UNIVERSIDAD DE GRANADA

Sistema Multi-Agente con Soporte a Procesos de Consenso

Trabajo Tutelado Fin de Master

Alumno: Iván Palomares Carrascosa

Tutor: Dr. Luis Martínez López

Curso 2010/2011

Granada, Septiembre 2011

Índice general

Índice de figuras

Capítulo 1

Introducción

La presente documentación constituye la memoria del trabajo de investigación titulado *Sistema Multi-Agente con Soporte a Procesos de Consenso* y que sirve como Trabajo Tutelado de Fin de Master para la obtención del título de *Master Universitario en Soft Computing y Sistemas Inteligentes*, impartido en la Universidad de Granada. La memoria comienza con un primer capítulo en el que exponemos la motivación del trabajo de investigación elegido, haciendo un breve repaso general al estado del arte en Toma de Decisiones, Procesos de Consenso y Sistemas Multi-Agente, para seguidamente fijar los objetivos de dicho trabajo y definir la estructura que se seguirá en el resto de la memoria.

1.1. Motivación

La Toma de Decisiones es una de las actividades cotidianas de los seres humanos, ya que constantemente nos enfrentamos a situaciones en las que existen varias alternativas, y en ocasiones debemos decidir cuál de ellas es mejor o cuál se adapta mejor a nuestras necesidades.

El proceso de Toma de Decisión es un proceso complejo, ya que es necesario realizar previamente un análisis detallado de las ventajas e inconvenientes asociados a cada alternativa. Según la *Teoría Clásica de la Decisión* [58, 70], existen diferentes tipos de problemas que se clasificarán atendiendo a distintos puntos de vista: criterios, expertos y contexto de definición. Según el número de expertos que participen en un problema de Decisión, podemos clasificar dichos problemas en: *Toma de Decisión Individual* y *Toma de Decisión en Grupo*.

Un problema de Toma de Decisión en Grupo (TDG) puede definirse como un problema de decisión en el que dos o más decisores o expertos intentan lograr una solución común para dicho problema, teniendo en cuenta las opiniones o preferencias de todos ellos. Tradicionalmente, los problemas de TDG se han resuelto llevando a cabo un proceso de selección en el que la mejor alternativa es elegida, sin tener en cuenta un acuerdo previo entre los expertos [37]. Así, este proceso de selección de alternativas puede dar lugar en ocasiones a soluciones que no sean aceptadas como buenas por parte de algunos expertos [73], debido a que puedan considerar que sus preferencias individuales no se han tenido en cuenta para obtener dicha solución.

Para evitar tales situaciones, es aconsejable llevar a cabo un proceso de *consenso*, en el que los expertos discuten y modifican sus preferencias de cara a alcanzar un nivel de acuerdo suficiente antes de tomar una decisión. El estudio de los procesos de consenso se ha convertido en un campo de investigación de gran importancia dentro de la Toma de Decisiones. Por ello, se han propuesto en la literatura varios enfoques de *modelos de consenso* [38, 39, 40, 42, 43, 47, 48, 52, 63]. El *consenso* se ha definido clásicamente como el acuerdo total y unánime de todos los expertos participantes en el problema, algo que resulta inusual en problemas de TDG reales. Por esta razón, las medidas de consenso absoluto seguidas en el enfoque clásico fueron evolucionando hacia medidas más flexibles [73], donde el objetivo es alcanzar un estado de acuerdo mutuo en el grupo de expertos, habiendo expresado cada uno de ellos sus preferencias y habiendo sido tenidas en cuenta, de modo que todos los expertos acepten la solución del problema. Esta interpretación flexible de consenso, la cual ha sido considerada en el modelo de consenso presentado en este trabajo, ha sido adoptada en multitud de enfoques, destacando la noción denominada *soft consensus* [46, 49, 50] basada en el concepto de mayoría difusa.

El consenso es un proceso dinámico e iterativo, consistente en una serie de rondas, donde los expertos expresan y discuten sus preferencias acerca de las alternativas del problema. Este proceso es tradicionalmente supervisado y coordinado por un *moderador* que ayuda a los expertos participantes en el problema a acercar sus opiniones [4, 53]. En la literatura se han propuesto algunos modelos de consenso con el objetivo de automatizar el proceso de alcance de consenso [43, 63].

Los procesos de consenso presentan actualmente algunas debilidades y problemas adicionales, como son la dificultad para trabajar con grandes grupos de expertos y la necesidad de una supervisión constante del problema a lo largo del mismo. Para superar algunas de estas dificultades, una de las principales líneas de actuación y mejora que se han propuesto sobre los modelos de consenso es la de ampliar el grado de automatización de los mismos, de manera que la necesidad de supervisión humana del proceso sea mínima o incluso nula. Una opción para conseguir esta automatización es mediante el paradigma de agentes software y *Sistemas Multi-Agente* (SMA) [62], que supondrá el modelo que se propone seguir en este trabajo de investigación.

Un *Agente Software* es una entidad capaz de percibir su entorno, procesar tales percepciones y responder o actuar en su entorno de manera racional y correcta, tendiendo a maximizar un resultado esperado. Una de las características más atractivas de los agentes software es su carácter autónomo, es decir, la capacidad para realizar las tareas que le permitan alcanzar un objetivo sin supervisión humana. Un Sistema Multi-Agente es un sistema distribuido compuesto por agentes software, donde la acción combinada de cada uno de ellos persigue la consecución de un objetivo común, o bien cada uno de ellos persigue su propio objetivo individual.

