma de decisiones. Este elemento no tiene sentido por sí mismo, asociando su existencia siempre a un problema de toma de decisión concreto.

- **Valoración**

Clase que representa un criterio concreto definido dentro de un problema de toma de decisiones. Este elemento no tiene sentido por sí mismo, asociando su existencia siempre a un problema de toma de decisión concreto.

Capítulo 5:

Diseño del software

No cabe duda de que realizar de forma adecuada cada una de las actividades que conlleva la Ingeniería del Software es imprescindible para la realización de un proyecto de calidad. Por ello, no se puede afirmar que ninguna de estas actividades sea más importante que el resto. Sin embargo, sí podemos decir que la actividad de diseño es la más delicada y laboriosa de realizar.

- Es delicada ya que, de no hacerse correctamente, se hace imposible el codificar de manera correcta en la implementación el modelo obtenido en la fase de análisis del sistema.
- Es laboriosa ya que las estrategias a seguir para que la traducción entre modelo y código se lleve a cabo correctamente son muy diversas y bastante complejas.

Se puede decir, que el diseño del sistema es la actividad de la Ingeniería del Software en la que se identifican los objetivos finales del sistema y se plantean diversas estrategias para alcanzarlos en la actividad de implementación. [18]

Sin embargo, el sistema no se suele diseñar de una sola pasada, sino que hemos de diferenciar entre el diseño y estructura de los datos que se van a manejar, y el diseño de la interfaz entre la aplicación y el usuario. Estas dos fases de diseño se realizan de forma paralela y se finalizan de forma simultánea. [18]

Por lo tanto, una vez realizada la especificación de requerimientos y el análisis del sistema (capítulo 4), se pasará a realizar el diseño del mismo. A continuación se expone el proceso a seguir para conseguir dicho objetivo, en el que se establecerán las estrategias a llevar a cabo para conseguir el propósito definido al inicio del proyecto.

En los siguientes apartados, se verán cada una de las fases o etapas del diseño del sistema.

5.1 Diagrama de clases de diseño

Un diagrama de clases de diseño es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, con sus atributos y operaciones, y las relaciones entre ellos. Estos diagramas son utilizados durante el proceso de diseño del software, donde a partir de los datos recopilados en la fase anterior de análisis, se realiza un diseño de la estructura que tendrá el sistema y de los principales componentes que tendrán para aportar la funcionalidad necesaria. A lo largo del proceso, este diagrama puede sufrir modificaciones para adaptarlo al resultado final de la aplicación. [18]

Un diagrama de clases de diseño muestra la especificación para las clases software de una aplicación. Los principales componentes de un diagrama de clases de diseño son los siguientes:

- Clases: Se corresponde con el componente principal de este tipo de diagramas. Están compuestos a su vez por un nombre descriptivo del elemento que representan, un conjunto de atributos y las operaciones que ofrecen al resto de elementos.

- Relaciones: Representa una conexión semántica entre elementos del problema. Existen 4 tipos de relaciones:

    - Generalización: Es una relación de especialización.
    - Asociación: Es una relación estructural. Pueden existir dos tipos: agregación y composición.
    - Realización: Es una relación contractual, en la que una clase requiere una serie de condiciones para que exista la relación.
    - Dependencia: Representa una relación de uso.

A diferencia del modelo de dominio o modelo conceptual (Ilustración 4.6) visto en el apartado 4.2.4, un diagrama de clases de diseño muestra definiciones de entidades software en lugar de los conceptos del mundo real mostrados por el modelo de dominio.

De esta forma, puede existir conceptos del mundo real que no tengan correspondencia en el diagrama de clases, sólo aparecerán aquellas que tengan asignada alguna funcionalidad o responsabilidad en la funcionalidad que debe aportar el sistema. Así mismo, pueden surgir nuevas clases software que sean necesarias añadir y que no tengan un objeto del mundo real relacionado.

A la hora de realizar el diagrama de clases del sistema, se debe recurrir a patrones de diseño que solucionen o sirvan para la modelización del mismo, siempre que sea posible. Estos patrones son soluciones a problemas de diseño que podemos reutilizar y que facilitan mucho la tarea de diseño e implementación posterior.

La aplicación "Flinstones", como se ha dicho en apartados anteriores, realiza una separación de responsabilidades para conseguir un código estructurado y que permita extender la funcionalidad del sistema de manera fácil y sencilla.

La estructuración realizada en "Flinstones" consiste en definir diferentes módulos que permiten separar los componentes principales de la aplicación, de esta forma existirá una separación entre la lógica principal del DSS, el conjunto de elementos de representación de la interfaz, los métodos de resolución incluidos y los operadores de agregación implementados para resolver los problemas de toma de decisiones.

Los principales módulos básicos que forman parte de la aplicación son los siguientes:

- Bibliotecas o librerías: Estas librerías ofrecen las estructuras y procedimientos con el objetivo de apoyar y facilitar la resolución de problemas de decisión. Estas librerías incluyen elementos tales como: los expertos, los criterios, las alternativas y los dominios de expresión.

- Interfaz gráfica de usuario: La interfaz de usuario (GUI) permite a los usuarios interactuar con el sistema de soporte a la decisión.

- Método de resolución: Estos métodos desarrollan las diferentes metodologías basadas en distintos tipos de datos (numéricos, lingüísticos, etc.), con el fin de resolver los problemas de toma de decisiones.

- Operadores de agregación: Estos operadores implementan el conjunto de operadores de agregación que se han utilizado para agregar la distinta información involucrada en el problema de decisión.

Las principales ventajas que aporta la separación de los principales componentes del sistema software según la funcionalidad que aporte son:

- Permite agregar métodos de resolución y atributos de manera sencilla y sin necesidad de realizar modificaciones en la interfaz del sistema.

- Del mismo modo, permite realizar modificaciones en la interfaz del sistema, modificando unos paquetes concretos sin necesidad de tener que cambiar otros componentes que no estén relacionados.

- Con esto, se consigue reducir el tiempo dedicado a las labores de mantenimiento del software, ya que se separa claramente los diferentes componentes del mismo.

- Así mismo, permite extender el software y realizar cambios en el mismo sin necesidad de paralizar el sistema por completo, también se pueden crear nuevos operadores y métodos de agregación de forma independiente y, posteriormente, añadirlos al sistema de forma fácil y rápida.

La estructura básica del sistema anteriormente comentada se resume en el siguiente diagrama de paquetes:

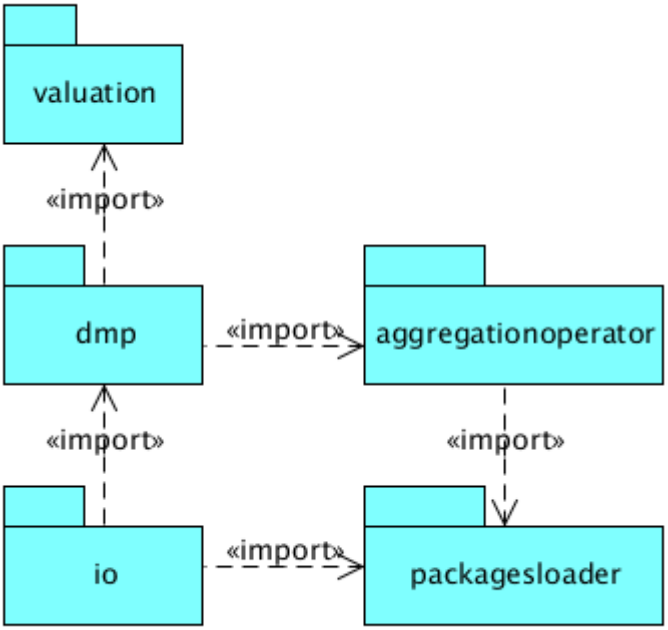


Ilustración 5.1 Diagrama de Paquetes

Como se puede ver en la imagen anterior, existen varios paquetes que contienen los distintos elementos que forman la estructura interna de la aplicación. A continuación, se detalla el contenido de cada paquete:

- Valuation: Contiene las clases relacionadas con las evaluaciones del sistema, así como, los diferentes dominios que se pueden utilizar.

- Dmp: Este paquete contiene los diferentes métodos incluidos en el sistema, así como, los pasos en los que se dividen dichos métodos, también contiene la representación gráfica asociada a cada uno de estos pasos.

- IO: Contiene clases necesarias para la lectura y escritura de los datos en archivo.

- AggregationOperator: Este paquete contiene los diferentes operadores de agregación incluidos en el sistema.

- PackageLoader: Contiene clases encargadas de gestionar la interacción entre el resto de paquetes.

Una vez vista la estructura básica del sistema, se expondrá de manera más detallada aquellas partes del diagrama de clases que han sido modificadas para añadir la funcionalidad desarrollada a lo largo de la realización del proyecto.

En primer lugar, se verán las modificaciones realizadas en el diagrama de clases (Ilustración 2.5 del capítulo 2.5.4.1) para añadir los nuevos operadores de agregación. A continuación se muestra el diagrama de clases modificado:

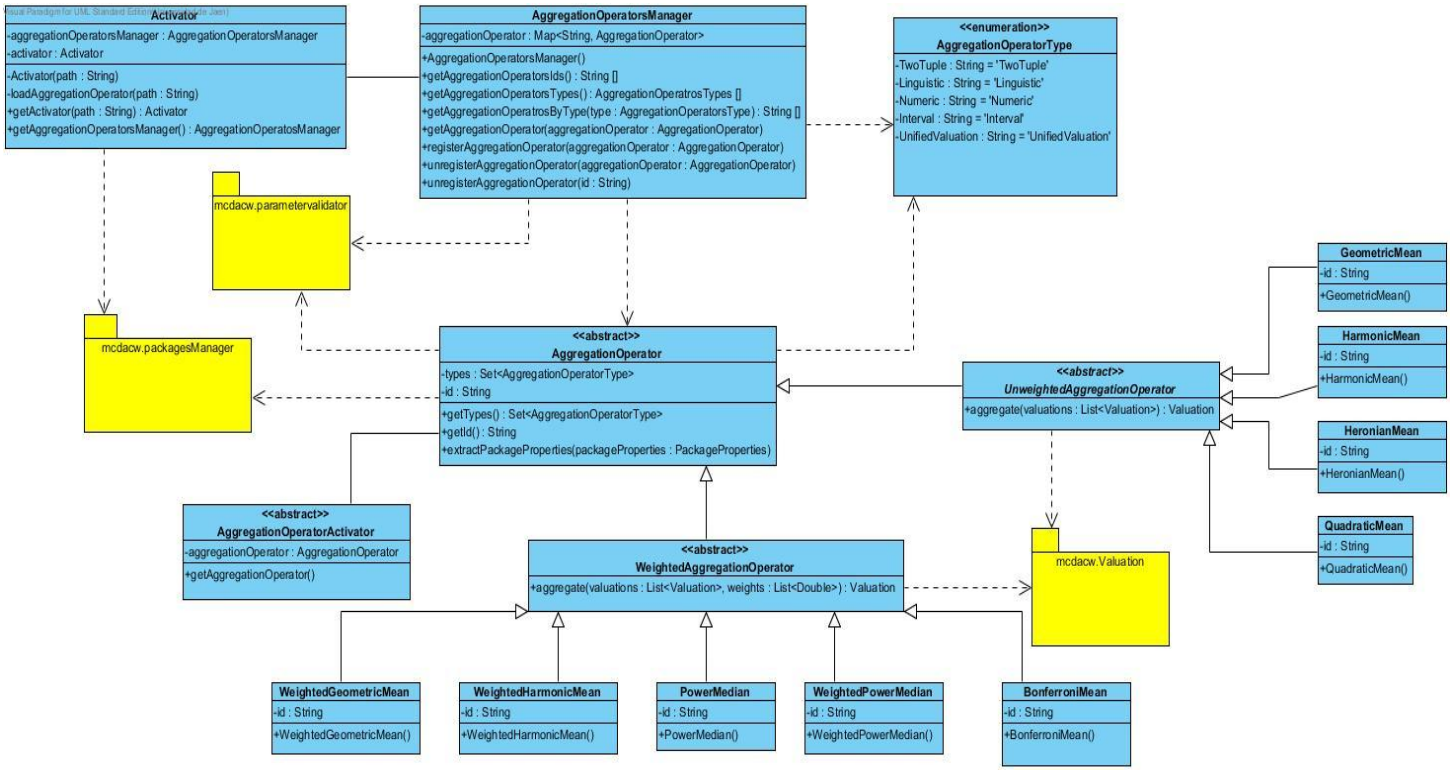


Ilustración 5.2 Diagrama de clases Operador de Agregación Modificado

En esta imagen se puede como se han añadido los nuevos operadores de agregación al sistema. Como se ha dicho en apartados anteriores, los operadores implementados se dividen en dos tipos: operadores que requieren pesos o parámetros y operadores simples, en la imagen se puede observar que cada operador implementando extiende de una clase más genérica según pertenezca a un tipo u otro.

El hecho de que los operadores extiendan de otras clases hace que sean necesarias definir una serie de operaciones de forma obligatoria posteriormente. Esto debe ser tenido en cuenta a la hora de implementar dichos operadores de agregación.

Una vez visto las modificaciones llevadas a cabo en el diagrama de clases para añadir los nuevos operadores de agregación, se pasa a exponer las modificaciones realizadas en el diagrama de clases (Ilustración 2.7 del capítulo 2.5.4.3) para añadir el nuevo método de resolución. A continuación se muestra el diagrama de clases modificado:

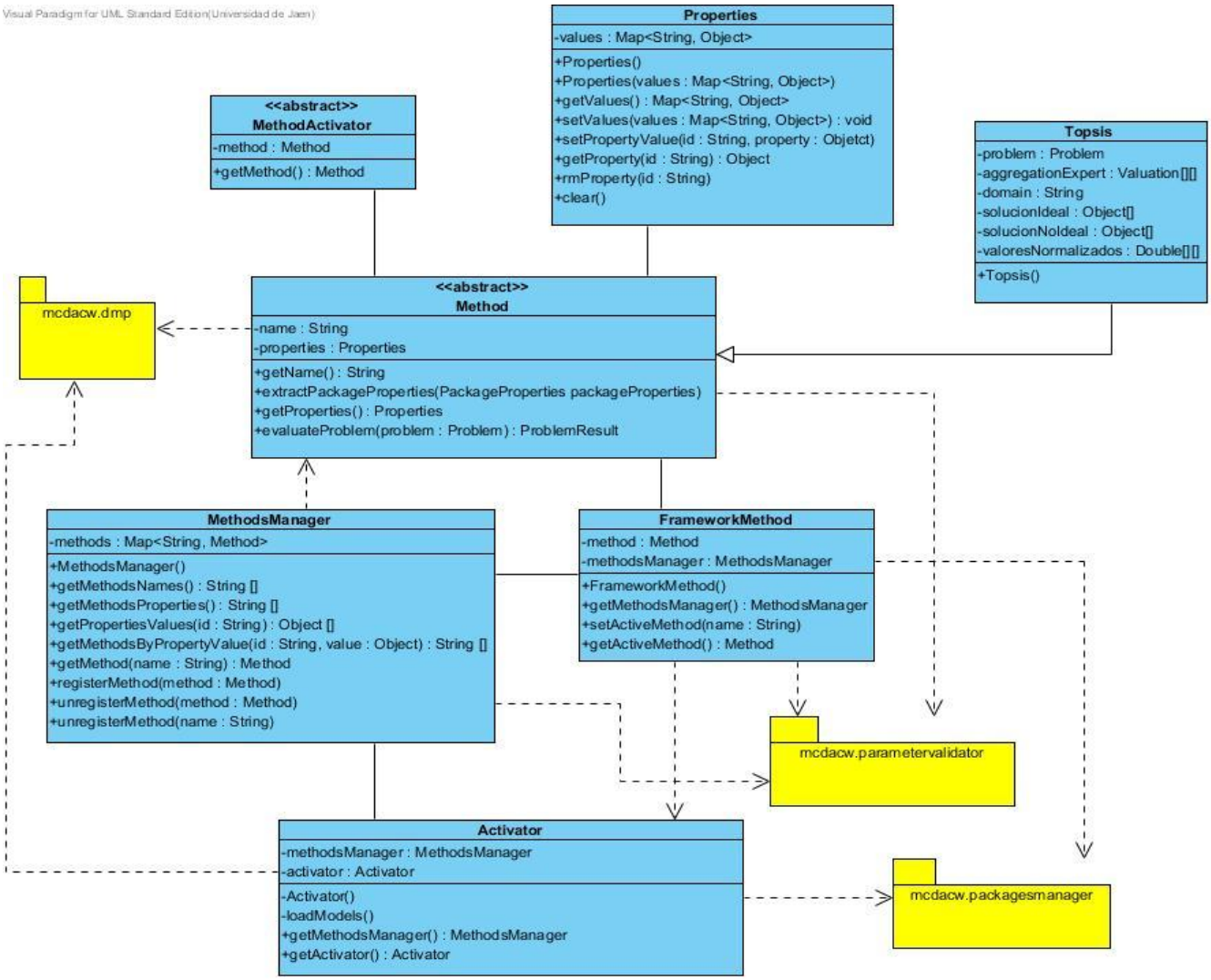


Ilustración 5.3 Diagrama de clases Método de Resolución Modificado

En la imagen superior se puede como se ha añadido el nuevo método de resolución al diagrama de clases que representa el sistema implementado. El hecho de que los métodos extiendan de otras clases hace que sean necesarias definir una serie de operaciones de forma obligatoria posteriormente. Esto debe ser tenido en cuenta a la hora de implementar el método de resolución.

En el caso de los métodos de resolución es necesario incorporar otra serie de clases relacionadas con los distintos pasos que componen dicho método. A continuación se muestran las modificaciones realizadas en el diagrama de clases (Ilustración 2.8 del capítulo 2.5.4.3) que representan los pasos de los distintos métodos de resolución.

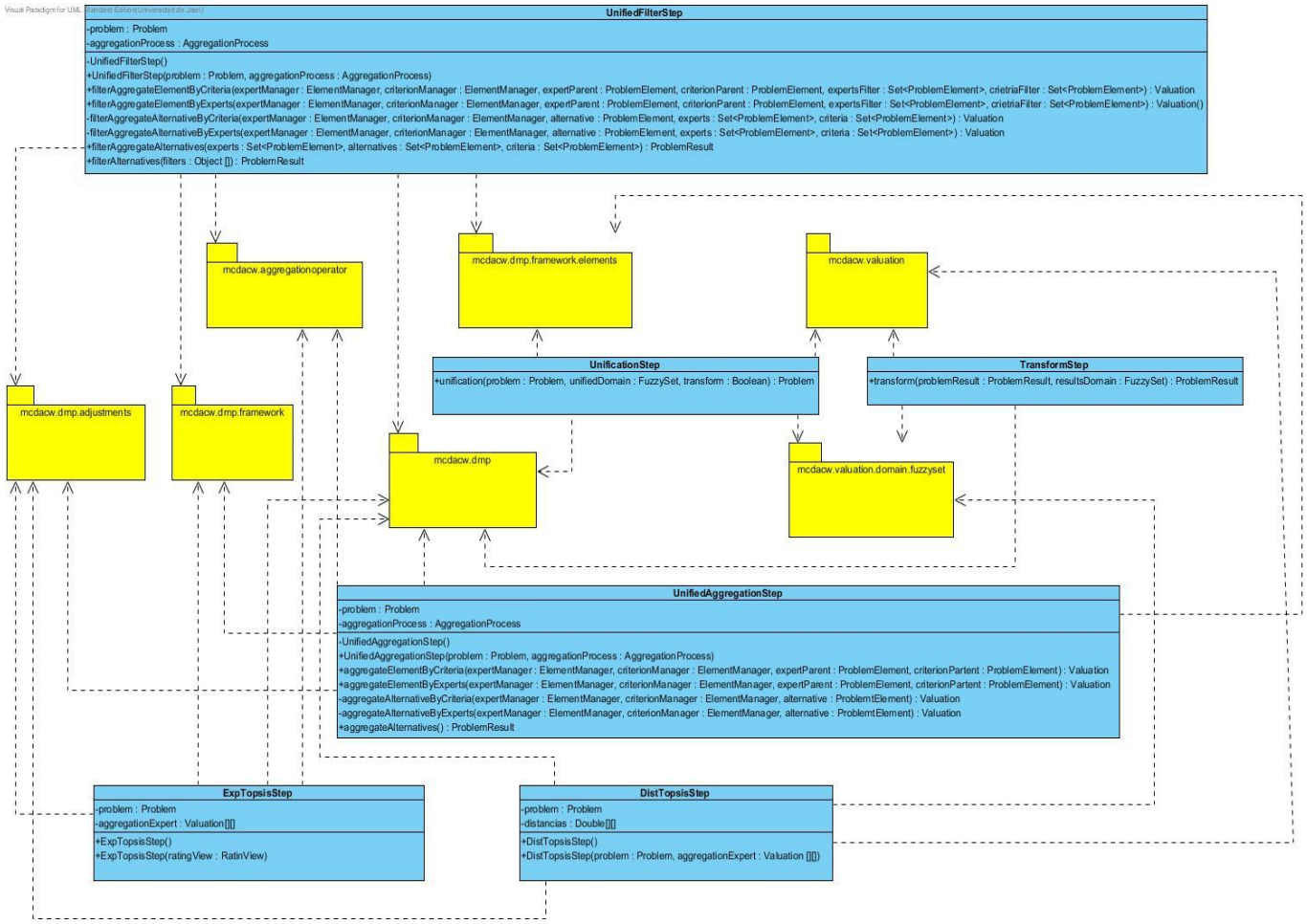


Ilustración 5.4 Diagrama de clases pasos Método Resolución Modificado

En esta imagen podemos ver como se han añadido los dos pasos en los que se divide el método de resolución Topsis, ambos pasos tiene una serie de dependencias con otros paquetes que son necesarias para realizar correctamente las diferentes operaciones involucradas en el proceso de resolución. El primer paso "ExpTopsisStep" se corresponde con la agregación por expertos que es necesario realizar mediante el operador de agregación seleccionado, mientras que el segundo paso se encarga de realizar el cálculo de los coeficientes de cercanía entre las alternativas y la solución ideal con el operador de distancia seleccionado.

5.2 Diagramas de interacción (secuencia)

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso. Mientras que el diagrama de casos de uso permite el modelado de una vista del escenario, el diagrama de secuencia contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario y mensajes intercambiados entre los objetos. [18]

Típicamente se examina la descripción de un caso de uso para determinar qué objetos son necesarios para la implementación del escenario. Si se dispone de la descripción de cada caso de uso como una secuencia de varios pasos, entonces se puede "caminar sobre" esos pasos para descubrir qué objetos son necesarios para que se puedan seguir los pasos. Un diagrama de secuencia muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales.

Existen dos tipos de mensajes: sincrónicos y asincrónicos.

- Los mensajes sincrónicos se corresponden con llamadas a métodos del objeto que recibe el mensaje. El objeto que envía el mensaje queda bloqueado hasta que termina la llamada. Este tipo de mensajes se representan con flechas con la punta rellena.
- Los mensajes asincrónicos terminan inmediatamente, y crean un nuevo hilo de ejecución dentro de la secuencia. Se representan con flechas con la cabeza abierta. También se representa la respuesta a un mensaje con una flecha discontinua.

Los mensajes se dibujan cronológicamente desde la parte superior del diagrama a la parte inferior; la distribución horizontal de los objetos es arbitraria. Más tarde, durante el diseño, el nombre es reemplazado con el nombre del método que está siendo llamado por un objeto en el otro. El método llamado, o invocado, pertenece a la definición de la clase instanciada por el objeto en la recepción final del mensaje.

En el proyecto, se han realizado 4 diagramas de secuencia correspondientes a los casos de uso más importantes de los bloques funcionales definidos en la especificación de requerimientos. A continuación se puede ver los diagramas de secuencia:

Diagrama de secuencia "Crear Problema" (Ilustración 5.5):

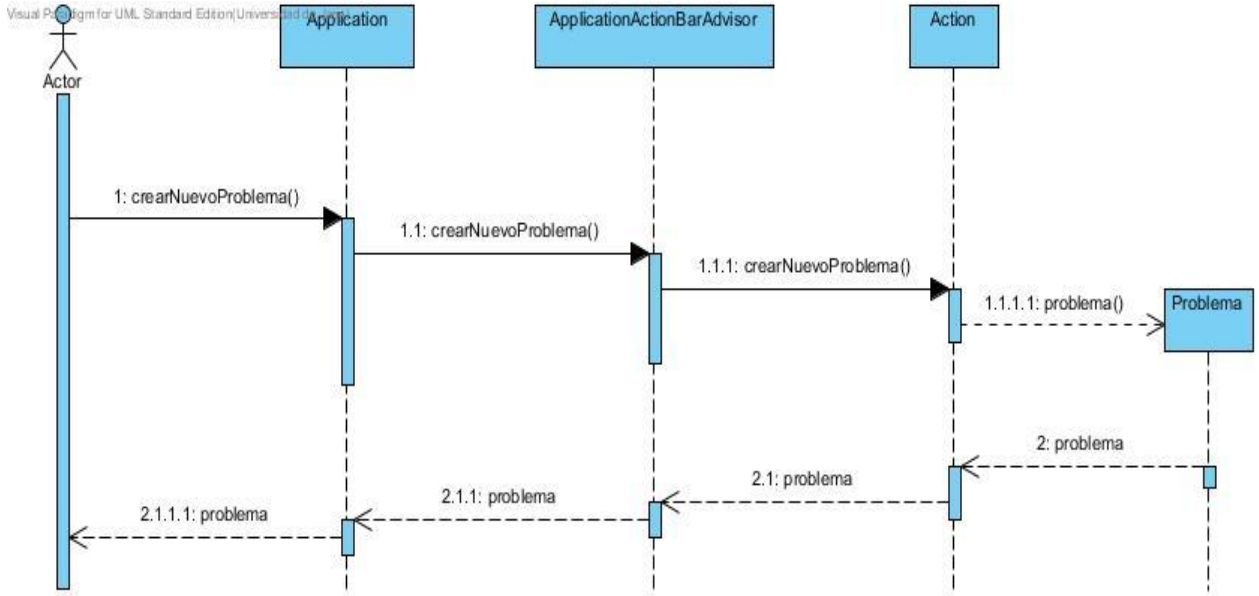


Ilustración 5.5 Diagrama de secuencia "Crear Problema"

Diagrama de secuencia "Añadir Criterio" (Ilustración 5.6):

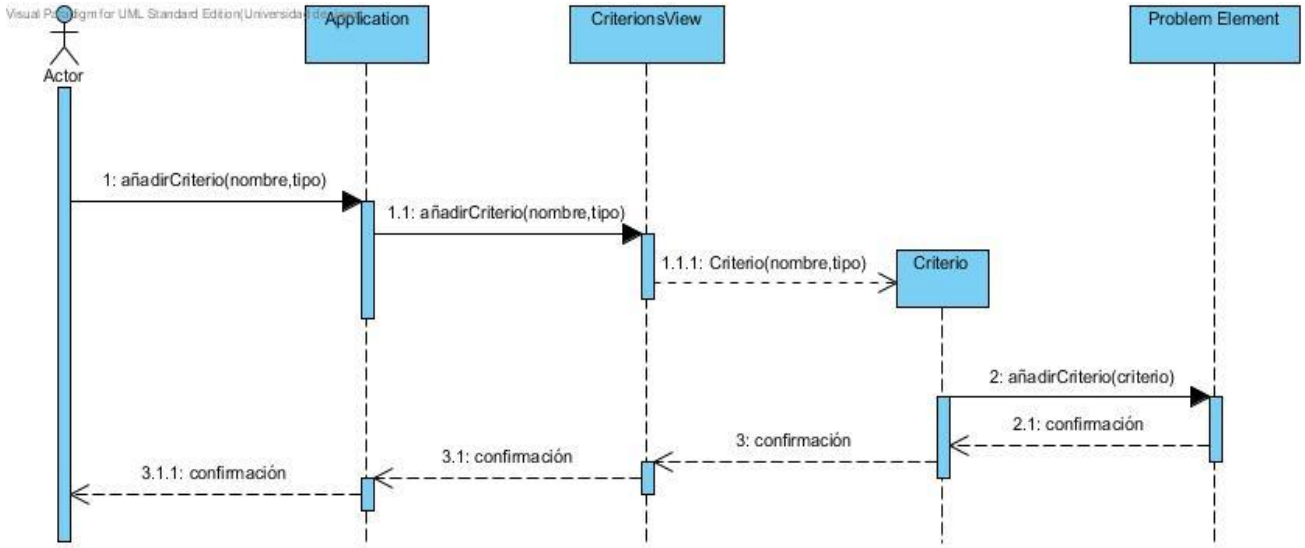


Ilustración 5.6 Diagrama de secuencia "Añadir Criterio"

Nota: Para añadir nuevas alternativas o expertos el proceso es similar para los criterios, únicamente varía el hecho de que no es necesario incluir el tipo.

Diagrama de secuencia "Valorar Alternativa" (Ilustración 5.7):

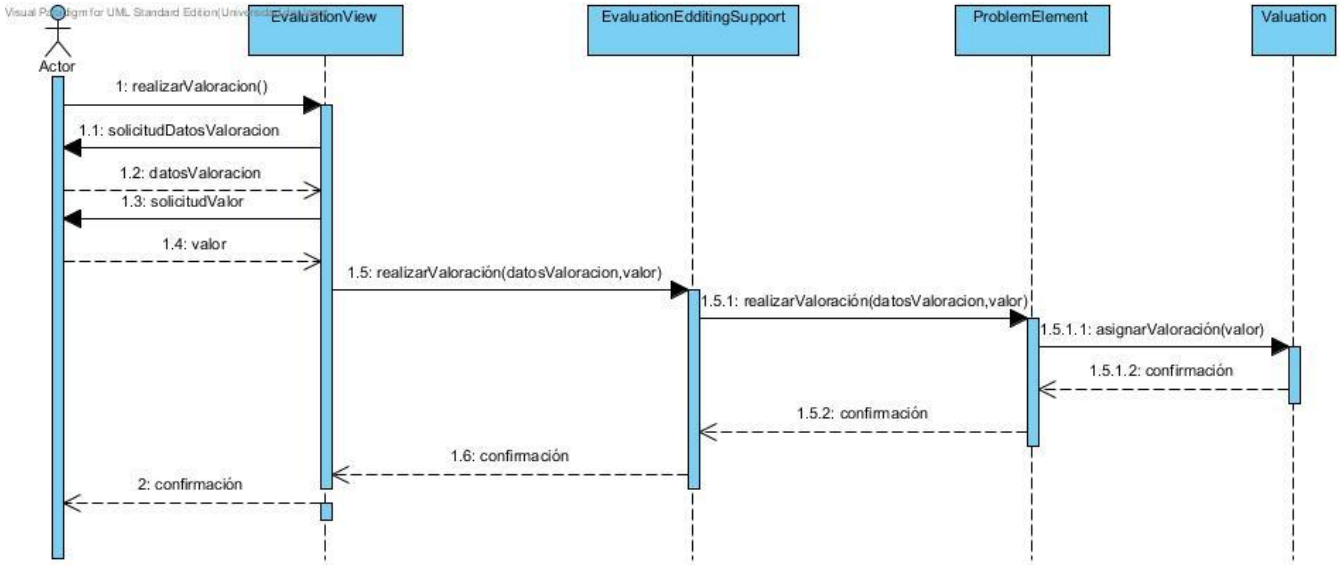


Ilustración 5.7 Diagrama de secuencia "Valorar Alternativa"

Diagrama de secuencia "Resolución Mediante Topsis" (Ilustración 5.8):

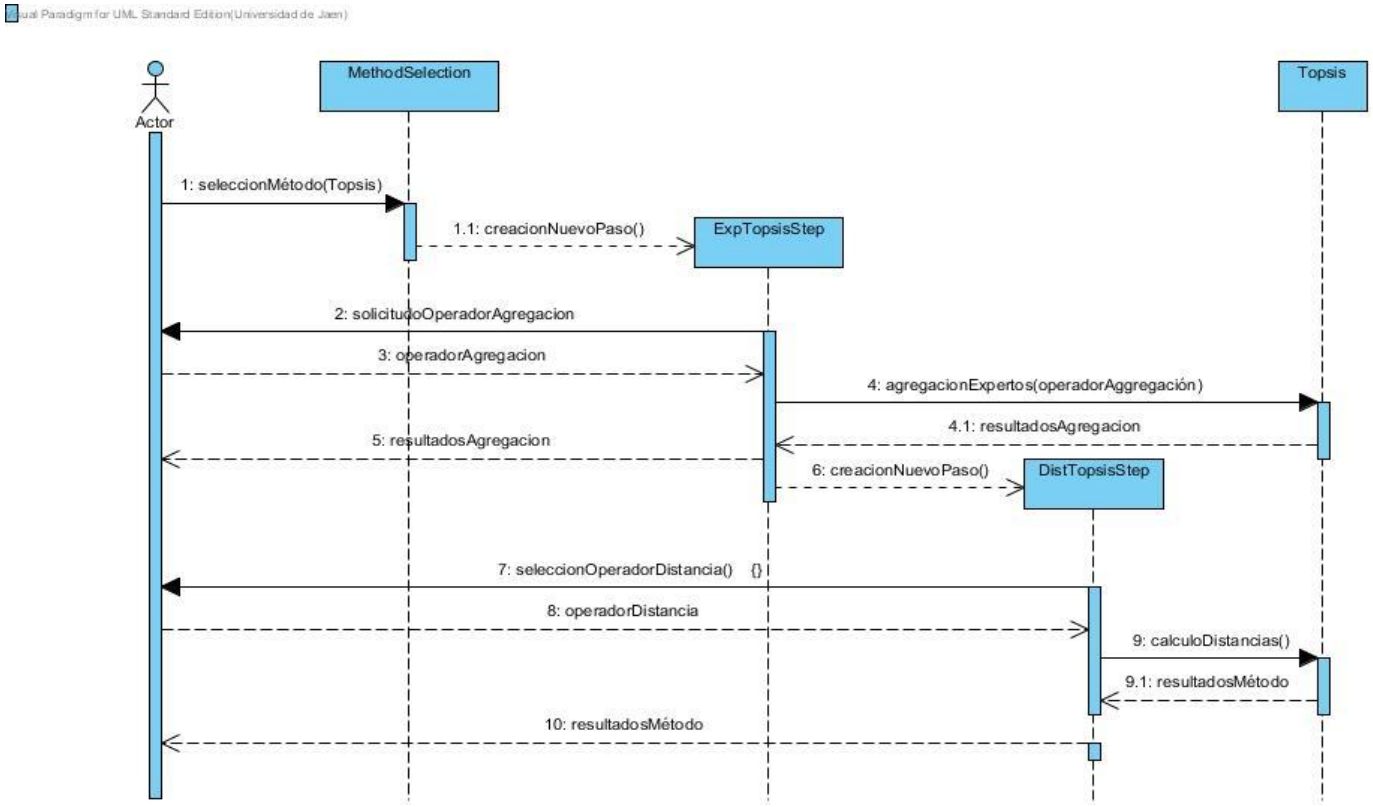


Ilustración 5.8 Diagrama de secuencia "Resolución Mediante Topsis"

5.3 Diseño de la interfaz

Una vez realizado el análisis de la aplicación y diseño de la estructura de la misma, se pasa al diseño de la interfaz. Esta etapa es importante en el proceso de desarrollo de cualquier software, ya que en ella se define la apariencia o aspecto visual que tendrá dicho software. Es fundamental realizar un diseño correcto de la interfaz ya que es el elemento con el que interactuará el usuario de la aplicación, por tanto, debe ser atractiva visualmente pero también debe resultar fácil entender como función y trabajar sobre ella. [18]

En muchas ocasiones programas que desde un punto de vista funcional estaban muy bien hechos, han sido descartados por tener una interfaz poco agradable para los usuarios o muy difícil de utilizar. En cambio, otros programas más simples o peor desarrollados desde un punto de vista técnico han sido muy utilizados por disponer de una interfaz adecuada a los usuarios que utilizarán dicho programa.

Dentro de este contexto se debe tener en cuenta los requerimientos que debe cumplir la interfaz para adecuarse a los requisitos tanto funcionales como no funcionales definidos al inicio del proyecto.

Los requerimientos de la interfaz gráfica entre la aplicación y el usuario están íntimamente ligados a la usabilidad y sus principios. La usabilidad se puede definir de varias formas:

- Usabilidad se define coloquialmente como facilidad de uso, ya sea de una página Web, una aplicación informática o cualquier otro sistema que interactúe con un usuario.
- Usabilidad se refiere a la capacidad de un software de ser comprendido, aprendido, usado y ser atractivo para el usuario, en condiciones específicas de uso.
- Usabilidad es la efectividad, eficiencia y satisfacción con la que un producto permite alcanzar objetivos específicos a usuarios específicos en un contexto de uso específico.

A partir de estas tres definiciones se pueden obtener los principios básicos de la usabilidad, los cuales se asociarán a los requerimientos no funcionales que deberá cumplir la interfaz gráfica:

- Facilidad de aprendizaje: Se refiere a la facilidad con la que nuevos usuarios pueden tener una interacción efectiva. Depende de los siguientes factores:

    - **Predictibilidad**: Una vez conocida la aplicación, se debe saber en cada momento a qué estado se pasará en función de la tarea que se realice.
    - **Síntesis**: Los cambios de estado tras una acción deben ser fácilmente captados.
    - **Generalización**: Las tareas semejantes se resuelven de modo parecido.
    - **Familiaridad**: El aspecto de la interfaz tiene que resultar conocido y familiar para el usuario.
    - **Consistencia**: Siempre se han de seguir una misma serie de pasos para realizar una tarea determinada.

- Flexibilidad: Relativa a la variedad de posibilidades con las que el usuario y el sistema pueden intercambiar información. También abarca la posibilidad de diálogo, la multiplicidad de vías para realizar la tarea, similitud con tareas anteriores y la optimización entre el usuario y el sistema.

- Robustez: Es el nivel de apoyo al usuario que facilita el cumplimiento de sus objetivos o, también, la capacidad del sistema para tolerar fallos. Está relacionada con los siguientes factores:

    - **Navegable**: El usuario debe poder observar el estado del sistema sin que esta observación repercuta de forma negativa en él.
    - **Recuperación de información**: La aplicación debe permitir volver a un estado anterior.
    - **Tiempo de respuesta:** Es el tiempo necesario para que el sistema pueda mostrar los cambios realizados por el usuario.

Por tanto, a la hora de realizar el diseño de la interfaz de usuario se deben tener en cuenta los principios de usabilidad previamente establecidos para adaptar la interfaz al usuario, evitando problemas de interacción que pueda tener al usar la aplicación.

Una vez vista está introducción sobre la importancia del diseño de la interfaz, así como, los principios que se deben tener en cuenta a la hora de realizar dicho diseño, se pasa a realizar el diseño de la interfaz. En esta etapa se llevarán a cabo las siguientes tareas:

- Definir el estilo de la aplicación.
- Definir las metáforas que se emplearan en la misma.
- Realizar el diseño de las diferentes pantallas.
- Establecer los caminos de navegación (storyboards) empleados para el desarrollo de la interfaz de nuestra aplicación.
- Definir los mensajes de error y las notificaciones empleadas.

Cada uno de los apartados anteriores se verán a continuación de forma más detalla.

5.3.1 Estilo

Un paso previo a la realización del diseño de la interfaz de usuario debe ser la definición del estilo de la misma. Esta fase es aún más importante cuando el proceso de diseño va a ser responsabilidad de varias personas, ya que ayuda a mantener la coherencia interna de la interfaz.

Así mismo, aunque las razones no son tan obvias, también es de utilizad definir la guía de estilo cuando sólo hay un diseñador encargado de la interfaz. Las siguientes razones justifican este hecho:

- En ocasiones es complicado mantener la coherencia y consistencia de una interfaz si es muy grande o incluye muchos elementos, incluso cuando el diseño es responsabilidad de una única persona.

- La persona encargada de realizar el diseño de la interfaz puede, por diversas razones, abandonar el diseño y la guía de estilo es de utilizad para que los sustitutos tengan una referencia con la que comenzar. De la misma forma ocurre si los encargados de mantenimiento y actualización de la interfaz son distintos de los diseñadores de la misma.

Una vez vista la utilidad de la guía de estilo, se definirá a continuación las reglas y normas por las que se regirá el diseño de la interfaz correspondiente a este proyecto:

- ***Fuentes***:
    - Para escribir los títulos correspondientes a las diferentes secciones o tablas mostradas en las distintas páginas o texto resaltado:
        - Tipo de letra: Cantarell.
        - Tamaño: 10 px.
        - Color de la letra: Negro
        - Estilo letra: Sencillo
        - Alineación del texto: Centrado

    - Para el título de la pestaña, el contenido de las tablas, así como, el resto de texto de las distintas páginas:
        - Tipo de letra: Cantarell.
        - Tamaño: 10 px.
        - Color de la letra: Negro
        - Estilo letra: Negrita
        - Alineación del texto: Centrado

- ***Colores de fondo***:
    - Paneles correspondiente al método:
        - Cuerpo: Gris, #D8D8D8
        - Botones: Gris Claro, #E6E6E6
    - Cuadros de diálogo:
        - Cuerpo: Gris, #D8D8D8
        - Botones: Gris Claro, #E6E6E6
    - Tablas:
        - Cabecera: Gris Claro, #D8D8D8
        - Cuerpo: Blanco, #FFFFFF

Nota: Al trabajar para añadir funcionalidad a una aplicación ya creada se debe seguir el mismo estilo de la misma para evitar diferencias entre partes de una misma aplicación.

5.3.2 Metáforas

Se define como metáfora el empleo de un objeto con un significado dentro de un contexto diferente al habitual. El uso de metáforas en el diseño de una interfaz gráfica es muy útil ya que ayudan al usuario mediante una asociación con otro objeto con el que está muy habituado en un contexto determinado, de esta forma el usuario realizará determinadas tareas de una forma más intuitiva al utilizar dicha interfaz. A pesar de ello, las metáforas se deben utilizar con cuidado, pudiendo ser un factor fundamental en la aceptación o rechazo de una interfaz.

Para que una metáfora sea útil y aporte los beneficios buscados, el desarrollador de la aplicación y el usuario final deben tener una base social y cultural similar, ya que de no ser así la metáfora puede no ser bien entendida por el usuario o no conseguir el efecto deseado por el diseñador y llevar a errores en la interacción. Por tanto, hay que intentar que las metáforas sean lo más internacionales posibles, es decir, estén totalmente adaptadas al público potencial de la aplicación.

Existen metáforas que son propias de un determinado sistema operativo o de un determinado software y otras que son universales en un contexto determinado, utilizándose en multitud de software de diferentes desarrolladores. Así mismo, las metáforas no sólo dependen del tipo de aplicación (escritorio o Web) sino también del ámbito de la misma.

Las características que debe poseer una buena metáfora son las siguientes:

- Debe ser consistente aplicándose de forma coherente y sólida a lo largo de todo el sistema.
- Auto-explicativa: Debe facilitar al usuario conocer las acciones posibles en cada momento.
- Debe hacer visibles partes y procesos invisibles al usuario, incluyendo el modelo conceptual del sistema las acciones alternativas y el resultado de las acciones.
- Debe informar con claridad al usuario acerca del estado actual del sistema y de las consecuencias de sus acciones.
- No desbordar al usuario con información superflua o innecesaria.

A continuación, se mostrarán las metáforas que se han utilizado en el software desarrollado, así como, las más importantes del software inicial:

- Icono representativo de la acción para "crear un nuevo problema":

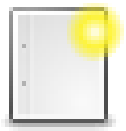


Ilustración 5.9 Metáfora "Crear Nuevo Problema"

- Icono representativo de la acción para "guardar un problema":

Ilustración 5.10 Metáfora "Guardar Problema"

- Icono representativo de la acción para "guardar un problema como":

Ilustración 5.11 Metáfora "Guardar Problema Como"

- Icono representativo de "expertos" y "experto":

Ilustración 5.12 Metáfora "Expertos"

Ilustración 5.13 Metáfora "Experto"

- Icono representativos de "alternativas" y "alternativa":

Ilustración 5.14 Metáfora "Alternativas"

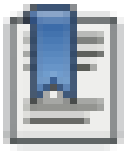


Ilustración 5.15 Metáfora "Alternativa"

- Icono representativo de "criterios" y "criterio":

Ilustración 5.16 Metáfora "Criterios"

Ilustración 5.17 Metáfora "Criterio"

- Icono representativo de los distintos tipos de criterio "beneficio" y "coste":

Ilustración 5.18 Metáfora "Beneficio"

Ilustración 5.19 Metáfora "Coste"

- Icono representativo de "dominio de expresión":

Ilustración 5.20 Metáfora "Dominio de expresión"

- Icono representativo de la acción "Añadir" y "Eliminar" utilizado en todos los elementos correspondientes a un problema determinado:

Ilustración 5.21 Metáfora "Añadir"

Ilustración 5.22 Metáfora "Eliminar"

Como se puede ver, las metáforas empleadas en la aplicación cumplen con las características que debe tener una buena metáfora: son coherentes con la acción o elemento que representan, facilitan la labor del usuario aportando información adicional y ayudando a comprender de manera fácil y rápida determinadas tareas y elementos.

5.3.3 Diseño de las pantallas de la aplicación

Una vez realizada la definición del estilo usado en la aplicación, así como, las principales metáforas usadas, se va a pasar a realizar el diseño de las distintas pantallas. Esta fase es muy importante en el desarrollo de cualquier software y es necesario dedicarle el tiempo y los recursos necesarios.

El diseño de la interfaz de usuario, como se ha dicho anteriormente, tiene un papel fundamental en el éxito o fracaso de una aplicación. Una buena distribución y ordenación de los elementos, así como, un uso de colores correctos hace que la interfaz sea más clara y fácil de usar para los distintos usuarios. Por el contrario, una ordenación incorrecta puede aumentar la complejidad de utilización de la interfaz.

Un principio esencial que se debe tener en cuenta cuando se realiza el diseño de la interfaz es que los elementos deben estar colocados de forma que el usuario pueda recorrer la interfaz de forma similar a la manera en la que se lee un texto. Este principio será tenido en cuenta en el diseño de las pantallas para la aplicación.

Antes de pasar mostrar el diseño de alto nivel de las pantallas creadas para la aplicación se darán una serie de pautas que se deben tener en cuenta para realizar un diseño correcto:

- El usuario es un usuario habitual de diferentes aplicaciones de escritorio, además de poseer conocimientos básicos sobre problemas de toma de decisión.
- El usuario usa la aplicación normalmente centrando toda su atención en ella por lo que debemos realizar un diseño a pantalla completa. Otra razón para elegir el diseño a pantalla completa es la gran cantidad de elementos que se deben incluir en ella.
- Todas las operaciones de interacción con el sistema deben hacerse de forma rápida y fiable de acuerdo con uno de los requisitos no funcionales definido en el apartado 4.1.2.
- La navegación a través de la aplicación debe ser fluida, evitando las funcionalidades rígidas. Se debe buscar la orientación del usuario hacia las distintas opciones con una jerarquía visual clara. Debe haber consistencia en las acciones.
- No se debe mostrar una interfaz muy recargada que pueda agobiar al usuario y ocultarle alguna funcionalidad. Cualquier contenido de texto debe ser concreto y preciso.

- No hay restricciones a la hora de usar el teclado y el ratón, ya que el usuario normalmente no tendrá las manos ocupadas.

Siguiendo los principios de usabilidad expuestos anteriormente, así como, los principios fundamentales de diseño, se pasa a realizar el diseño de las diferentes pantallas de la aplicación.

Las pantallas diseñadas se van a dividir en dos grupos: las pantallas correspondientes al método de resolución Topsis y los cuadros de diálogo correspondientes a los operadores de agregación que requieren pesos o parámetros.

5.3.3.1 Pantallas método Topsis

Para realizar el diseño de las pantallas del método Topsis se partió de los pasos que se deben realizar para llevar a cabo dicho método y que viene definidos en el documento *"Gui-Wu Wei, Extension of TOPSIS method for 2-tuple linguistic multiple attribute group decision making with incomplete weight information (Springer-Verlag, 2009)"* [13].

Partiendo de dicha división del método en pasos se realizo el siguiente diseño preliminar formado por 4 pantallas diferentes:

1- Selección del método: Está pantalla será similar a las empleadas para seleccionar cualquier otro método del sistema para que la interacción con el nuevo método sea similar a la interacción con el resto de métodos.

   Se debe tener en cuenta que a la hora de seleccionar un método determinado se deben cumplir una serie de requisitos para poder emplear dicho método sobre el conjunto de datos de entrada, de no cumplirse se debe informar del error al usuario.

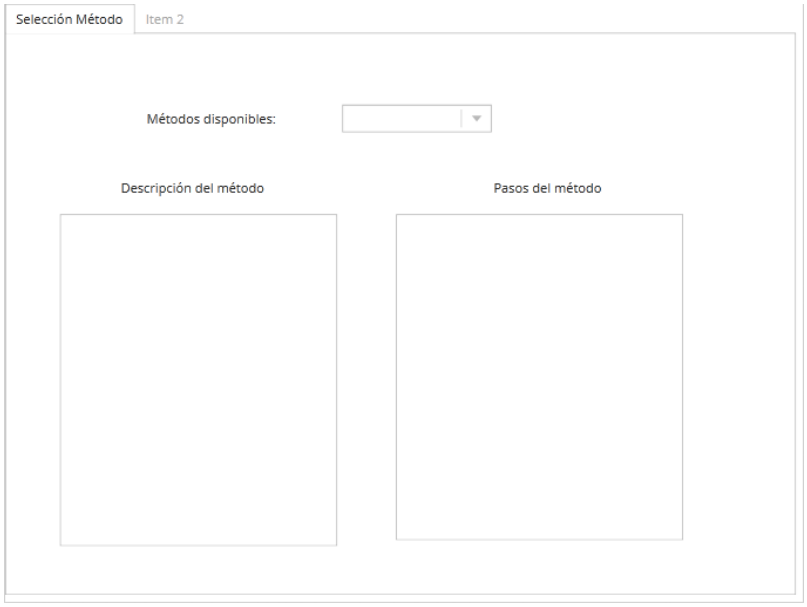


Ilustración 5.23 Pantalla "Selección del Método"

Como se puede ver en la ilustración anterior, la pantalla está compuesta fundamentalmente por una parte donde se selecciona el método y otra parte donde se ofrece información sobre el mismo.

2- Selección de operador de agregación para expertos y muestra de solución ideal y no ideal.

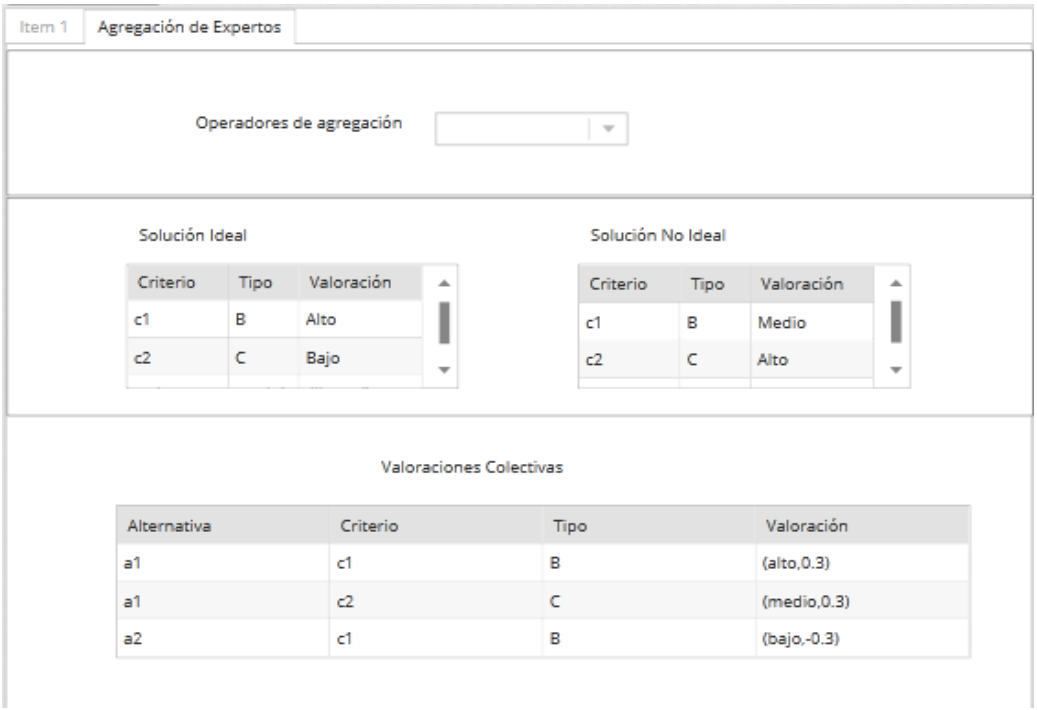


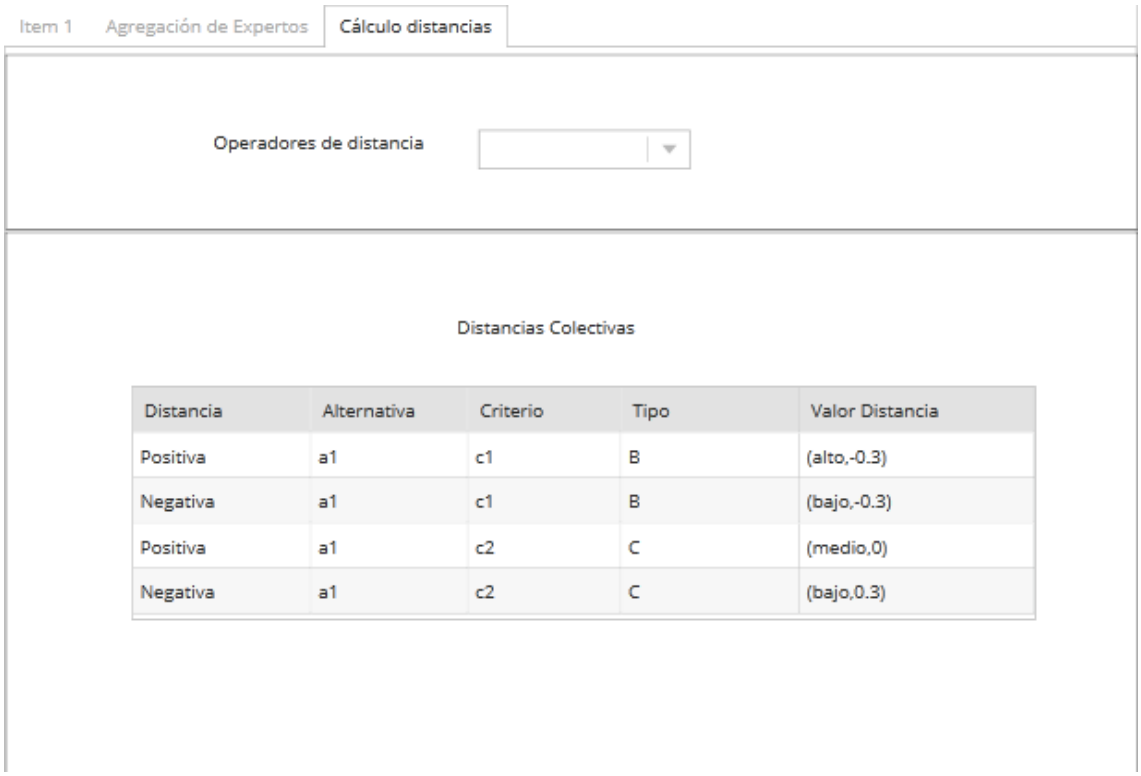
Ilustración 5.24 Pantalla "Agregación Expertos"

Como se puede ver la pantalla anterior se divide en 3 partes fundamentalmente:

a. La parte superior se corresponde a la selección del operador de agregación.
b. La parte central se muestra la solución ideal a la izquierda y la solución no ideal a la derecha.
c. En la parte inferior, aparece la información agregada por expertos.

Es obvio que si se modifica el operador de agregación, las tablas que contienen los resultados deben refrescarse con la nueva información generada.

3- Cálculo de las distancias positivas y negativas de cada valoración colectiva según el operador de distancia seleccionado.



Distancia	Alternativa	Criterio	Tipo	Valor Distancia
Positiva	a1	c1	B	(alto,-0.3)
Negativa	a1	c1	B	(bajo,-0.3)
Positiva	a1	c2	C	(medio,0)
Negativa	a1	c2	C	(bajo,0.3)

Ilustración 5.25 Pantalla "Cálculo de Distancias"

En la pantalla anterior se pueden ver fundamentalmente dos partes. La parte superior se corresponde con la selección del operador de distancia que será utilizado para realizar los cálculos y en la zona inferior, se mostrará las distancias colectivas calculadas utilizando dicho operador de agregación.

4- Ranking de alternativas según los coeficientes de cercanía calculados a partir de las distancias obtenidas en el paso anterior.

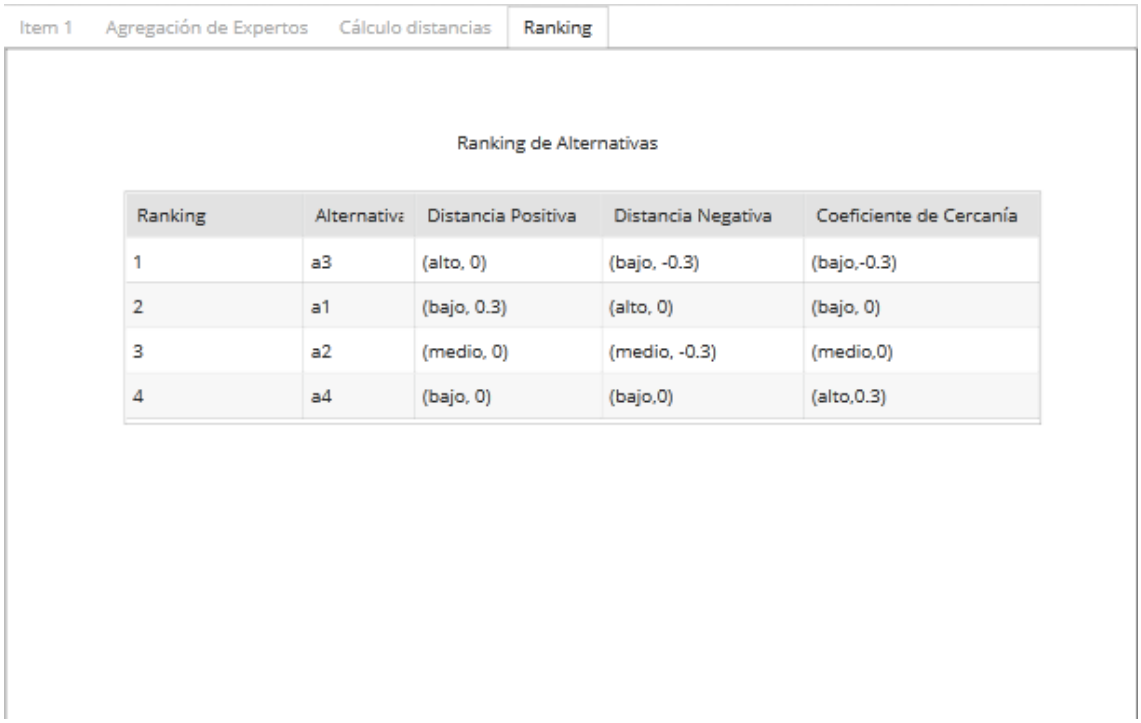


Ilustración 5.26 Pantalla "Ranking Alternativas"

En la pantalla anterior se ve que existe una sola tabla con la información ordenada según el coeficiente de cercanía calculado a partir de las distancias obtenidas en el paso previo. Esta pantalla se corresponde con el resultado del método de resolución Topsis y es la última pantalla mostrada.

Estas pantallas constituyen el diseño inicial realizado para realizar el método Topsis, pero algo fundamental en cualquier proceso de diseño es la realimentación que se debe producir a lo largo del mismo, ya que se trata de un proceso iterativo. Por ello, es esencial realizar análisis detallados de los diseños realizados para intentar en la medida de lo posible mejorar los diseños para obtener unos resultados mejores.

Una vez analizadas las pantallas anteriores se llegó a la conclusión de que las pantallas 3 y 4 se podrían unir en una única pantalla, ya que muestran información muy relacionada y el hecho de unirlas no supone obtener una pantalla extremadamente sobrecargada. El diseño resultante es el siguiente y se corresponde con la tercera y última pantalla del método:

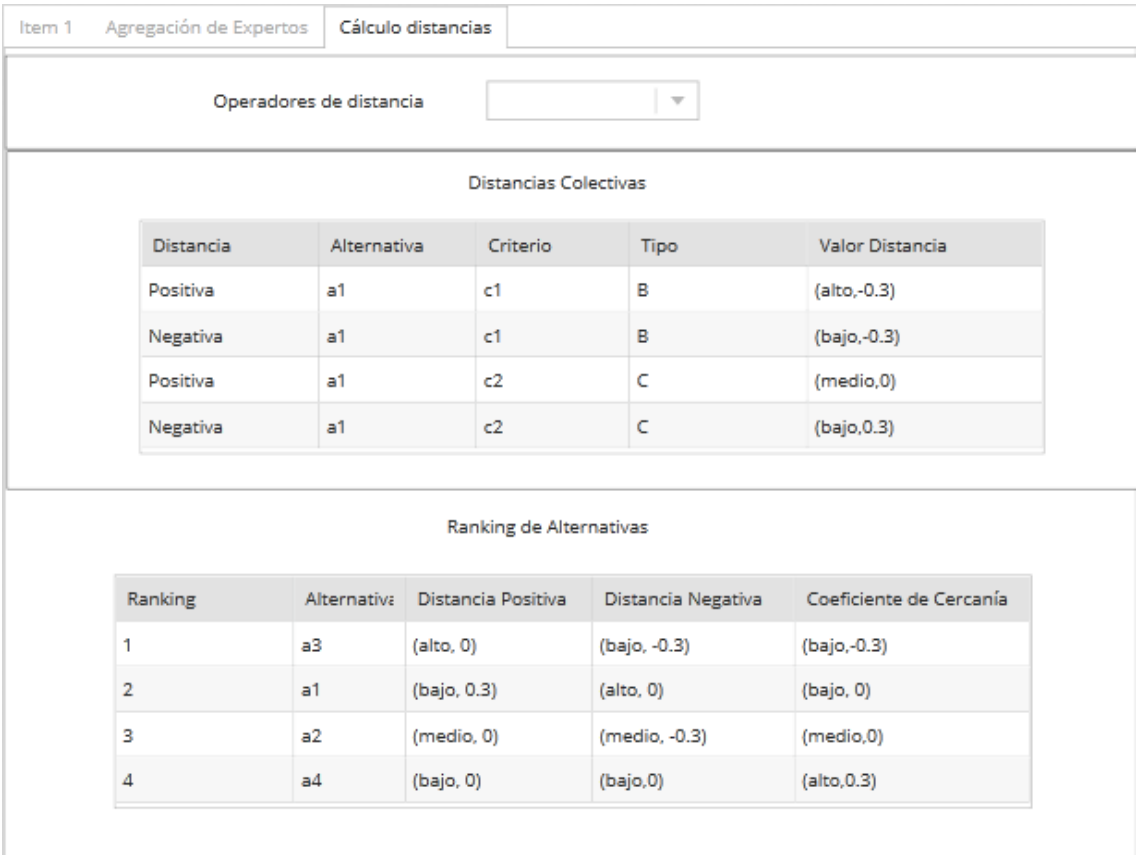


Ilustración 5.27 Pantalla "Distancias y Ranking"

Como se puede ver en la ilustración anterior, la pantalla agrega la información anteriormente expuesta en las pantallas 3 y 4 del diseño previo. De esta forma, el usuario visualizará toda la información relativa a los resultados del método en una única pantalla, sin ser necesarias dos pantallas distintas para mostrar información relacionada.

Por tanto, el diseño definitivo de la interfaz del método Topsis estará formado por 3 pantallas:

- La primera para seleccionar el método.
- La segunda para realizar la agregación por expertos y el cálculo de la solución ideal y no ideal.
- La tercera que mostrará el cálculo de las distancias, así como, el ranking de alternativas.

Partiendo del diseño anterior, se ha realizado la implementación y desarrollo de las pantallas que de soporte a dicho diseño. A continuación, vemos el resultado definitivo:

1- Selección del método:

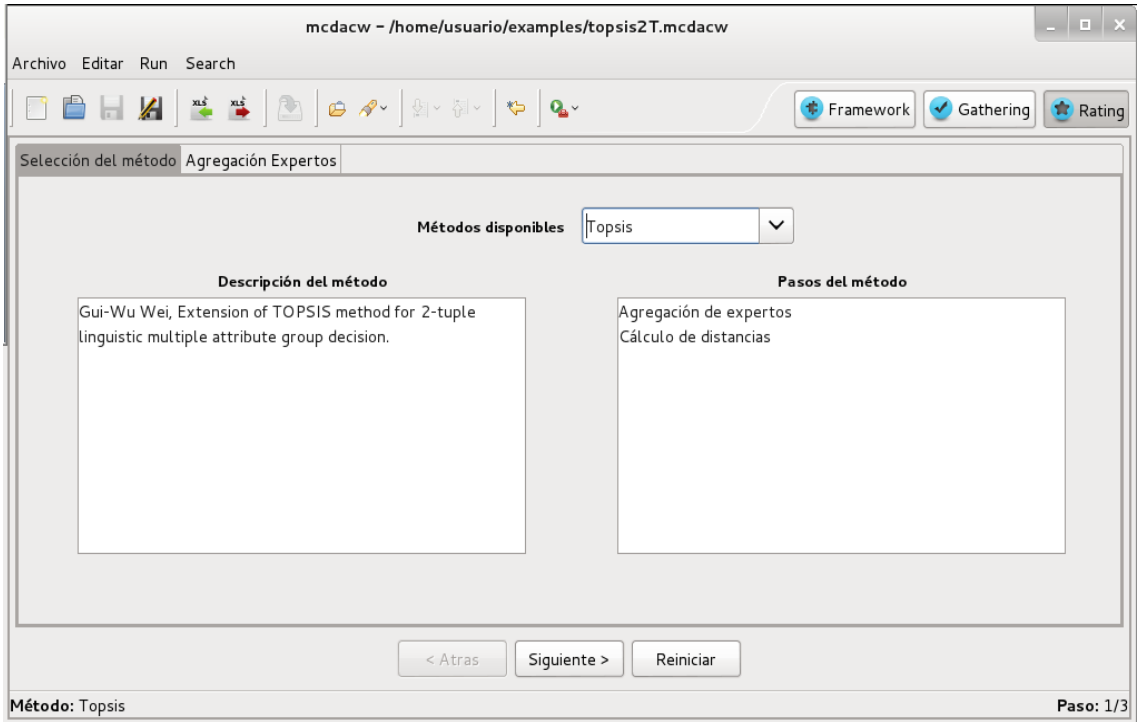


Ilustración 5.28 Pantalla definitiva "Selección del método"

2- Agregación de expertos:

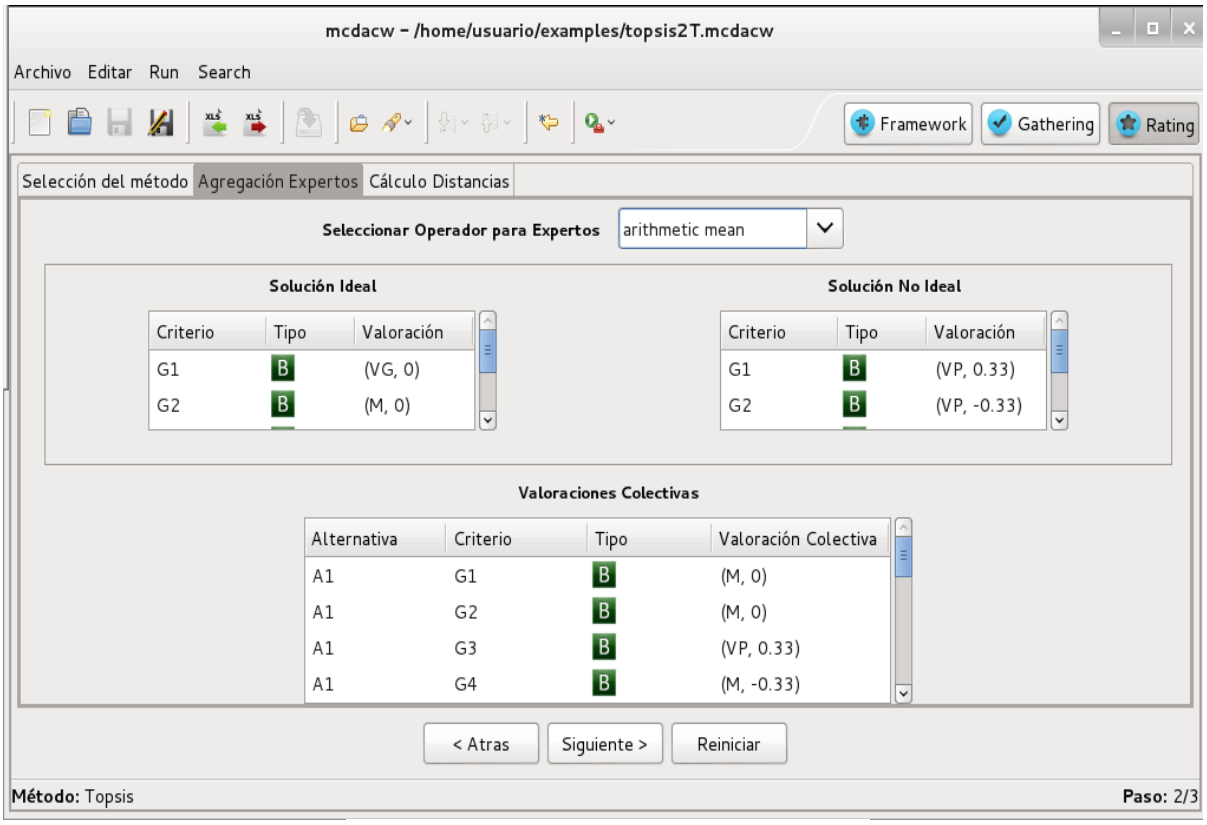


Ilustración 5.29 Pantalla Definitiva "Agregación de expertos"

3- Cálculo de distancias y ranking de alternativas:

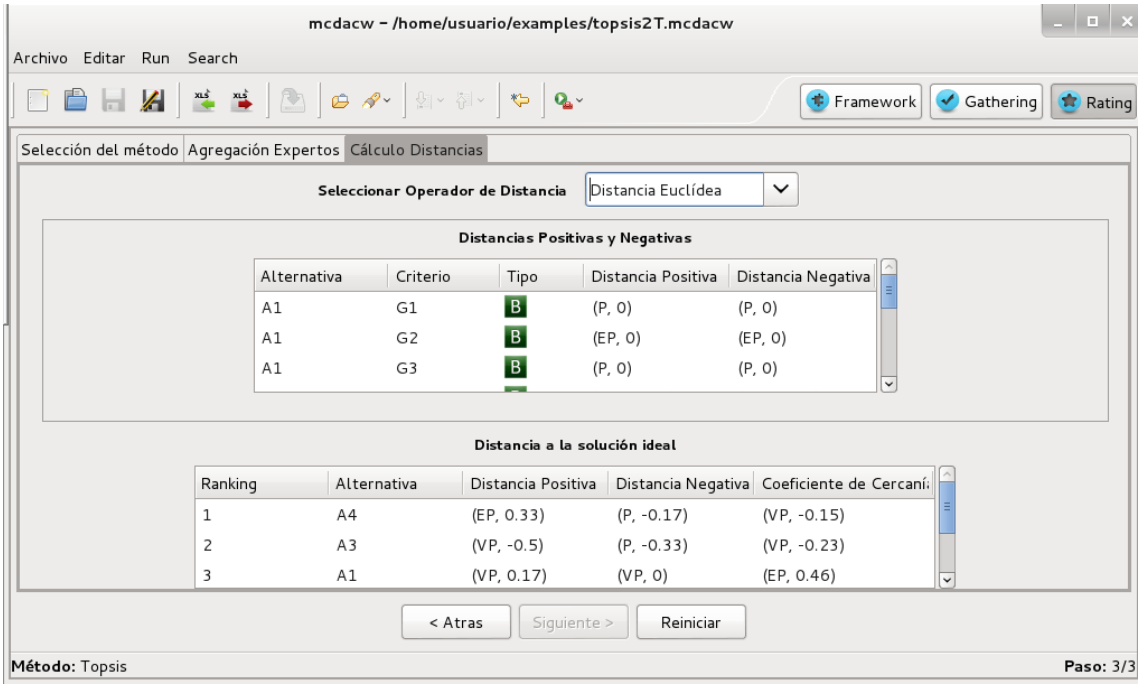


Ilustración 5.30 Pantalla Definitiva "Cálculo de distancias y ranking"

5.3.3.2 Cuadros de diálogo para los operadores de agregación.

Para realizar el diseño de los cuadros de diálogo con los que la aplicación solicitará a los usuarios los pesos o los parámetros que requieran los distintos operadores de agregación, se ha partido de los cuadros de diálogo similares que la aplicación "Flinstones" emplea para otros operadores ya creados.

Es obvio que los operadores de agregación que no requieran ni pesos ni parámetros para su funcionamiento no influirán en la labor de diseño de la interfaz, ya que estos operadores no necesitan solicitar ningún tipo de dato al usuario, sólo son seleccionados y la aplicación realiza los cálculos de forma interna.

A continuación, se ha realizado una división de los operadores de agregación según los elementos que soliciten al usuario, para cada uno de los tipos de operadores creado se expondrá el diseño preliminar realizado, así como el resultado definitivo obtenido.

- **Operadores que requieren pesos:** Estos operadores requieren únicamente pesos para su funcionamiento, entre estos operadores se encuentran: "*Media geométrica pesada" y "Media harmónica pesada".* El cuadro de diálogo debe solicitar dichos pesos al usuario.

  El diseño preliminar es el siguiente:

Ilustración 5.31 Cuadro de Diálogo "Operador con Pesos"

El resultado definitivo es el siguiente:

Ilustración 5.32 Cuadro de Diálogo Definitivo "Operador con Pesos"

- **Operador Power Median:** Este operador requiere únicamente un parámetro para funcionar. El cuadro de diálogo debe solicitar dicho parámetro al usuario.

  El diseño preliminar es el siguiente:

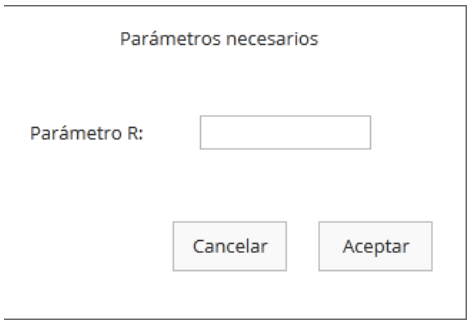


Ilustración 5.33 Cuadro de Diálogo "Operador Power Median"

El resultado definitivo es el siguiente:

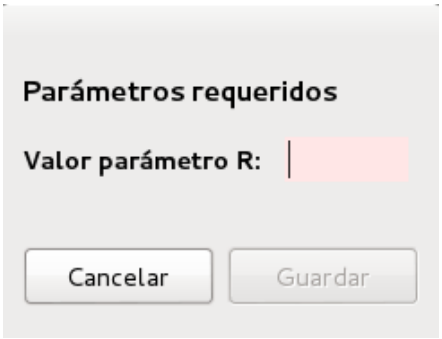


Ilustración 5.34 Cuadro de Diálogo Definitivo "Operador Power Median"

- **Operador Power Median Pesada:** Este operador requiere un parámetro y los pesos asociados a cada uno de los expertos para funcionar. El cuadro de diálogo debe solicitar tanto el parámetro como los pesos al usuario.

El diseño preliminar es el siguiente:

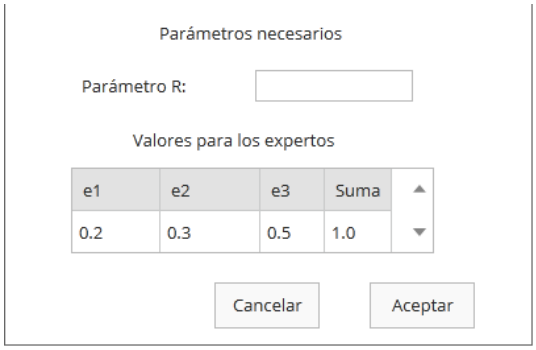


Ilustración 5.35 Cuadro de Diálogo "Operador Power Median Pesada"

El resultado definitivo es el siguiente:

Ilustración 5.36 Cuadro de Diálogo Definitivo "Operador Power Median Pesada"

- **Operador Bonferroni Means:** Este operador requiere dos parámetros para funcionar y poder realizar los cálculos asociados. El cuadro de diálogo debe solicitar ambos parámetros al usuario.

  El diseño preliminar es el siguiente:

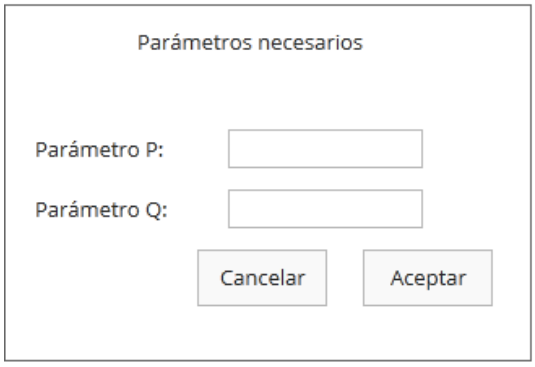


Ilustración 5.37 Cuadro de Diálogo "Operador Bonferroni Means"

  El resultado definitivo es el siguiente:

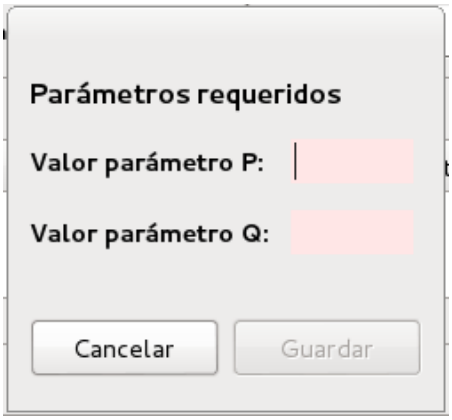


Ilustración 5.38 Cuadro de Diálogo Definitivo "Bonferroni Means"

5.3.4 Storyboards

En los apartados anteriores se ha realizado un diseño de la interfaz visual de la aplicación desde un punto de vista estático, es decir, cada pantalla se ha diseñado de forma individual, pero aún no se tiene una idea de si la interacción resultado de combinar dichas pantallas va a ser adecuada y compresible para el usuario, facilitando el uso del software tanto como se pueda. Para ello, en este apartado se va a realizar un diseño de la interfaz en movimiento, comprobando si es usable.

Para realizar el análisis y estudio de la secuencia de interacción que seguirá el usuario a partir de las pantallas anteriormente definida se empleará una herramienta llamada *storyboards.* Esta herramienta se encarga de mostrar, a modo de secuencia, los pasos por los que pasará el usuario al llevar a cabo una determinada acción sobre la aplicación diseñada. Los *storyboard* están muy ligados a los escenarios vistos en el capítulo de análisis.

Los *storyboards* emplean una serie de imágenes representativas de cada uno de los pasos que componen el proceso de interacción y mediante flechas, que unen dichas imágenes, se establecerá el camino que seguirá la interacción representada.

La principal utilidad de los *storyboards* es que permiten evaluar el diseño realizado y detectar errores en la interacción en fases tempranas del desarrollo del sistema, antes de que se tenga un producto ya implementado. La ventaja de esto es que solucionar errores detectados en estas fases iniciales es más fácil y requiere menos tiempo y recursos que el solucionarlos en fases posteriores donde los efectos pueden ser mucho más graves.

A continuación se muestran los *storyboards* realizados para evaluar la interacción del método Topsis y los distintos tipos de operadores de agregación implementados en este proyecto:

- *Storyboard* Método Topsis, método de agregación simple y Distancia Euclídea.
- *Storyboard* Método Topsis, método de agregación con parámetros y Distancia Pesada.
- *Storyboard* Método Topsis, método de agregación con pesos y Distancia Euclídea.

Storyboard Método Topsis, método de agregación simple y Distancia Euclídea:

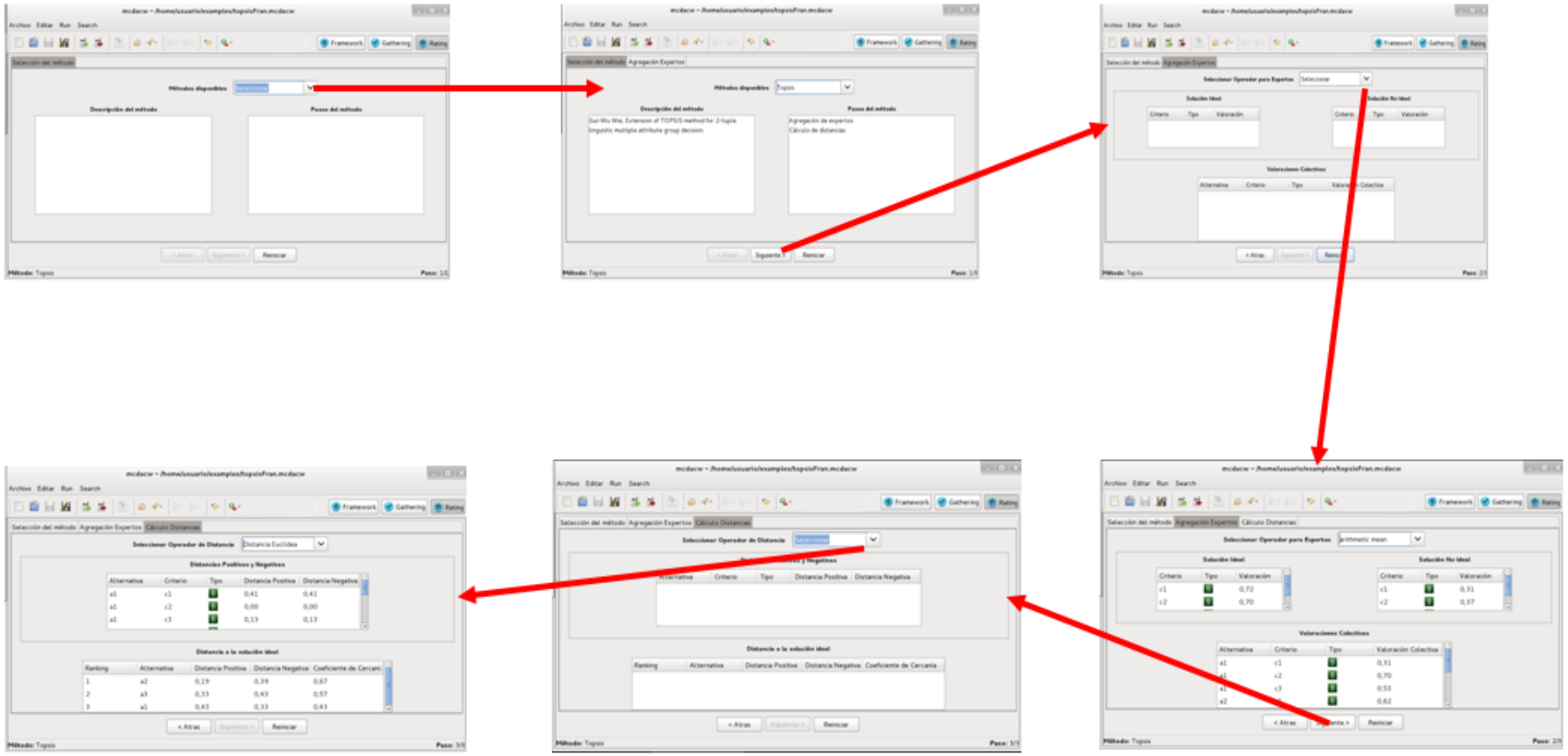


Ilustración 5.39 Storyboards Método Topsis, método de agregación simple y Distancia Euclídea

Storyboard Método Topsis, método de agregación con parámetros y Distancia Pesada:

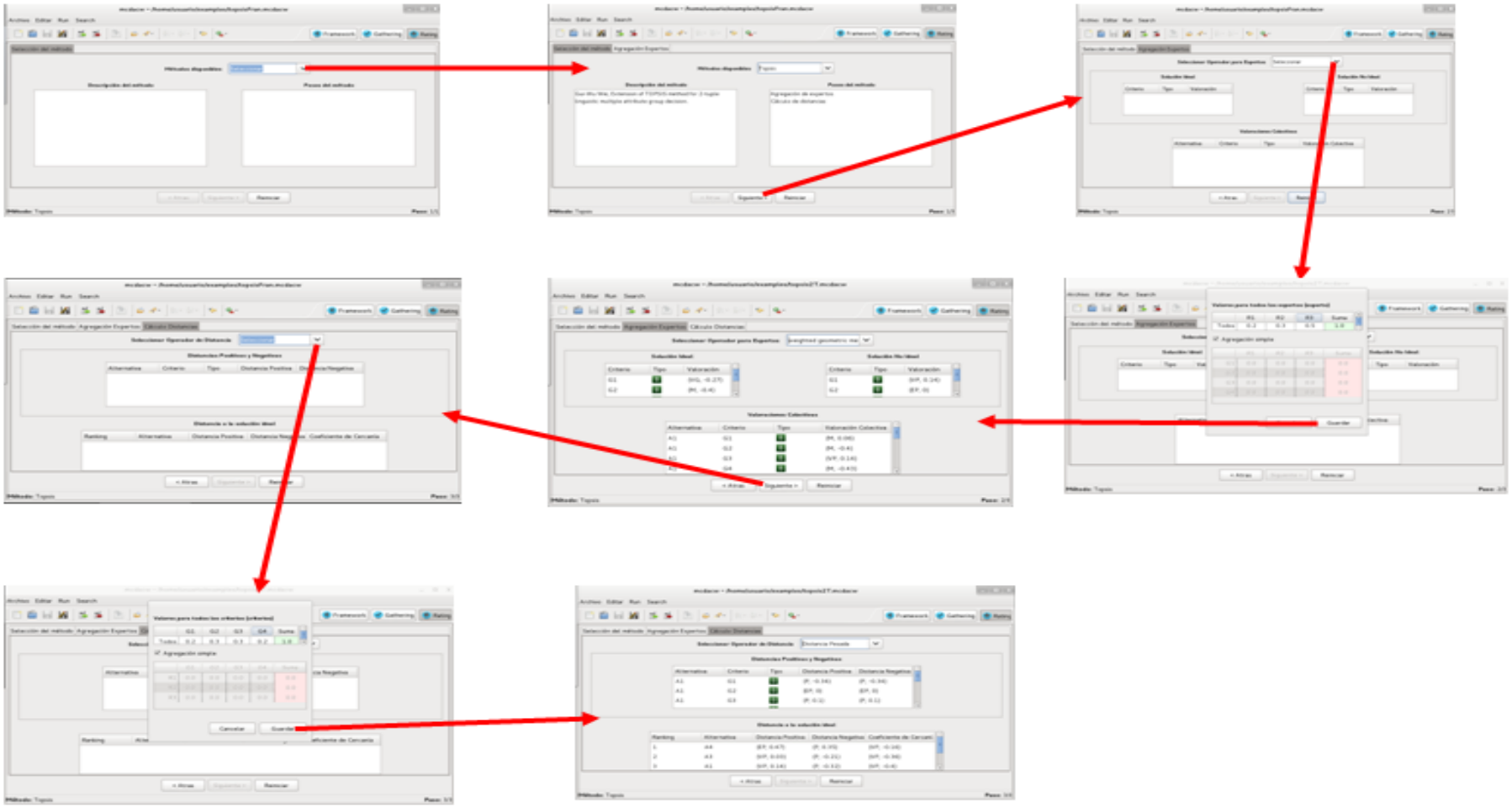


Ilustración 5.40 Storyboards Método Topsis, método de agregación con parámetros y Distancia Pesada

Storyboard Método Topsis, método de agregación con pesos y Distancia Euclídea:

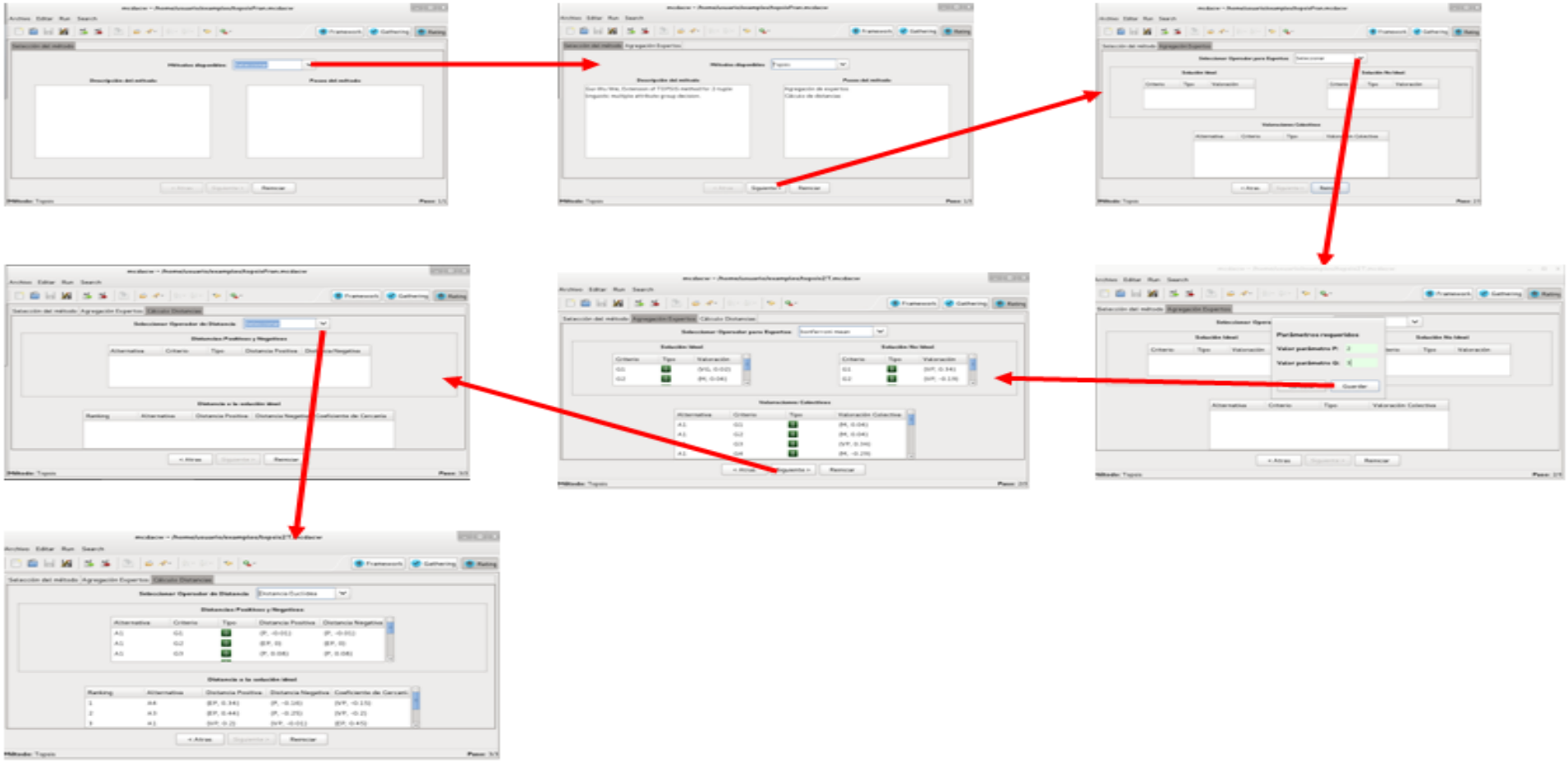


Ilustración 5.41 Storyboards Método Topsis, método de agregación con pesos y Distancia Euclídea

5.3.5 Notificaciones y mensajes de error

Hasta este momento se han realizado los diseños de las pantallas que componen los elementos implementados en el desarrollo del proyecto, así como, las diferentes secuencias de interacción que se pueden dar, por tanto, se tiene una idea clara de la forma en la que se producirá la interacción entre el usuario y el sistema. A lo largo de dicha interacción el sistema puede necesitar informar al usuario de determinadas circunstancias o errores, por ello, el siguiente paso será definir tanto las notificaciones como los mensajes de error que se suministran al usuario.

5.3.5.1 Notificaciones.

Las notificaciones son un elemento fundamental de cara a la usabilidad, ya que si se diseñan de forma correcta permiten aumentar la confianza del usuario en el uso del sistema cuando realiza cualquier operación. Las notificaciones se deben diseñar atendiendo a una serie de características:

- **Ser breves:** El objetivo de que sean breves es que el usuario las perciba de forma rápida y no las descarte por ser demasiado extensas, por ello, las notificaciones tienen que ser tan breves y claras como sea posible.

- **Formato físico apropiado:** El formato utilizado en el mensaje debe favorecer la lectura del mismo y no dificultarla usando un formato poco legible.

- **Colores adecuados:** Se deben usar colores acorde con el mensaje que se quiera transmitir al usuario, es decir, rojo si es un mensaje de error o verde si se trata de un mensaje de confirmación.

Algunos de los mensajes de notificación utilizados en el sistema diseñado son los siguientes:

- Creación de un nuevo experto: si el nombre del experto está vació o el nombre se corresponde con otro experto introducido previamente el sistema informa de esa circunstancia al usuario para que la solucione.

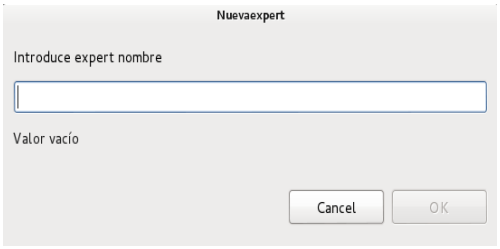


Ilustración 5.42 Notificación valor vacío

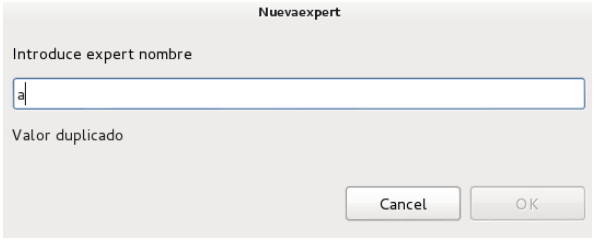


Ilustración 5.43 Notificación valor duplicado

Nota: El mensaje anterior es válido también para criterios y alternativas.

- Archivo guardado: si el archivo se guarda de forma correcta aparecerá la siguiente notificación de confirmación.

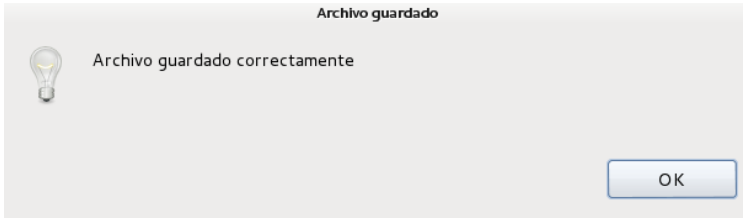


Ilustración 5.44 Notificación archivo guardado

- Confirmación de cancelación: si se solicita cambiar el problema o volver a un paso anterior el sistema solicitará la confirmación para descartar los datos actuales.

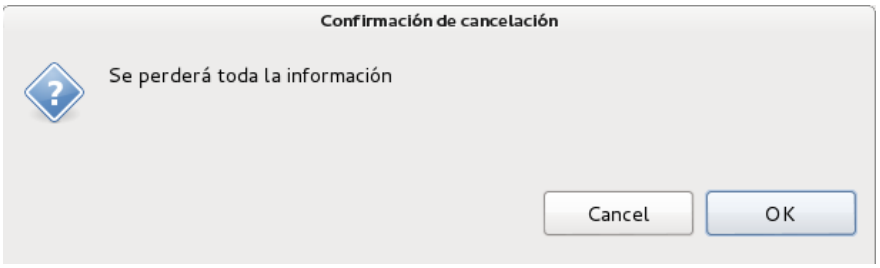


Ilustración 5.45 Notificación confirmación de cancelación

- Método seleccionado: si los datos del problema no cumplen con los requisitos del método seleccionado se mostrará la siguiente notificación informando del error.

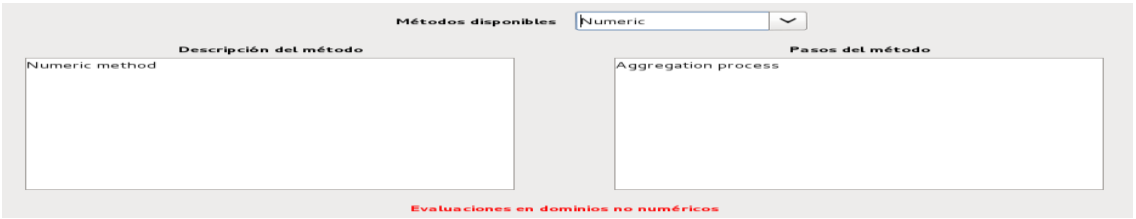


Ilustración 5.46 Notificación método erróneo

5.3.5.2 Mensajes de error.

El hecho de conocer la interacción entre el usuario y el sistema incluye también los errores que se puedan dar en dicha interacción. Por ello, es necesario definir la forma en la que se comunicarán dichos errores al usuario y diseñar los mensajes que se utilizarán.

Los mensajes de error permiten al sistema informar al usuario de que se ha producido algún tipo de error en la interacción. Estos errores se suelen producir debido a que el usuario no dispone del suficiente conocimiento sobre la interfaz, porque no se ha entendido el estado actual del sistema o por un error involuntario. El efecto principal de esto es que el usuario puede sentirse confuso, pudiendo llegar a abandonar el uso del sistema.

Para ayudar al usuario a solucionar los posibles errores que pueda cometer se usan los mensajes de error. A la hora de diseñar los diferentes mensajes de error que se utilizarán se deben seguir una serie de reglas:

- **Ser breves:** El usuario debe adquirir de forma rápida y breve la idea que el mensaje de error debe transmitir.

- **Ser específicos:** El mensaje utilizado debe indicar claramente el motivo que ha llevado al sistema a una situación errónea.

- **Usar un tono positivo:** El mensaje no debe alarmar al usuario en exceso y debe de ofrecer posibles soluciones siempre que sea posible.

- **Usar un formato apropiado:** El formato del mensaje debe facilitar la lectura y comprensión del mismo.

Los mensajes de error usados en el sistema son los siguientes:

- Valoraciones iguales a 0: Existe determinados métodos que no permiten que las valoraciones dadas sean igual a 0 debido a los cálculos que realizan. El mensaje de error dado será el que se muestra a continuación.

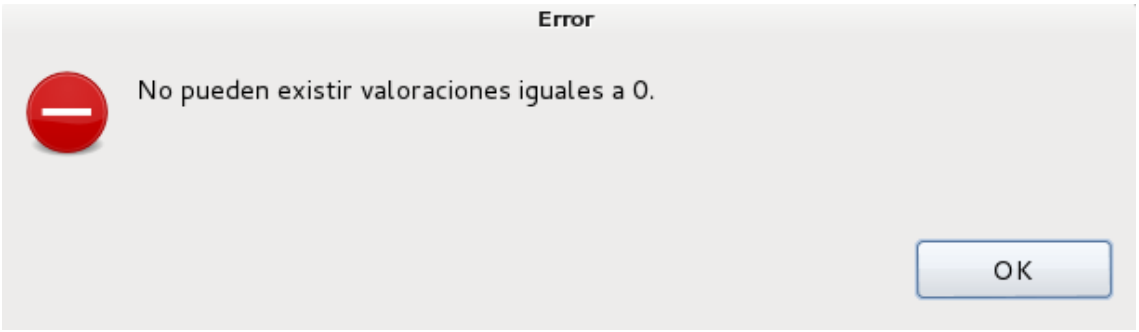


Ilustración 5.47 Valoraciones iguales a 0

- Valoraciones sin evaluar: El sistema muestra un mensaje de error si se intenta seleccionar un método y todas las valoraciones están sin evaluar, ya que en este caso no se puede aplicar ningún método de decisión.

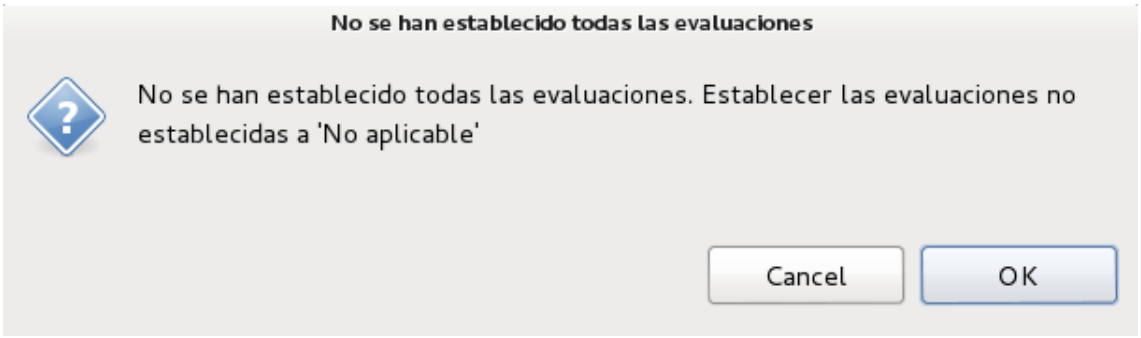


Ilustración 5.48 Valoraciones sin evaular

Capítulo 6:

Implementación, Validación y Pruebas del Software

6.1 Implementación

La implementación es la actividad final de la Ingeniería del Software que tiene relación con el desarrollo del sistema, tras esta actividad, lo único que se debe realizar son las pruebas para validar dicho software (capítulo 7).

El objetivo de la implementación es transformar el modelo obtenido en las actividades anteriores en código fuente, dichas actividades previas se han desarrollado en los capítulos anteriores: análisis del sistema (capítulo 4) y diseño (capítulo 5). Para ello algo fundamental es elegir un lenguaje de programación adecuado que se empleará para la codificación, así como, la herramienta utilizada para generarla.

A continuación se detallan varios aspectos relacionados con la implementación, en primer lugar se verá el lenguaje de programación utilizado para la codificación y, posteriormente, se expondrá la herramienta de desarrollo empleada.

6.1.1 Lenguajes de programación

La implementación llevada a cabo en el proyecto se ha realizado utilizando Java, que es un lenguaje de programación orientado a objetos, ya que es el lenguaje de programación utilizado para desarrollar la aplicación "Flinstones".

Java fue originalmente desarrollado por James Gosling de Sun Mircosystems a principios de los años 90. El lenguaje deriva mucha de su sintaxis de C y C++, pero tiene menos facilidades de bajo nivel que cualquiera de ellos, lo que ofrece una abstracción mayor aportando herramientas que eviten muchos de los errores provocados por el hecho de interactuar a tan bajo nivel. [4]

Las aplicaciones Java son compiladas a bytecode que puede correr en cualquier máquina virtual Java sin importar la arquitectura de la computadora. Su intención es que los desarrolladores de aplicaciones escriban el código una vez y lo ejecuten en cualquier dispositivo sin tener que ser recompilado. [4]

Las principales características del lenguaje Java son las siguientes:

- **Lenguaje simple:** La curva de aprendizaje de Java es muy rápida, resulta sencillo escribir código relativamente complejo en poco tiempo. Además, migrar desde otros lenguajes como C o C++ no resulta muy complicado.

- **Orientado a objetos:** Desde su aparición, Java fue diseñado como un lenguaje orientado a objetos. Dichos objetos agrupan en estructuras encapsuladas tanto sus datos como los métodos para manipularlos.

- **Distribuido:** Java proporciona un conjunto de clases para su uso en aplicaciones de red, entre otras cosas permite abrir sockets, abrir y aceptar conexiones con servidores, etc.

- **Interpretado y compilado a la vez:** Java es compilado, ya que su código fuente se transforma en una especie de código máquina. Por otra parte, es interpretado ya que los bytecodes se pueden ejecutar en cualquier máquina que disponga del intérprete.

- **Robusto:** Java proporciona numerosas comprobaciones tanto en compilación como en tiempo de ejecución. Adicionalmente, sus características liberan a los programadores de gran cantidad de comprobaciones y posibles errores.

6.1.2 Herramientas de desarrollo

La herramienta de desarrollo que se ha empleado para la implementación del proyecto ha sido Eclipse, esta decisión se debe a que este entorno ha sido utilizado para desarrollar la aplicación "Flinstones" y se adapta a la características que debe cumplir el software desarrollado para integrarlo correctamente en la aplicación.

Eclipse es un programa informático compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar "aplicaciones de cliente enriquecido". Esta plataforma ha sido usada para desarrollar entornos de desarrollo integrados (IDE), así como, para otros tipos de aplicaciones cliente.

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. En la actualidad, Eclipse es desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

Una característica fundamental que aporta el uso de Eclipse y que ha sido utilizado para el desarrollo de la aplicación "Flinstones" es la plataforma RCP (plataforma de cliente enriquecido). Esta herramienta se basa en un conjunto de plug-ins dinámicos incorporados a lo largo del desarrollo del proyecto, así como, en las API utilizadas para el desarrollo del software implementado. La principal utilidad que aporta RCP es que permite desarrollar rápidamente aplicaciones complejas, además de facilitar el mantenimiento y extensión del software. Por otra parte, los componentes que forman parte del RCP son de alta calidad, se mantiene de forma activa y son de código abierto.

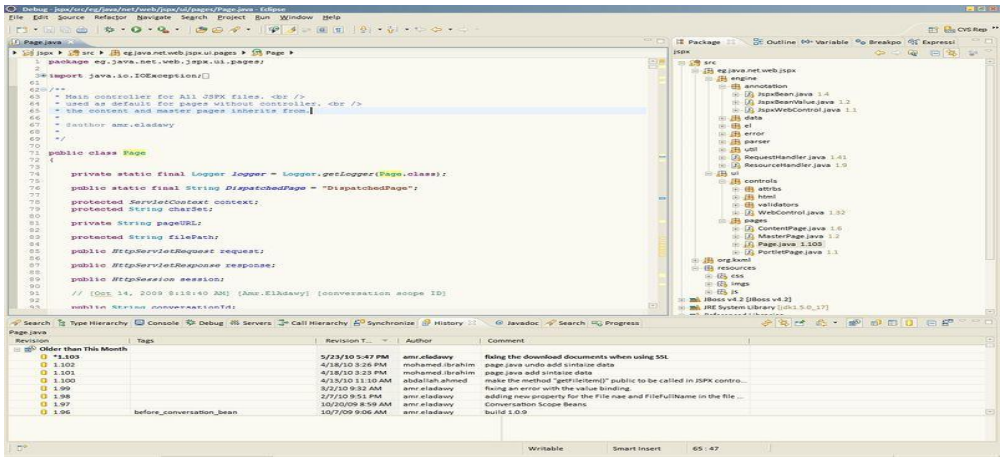


Ilustración 6.1. Entorno de trabajo Eclipse

6.2 Validación y pruebas del software

Una vez se ha realizado el diseño y desarrollo del software relativo al proyecto tratado, es necesario realizar una serie de pruebas sobre el mismo. Estas pruebas están orientadas a validar el software desarrollado, para ello se comprobará que dicho software cumpla con los requisitos funcionales previamente establecidos de forma correcta y sin errores en la ejecución.

El mantenimiento de software involucra cualquier tipo de pruebas del software realizadas, las cuales son un elemento crítico para la garantía de calidad del software y representan una revisión final de las especificaciones, del diseño y de la codificación.

El objetivo de esta fase es, una vez desarrollado el sistema, realizar un conjunto de pruebas sobre el mismo, para detectar los errores que puedan existir y solucionarlos antes de proporcionar el software definitivo. Con esto se intentará conseguir llegar a un sistema sin errores garantizando, como se ha dicho, la calidad del software. Para comprobar esto se realizará una serie de casos de test sobre el software desarrollado.

6.2.1 Casos de test

Los test diseñados para evaluar el software desarrollado durante el transcurso del proyecto son los siguientes:

Test 1: Creación de nuevo experto

- *Requisitos testeados*: RF-01
- *Acción*: Un usuario solicita la opción de añadir un nuevo experto.
- *Checkpoint 1*: El sistema muestra el formulario con los datos que se deben aportar.
- *Acción*: Un usuario completa los campos y pulsa aceptar
- *Checkpoint 2:* El sistema muestra la confirmación de que el experto se ha añadido correctamente al sistema.

Test 2: Creación de nueva alternativa

- *Requisitos testeados*: RF-02
- *Acción*: Un usuario solicita la opción de añadir una nueva alternativa.
- *Checkpoint 1*: El sistema muestra el formulario con los datos que se deben aportar.
- *Acción*: Un usuario completa los campos y pulsa aceptar
- *Checkpoint 2:* El sistema muestra la confirmación de que la alternativa se ha añadido correctamente al sistema.

Test 3: Creación de nuevo criterio

- *Requisitos testeados*: RF-03
- *Acción*: Un usuario solicita la opción de añadir un nuevo criterio.
- *Checkpoint 1*: El sistema muestra el formulario con los datos que se deben aportar.
- *Acción*: Un usuario completa los campos y pulsa aceptar
- *Checkpoint 2:* El sistema muestra la confirmación de que el criterio se ha añadido correctamente al sistema.

Test 4: Creación de nuevo dominio de expresión

- *Requisitos testeados*: RF-04
- *Acción*: Un usuario solicita la opción de añadir un nuevo dominio de expresión.
- *Checkpoint 1*: El sistema solicita el tipo de dominio que desea añadir.
- *Acción*: Un usuario selecciona el tipo de dominio y pasa al siguiente paso.
- *Checkpoint 2:* El sistema solicita los datos concretos según el dominio de expresión solicitado.
- *Acción:* Un usuario completa los datos y pulsa confirmar.
- *Checkpoint 3:* El sistema muestra la confirmación de que el nuevo dominio de expresión se ha añadido correctamente al sistema.

Test 5: Asignación de dominio de expresión

- *Requisitos testeados*: RF-05
- *Acción*: El usuario selecciona los elementos a los que les desea asignar el dominio de expresión correspondiente.

- *Checkpoint 1*: El sistema debe mostrar la confirmación de que el dominio de expresión se ha asignado correctamente.

Test 6: Cargar problema guardado

- *Requisitos testeados*: RF-06
- *Acción*: Un usuario solicita la opción de cargar problema existente.
- *Checkpoint 1*: El sistema solicita el archivo que debe cargar.
- *Acción*: Un usuario selecciona el archivo y pulsa aceptar.
- *Checkpoint 2:* El sistema muestra los datos del problema cargado en la pantalla relativa al marco del problema.

Test 7: Guardar problema

- *Requisitos testeados*: RF-07
- *Acción*: Un usuario pulsa la opción de guardar problema.
- *Checkpoint 1*: El sistema solicita el nombre del archivo.
- *Acción*: El usuario introduce el nombre y pulsa aceptar.
- *Checkpoint 2*: El sistema debe mostrar un mensaje informando de que el problema se ha guardado correctamente.

Test 8: Valorar alternativa

- *Requisitos testeados*: RF-08
- *Acción*: Un usuario pulsa la opción para valorar la alternativa deseada.
- *Checkpoint 1*: El sistema debe mostrar los campos que el usuario debe completar para realizar la valoración.
- *Acción*: El usuario completa los campos y pulsa aceptar.
- *Checkpoint 2*: El sistema debe mostrar una confirmación visual de que la valoración se ha realizado correctamente.

Test 9: Selección del método de resolución

- *Requisitos testeados*: RF-09
- *Acción*: El usuario selecciona la pestaña correspondiente a la selección del método de resolución.
- *Checkpoint 1*: El sistema muestra la pestaña correspondiente.
- *Acción*: El usuario selecciona el método de resolución que desee.
- *Checkpoint 2*: El sistema debe mostrar la información del método correspondiente, habilitando las siguientes pestañas o mostrando el mensaje del error producido.

Test 10: Resolución mediante método de decisión Topsis y operador de agregación simple

- *Requisitos testeados*: RF-10, RF-11, RF-12
- *Precondiciones*: Un usuario ha seleccionado el método Topsis y el problema se puede resolver con dicho método.
- *Acción*: El usuario selecciona la pestaña correspondiente al paso en el que se realiza la agregación de expertos.
- *Checkpoint 1*: El sistema muestra la pestaña correspondiente.
- *Acción*: El usuario selecciona el operador de agregación sin pesos ni parámetros que desee.
- *Checkpoint 2*: El sistema realiza la agregación, muestra los resultados y habilita el siguiente paso del método.
- *Acción*: El usuario selecciona la pestaña correspondiente al segundo paso del método.
- *Checkpoint 3:* El sistema muestra la pestaña seleccionada.
- Acción: El usuario selecciona el operador de distancias deseado.
- Checkpoint 4: El sistema realiza los cálculos y muestra los resultados obtenidos.

Test 11: Resolución mediante método de decisión Topsis y operador de agregación con parámetros

- *Requisitos testeados*: RF-10, RF-11, RF-13
- *Precondiciones*: Un usuario ha seleccionado el método Topsis y el problema se puede resolver con dicho método.
- *Acción*: El usuario selecciona la pestaña correspondiente al paso en el que se realiza la agregación de expertos.
- *Checkpoint 1*: El sistema muestra la pestaña correspondiente.

- *Acción*: El usuario selecciona el operador de agregación con parámetros que desee.
- *Checkpoint 2*: El sistema solicita los parámetros al usuario.
- Acción: El usuario introduce los parámetros y pulsa aceptar.
- Checkpoint 3: El sistema realiza la agregación, muestra los resultados y habilita el siguiente paso del método.
- *Acción*: El usuario selecciona la pestaña correspondiente al segundo paso del método.
- *Checkpoint 4:* El sistema muestra la pestaña seleccionada.
- Acción: El usuario selecciona el operador de distancias pesado.
- Checkpoint 5: El sistema solicita los pesos para el cálculo de las distancias.
- Acción: El usuario introduce los pesos y pulsa aceptar.
- Checkpoint 6: El sistema realiza los cálculos y muestra los resultados obtenidos.

Test 12: Resolución mediante método de decisión Topsis y operador de agregación con pesos

- *Requisitos testeados*: RF-10, RF-11, RF-14
- *Precondiciones*: Un usuario ha seleccionado el método Topsis y el problema se puede resolver con dicho método.
- *Acción*: El usuario selecciona la pestaña correspondiente al paso en el que se realiza la agregación de expertos.
- *Checkpoint 1*: El sistema muestra la pestaña correspondiente.
- *Acción*: El usuario selecciona el operador de agregación con pesos que desee.
- *Checkpoint 2*: El sistema solicita los pesos al usuario.
- Acción: El usuario introduce los pesos y pulsa aceptar.
- Checkpoint 3: El sistema realiza la agregación, muestra los resultados y habilita el siguiente paso del método.
- *Acción*: El usuario selecciona la pestaña correspondiente al segundo paso del método.
- *Checkpoint 4:* El sistema muestra la pestaña seleccionada.
- Acción: El usuario selecciona el operador de distancias deseado.
- Checkpoint 5: El sistema realiza los cálculos y muestra los resultados obtenidos.

6.2.2 Resultados de las pruebas de validación

A continuación se muestra una tabla con los resultados obtenidos, después de realizar los test diseñados sobre el software desarrollado en el transcurso del proyecto:

TEST	RESULTADO
Test 1	
Checkpoint 1	OK
Checkpoint 2	OK
Test 2	
Checkpoint 1	OK
Checkpoint 2	OK
Test 3	
Checkpoint 1	OK
Checkpoint 2	OK
Test 4	
Checkpoint 1	OK
Checkpoint 2	OK
Checkpoint 3	OK
Test 5	
Checkpoint 1	OK
Test 6	
Checkpoint 1	OK
Checkpoint 2	OK
Test 7	
Checkpoint 1	OK
Checkpoint 2	OK
Test 8	
Checkpoint 1	OK
Checkpoint 2	OK
Test 9	
Checkpoint 1	OK
Checkpoint 2	OK
Test 10	
Checkpoint 1	OK
Checkpoint 2	OK
Checkpoint 3	OK
Checkpoint 4	OK
Test 11	
Checkpoint 1	OK
Checkpoint 2	OK
Checkpoint 3	OK
Checkpoint 4	OK
Checkpoint 5	OK
Checkpoint 6	OK
Test 12	

Checkpoint 1	OK
Checkpoint 2	OK
Checkpoint 3	OK
Checkpoint 4	OK
Checkpoint 5	OK

Tabla 7.1 Resultados de pruebas de validación

La tabla anterior muestra que todos los test realizados sobre el software desarrollado en el transcurso del proyecto ofrecen resultados correctos, por tanto, se puede concluir que el software proporcionado está libre de errores y que proporciona al usuario la funcionalidad deseada, ya que los test que se han definido cubren todos los requisitos funcionales que deben proporcionar el software a los distintos usuarios del sistema.

Capítulo 7:

Conclusiones y Trabajos Futuros

7.1 Conclusiones

Día a día, los seres humanos tenemos la necesidad de tomar diversas decisiones que influyen, de forma más o menos importante, en diferentes aspectos de nuestra vida. Es habitual que tengamos que seleccionar la mejor opción entre un conjunto más o menos amplio de alternativas, siendo está decisión de vital importancia en muchas ocasiones.

A la hora de tomar una decisión, el primer paso consiste en fijar el objetivo que se desea conseguir. Posteriormente, se pasa a la recopilación de información necesaria que tenga influencia en el objetivo que se desea cumplir. Finalmente, se debe procesar la información recopilada y analizar los resultados obtenidos para poder elegir la alternativa que proporcione mejores resultados para el contexto del problema planteado.

Anteriormente, se ha dado una breve descripción de lo que sería el proceso de toma de decisión. Este proceso será más o menos complejo según la magnitud del problema que se desee solucionar, así como, de la importancia y repercusión de la decisión que se debe alcanzar. De cualquier forma, la toma de decisión suele ser un proceso complicado que requiere la división del mismo en diferentes fases para facilitar el trabajo que se debe realizar. Para facilitar este complejo trabajo aparecen diversos sistemas que ofrecen un apoyo a la hora de resolver problemas de toma de decisión.

En este contexto surgen los Sistemas de Apoyo a la Decisión (DSS). El objetivo fundamental de este tipo de sistemas no es resolver por sí mismos los problemas de toma de decisión, sino ejercer de apoyo para las personas encargadas en el proceso de toma de decisión. El sistema ayuda al usuario en la realización de multitud de tareas como son fundamentalmente: recogida de información, generación de alternativas, evaluación y comparación de las mismas.

Este proyecto surgió con el propósito de desarrollar una serie de operadores de agregación, así como, de diseñar y desarrollar el método de decisión Topsis para incorporarlos a la aplicación "Flinstones", que se corresponde con una aplicación informática de apoyo a la toma de decisiones en ambientes de incertidumbre.

El hecho de tener que incorporar tanto los operadores de agregación como el método implementados en el proyecto a una herramienta ya desarrollada ha supuesto algunos problemas adicionales puesto que ha sido necesaria la realización de un análisis exhaustivo de dicha herramienta de forma previa al desarrollo de los diferentes elementos. Este análisis es fundamental ya que es necesario entender la estructura interna de la aplicación para poder adaptar los elementos desarrollados y que estos funcionen correctamente.

Personalmente, creo que el hecho de tener que realizar el análisis de la aplicación de forma previa es complicado, pero es una acción necesaria a lo largo de la vida laboral de cualquier ingeniero informático, por lo que este proyecto me ha aportado conocimiento y experiencia en ese sentido.

Este proyecto me ha servido, además de lo dicho anteriormente, para introducirme en la teoría de decisión un proceso muy complejo, pero que en la actualidad tiene gran difusión y es de vital importancia en muchas ocasiones. Así mismo, conocer el funcionamiento y la estructura general de las herramientas de apoyo a la toma de decisión.

Desde el punto de vista del desarrollo, se han seguido las fases de la Ingeniería del Software, ya que al ser un proyecto software es fundamental realizar determinadas fases para obtener un producto de calidad. En primer lugar, se define de forma precisa los requerimientos que debe cumplir el sistema, esto es algo fundamental para poder realizar el análisis y diseño posterior. Posteriormente, se realiza el análisis del sistema obteniendo un modelo que sea correcto, completo, consistente y verificable, para, a continuación, pasar a realizar el diseño detallado del mismo. Finalmente, se procede a la fase de implementación del software y realizar las pruebas para verificar y validar el funcionamiento del sistema final.

En dicho desarrollo, se han encontrado más complejidad en el diseño y desarrollo del método Topsis que en el desarrollo de los operadores de agregación. Esto es algo obvio debido a que el método requiere un análisis mucho más complejo, así como, un proceso de resolución más largo en el que se deben tener en cuenta multitud de factores.

Como conclusión, indicar que desde un punto de vista personal creo que he cubierto el propósito definido al inicio del proyecto y he cumplido con las expectativas personales puestas sobre la realización del mismo.

7.2 Trabajos Futuros

Dado que la teoría de decisión es un área importante y de gran utilidad tanto a nivel empresarial como comercial y los sistemas de apoyo a la decisión están adquiriendo cada vez más importancia para ayudar a la solución de este tipo de procesos, las evoluciones en este tipo de sistemas no se detienen y se centrarán fundamentalmente en los siguientes aspectos:

- **Apoyo a las decisiones simultaneas:** El gran desarrollo e innovación que se ha producido en las comunicaciones de datos, en funciones tales como correo electrónico, redes locales y teleconferencias, facilitan el hecho de que los sistemas de apoyo tengan tendencia a la realización del proceso de toma de decisión es grupo y de forma simultánea.

- **Sistemas distribuidos de apoyo a las decisiones:** Este punto tiene relación con el anterior, debido que el hecho de tomar decisiones simultaneas implica la necesidad de coordinar las acciones de diversas localidades remotas.

- **Apoyo gráfico:** Se aplicarán las grandes evoluciones dadas en el área gráfica de los sistemas informáticos para mejorar la visualización de la información, proporcionando gráficos y nuevos elementos que faciliten el análisis de los resultados aportados.

- **Reconocimiento de voz:** Se tenderá al desarrollo de sistemas compatibles con los nuevos avances tecnológicos, como por ejemplo el reconocimiento de voz, lo que facilitará el suministro de información llevado a cabo por el usuario del sistema.

- **Descentralización del proceso de toma de decisiones:** El proceso de toma de decisiones ha estado centralizado en la mayoría de las organizaciones empresariales tradicionalmente. La aparición de estándares de trabajo que imponen técnicas de calidad y reingeniería de procesos presuponen que las decisiones deben tomarse en el nivel más bajo posible dentro del ámbito de la decisión que se está tomando, para poder ajustarse rápidamente a los posibles cambios en las condiciones del problema. Esto hace que se deba descentralizar el proceso de toma de decisiones, para que las personas responsables de dicha tarea dispongan de información fresca y actualizada, sea cual sea el nivel al que pertenece.

Capítulo 8:

Referencias Bibliográficas

8. Referencias Bibliográficas

[1] Finlay, P. N. (1994). Introducing decision support systems. Oxford, UK Cambridge, Mass., NCC Blackwell; Blackwell Publishers.

[2] Druzdzel, M. J. and R. R. Flynn (1999). Decision Support Systems. Encyclopedia of Library and Information Science. A. Kent, Marcel Dekker, Inc.

[3] Gui-Wu Wei (2009). Extension of TOPSIS method 2-tuple linguistic multiple attribute group decision making with incomplete weight information.

[4] ECKEL B. (2006) Thinking in Java, Prentice-Hall International Edition.

[5] After, S.L. (1980). Decision support systems: current practice and continuing challenges. Reading, Mass., Addison-Wesley Pub.

[6] Druzdzel, M. J. and R. R. Flynn (1999). Decision Support Systems. Encyclopedia of Library and Information Science. A. Kent, Marcel Dekker, Inc.

[7] Página Web del proyecto "Flinstones": http://serezade.ujaen.es/flinstones/

[8] R.R. Yager, On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decision Making, IEEE Transaction on Systems, Man and Cybernetics, 18 (1988) 183-190.

[9] D. Dubois, H. Prade, A Review of Fuzzy Sets Aggregation Connectives, Information Sciences, 36 (1985) 85-121.

[10] D. Dubois, J.-L. Koning, Social Choice Axioms for Fuzzy Sets Aggregation, Fuzzy Sets and Systems, 43 (1991) 257-274.

[11] J. Kacprzyk, M. Roubens, Non-Conventional Preference Relations in Decision Making (Springer-Verlag, 1988).

[12] Gleb Beliakov, Ana Pradera, Tomasa Calvo: Aggregation Functions: A Guide for Practitioners. Studies in Fuzziness and Soft Computing 221, Springer 2007, ISBN 978-3-540-73720-9, pp. 1-304

[13] Gui-Wu Wei, Extension of TOPSIS method for 2-tuple linguistic multiple attribute group decision making with incomplete weight information (Springer-Verlag, 2009).

[14] Lai YJ, Lui TY, Hwang CL (1994) TOPSIS for MODM. Eur J Oper Res 76 (3):486-500.

[15] Hwang CL, Yoon K (1981) Multiple attribute decision making methods and applications. Springer, Berlin.

[16] Dyer JS, Fishburn PC, Steuer RE (1992) Multiple criteria decision making, multiattribute utility theory: the next ten years. Manag Sci 38(5):645-654.

[17] Abo-Sinna MA, Amer AH (2005) Extensions of TOPSIS for multi-objetive large-scale nonlinear programming problems. Appl Math Comput 162:243-256.

[18] Pressman, Roger S. (2003), "Ingeniería del Software. Un enfoque práctico", Quinta edición, México: Mc Graw Hill.

[19] Larman, Craig (2003), "UML y Patrones", Segunda Edición, Prentice Hall.

[20] M. Espinilla, "Nuevos Modelos de Evaluación Sensorial con Información Lingüística". Memoria de Tesis Doctoral, Universidad de Jaén, 2009.

[21] C. Romero. "Teoría de la decisión multicriterio: conceptos, técnicas y aplicaciones". Alianza Universal, Madrid, 1993.

[22] C.H. Cheng and Y. Lin. "Evaluating the best main battle tank using fuzzy decision theory with linguistic criteria evaluation". European Journal of Operational Research, 142(1):174-186, 2002.

[23] T. Bilgiç. "Interval-valued preference structures". European Journal of Operational Research, 105(1):162-183, 1998.

[24] F. Herrera and L. Martinez. "The 2-tuple linguistic computational model. Advantages of its linguistic description, accuracy and consistency". International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 9(Suppl.):33-49, 2001.

- **Métodos implementados en Flinstone:**

[25] F. Herrera, E. Herrera-Viedma, L. Martínez, A fuzzy linguistic methodology to deal with unbalanced linguistic term sets, IEEE Transactions on Fuzzy Systems 16 (2) (2008) 354–370.

[26] F. Herrera, L. Martínez, A model based on linguistic 2-tuples for dealing with multigranularity hierarchical linguistic contexts in multiexpert decision-making, IEEE Transactions on Systems, Man and Cybernetics. Part B: Cybernetics 31 (2) (2001) 227–234.

[27] F. Herrera, E. Herrera-Viedma, L. Martínez, A fusion approach for managing multi-granularity linguistic term sets in decision making, Fuzzy Sets and Systems 114 (1) (2000) 43–58.

[28] M. Espinilla, J. Liu, L. Martínez, An extended hierarchical linguistic model for decision-making problems, Computational Intelligence 27 (3) (2011) 489–512.

[29] F. Herrera, L. Martínez, P. Sánchez, Managing non-homogeneous information in group decision making, European Journal of Operational Research 166 (1) (2005) 115–132.

- **Enfoque lingüístico difuso y lógica difusa**

[30] L. Zadeh, The concept of a linguistic variable and its applications to approximate reasoning, Information Sciences, Part I, II, III 8,8,9 (1975) 199–249,301–357,43–80.

[31] L. Zadeh, Fuzzy sets, Information and Control 8 (3) (1965) 338–353.

- **2-tuplas:**

[32] F. Herrera, L. Martíınez, A 2-tuple fuzzy linguistic representation model for computing with words, IEEE Transactions on Fuzzy Systems 8 (6) (2000) 746–752.

[33] F. Herrera, L. Martínez, The 2-tuple linguistic computational model. Advantages of its linguistic description, accuracy and consistency, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 9 (Suppl.) (2001) 33–49.

- **Soporte a la decisión**

[34] J. Lu, G. Zhang, D. Ruan, Multi-objective Group Decision Making: Methods, Software and Applications With Fuzzy Set Techniques, Imperial College Press, London, UK, UK, 2007.

[35] J. Ma, J. Lu and G. Zhang., Decider: A fuzzy multi-criteria group decision support system, Knowledge-Based Systems, Vol. 23, No.1, (2010) 23-31.

Apéndice A

Manual de Usuario

A. Manual de usuario

El objetivo de este apartado es mostrar un manual en el que se explicará el funcionamiento de la aplicación "Flinstones", se pretende así, que el usuario pueda aprender a utilizar la aplicación rápidamente y sacar el máximo partido de ésta de la forma más sencilla posible.

Para realizar el manual se ha optado por mostrar la resolución paso a paso de un problema concreto de toma de decisión. De esta forma, el lector del manual sabrá los pasos fundamentales que debe seguir para resolver este tipo de problemas usando la aplicación.

La aplicación "Flinstones" permite la resolución de problemas de toma de decisión utilizando múltiples métodos, en este caso, se va a mostrar el ejemplo utilizando el método de decisión Topsis, ya que ha sido el que se ha desarrollado en el proyecto. Algunos pasos de los métodos de decisión, como la definición de los elementos del problema y evaluación de las alternativas, son comunes a todos ellos, aun así, una vez se seleccione el método el proceso suele variar según dicha selección. A pesar de ello, una vez explicado un método de decisión es fácil aprender a utilizar el resto, puesto que la interacción es parecida.

El ejemplo con el que se ilustrará el uso de la aplicación será el mismo que se ha expuesto en el apartado 3.2.1 de esta memoria, aplicando el método de decisión Topsis para 2-tuplas lingüísticas. De esta forma, a la vez que se expone el funcionamiento del sistema se demuestra que los resultados obtenidos mediante el método desarrollado son correctos.

Antes de pasar a resolver paso a paso el problema, se expondrá el contexto del mismo, así como, los componentes que deberán tenerse en cuenta a la hora de realizar la resolución. A continuación se exponen las condiciones del problema:

El ejemplo a resolver plantea un problema en el que se supone que se está realizando una investigación para descubrir que alternativa es mejor para realizar una inversión de capital. Las 5 posibles alternativas son: A_1 una compañía de coches, A_2 una compañía de alimentos, A_3 una compañía de ordenadores, A_4 una compañía de armas y A_5 una compañía de televisión. La decisión debe ser tomada atendiendo a 4 criterios: G_1 es el análisis de riesgo, G_2 es el análisis de expansión, G_3 es el impacto socio-político, G_4 es el análisis del impacto ambiental.

Las valoraciones realizadas por los 3 expertos son las siguientes:

	G1	*G2*	*G3*	*G4*
A1	M	G	P	P
A2	P	VP	M	P
A3	G	M	G	EP
A4	VG	P	P	G
A5	EG	EP	VP	M

Tabla A1.1. Valoraciones Experto 1

	G1	*G2*	*G3*	*G4*
A1	P	M	VP	VP
A2	VP	EP	G	G
A3	M	G	P	EG
A4	EG	VP	VP	M
A5	P	VP	M	VP

Tabla A1.2. Valoraciones Experto 2

	G1	*G2*	*G3*	*G4*
A1	G	P	VP	VG
A2	VP	G	P	G
A3	VG	VP	G	P
A4	G	VG	EG	VP
A5	M	VP	M	G

Tabla A1.3. Valoraciones Experto 3

Una vez presentado el problema, se pasa a la resolución del mismo. La resolución de problemas de toma de decisión utilizando el DSS "Flinstones" se compone de una serie de pasos que se exponen a continuación:

Paso 1: Marco de evaluación

Se corresponde con el primer paso que se debe dar a la hora de resolver cualquier problema de toma de decisión. En este primer paso, se pretende establecer el marco de evaluación del problema que se desea resolver, es decir, se define la estructura que tendrá el problema, así como, los elementos principales que lo compondrán.

La primera pantalla que se mostrará cuando se ejecute el software pretende cumplir este primer paso esencial en cualquier problema de toma de decisiones, dicha pantalla es la siguiente:

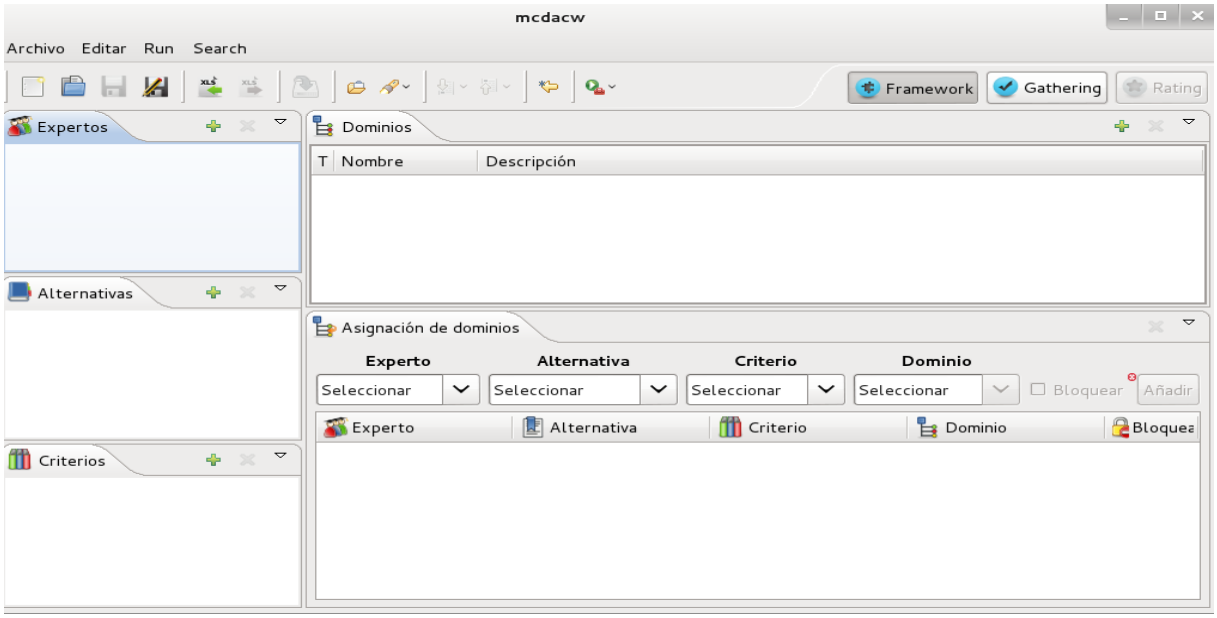


Ilustración A1.1 Pantalla Inicial de la Aplicación

En esta ventana, se define el marco de evaluación del problema, por tanto, se deben incluir todos los elementos involucrados en el mismo. A continuación, se muestra paso a paso como se deben introducir cada uno de estos elementos:

- **Expertos:**

Para añadir un nuevo experto en el marco del problema se debe pulsar sobre el botón con el signo '+' situado en el recuadro de la parte izquierda de la pantalla correspondiente a los expertos del problema abierto. En la siguiente imagen se muestra dicho botón:

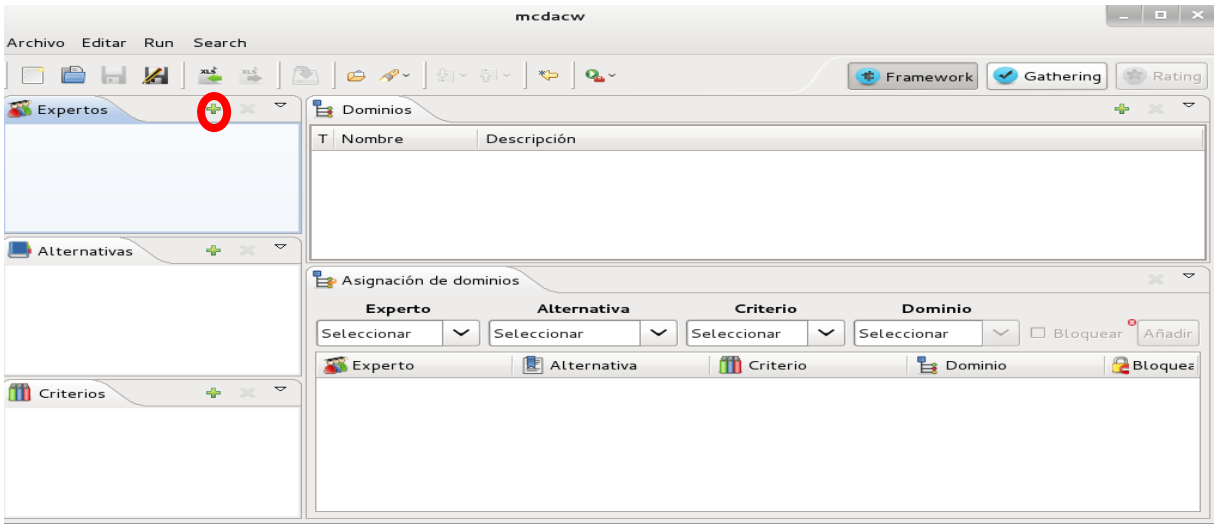


Ilustración A1.2 Añadir Expertos

Una vez pulsado dicho botón, el sistema solicitará la información necesaria para crear un nuevo experto:

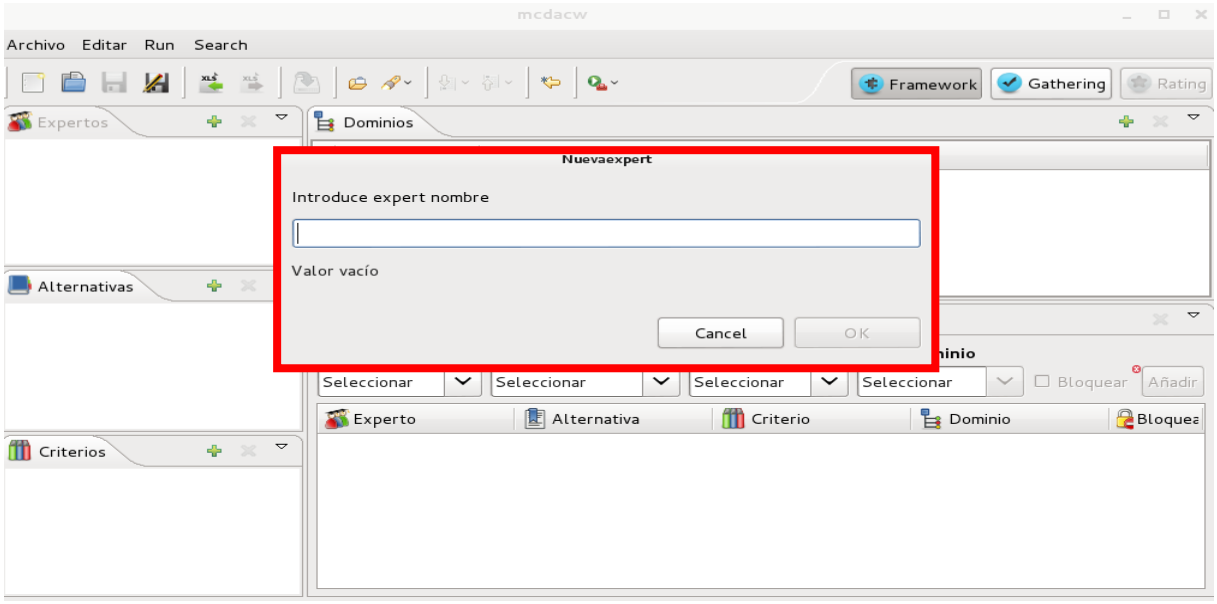
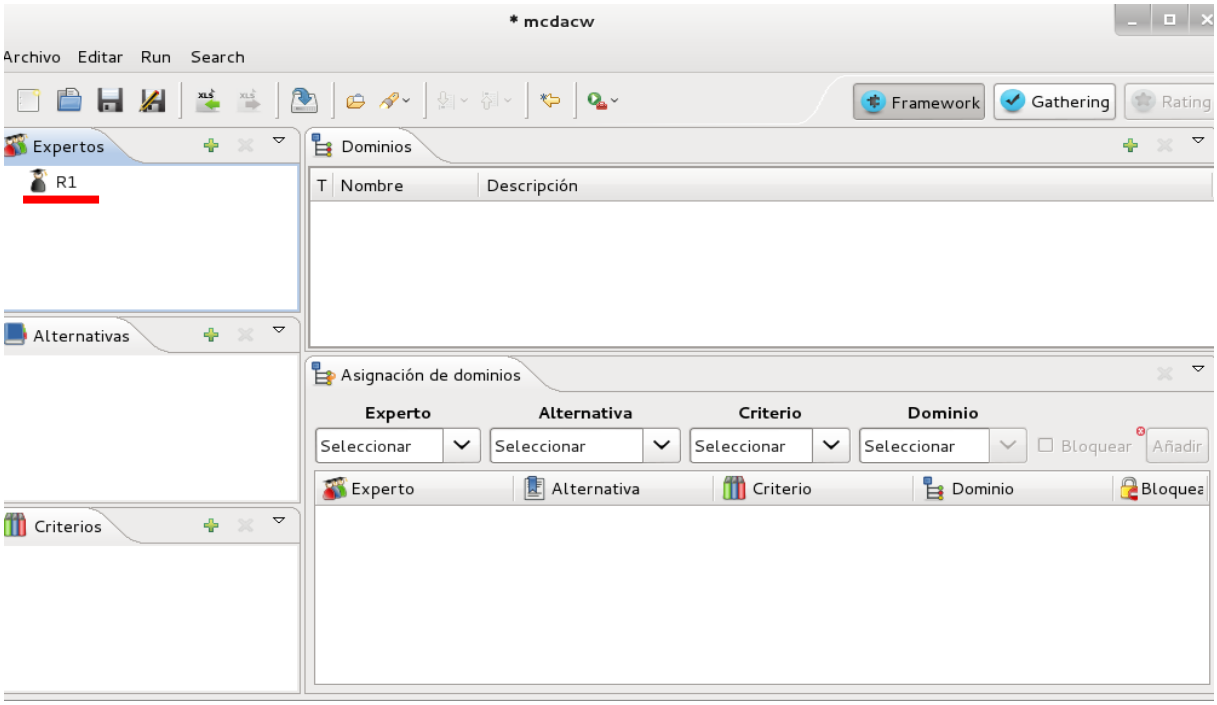


Ilustración A1.3 Datos del experto

A la hora de rellenar el nombre del experto se debe tener en cuenta que no puede quedarse vacio ni coincidir con un nombre de experto ya creado. Una vez introducidos los datos de forma correcta, se debe pulsar el botón de confirmación 'OK' y se podrá visualizar el nuevo experto en el lado izquierdo de la pantalla:



A1.4 Nuevo experto

Como se puede ver, se ha añadido de forma correcta el nuevo experto a los elementos que componen el problema de toma de decisión. En la definición del ejemplo existen tres expertos, por tanto, este proceso habrá que repetirlo dos veces más:

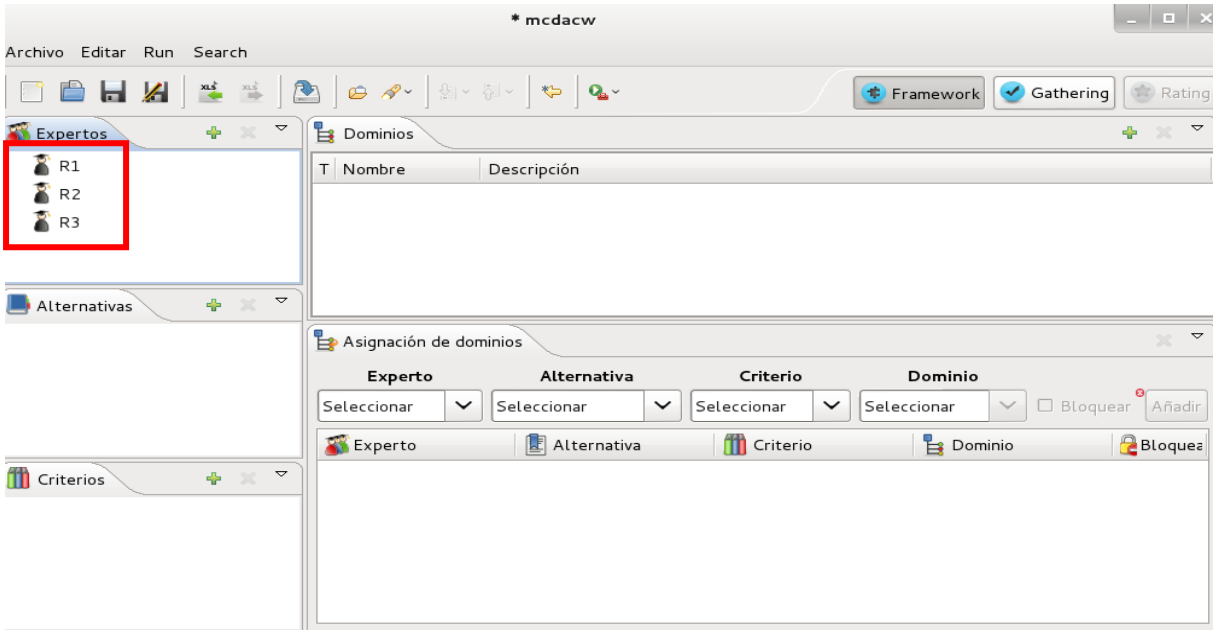


Ilustración A1.5 Expertos del problema

- **Alternativas:**

Para añadir una nueva alternativa en el marco del problema se debe pulsar sobre el botón con el signo '+' situado en el recuadro de la parte izquierda de la pantalla correspondiente a las alternativas del problema abierto. En la siguiente imagen se muestra dicho botón:

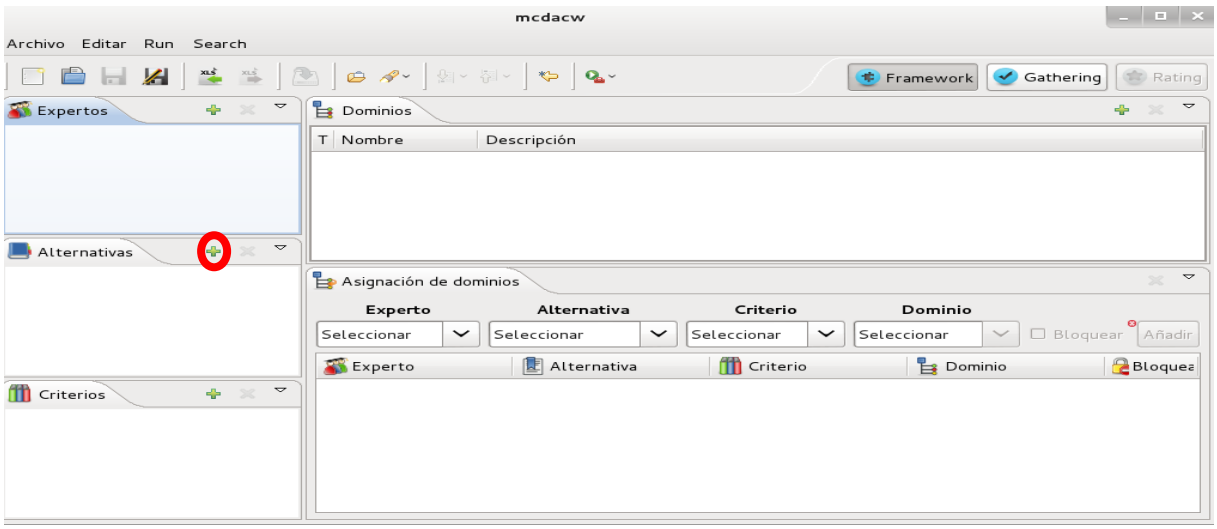


Ilustración A1.6 Añadir Alternativa

Una vez pulsado dicho botón, el sistema solicitará la información necesaria para crear una nueva alternativa:

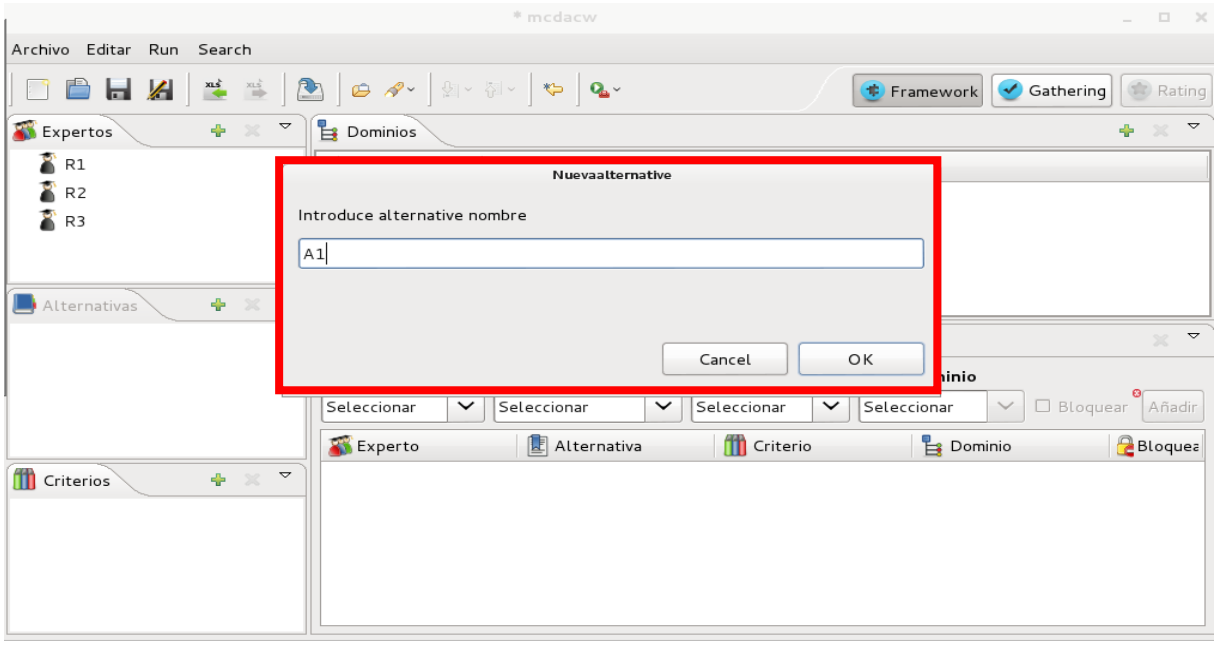
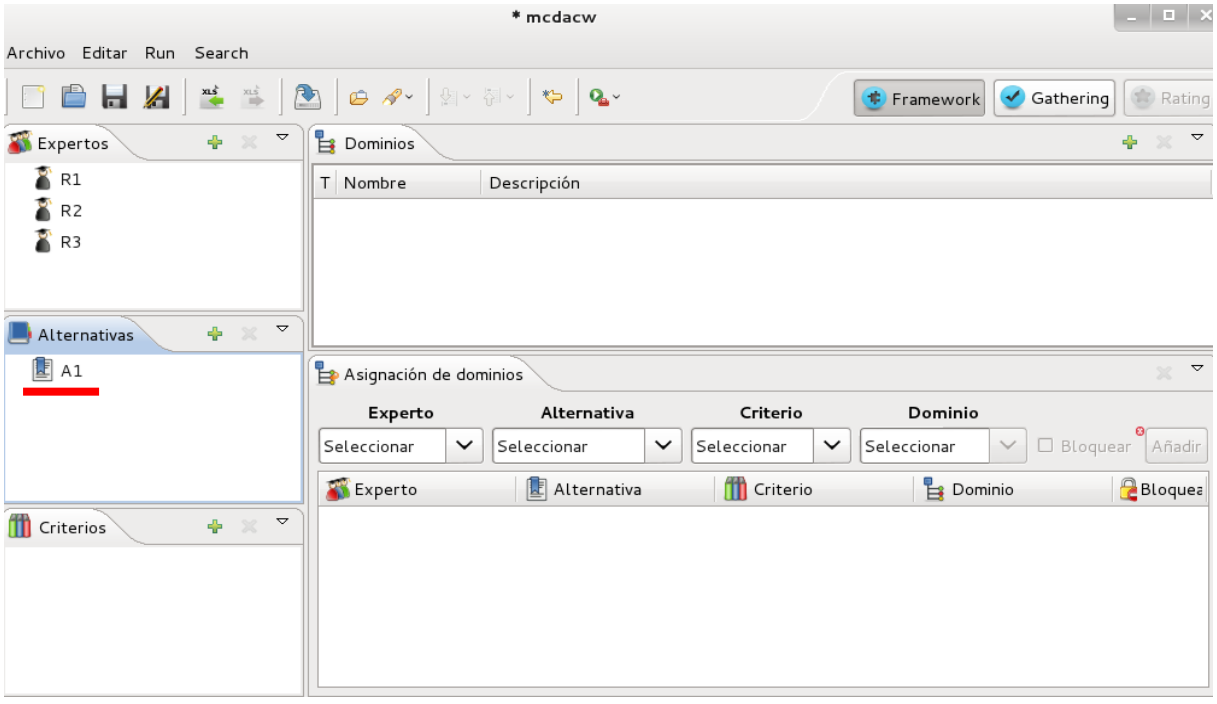


Ilustración A1.7 Datos de la alternativa

A la hora de rellenar el nombre de la alternativa se debe tener en cuenta que no puede quedarse vacio ni coincidir con un nombre de alternativa ya creada. Una vez introducidos los datos de forma correcta, se debe pulsar el botón de confirmación 'OK' y se podrá visualizar la nueva alternativa en el lado izquierdo de la pantalla:



A1.8 Nueva alternativa

Como se puede ver, se ha añadido de forma correcta la nueva alternativa a los elementos que componen el problema de toma de decisión. En la definición del ejemplo existen cinco alternativas, por tanto, este proceso habrá que repetirlo cuatro veces más:

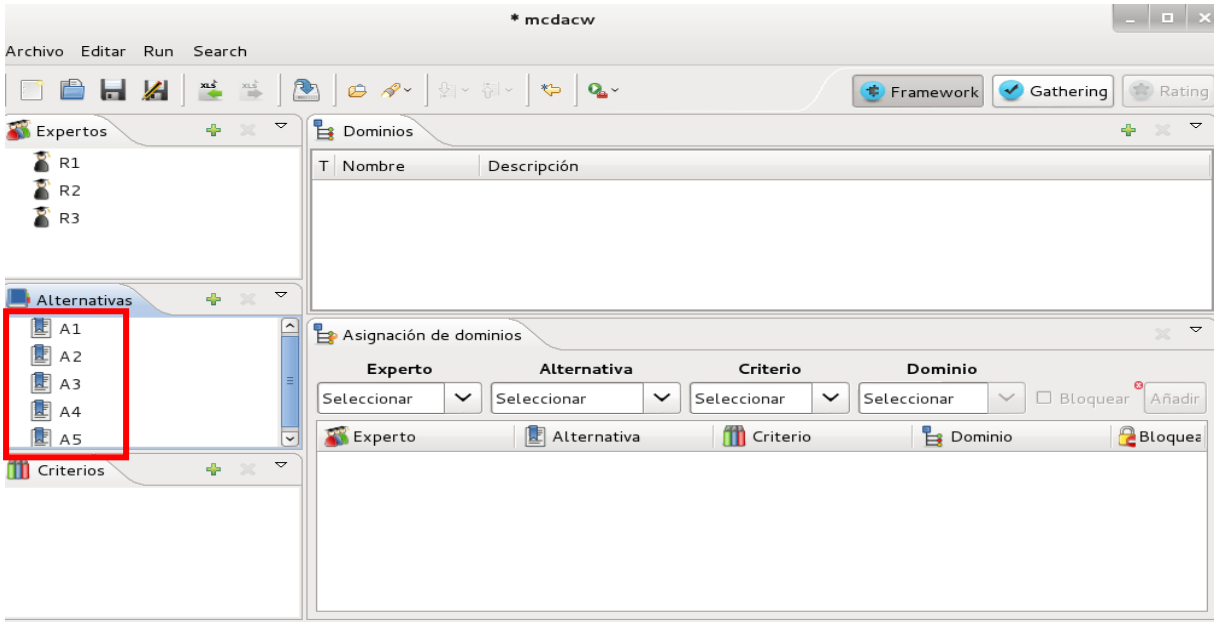


Ilustración A1.9 Alternativas del problema

- **Criterios:**

Para añadir un nuevo criterio en el marco del problema se debe pulsar sobre el botón con el signo '+' situado en el recuadro de la parte izquierda de la pantalla correspondiente a los criterios del problema abierto. En la siguiente imagen se muestra dicho botón:

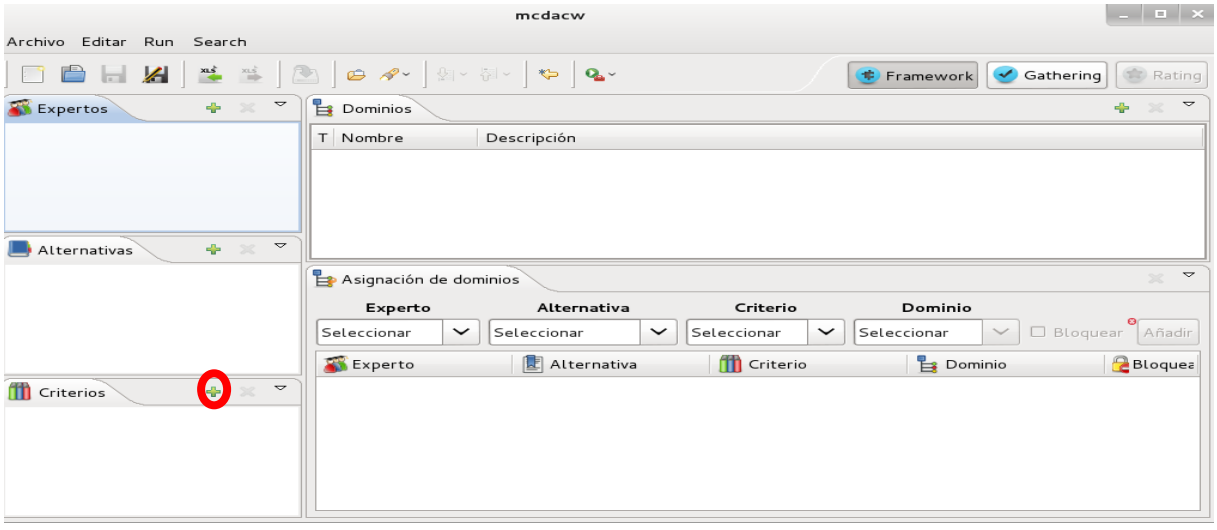


Ilustración A1.10 Añadir Criterio

Una vez pulsado dicho botón, el sistema solicitará la información necesaria para crear un nuevo criterio:

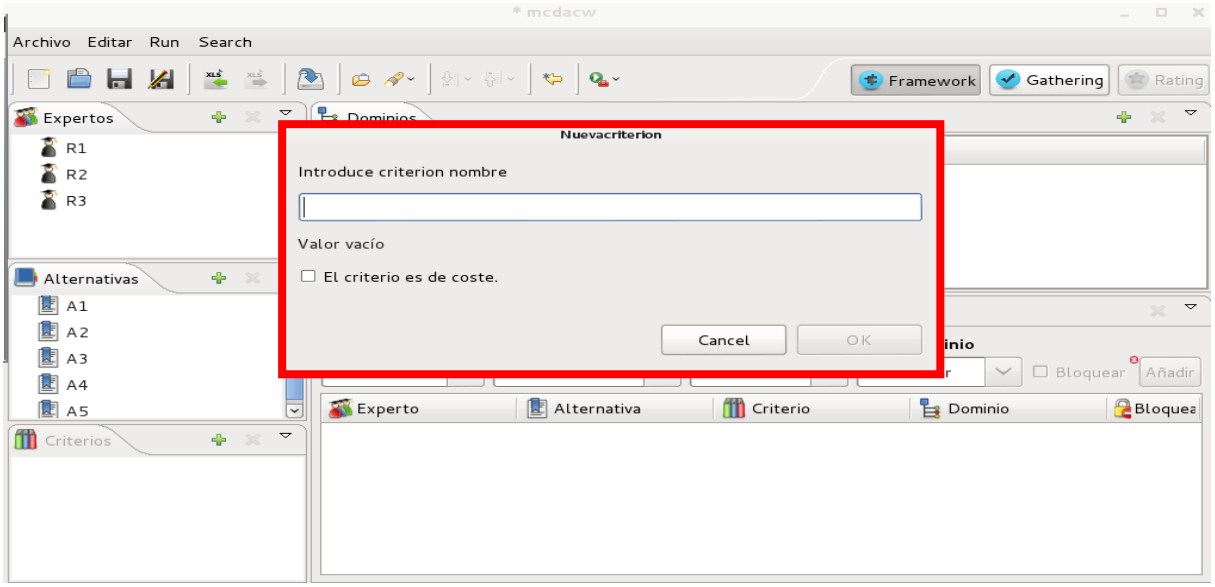
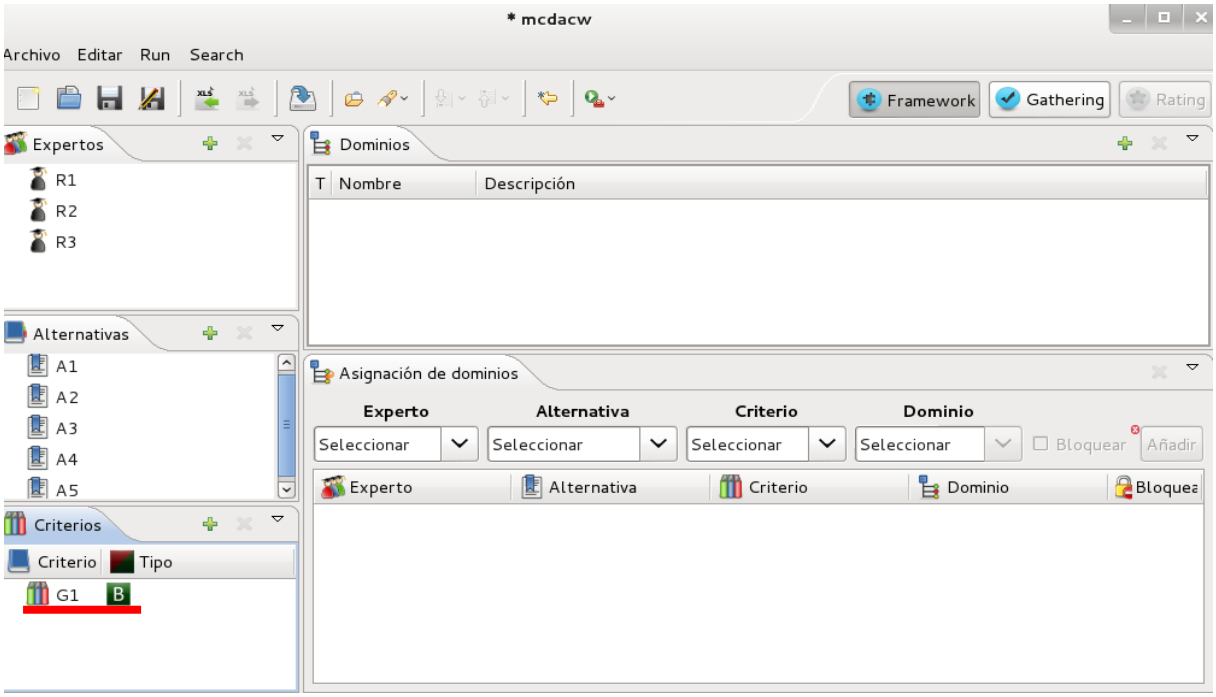


Ilustración A1.11 Datos del criterio

A la hora de rellenar el nombre del criterio se debe tener en cuenta que no puede quedarse vacio ni coincidir con un nombre de criterio ya creado. A diferencia de los expertos y las alternativas, en este caso se solicita también el tipo del criterio (beneficio o coste). Una vez introducidos los datos de forma correcta, se debe pulsar el botón de confirmación 'OK' y se podrá visualizar el nuevo criterio en el lado izquierdo de la pantalla:



A1.12 Nuevo criterio

Como se puede ver, se ha añadido de forma correcta el nuevo criterio, cuyo tipo es beneficio, a los elementos que componen el problema de toma de decisión. En la definición del ejemplo existen cuatro criterios todos ellos de beneficio, por tanto, este proceso habrá que repetirlo tres veces más:

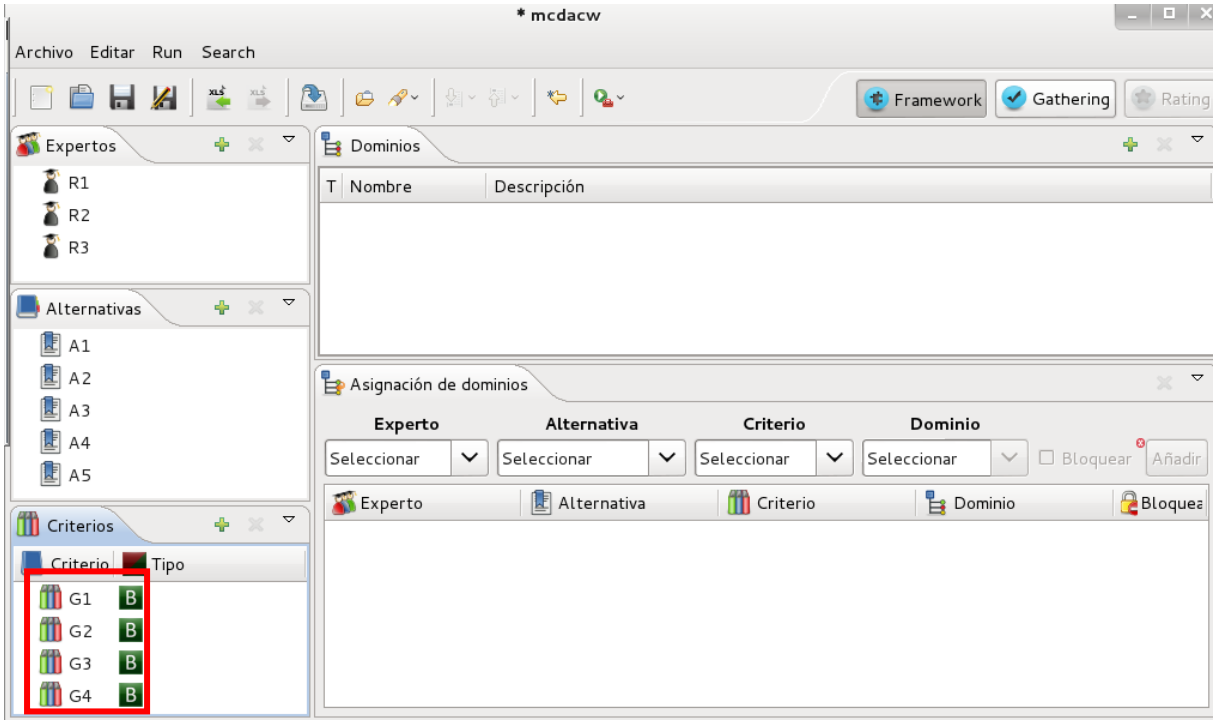


Ilustración A1.13 Criterios del problema

Nota: Cualquiera de los elementos anteriormente definidos se pueden eliminar si se desea, para ello basta con seleccionarlos y pulsar sobre el aspa roja que se activará cuando se produzca dicha selección, situada junto a la cruz pulsada para añadir los distintos elementos:

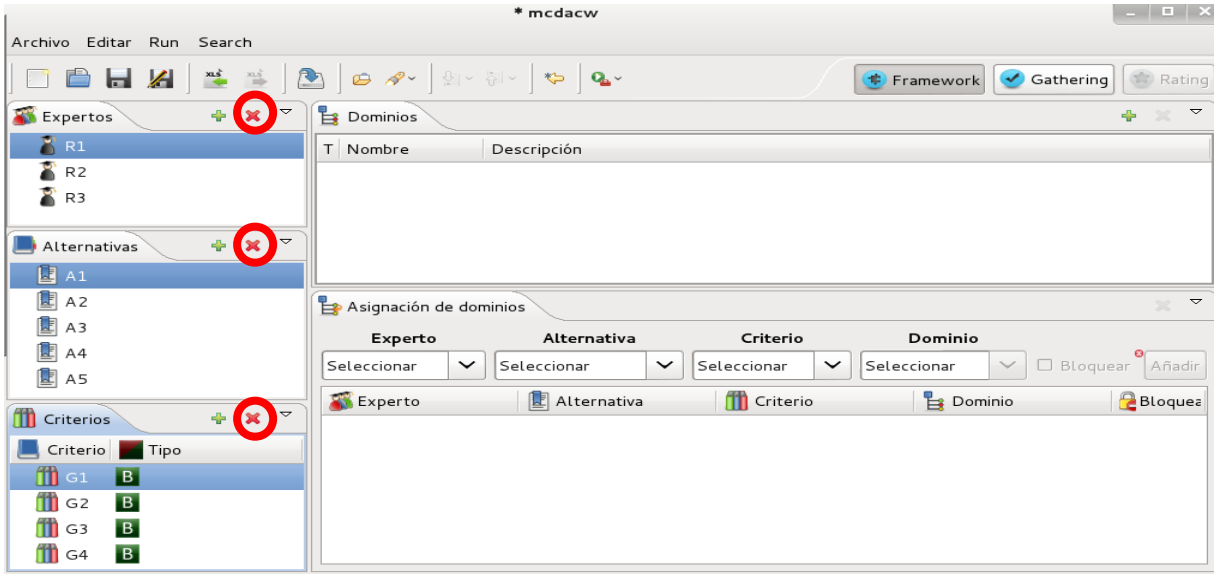


Ilustración A1.14 Eliminación de elementos

- **Dominios de expresión:**

Para crear un nuevo dominio de expresión en el marco del problema se debe pulsar sobre el botón con el signo '+' situado en el recuadro de superior de la zona central de la pantalla correspondiente a los diferentes dominios. En la siguiente imagen se muestra dicho botón:

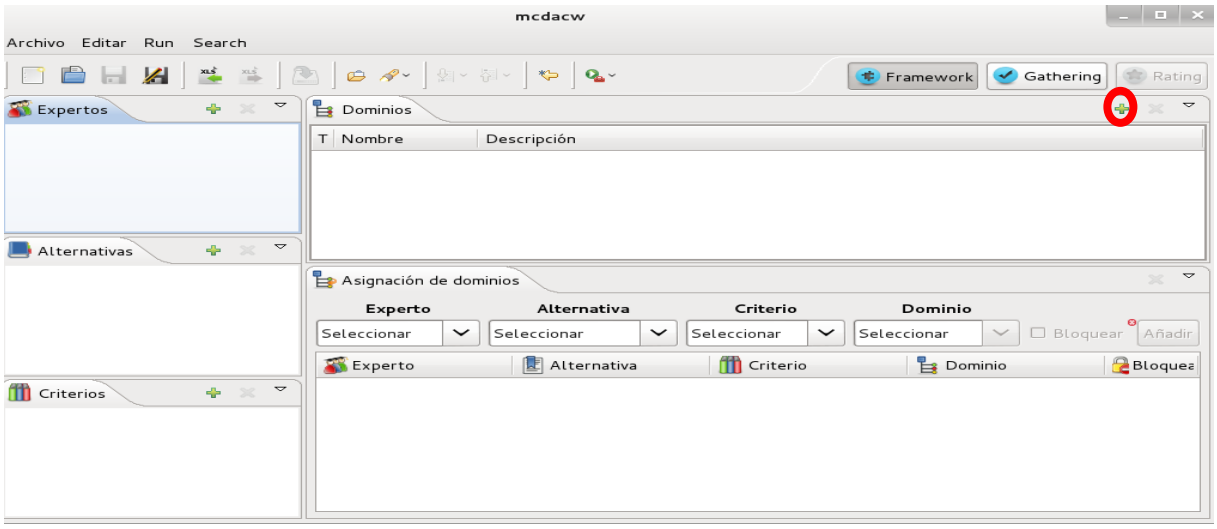


Ilustración A1.15 Añadir Dominio de Expresión

Una vez pulsado dicho botón, el sistema solicitará la información necesaria para crear un nuevo dominio de expresión:

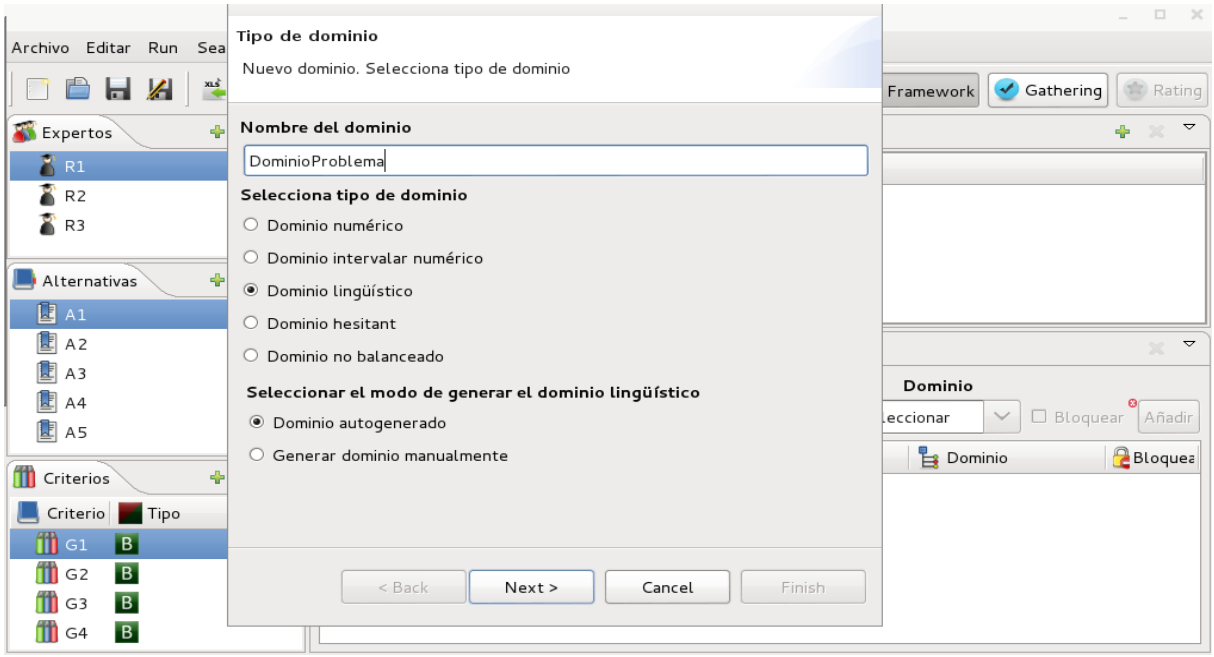


Ilustración A1.16 Datos del dominio de expresión

Para crear el dominio de expresión se debe indicar el nombre del mismo y seleccionar el tipo del dominio que se desea generar, en este caso, dominio lingüístico autogenerado. Si todo es correcto se podrá pasar al siguiente paso pulsando el botón para continuar:

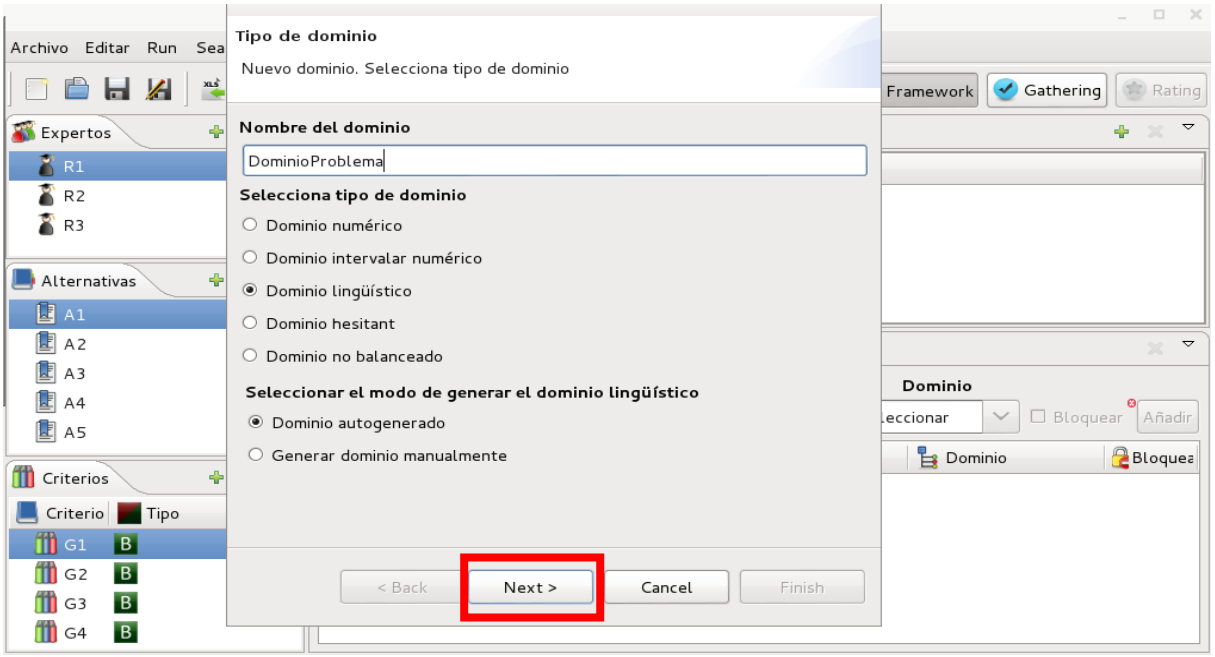


Ilustración A1.17 Definición del dominio de expresión

Una vez pulsado el botón anteriormente mostrado, se pasará a definir las etiquetas del dominio que se desea definir:

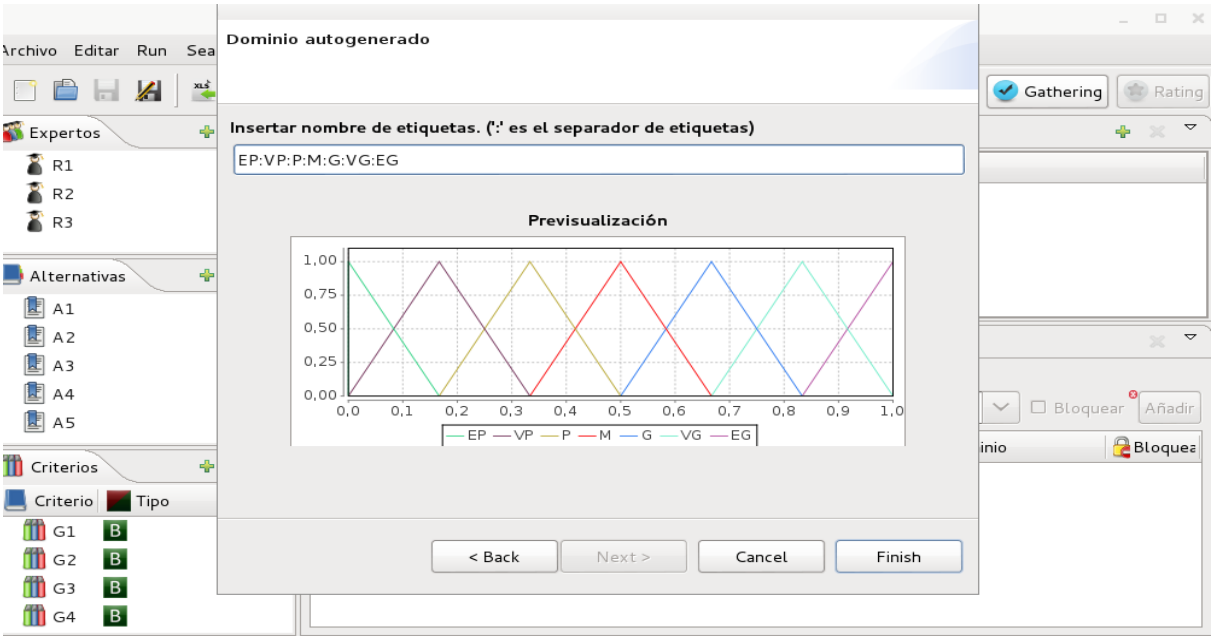
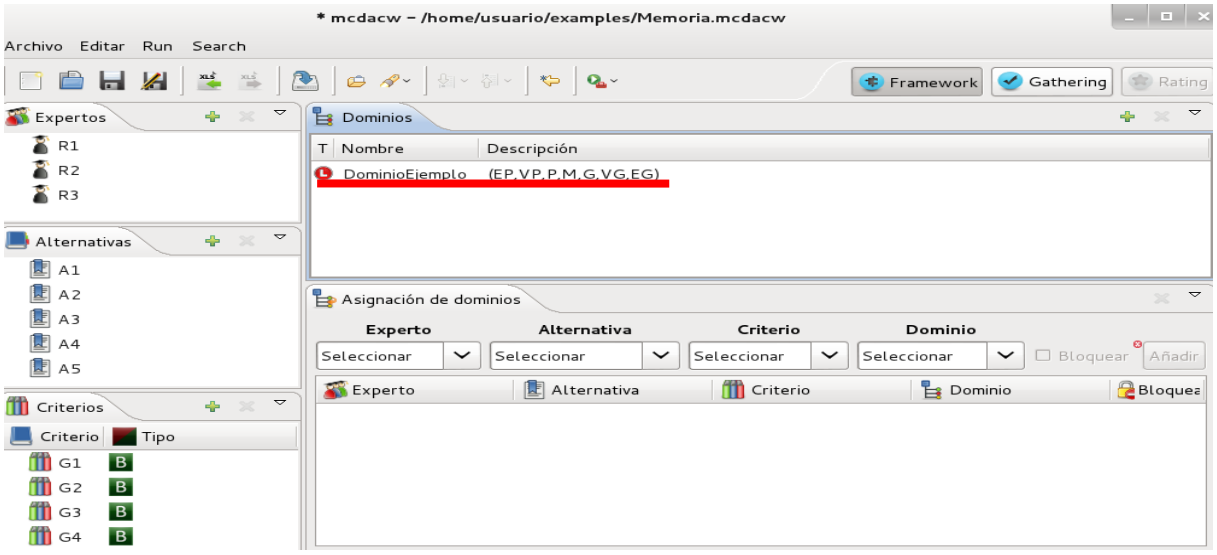


Ilustración A1.18 Definición del dominio de expresión 2

Para definir dichas etiquetas basta con introducir su nombre en el campo de texto superior y, al ser autogenerado, el sistema genera directamente el dominio de expresión, así como, el gráfico de muestra. Una vez se introduzcan las etiquetas y se pulse el botón para finalizar, se podrá visualizar el dominio definido en la zona centrar correspondientes a todos los dominios introducidos:



A1.19 Nuevo Dominio de Expresión

- **Asignación de dominios de expresión:**

Una vez definidos tanto los elementos del problema como los dominios de expresión que se van a utilizar, el último paso que se debe realizar para completar la definición del marco del problema es asignar a cada elemento su dominio de expresión correspondiente. Esto se realiza en la zona inferior de la parte central de la pantalla:

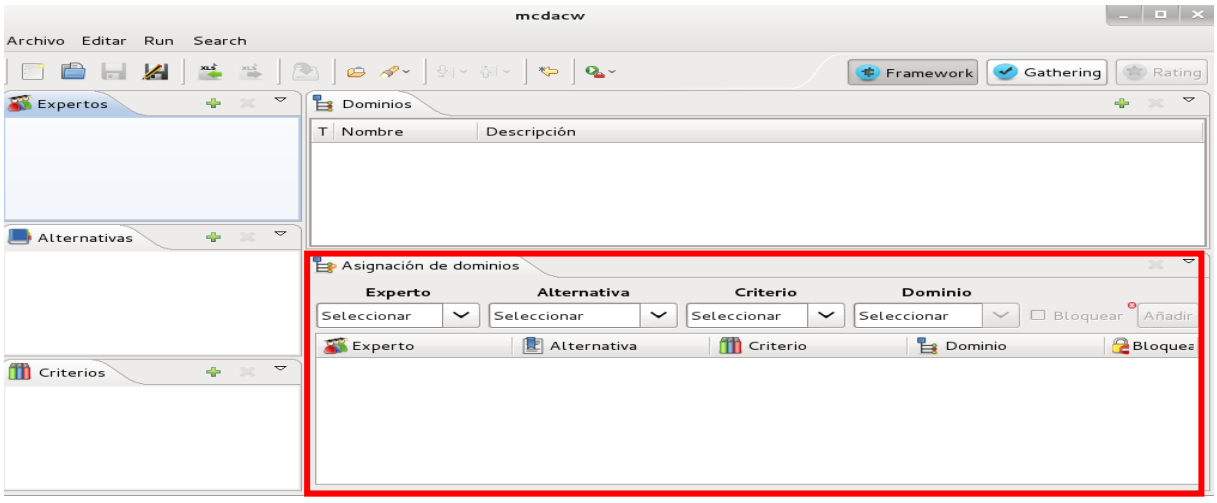


Ilustración A1.20 Asignar Dominio de Expresión

En el ejemplo que se está llevando a cabo sólo hemos definido un dominio de expresión, por tanto, se debe asignar dicho dominio a todos los elementos del problema. Para ello, debemos seleccionar todos los elementos del problema, el dominio de expresión y pulsar el botón para añadir:

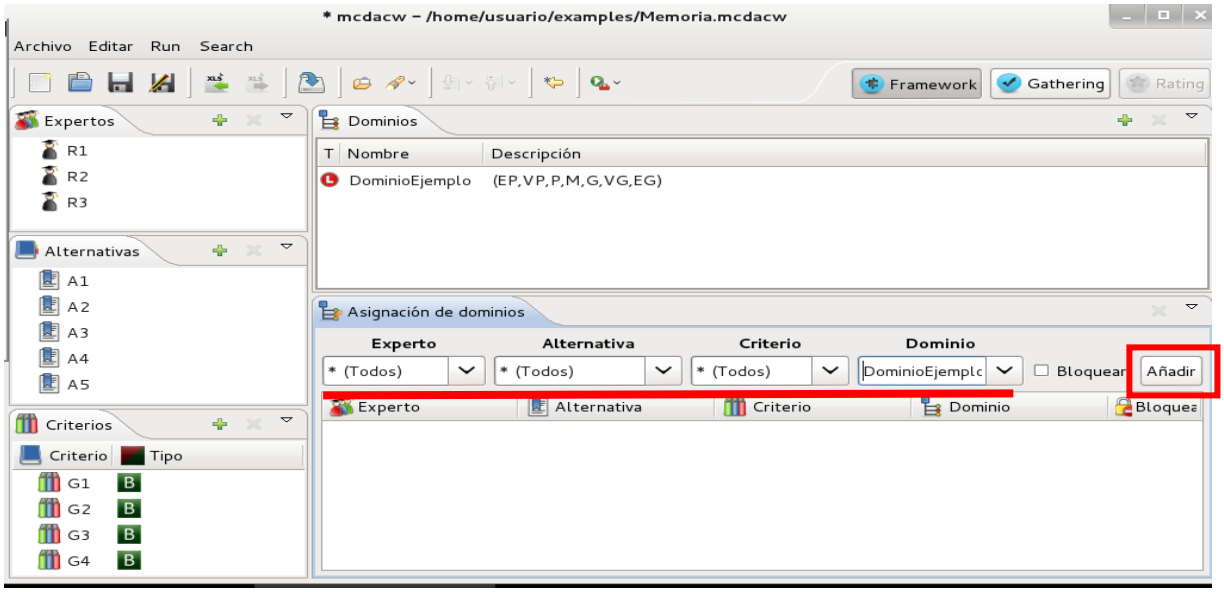


Ilustración A1.21 Asignar Dominio de Expresión 2

Una vez hecho pulsado dicho botón, se podrá visualizar la asignación realizada en la zona correspondiente a las asignaciones de dominio realizadas en el problema:

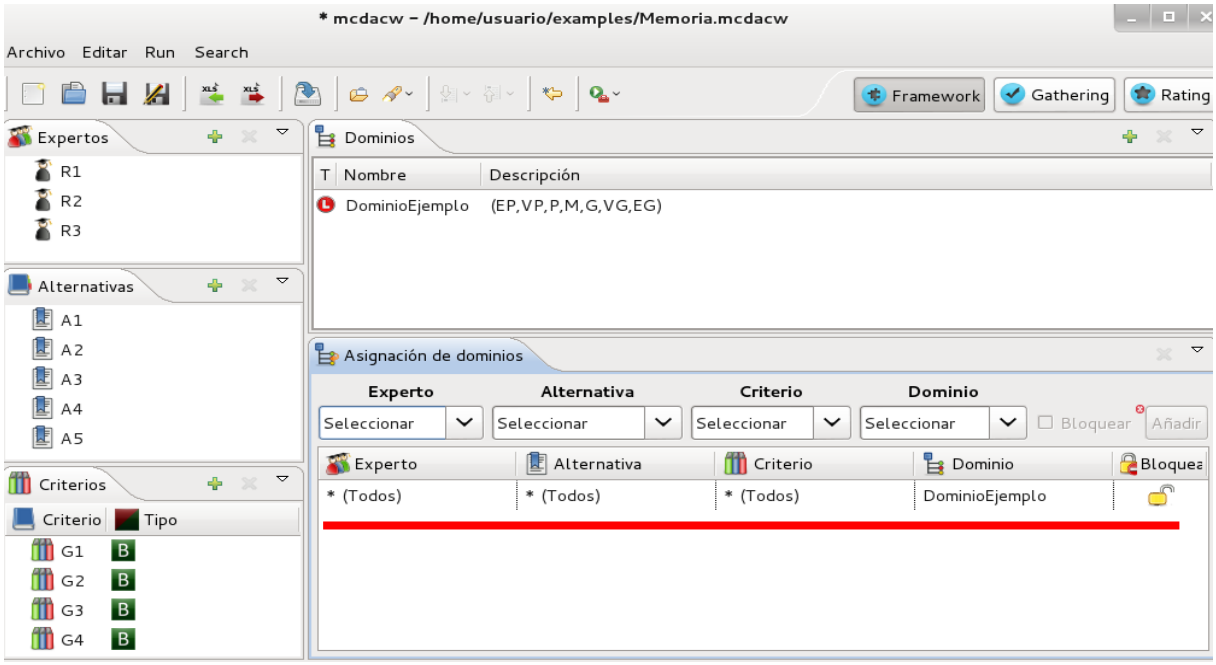


Ilustración A1.22 Resultado de la asignación del dominio de expresión

Paso 2: Recogida de información

Una vez definido el marco del problema de decisión que se desea solucionar con todos los elementos que se deben tener en cuenta, se pasa a introducir la información relativa a las diferentes valoraciones proporcionadas por los distintos expertos. Para ello, el usuario debe seleccionar la pestaña correspondiente a la recogida de la información asociada al problema:

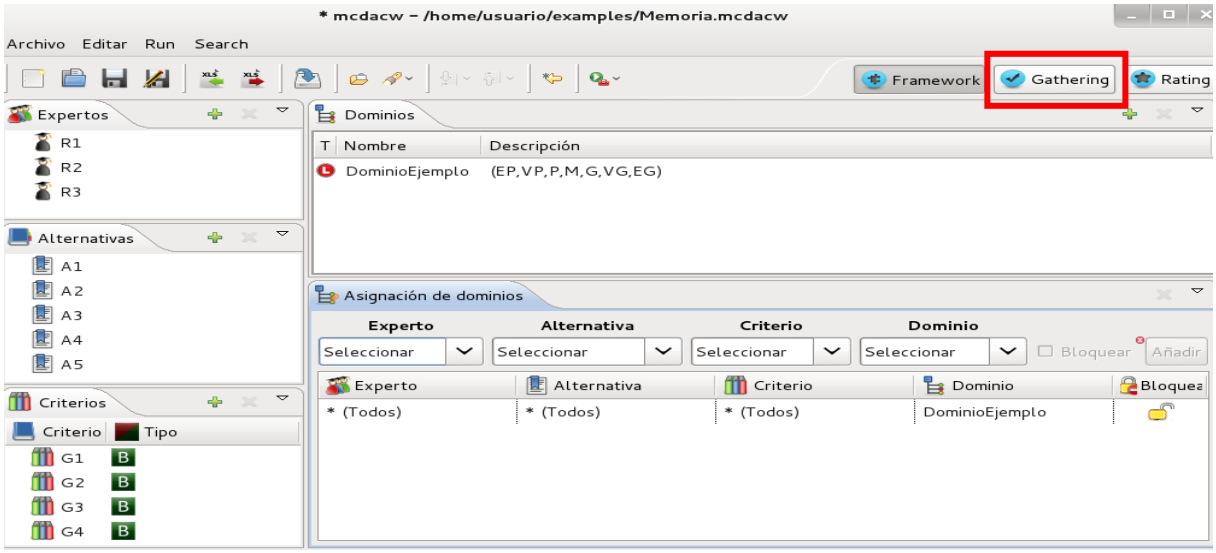


Ilustración A1.23 Botón para activar pantalla de Recogida de Información

Una vez el usuario seleccione el elemento anteriormente resaltado en la imagen, el sistema mostrará el conjunto de elementos que pueden ser valorados por el usuario:

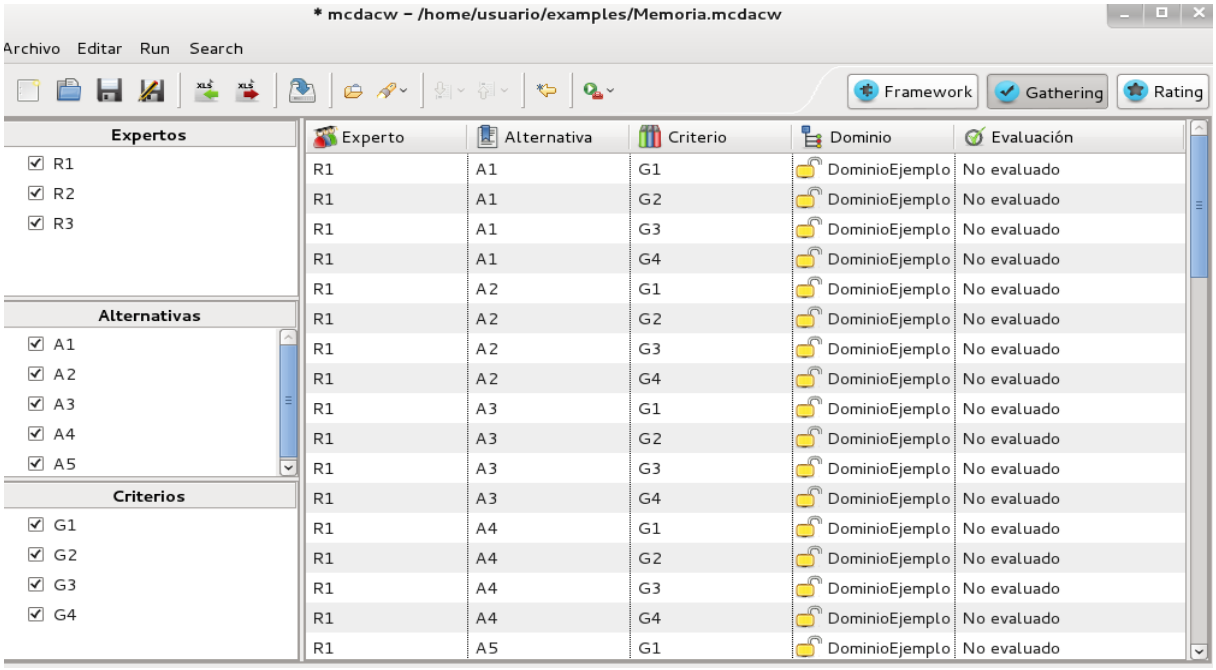


Ilustración A1.24 Pantalla Recogida de Información

Como se puede ver en la imagen anterior, en la zona central de la pantalla aparecen todas las alternativas que deben ser evaluadas en el problema, además, aparece una zona a la izquierda para filtrar las alternativas y reducir la información mostrada. Para evaluar uno de los elementos, el primer paso será seleccionarlo, una vez se seleccione se habilitará una zona para realizar la evaluación:

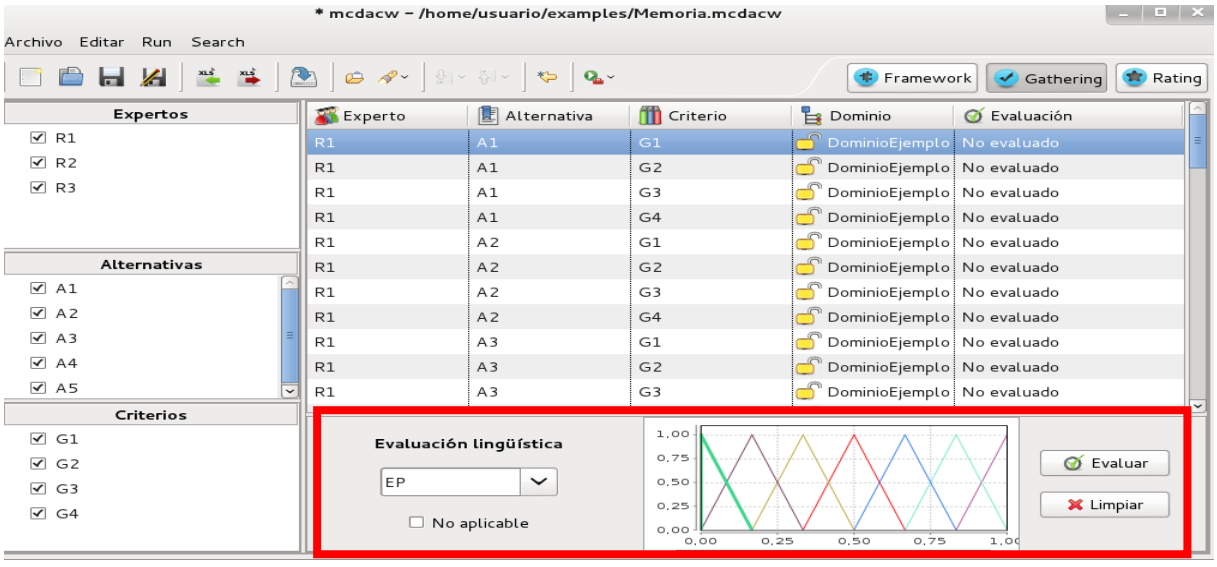


Ilustración A1.25 Zona de Evaluación de Alternativas

Esta zona variará según el dominio de expresión utilizado, pero el proceso para realizar la evaluación es sencillo. En este caso, basta con seleccionar en la pestaña de la izquierda el valor de la

evaluación y pulsar el botón "Evaluar". Una vez pulsado el botón, se mostrará en la zona central la alternativa evaluada con la nueva información suministrada:

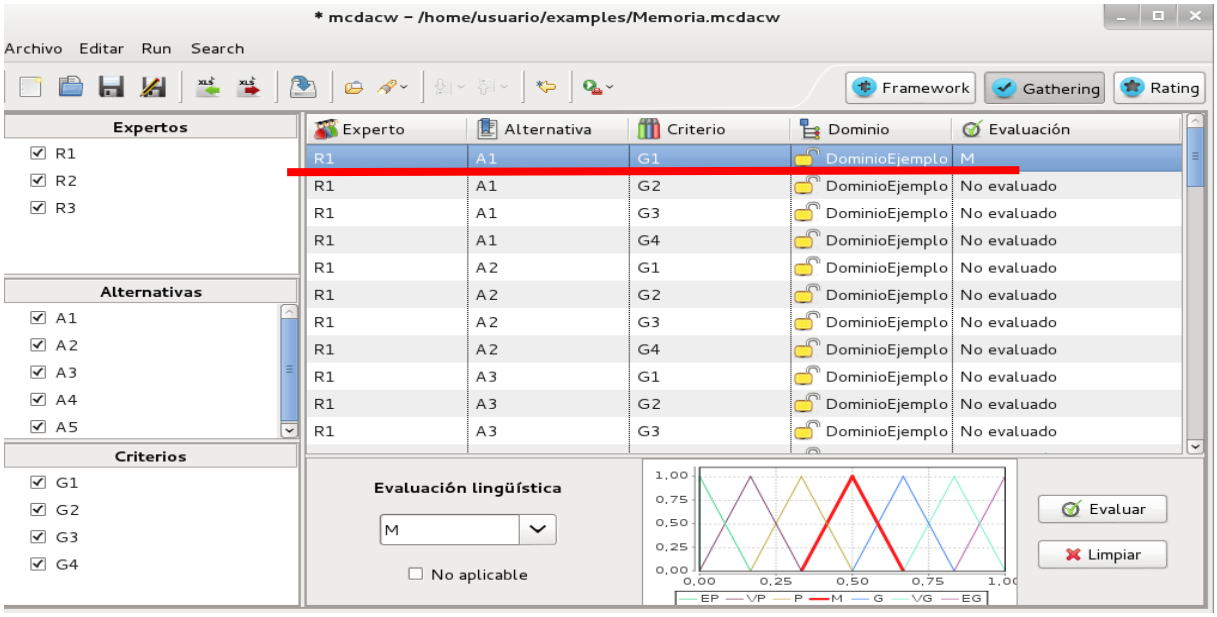


Ilustración A1.26 Evaluación de Alternativas

De esta forma se realiza la evaluación de una alternativa. Para poder realizar el proceso de resolución del problema de toma de decisión es necesario evaluar todas las alternativas o márcalas como "No aplicable" para que no sea tenida en cuenta:

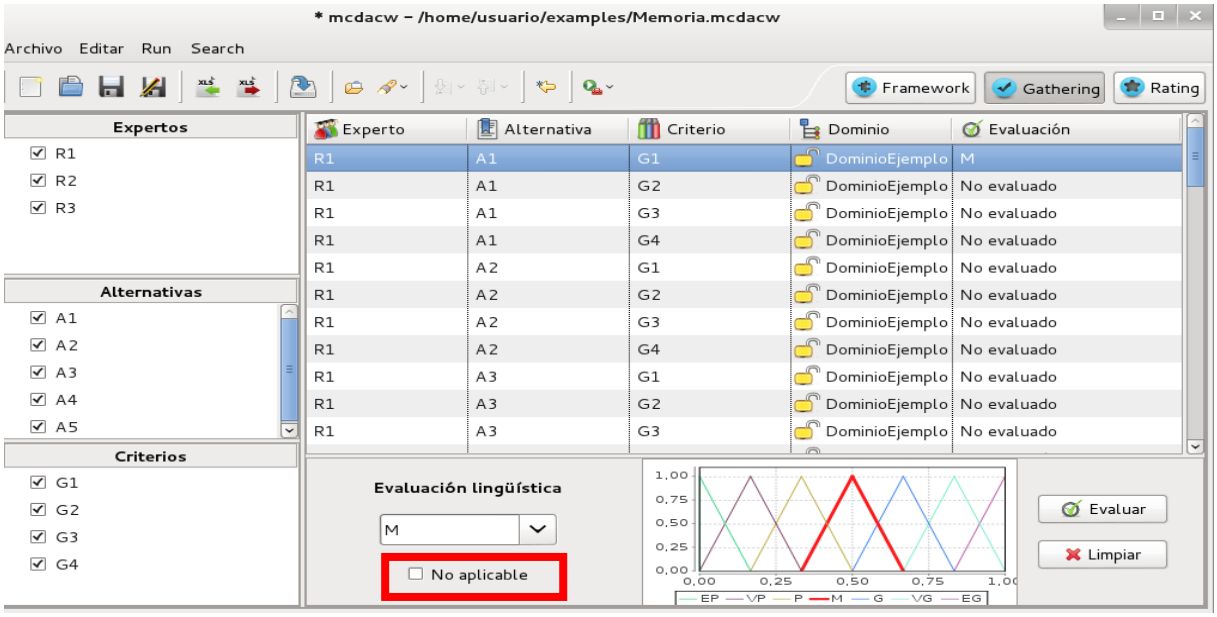


Ilustración A1.27 Alternativa No Aplicable

Para la realización del ejemplo se ha realizado la evaluación de todas las alternativas de acuerdo a la información mostrada en las tablas iniciales correspondientes a los 3 expertos:

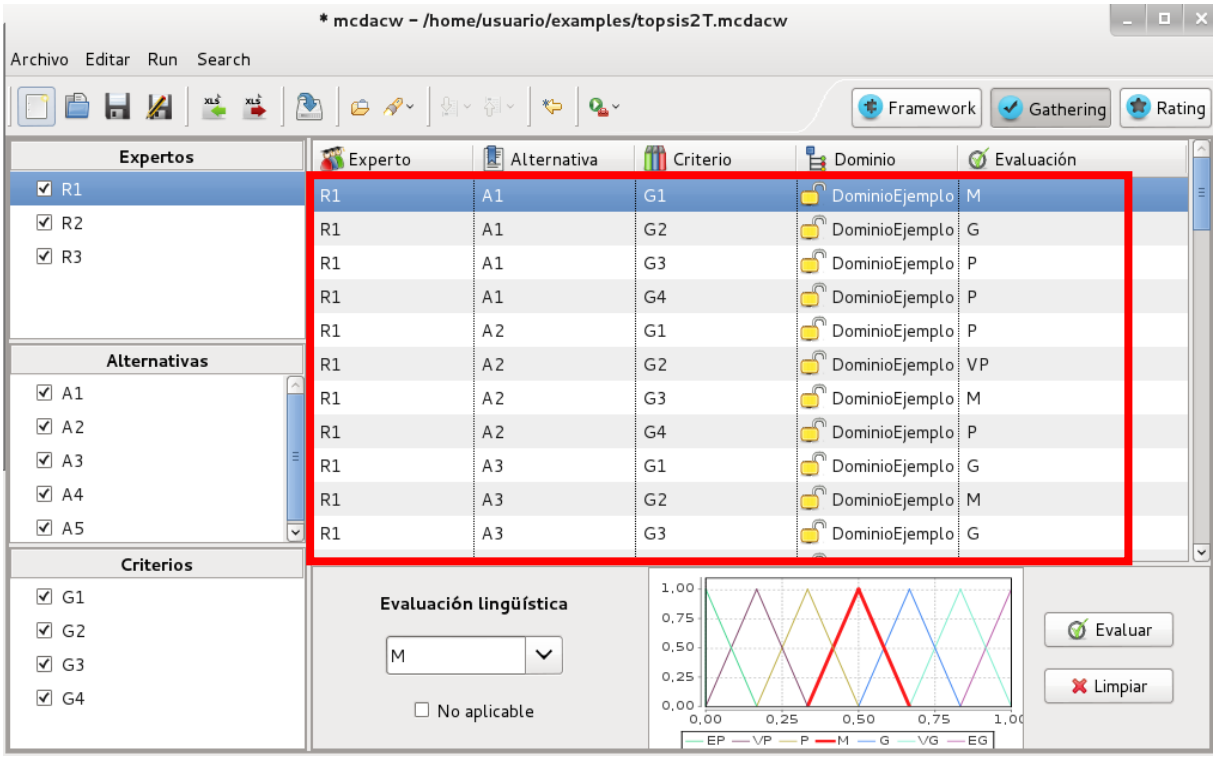


Ilustración A1.28 Valoración de Alternativas

Paso 3: Selección del método de decisión

Una vez asignadas las evaluaciones a todos los elementos de problema o indicar que no se deben aplicar para el cálculo de la solución, se pasa a la selección del método de decisión que se usará para resolver el problema. Para ello el usuario debe pulsar sobre la pestaña correspondiente a la selección del método de decisión:

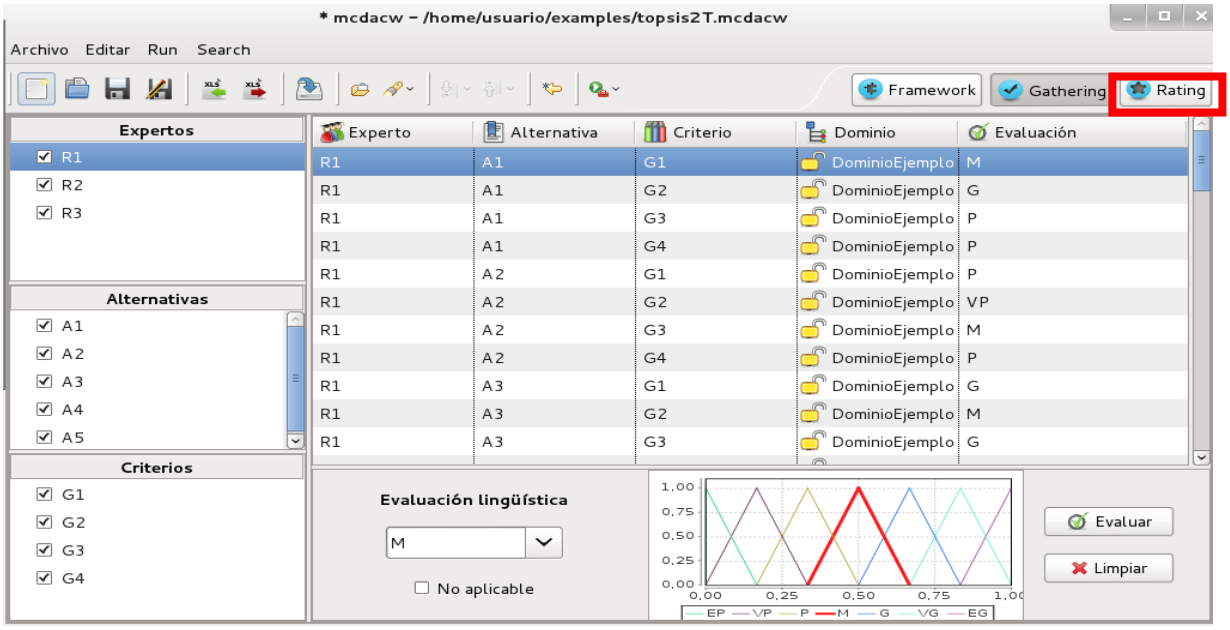


Ilustración A1.29 Pestaña para pasar a la Selección del Método

Una vez el usuario seleccione el elemento anteriormente resaltado en la imagen, el sistema mostrará una nueva pantalla donde el usuario podrá seleccionar el método de decisión que desee:

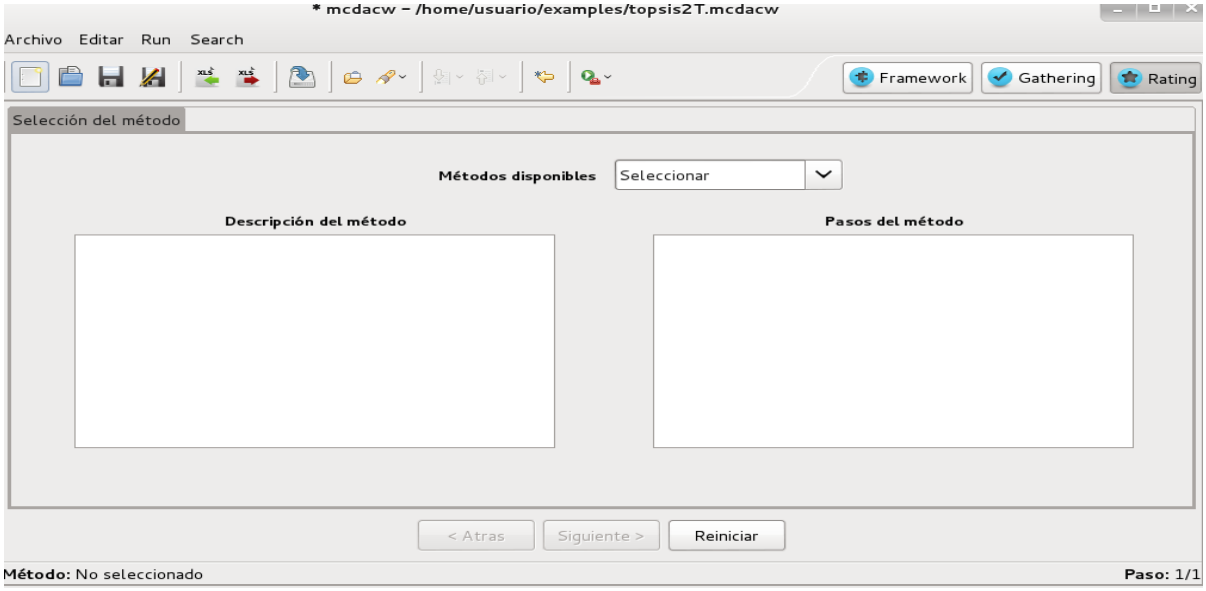


Ilustración A1.30 Pantalla Selección del Método

En la pantalla para seleccionar el método de decisión anteriormente mostrada, aparece un elemento para seleccionar los distintos métodos ofrecidos por el sistema:

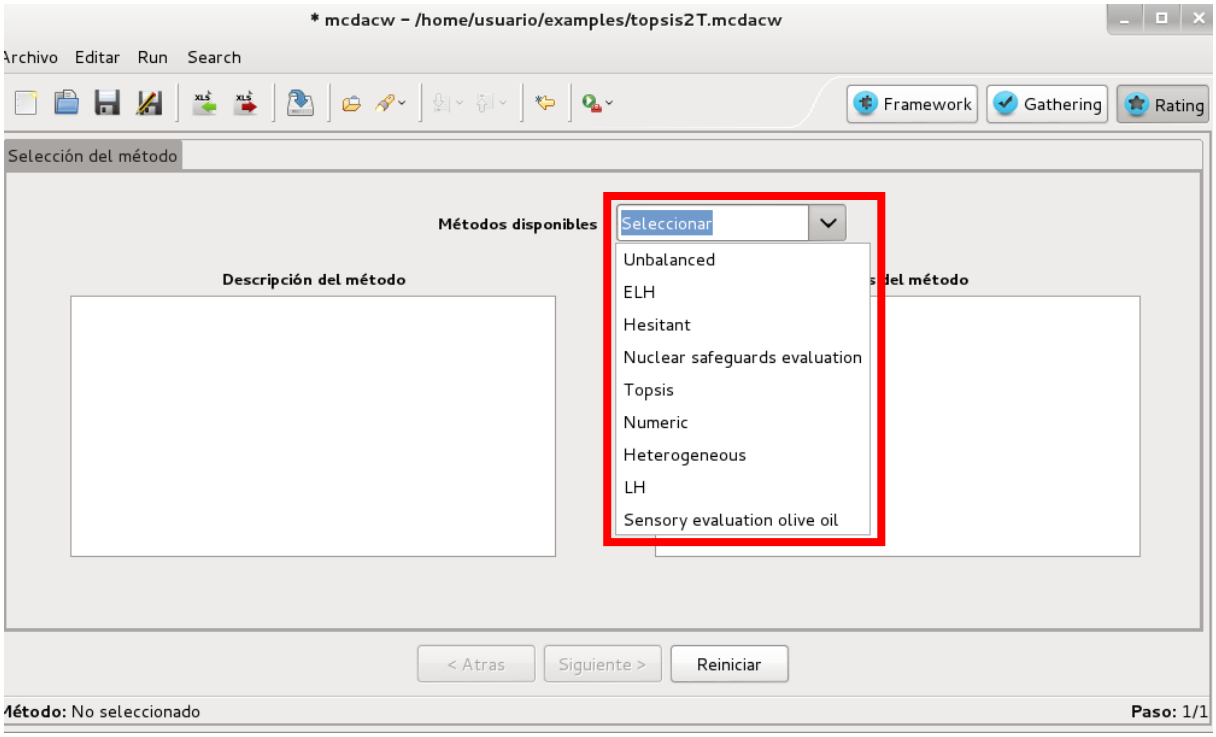


Ilustración A1.31 Zona de Selección del Método

Una vez seleccionado el método, en este caso Topsis, se mostrará la descripción del mismo, así como, los pasos que lo componen, además de habilitar el primer paso para seguir con el proceso:

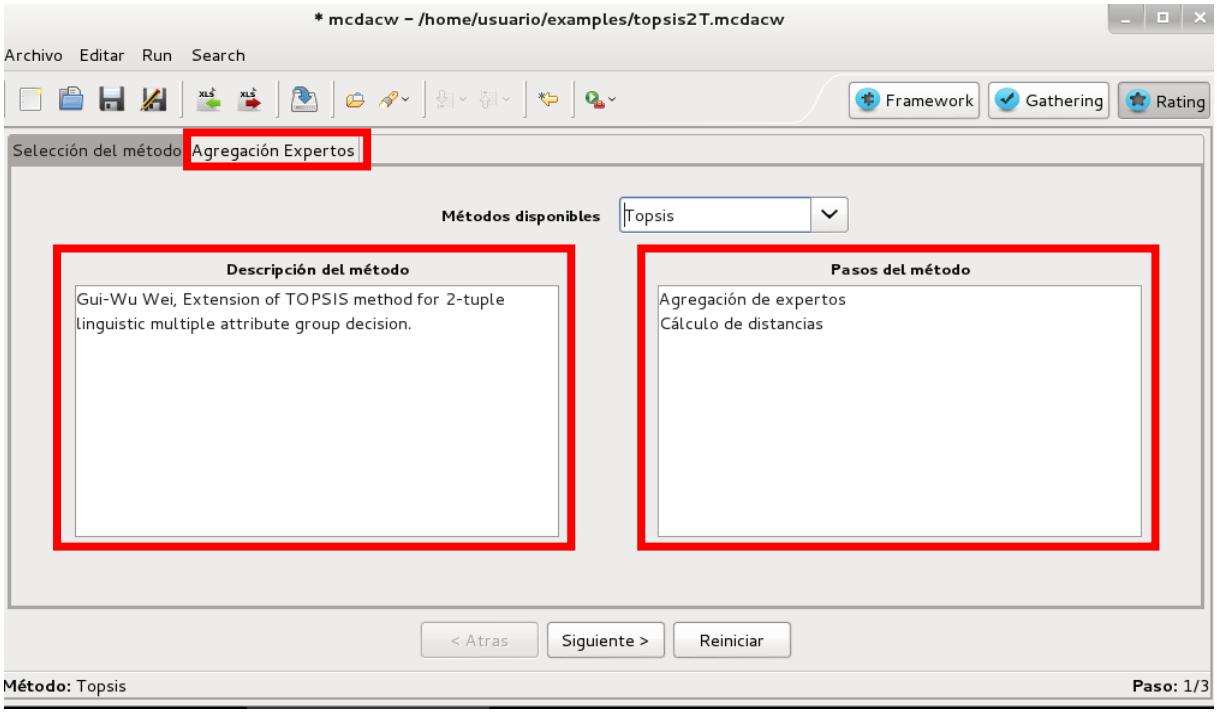


Ilustración A1.32 Método Topsis Seleccionado

Paso 4: Resolución mediante el método Topsis Paso 1: "Agregación de Expertos"

Una vez seleccionado el método de decisión que se desea aplicar para resolver el problema, en este caso Topsis, se habilitará el primer paso del mismo. Los distintos pasos de los métodos se irán habilitando a medida que se completen los pasos previos necesarios para realizar el proceso de resolución de forma correcta. Para pasar al primer paso del método se selecciona la pestaña correspondiente a dicho paso o se pulsa sobre el botón "Siguiente":

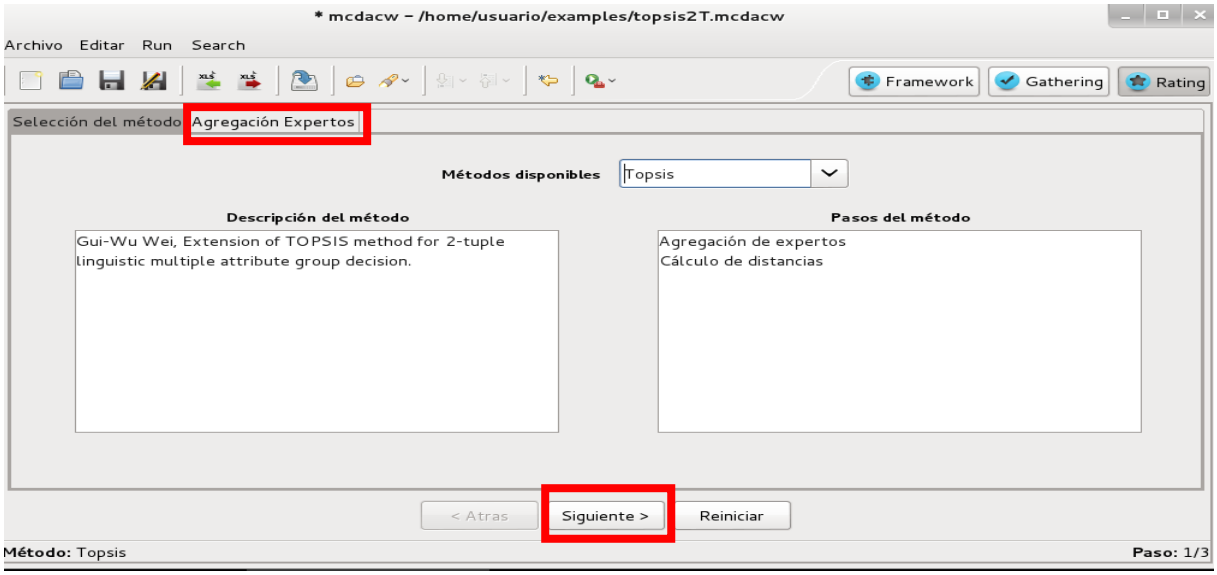


Ilustración A1.33 Elementos para pasar al siguiente paso del método

Una vez pulsamos uno de los elementos para pasar al primer paso del método, se mostrará la pantalla correspondiente a dicho paso:

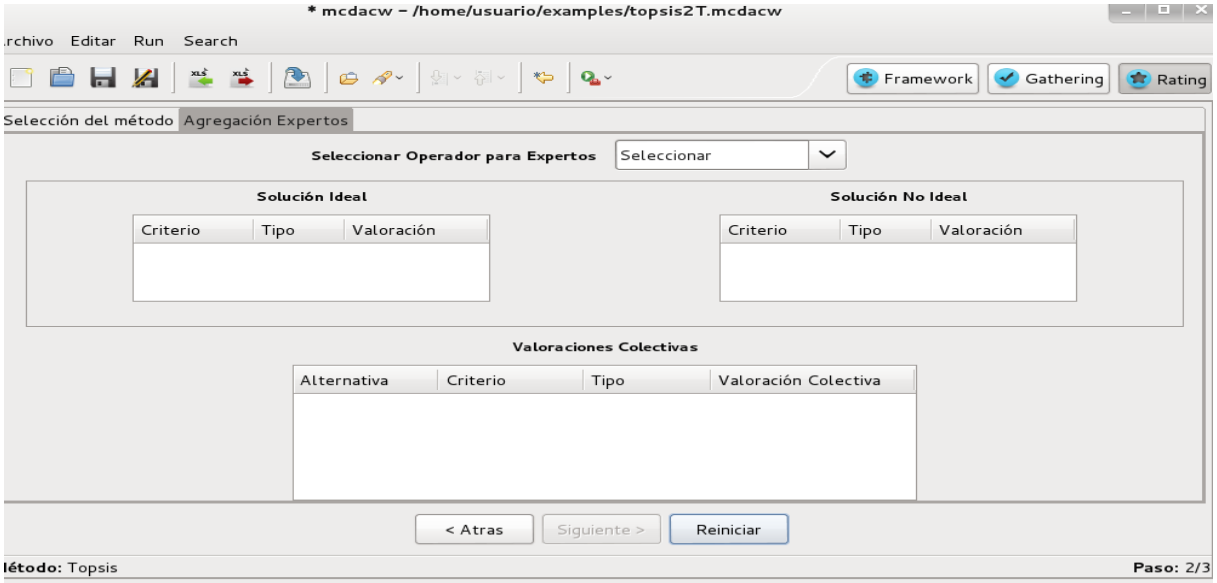


Ilustración A1.34 Primer paso Método Topsis: "Agregación de Expertos"

En este primer paso del método Topsis se realiza la agregación de la información proporcionada por los distintos expertos en una única matriz colectiva, además, se realiza el cálculo de la solución ideal y no ideal. Para ello, es necesario seleccionar el operador de agregación que se debe aplicar en el proceso:

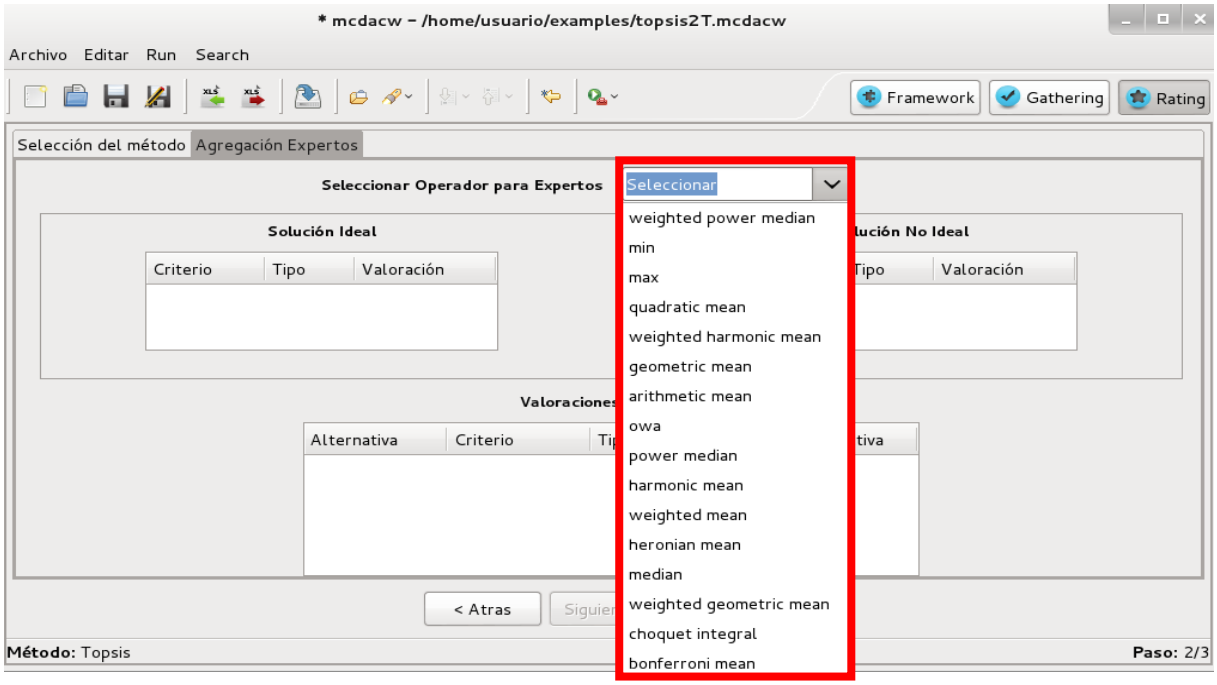


Ilustración A1.35 Selección Operador de Agregación

Una vez seleccionado el operador de agregación, en este caso la media aritmética para seguir con el mismo ejemplo, se muestran los resultados de dicho paso en las tablas y se activa el siguiente paso del método:

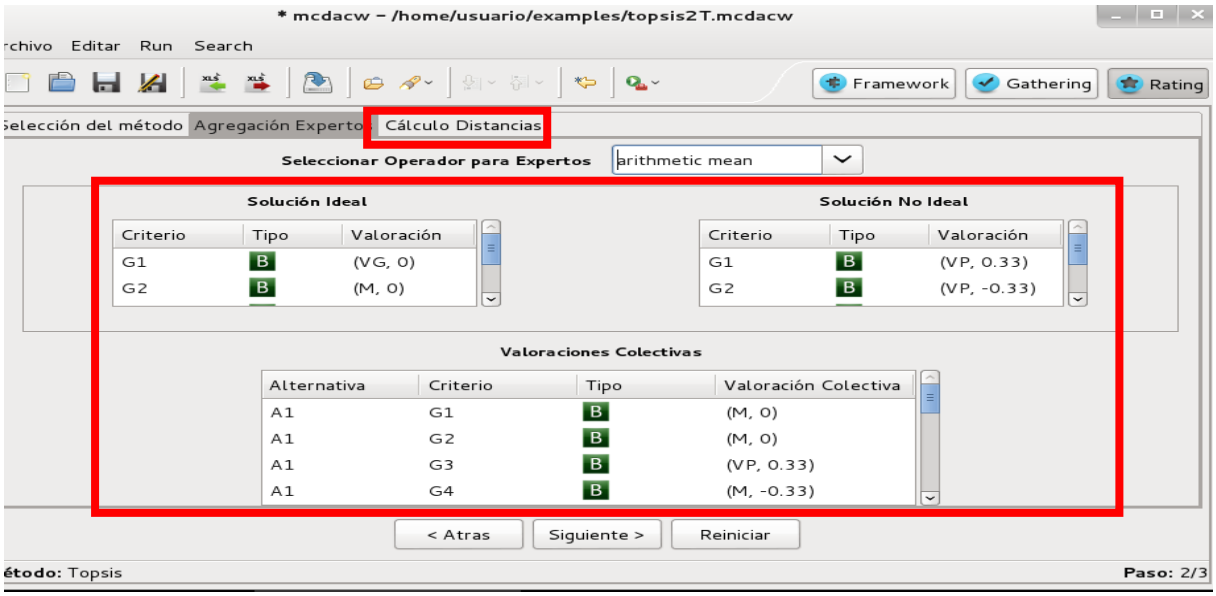


Ilustración A1.36 Resultado Agregación Expertos

Nota: Si el operador de agregación seleccionado requiere pesos o parámetros el sistema solicitará dichos datos al usuario que los deberá introducir para poder continuar con el proceso.

Como se puede ver, una vez seleccionado el operador de agregación se producen los cálculos asociados a este primer paso del método de decisión Topsis. Debido a que todos los datos suministrados por las tablas no se visualizan en la imagen anterior, se extraen en las siguientes tablas para poder ser comparados con los datos del ejemplo realizado en el capítulo 3.2.1:

- *Valoraciones colectivas:*

	G1	***G2***	***G3***	***G4***
A1	(M,0)	(M,0)	(VP,0.33)	(M,-0.33)
A2	(VP,0.33)	(P,-0.33)	(M,0)	(M,0.33)
A3	(G,0)	(M,-0.33)	(M,0.33)	(M,-0.33)
A4	(VG,0)	(M,-0.33)	(M,0)	(M,-0.33)
A5	(G,-0.33)	(VP,-0.33)	(P,0.33)	(M,-0.33)

Tabla A1.4 Resultado Aplicación Matriz Colectiva de Expertos

- *Solución ideal y no ideal:*

	C1	***C2***	***C3***	***C4***
Solución Ideal	(VG,0)	(M,0)	(M,0.33)	(M,0.33)
Solución No Ideal	(VP,0.33)	(VP,-0.33)	(VP,0.33)	(M,-0.33)

Tabla A1.5. Resultado Aplicación Solución Ideal y No Ideal

Nota: Hablar de "solución ideal" y "solución no ideal" es similar a hacerlo de "solución ideal positiva" y "solución ideal negativa" respectivamente.

Estos son los datos suministrados por la aplicación para el primer paso del método de decisión Topsis correspondiente a la agregación de expertos. Si se comparan estos datos con los obtenidos en los pasos 2 y 3 del ejemplo mostrado en el apartado 3.2.1, se podrá ver que son similares en ambos casos. Por tanto, se puede concluir que este primer paso del método de decisión Topsis llevado a cabo utilizando la aplicación "Flinstones" es correcto.

Paso 5: Resolución mediante el método Topsis Paso 2: "Cálculo de Distancias"

Una vez completado el primer paso del método de decisión Topsis correspondiente a la agregación de expertos, se habilitará el segundo paso del mismo. Al igual que para seleccionar el paso anterior, para pasar al siguiente paso del método se selecciona la pestaña correspondiente a dicho paso o se pulsa sobre el botón "Siguiente":

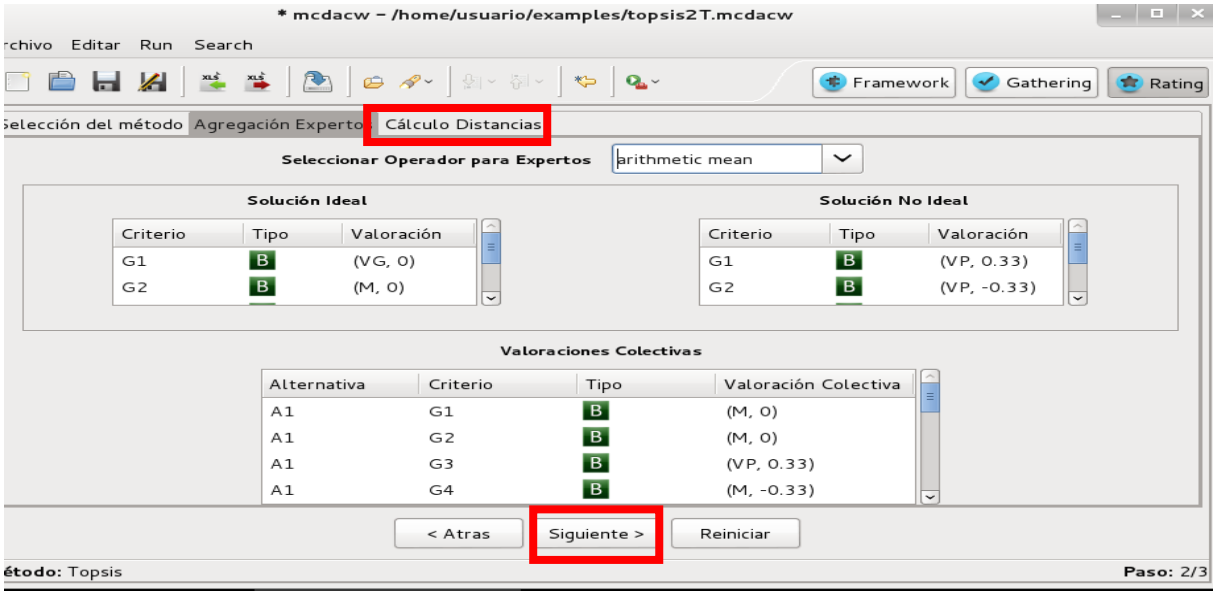


Ilustración A1.37 Elementos para pasar al siguiente paso del método

Una vez pulsamos uno de los elementos para pasar al siguiente paso del método, se mostrará la pantalla correspondiente a dicho paso:

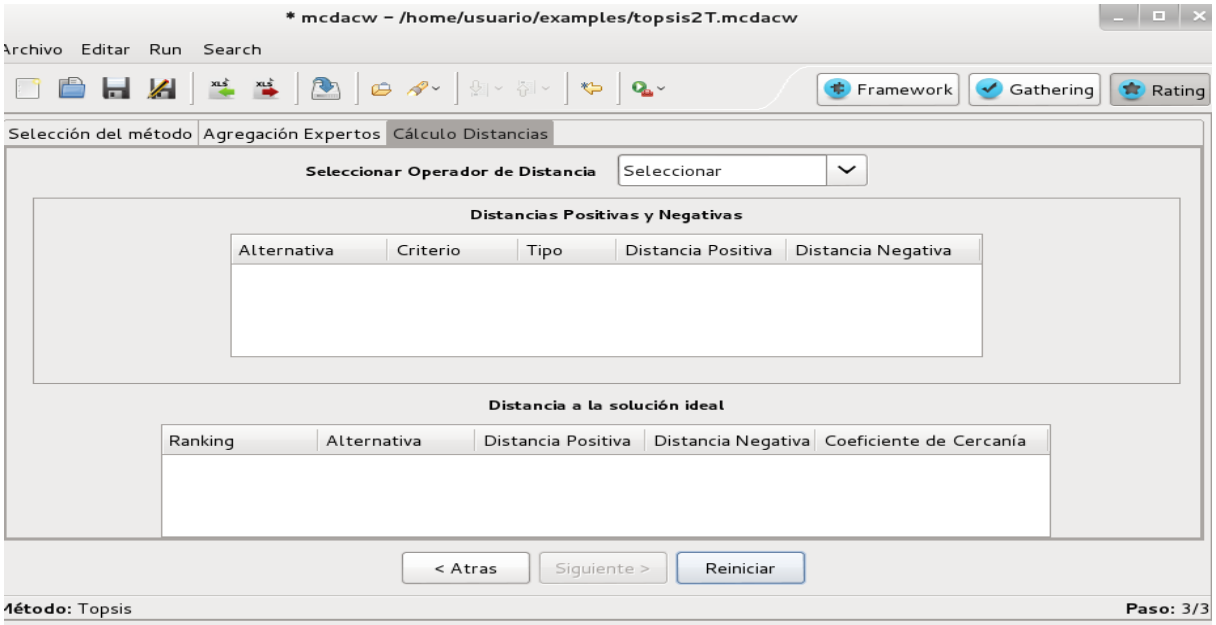


Ilustración A1.38 Segundo paso Método Topsis: "Cálculo de Distancias"

En este segundo paso del método Topsis se realiza el cálculo de las distancias de cada una de las alternativas con la solución ideal y no ideal, así como, el cálculo de los coeficientes de cercanía

para cada alternativa. Para ello, es necesario seleccionar el operador de distancias que se debe aplicar en el proceso:

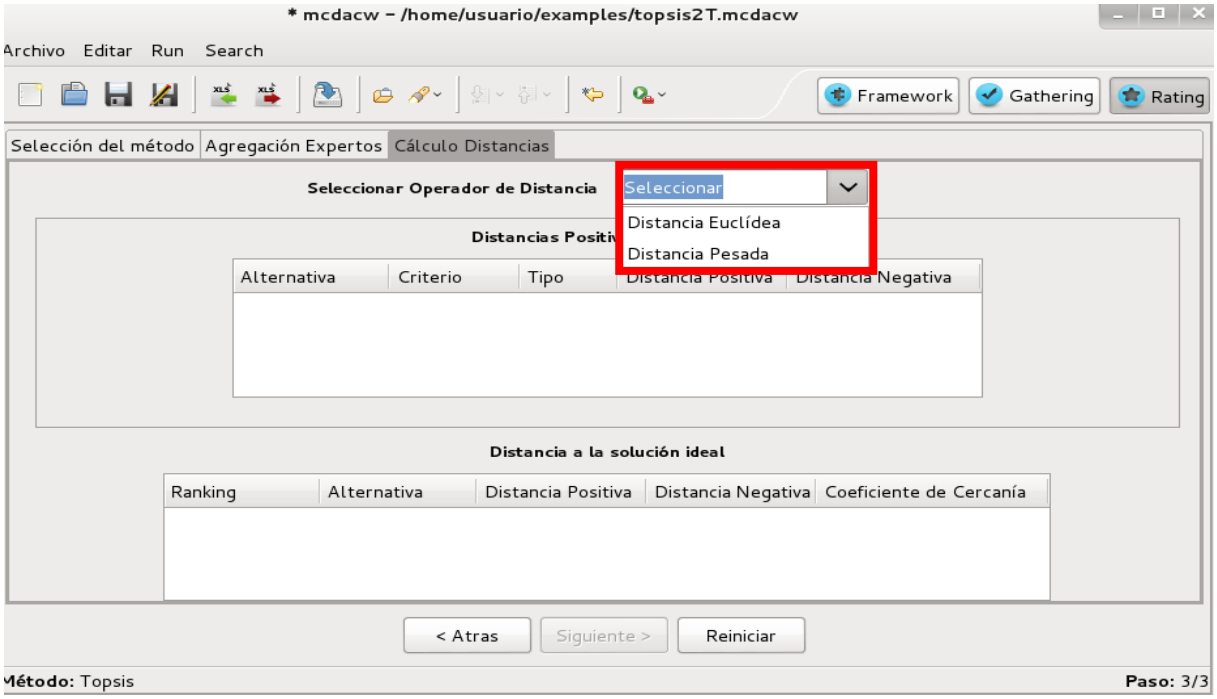


Ilustración A1.39 Selección Operador de Distancia

Una vez seleccionado el operador de distancias, en este caso el de distancias pesada para seguir con el mismo ejemplo, el sistema solicitará los pesos requeridos para seguir con los cálculos:

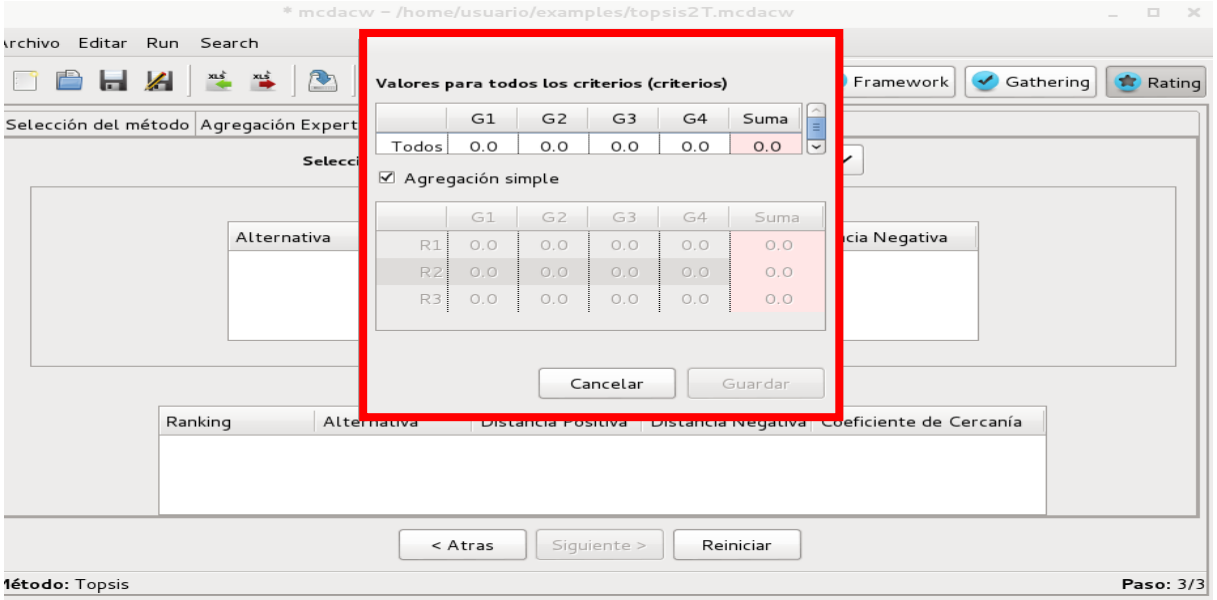


Ilustración A1.40 Solicitud de Pesos

Nota: Si el operador de distancias seleccionado es el de "Distancia Euclídea", el sistema no solicitará ningún dato puesto que no requiere ninguno para aplicarlo.

Cuando se introduzcan los pesos para cada uno de los criterios y se pulse el botón "Guardar", el sistema realizará los cálculos correspondientes a este paso del método, mostrando los datos obtenidos en las diferentes tablas:

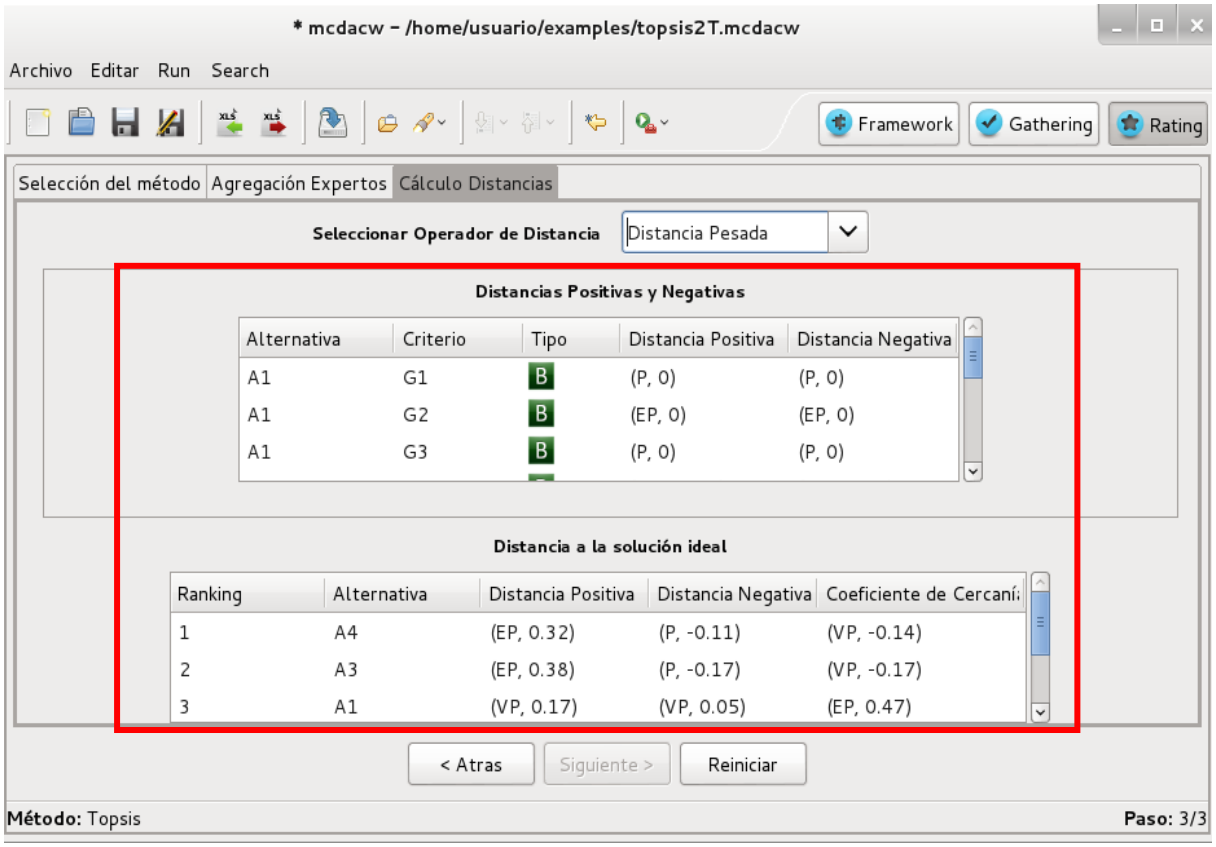


Ilustración A1.41 Resultados Obtenidos Paso 2: "Cálculo Distancias"

Como se puede ver, una vez seleccionado el operador de distancia e introducidos los pesos correspondientes se producen los cálculos asociados a este segundo paso del método de decisión Topsis. Debido a que todos los datos suministrados por las tablas no se visualizan en la imagen anterior, se extraen en las siguientes tablas para poder ser comparados con los datos del ejemplo realizado en el capítulo 3.2.1:

Nota: A continuación se muestra los datos correspondientes a la tabla inferior de la imagen anterior (Ilustración A1.41), ya que los datos de la tabla superior se corresponden con un paso previo para la obtención de los datos de la tabla inferior.

- *Distancia Solución Ideal, Distancia Solución No Ideal, Coeficiente de Cercanía:*

	A4	***A3***	***A1***	***A2***	***A5***
Distancia a Solución Ideal Positiva	(EP,0.32)	(EP,0.38)	(VP,0.17)	(VP,0.21)	(VP,0.44)
Distancia a Solución Ideal Negativa	(P,-0.11)	(P,-0.17)	(VP,0.05)	(VP,0.01)	(VP,-0.22)
Coeficiente de Cercanía	(VP,-0.14)	(VP,-0.17)	(EP,0.47)	(EP,0.45)	(EP,0.35)

Tabla A1.6 Distancias de cada alternativa a solución ideal positiva y negativa y Coeficiente de Cercanía

Nota: Las columnas están ordenadas según el ranking ofrecido por la aplicación. Por tanto, el sistema determina que la mejor alternativa es la A4.

Estos son los datos suministrados por la aplicación para el segundo y último paso del método de decisión Topsis correspondiente al cálculo de distancias. Si se comparan estos datos con los obtenidos en los pasos 4, 5 y 6 del ejemplo mostrado en el apartado 3.2.1, se podrá ver que los resultados obtenidos son similares en ambos casos. Por tanto, se puede concluir que el método de decisión Topsis llevado a cabo utilizando la aplicación "Flinstones" funciona de forma correcta.

Aclaraciones Finales:

Finalmente, indicar que a lo largo de todo el proceso se puede volver sobre los pasos anteriores del método de resolución para cambiar lo que se considere oportuno, en ese caso, se refrescaría la información ofrecida por la aplicación, para ello se pulsaría sobre la etiqueta correspondiente al paso sobre el que se desea volver o sobre el botón "Atrás", también se podría reiniciar todo el proceso de resolución pulsando sobre el botón "Reiniciar":

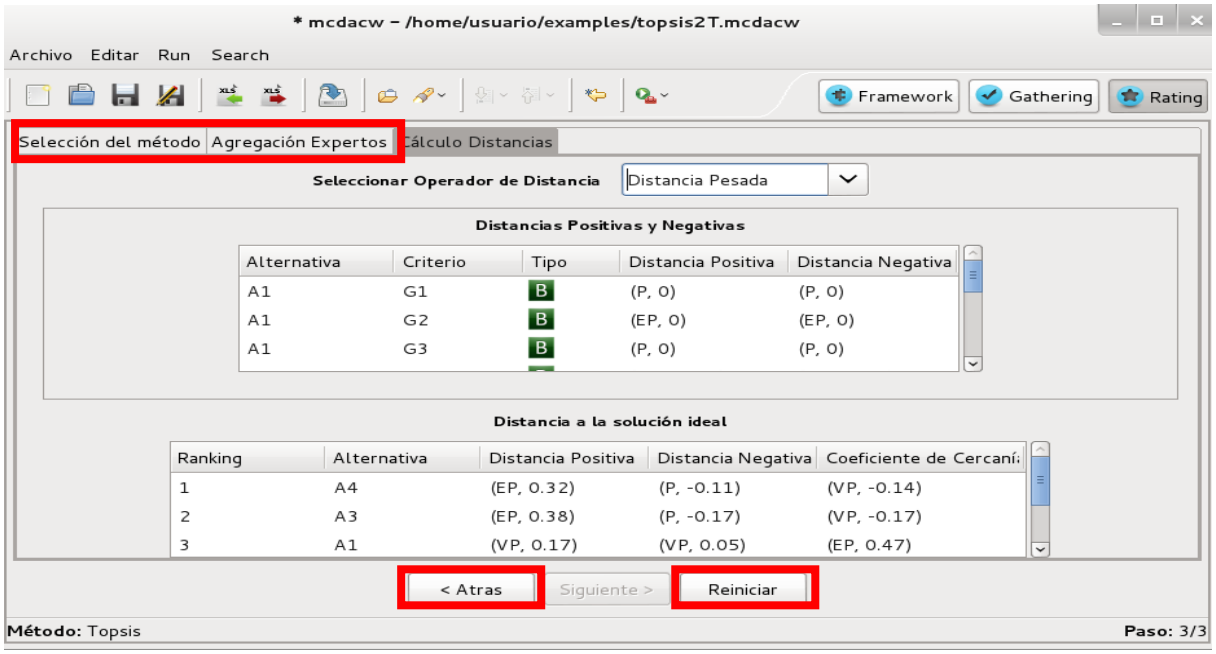


Ilustración A1.42 Volver a paso previo o Reiniciar

Esta sería la interacción que un usuario de la aplicación debería llevar a cabo si desease resolver un problema de decisión desde el inicio utilizando la aplicación "Flinstones" y empleando el método de decisión Topsis. Es evidente, que la interacción con otros métodos de decisión será distinta, pero muchas situaciones serán similares a lo expuesto en este ejemplo, puesto que los distintos métodos han sido diseñados para aumentar al máximo la familiaridad entre ellos, facilitando al usuario el paso de unos a otros.

Apéndice B

Manual de Instalación y Desarrollador

B. Manual de instalación y desarrollador.

Este apéndice está dedicado a mostrar las configuraciones que son necesarios llevar a cabo para poner en marcha la aplicación "Flinstones", así como, el proceso que debería realizar un desarrollador para aumentar la funcionalidad de dicha aplicación, añadiendo nuevos métodos, operadores de agregación u operadores de distancias.

Las únicas consideraciones previas que han de tenerse en cuenta es que durante todo el proceso expuesto en este manual se ha supuesto que la unidad principal de disco duro es C: y que la unidad principal de disco flexible es D:. Además, se debe tener en cuenta que el Sistema Operativo del equipo donde se han realizado todas las instalaciones y configuraciones es Microsoft Windows 7. El material necesario para realizar todo el proceso se encuentra en el CD que acompaña a esta memoria, si existe algún error en alguno de los archivos póngase en contacto con el responsable de la aplicación para solucionar el problema.

B.1 Manual de instalación

Este primer apartado se centra en explicar las configuraciones necesarias que se deben llevar a cabo para poder ejecutar la aplicación "Flinstones" en nuestro sistema. Se debe comprobar que existan los siguientes elementos en el CD-ROM:

- jdk-7u25-windows-x64
- Carpeta "VersionesAplicacion" con versiones de la aplicación para diferentes sistemas operativos.

Paso 1: Instalar Java

En caso de que el equipo donde se desea ejecutar la aplicación no disponga de máquina virtual Java se debe realizar la instalación de la misma, ya que es fundamental para realizar la ejecución.

El fichero de instalación se encuentra en el CD-ROM, pero también se puede descargar de la página oficial de Oracle: http://www.oracle.com/technetwork/es/java/javase/downloads/index.html. Para realizar la instalación se debe ejecutar el fichero "jdk-7u25-windows-x64.exe" (o el fichero descargado en su caso) y se iniciará el proceso de instalación:

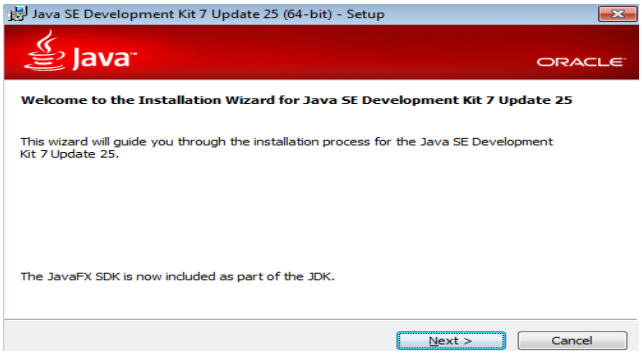


Ilustración A2.1 Inicio Instalación Java

El resto es un proceso sencillo, siguiendo los pasos marcados por el asistente se completa la instalación, realizando las configuraciones que cada usuario considere oportunas.

Paso 2: Ejecutar la aplicación

Una vez realizado el paso 1 o si ya se disponía de software necesario, se procede a ejecutar la aplicación. En la carpeta "VersionesAplicacion" se encuentran distintas carpetas con los ficheros necesarios para ejecutar la aplicación en diferentes Sistemas Operativos:

- *linux.gtk.x86*: Aplicación para Linux 32 bits.
- *linux.gtk.x86_64*: Aplicación para Linux 64 bits.
- *macosx.cocoa.x86_64*: Aplicación para Mac 64 bits.
- *win32.win32.x86*: Aplicación para Windows 32 bits.
- *win32.win32.x86_64*: Aplicación para Windows 64 bits.

Este ejemplo se ha realizado para Windows 64 bits, pero el proceso en todos ellos es similar. Para ejecutar la aplicación se debe entrar en la carpeta correspondiente a Windows 64 bits y, dentro de esta, en la carpeta "Eclipse". Una vez en la carpeta anterior, se debe ejecutar el archivo "mcdacw" y se debe iniciar la aplicación de forma correcta.

Nota: Para ver un ejemplo de uso de la aplicación ver el apéndice A.

B.2 Manual de desarrollador

Este apartado se centra en explicar las configuraciones necesarias que debe llevar a cabo un desarrollador que desee aumentar la funcionalidad de la aplicación "Flinstones", de forma similar a la llevada a cabo para realizar este proyecto.

Paso 1: Instalar Virtual Box

En este apartado se describe la instalación de la aplicación Virtual Box. Se trata de un software de virtualización que permite instalar Sistemas Operativos adicionales, conocidos como "sistemas invitados", dentro de otro Sistema Operativo denominado "anfitrión", proporcionado a cada uno su propio ambiente virtual.

El fichero de instalación se encuentra en el CD-ROM, pero también se puede descargar de la página de Virtual Box: https://www.virtualbox.org/wiki/Downloads. Para realizar la instalación se debe ejecutar el fichero "VirtualBox-4.2.16-86992-Win" (o el fichero descargado en su caso) y se iniciará el proceso de instalación:

Ilustración A2.2 Inicio Instalación Virtual Box

El resto es un proceso sencillo, siguiendo los pasos marcados por el asistente se completa la instalación, realizando las configuraciones que cada usuario considere oportunas.

Paso 2: Importar Imagen

A continuación se describe el proceso para incorporar una nueva imagen a Virtual Box, este proceso es similar tanto para la imagen inicial como para la imagen final. Antes de comenzar se debe realizar el paso 1 y se debe comprobar que en el CD-ROM se encuentren los siguientes archivos:

- "ImagenInicial", correspondiente a la imagen del sistema previa al inicio del proyecto.
- "ImagenFinal", correspondiente a la imagen del sistema una vez terminado el proyecto.

El primer paso consiste en ejecutar cualquiera de los dos archivos anteriores, una vez realizado se iniciará el proceso de importación de la imagen:

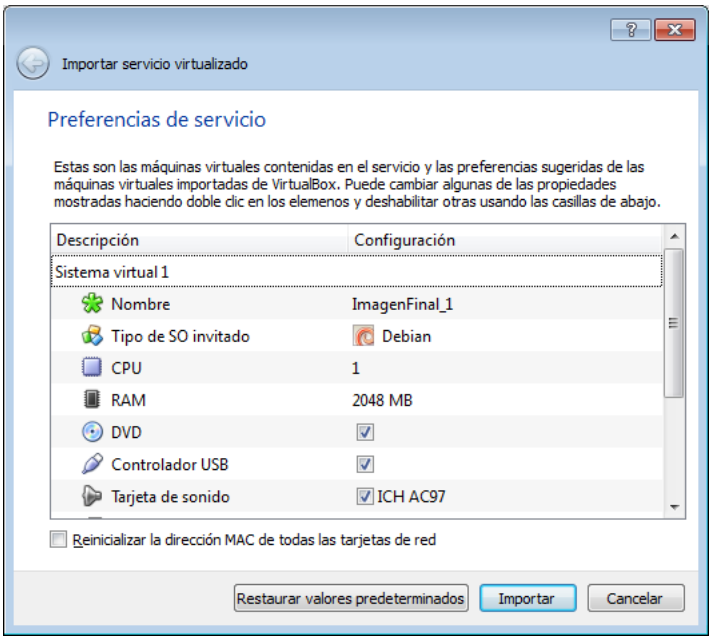


Ilustración A2.3 Inicio Importación Imagen

Una vez ejecutado el archivo de importación de la imagen se debe seguir el proceso marcado por el asistente. En dicho proceso se solicita una serie de opciones de configuración, estas opciones deberán ser modificadas por cada usuario según las características de su equipo y cubren diversos aspectos como son: memoria, pantalla, audio, red, etc. Si la imagen no funciona correctamente o se quiere modificar las opciones de configuración se puede realizar desde el menú de Virtual Box:

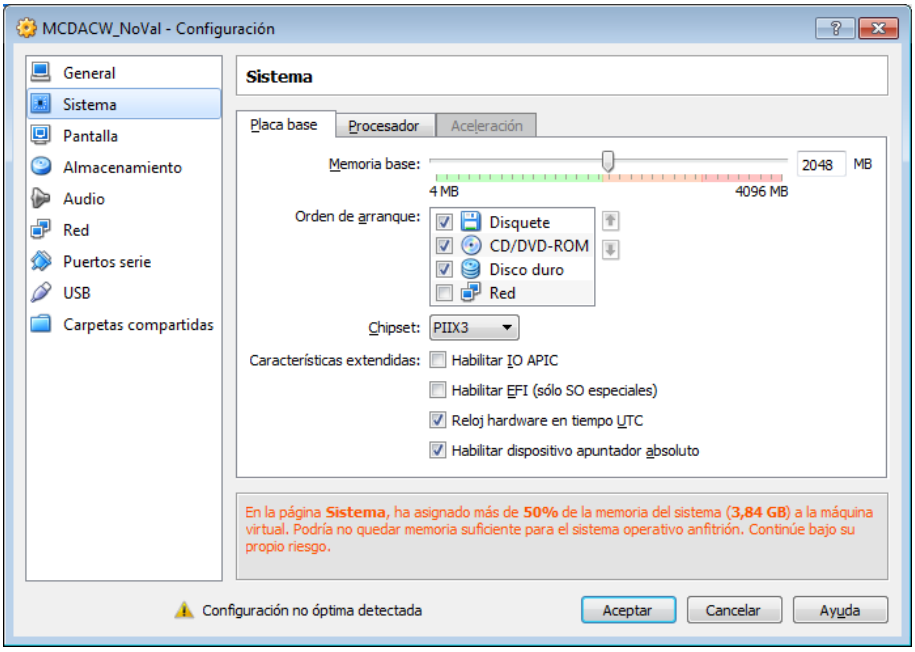


Ilustración A2.4 Pantalla Configuración Imagen Virtual Box

Cuando se haya instalado la imagen correctamente se podrá visualizar en el menú de Virtual Box y se podrá inicializar desde dicho menú:

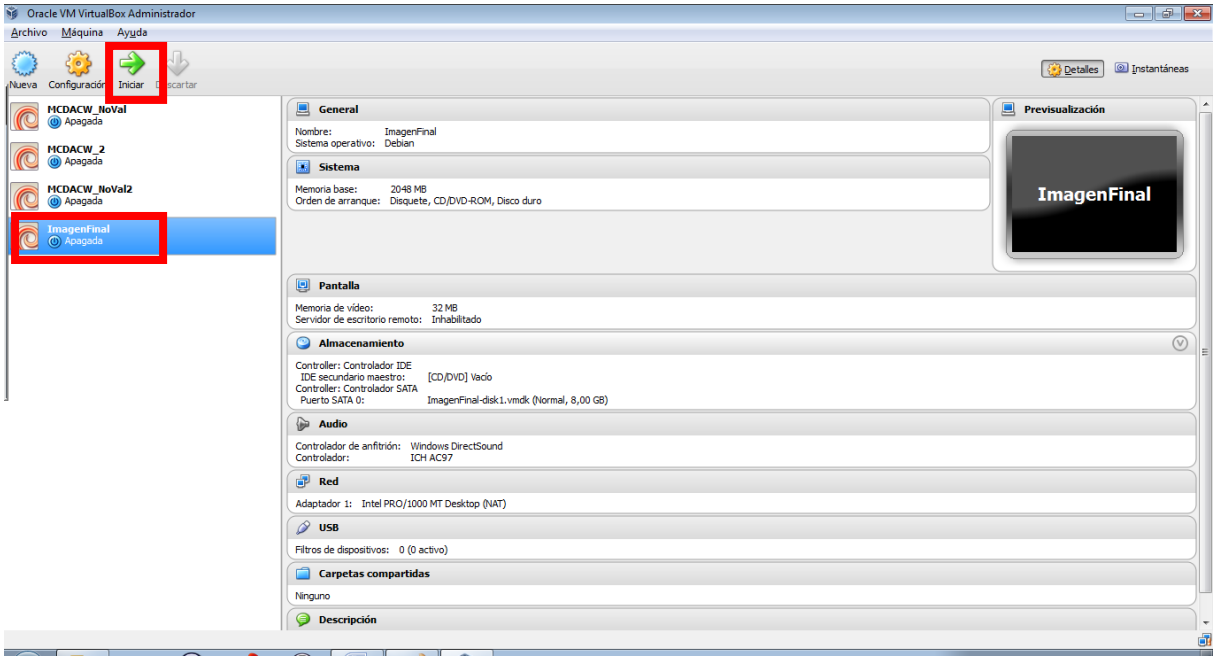


Ilustración A2.5 Menú Inicial Virtual Box

El siguiente paso una vez iniciada la virtualización consisten en lanzar el entorno de programación Eclipse dentro de la nueva imagen, para ello, la forma más fácil consiste en pulsar "Alt + f2" y, a continuación, escribir en el cuadro de diálogo "Eclipse":

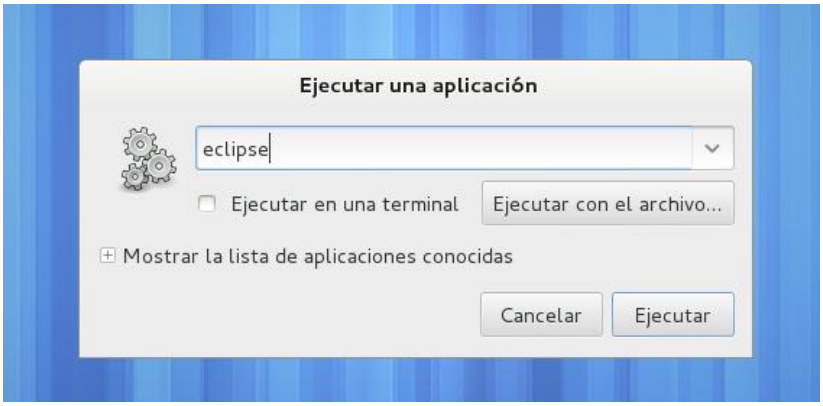


Ilustración A2.6 Lanzar Eclipse

Dentro del entorno de programación, se deben visualizar los distintos paquetes correspondientes con el proyecto realizado. En este paso pueden ocurrir algunos problemas y que los distintos paquetes no aparezcan, para solucionarlo se deben importar todos los paquetes existentes en la carpeta "workspace" y "workspace3" (ambas situadas en la carpeta "Carpeta Personal") a un nuevo proyecto de eclipse, de esta forma se tendrán todos los paquetes que están involucrados en el proyecto:

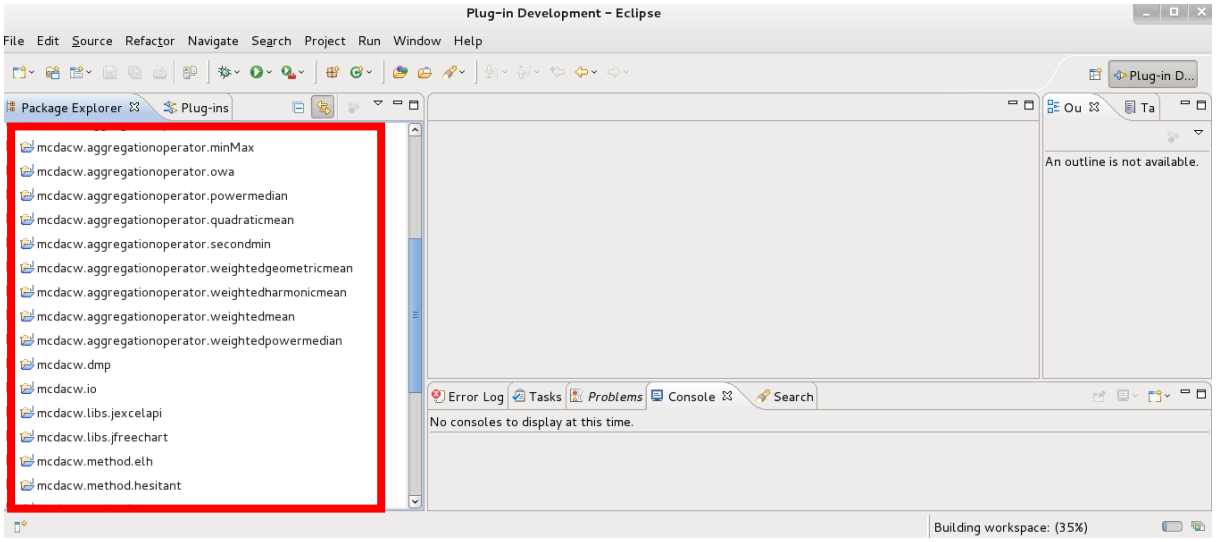


Ilustración A2.7 Paquetes Proyecto Eclipse

Nota: Se importarán los archivos de las dos carpetas si se instala la imagen final, si se instala la imagen inicial únicamente se importarán los archivos de la carpeta "workspace".

Paso 3: Aumentar la funcionalidad

En este apartado se describe el proceso que se debe seguir para aumentar la funcionalidad del sistema añadiendo nuevos operadores de agregación, métodos de decisión u operadores de distancia. Se supone que se realiza sobre una de las imágenes incluidas en el CD-ROM de la memoria y que se han realizado los pasos previos para importarla y el usuario se encuentra sobre el entorno de programación "Eclipse".

- **Nuevos operadores de agregación**

Para crear introducir un nuevo operador de agregación al sistema se deben llevar a cabo una serie de pasos que se enumeran a continuación:

1- Se debe crear un nuevo paquete dentro de la carpeta correspondiente a los operadores de agregación de forma similar al resto de operadores de agregación incluidos. Este paquete debe incluir fundamentalmente:

   a. Un paquete del mismo nombre del operador que incluya: la clase activador que se encargará de incorporar dicho paquete a la aplicación y otra clase del mismo nombre del operador que extenderá del tipo de operador apropiado y se encargará de realizar llamar a las funciones de evaluación adecuadas según el tipo de dato.
   b. Un paquete adicional denominado "Valuation" que contendrá tantas clases como tipos de datos permita el operador. Estas clases tendrán el nombre del tipo de dato y ofrecerán la funcionalidad apropiada en cada caso.
   c. Un fichero de meta-información en el que se indicará los tipos de datos que maneja el operador, el nombre que le queremos dar y la versión del mismo.

   La forma más sencilla de realizar este paso es copiar uno de los paquetes de los operadores que ya se encuentran incluidos en el sistema, cambiar el nombre del mismo y partir de dicho código para realizarlas modificaciones que se consideren oportunas.

   En la siguiente imagen se puede ver el contenido que tiene un paquete correspondiente a un operador de agregación ya creado:

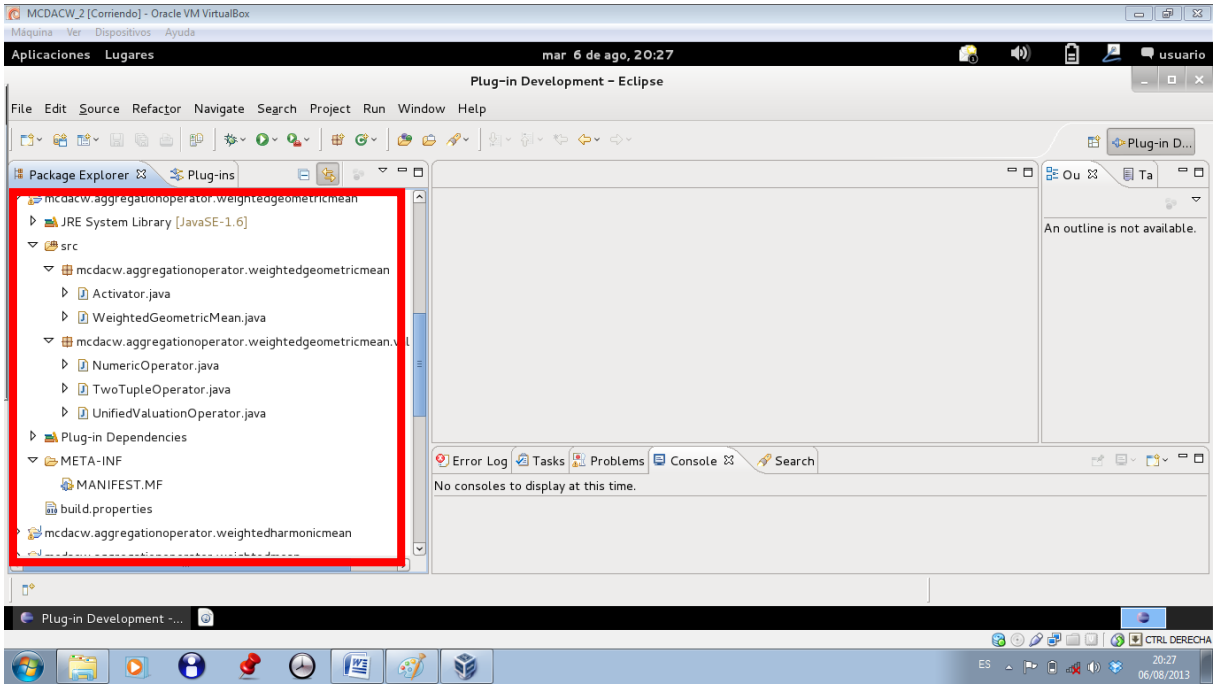


Ilustración A2.8 Contenido Paquete Operador de Agregación

2- Una vez creado el paquete anterior con todos sus elementos, se pasa a la implementación de la funcionalidad, dicha implementación será distinta para cada tipo de dato habilitado por el operador de agregación.

La implementación de cada uno de los tipos se realizara dentro del paquete "Valuation" y en la clase correspondiente al tipo de dato que se desee implementar:

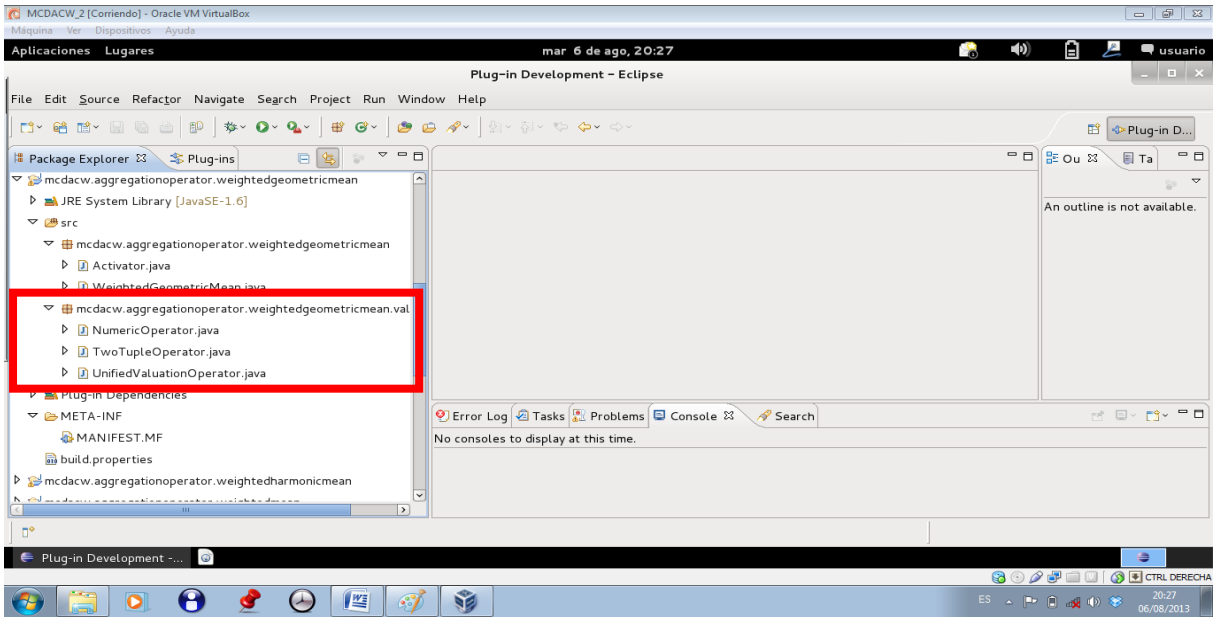


Ilustración A2.9 Clases Según Tipo Operador de Agregación

3- Una vez implementado el operador de agregación se exportará el mismo en forma de plugin para la aplicación y se incluirá en la carpeta correspondiente a los distintos plugins de los operadores de agregación. Esto se hará a partir del fichero "manifest.mf":

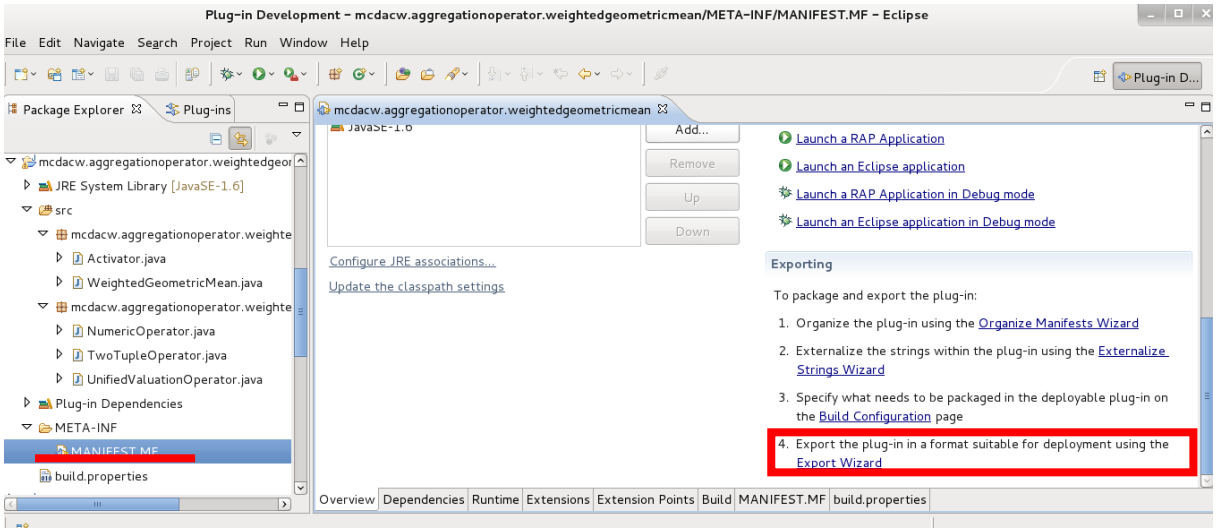


Ilustración A2.10 Exportación Operador de Agregación

La exportación del operador de agregación se realizará dentro de una carpeta "plugins" que será introducida en la carpeta "operators". El usuario deberá extraer el fichero contenido en la carpeta "plugins" correspondiente al nuevo operador e introducirlo directamente en la carpeta "operators", además, deberá eliminar cualquier fichero correspondiente al mismo operador si ya existía alguno.

Nota: El proceso anteriormente descrito es común tanto para operadores que requieran pesos como operadores que no los requieran, pero en el caso de operadores pesados el proceso debe incluir unos pasos adicionales.

En estos casos, se debe modificar ligeramente los pasos de los distintos métodos de resolución para que la aplicación solicite los distintos pesos cuando sea necesario, así mismo, ocurre cuando los operadores de agregación requieren algún tipo de parámetro adicional para su funcionamiento. Estas modificaciones se deben realizar siempre que la aplicación requiera al usuario la selección de un operador de agregación, ya que en ese momento será cuando se deben solicitar los pesos o parámetros necesarios.

- **Nuevos métodos de decisión**

Para incluir un nuevo método de decisión en el sistema se deben seguir una serie de pasos que se enumeran a continuación:

1- Se debe crear un nuevo paquete dentro de la carpeta correspondiente a los métodos de resolución. Este paquete debe incluir fundamentalmente:

   a. Un paquete del mismo nombre del método que incluya: la clase activador que se encargará de incorporar dicho paquete a la aplicación y otra clase del mismo nombre del método que extenderá de la clase abstracta vista anteriormente en el diagrama de clases correspondiente a los métodos de resolución.

   b. Un fichero de meta-información en el que se indicará los tipos de datos que maneja el método, el nombre que le queremos dar y la versión del mismo.

La forma más sencilla de realizar este paso es copiar uno paquetes de los métodos que ya se encuentran incluidos en el sistema, cambiar el nombre del mismo y partir de dicho código para realizarlas modificaciones que se requieran.

En la siguiente imagen se puede ver el contenido que tiene un paquete correspondiente a un método de decisión ya creado:

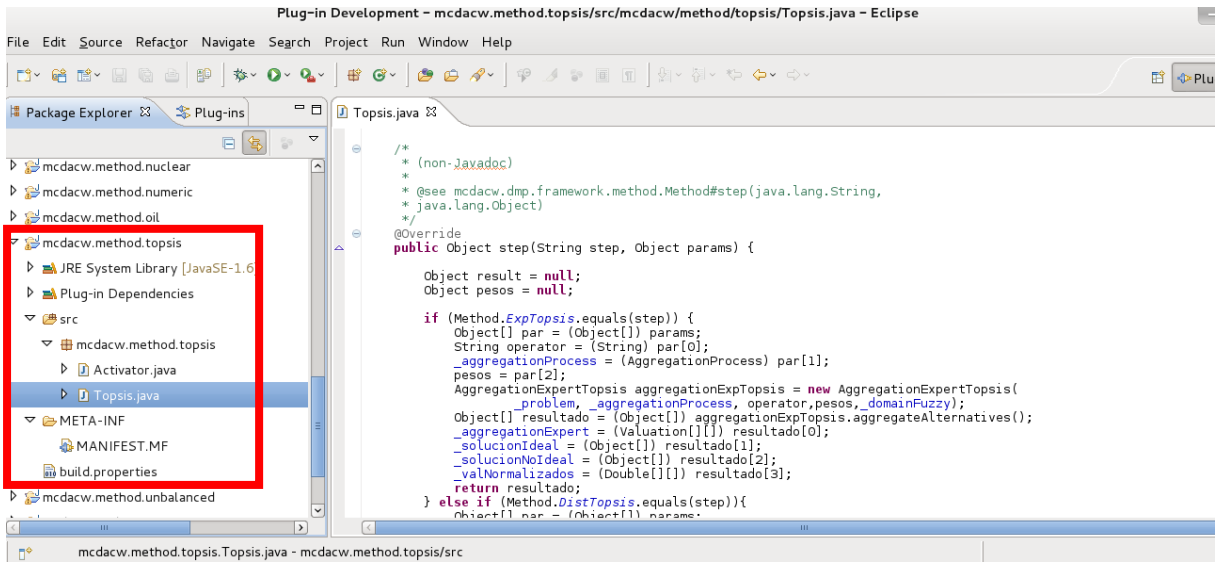


Ilustración A2.11 Paquetes Método de Decisión

2- Una vez creado el paquete anterior con todos sus elementos, se pasa a la implementación de la funcionalidad del método, dicha implementación será distinta para cada tipo de dato habilitado por el método e incorporará los cálculos necesarios para obtener los resultados adecuados.

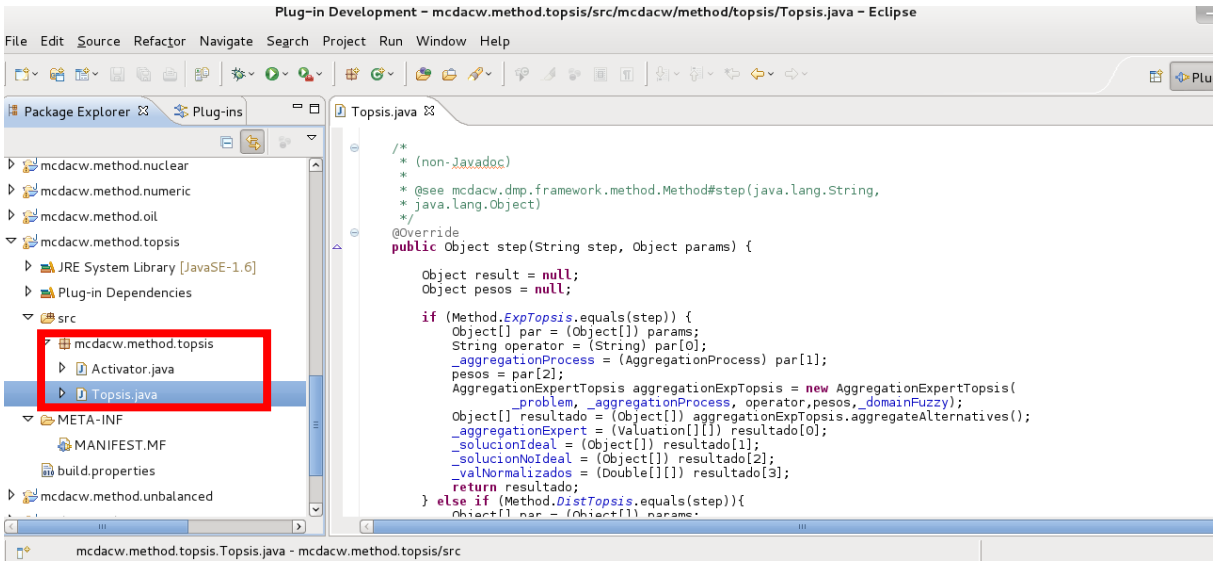


Ilustración A2.12 Clase correspondiente al Método

3- Una vez implementado el método de resolución se exportará el mismo en forma de plugin para la aplicación y se incluirá en la carpeta correspondiente a los distintos plugins del resto de métodos. Este proceso se hará de forma similar que con los operadores de agregación, pero cambiando en este caso se realizará sobre la carpeta "methods".

Como se ha dicho anteriormente, esto es solo una parte de la incorporación de un nuevo método a la aplicación. Para completarlo se deben crear los pasos correspondientes que definirán el comportamiento de la interfaz cuando se ejecute el método. Para relacionarlo con el método simplemente se le indicará en el constructor del mismo los pasos que debe seguir y se suministrarán las funciones necesarias para realizar los cálculos de dicho método.

En caso de que la implementación del método requiera la creación de nuevos pasos en lugar de la reutilización de algunos ya creados, se deben realizar lo siguiente:

1- Se crea un nuevo paquete dentro de la carpeta correspondiente a los distintos pasos implementados con el nombre del nuevo paso que contendrá una clase con el mismo nombre.

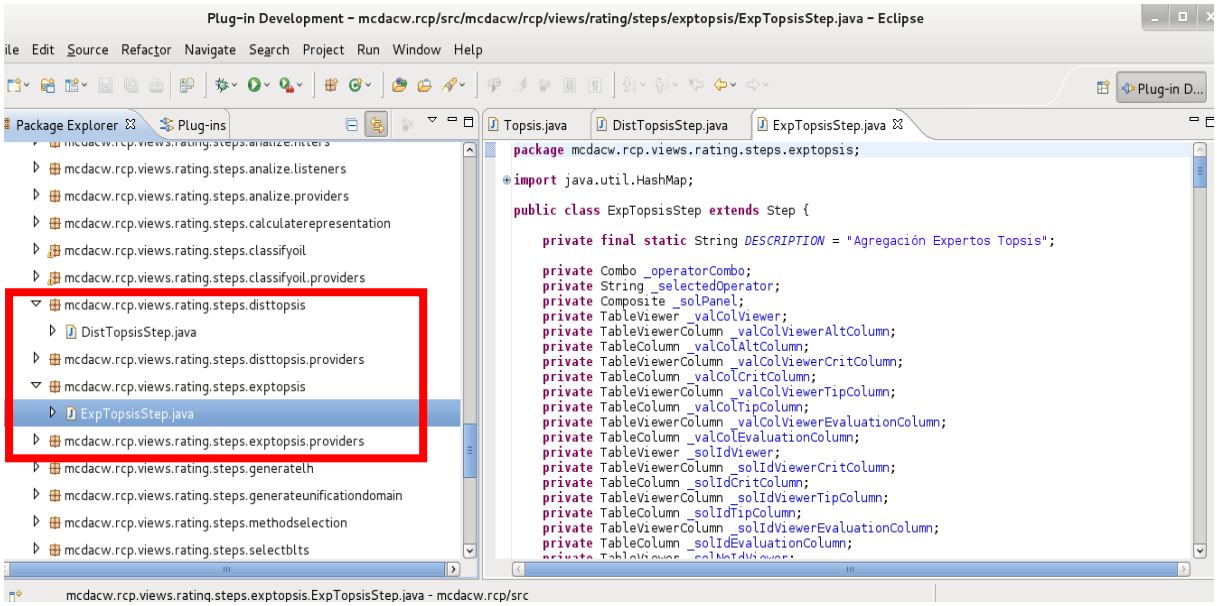


Ilustración A2.13 Ejemplos pasos Creados para Topsis

2- Se incorporará el contenido de dicha clase que contendrá la definición de la interfaz correspondiente a dicho paso, así como, las invocaciones al método siempre que sea necesario.

3- Además es necesario realizar algunas modificaciones en otras clases para el correcto funcionamiento de los métodos.

   a. La clase abstracta correspondiente a los métodos debe añadir el identificador de los nuevos pasos creados.
   b. En el paso ya creado correspondiente a la selección del método se deben añadir los nuevos pasos cuando sea necesario.

Nota: La interfaz invocará al método correspondiente para realizar los cálculos correspondientes cuando se den las condiciones precisas. Dichos cálculos se pueden incluir en la clase correspondiente al método o en clases separadas que serán llamadas por la clase del método.

- **Nuevos operadores de distancia**

Los operadores de distancia se utilizan únicamente en el método de decisión Topsis, por tanto, para explicar cómo incluir nuevos operadores de distancia se centrará la atención en dicho método y en los pasos definidos para llevarlo a cabo. En concreto, la intervención de los operadores de distancia se produce en el segundo paso del método Topsis. Para añadir un nuevo operador de distancia se debe realizar lo siguiente:

1- Se modificará la clase correspondiente al segundo paso del método (DistTopsisStep.java) en la que se define la estructura del a interfaz de dicho paso:

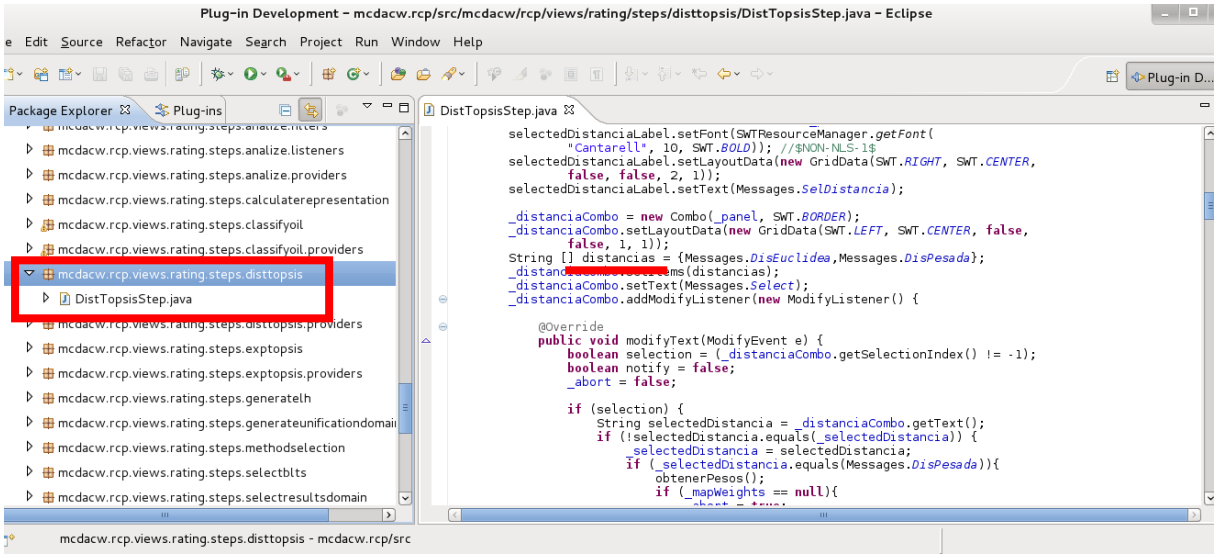


Ilustración A2.14 Paso Operadores de Distancia

En este caso, se introducen los elementos para que el usuario pueda seleccionar el nuevo paso de forma similar a como selecciona los ya incluidos.

2- Se modifica el cálculo del método Topsis para el segundo paso. Al implementar este cálculo se opto por extraerlo en una clase aparte para simplificar el código, por tanto, se debe modificar dicha clase:

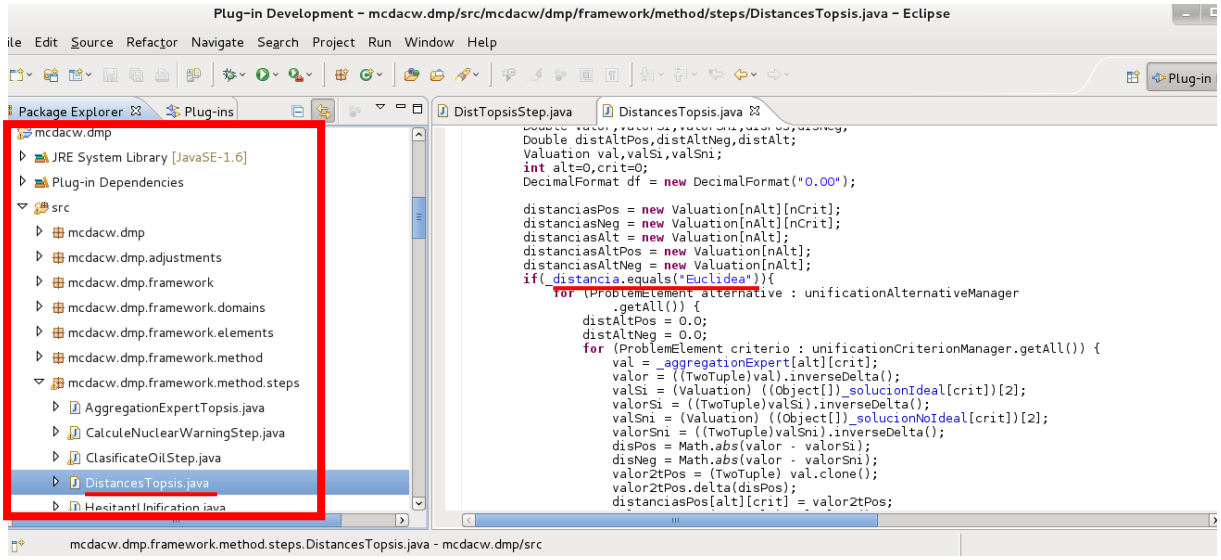


Ilustración A2.15 Cálculo según Operador Distancia

Esta clase se debe modificar para que cuando el operador de distancia seleccionado sea el nuevo operador añadido se realicen los cálculos correspondientes de forma muy parecida a como se realiza para los otros operadores de distancia.

Estos son los pasos a seguir para incorporar nueva funcionalidad al sistema: operadores de agregación, métodos de expresión y operadores de distancia. En los procesos descritos no se ha entrado en detalle en cuestiones de más profundas de implementación, ya que complicaría la lectura y comprensión del manual.