1.2. Objetivos

Este trabajo se centra en el estudio de problemas de TDG bajo incertidumbre, y el propósito del mismo consiste en desarrollar un *Sistema de Apoyo al Consenso (SAC) semi-supervisado* que, además de automatizar los procesos de consenso, permita dotarlos de cierta autonomía durante su realización. Para alcanzar dicho propósito, proponemos construir un SAC basado en un sistema multi-agente cooperativo en el que actúen diferentes tipos de agentes que asumen los diferentes roles habituales en este tipo de problema, donde cada agente tendrá su propia autonomía para contribuir a realizar el proceso de consenso, permitiendo no obstante la supervisión del experto humano, necesaria en algunas ocasiones. Este SMA intentará guiar los procesos de consenso para alcanzar un grado de acuerdo pre-determinado a priori, antes de que el grupo pase a tomar la decisión. Gracias al carácter autónomo e inteligente de los agentes, estos serán capaces de actuar sin supervisión constante de los expertos humanos correspondientes, acelerando en ocasiones la negociación de los procesos de consenso.

Para conseguir este propósito global, nos planteamos alcanzar los siguientes objetivos:

- **Revisión de los modelos de Consenso para problemas de TDG bajo incertidumbre**. En la literatura podemos encontrar diferentes modelos teóricos de consenso capaces de llevar a cabo procesos de TDG bajo incertidumbre, esto es, con información vaga o imprecisa. En ellos se describen tanto las estructuras y dominios empleados para la expresión de preferencias de los expertos como los operadores utilizados para trabajar con diferentes tipos de información.
- **Revisión del Estado de Arte en el área de los SMA y su relación con Procesos de Consenso**. Dado que el sistema que pretendemos desarrollar se apoyará sobre la tecnología multi-agente, será necesario realizar un estudio previo de las arquitecturas y modelos de SMA más comunes en la literatura, de cara a abordar los aspectos relativos a modelos de consenso previamente revisados desde una perspectiva del diseño y

funcionamiento de los agentes software.

- **Definición de un Modelo de Consenso adecuado para su Implementación mediante un SMA**: Es importante adoptar un modelo de consenso que permita conseguir un cierto grado de automatización, ya que no todos los modelos existentes en la literatura están diseñados para aplicar una automatización directa sobre ellos. Al modelo utilizado, se le añadirá la capacidad de semi-supervisión del proceso por parte del experto humano sobre su agente correspondiente.
- **Diseño de una Arquitectura Multi-Agente para ayudar al alcance del Consenso**. Para diseñar un SMA es necesario llevar a cabo un análisis exhaustivo del contexto en el que se implantará, con el propósito de identificar los diferentes agentes que participarán, lo que supone además dotarlos de una personalidad que dependerá de las funciones que queramos que lleven a cabo de manera autónoma en el proceso de consenso.
- **Diseño de una Ontología para facilitar la Comunicación entre Agentes**. Dado que una de las principales características de los Sistemas Multi-Agente es la capacidad de interacción y comunicación con otros agentes, es importante definir una ontología donde se especifique claramente el vocabulario y semántica de los términos utilizados por los agentes en dicha comunicación.
- **Implementación del Sistema Multi-Agente que dé Soporte a los procesos de Consenso bajo una plataforma FIPA**. La fase de implementación se llevará a cabo siguiendo el estándar FIPA[1] de desarrollo de SMA, y mediante la plataforma de desarrollo de Sistemas Multi-Agente JADE[2].

[1] http://www.FIPA.org

[2] http://jade.tilab.com

1.3. Estructura

Para alcanzar los objetivos que perseguimos, esta memoria se estructura en los siguientes capítulos:

- Capítulo 2: Presenta una revisión de los principales temas de investigación seguidos en este trabajo, incluyendo un repaso general a los problemas de TDG y una revisión de modelos de Consenso, haciendo especial hincapié en aquellos que utilizan medidas flexibles de consenso. A continuación, se revisará la tecnología de SMA, atendiendo a los tipos de agentes existentes, arquitecturas para construir SMA, estándares de desarrollo y comunicación entre agentes.
- Capítulo 3: En este capítulo, se presenta nuestra propuesta de SMA con soporte a procesos de consenso, donde describimos el modelo de consenso seguido, haciendo especial hincapié en la semi-supervisión de los agentes durante el proceso para conseguir el mayor grado de autonomía posible, los principales detalles relativos a la arquitectura del sistema, y el diseño de la ontología llevado a cabo. El capítulo finaliza con un breve ejemplo ilustrativo para mostrar el funcionamiento del sistema, así como el grado de autonomía y de supervisión humana obtenidos.
- Capítulo 4: Este capítulo concluye la memoria de investigación, presentando las conclusiones más relevantes de la investigación realizada, e indicando las líneas de actuación futuras a llevar a cabo.

Capítulo 2

Preliminares

Este capítulo realiza un estudio y visión general del consenso dentro del ámbito de la Toma de Decisión en Grupo (TDG), además de hacer una revisión del estado del arte en Sistemas Multi-Agente (SMA). Para ello, comenzamos sentando las bases y conceptos básicos de los problemas de TDG y, más concretamente, aquellos basados en consenso, mostrando a continuación un esquema general de resolución de procesos de consenso y dando una breve visión de algunos modelos de consenso existentes en la literatura. Seguidamente, se estudian las principales características de los SMA, incluyendo los tipos y arquitecturas de SMA existentes, estándares de desarrollo y mecanismos de comunicación entre agentes.

2.1. Problemas de Toma de Decisión en Grupo (TDG)

Tomar una decisión consiste en elegir la mejor opción de entre un conjunto de alternativas posibles [10, 46, 70, 80]. A menudo nos enfrentamos a situaciones en las que debemos decidir qué alternativa tomar en función del entorno en el que nos encontramos.

Los problemas clásicos de decisión presentan los siguientes elementos básicos:

1. Uno o varios objetivos por resolver.

2. Un conjunto de alternativas o decisiones posibles para alcanzar dichos objetivos.

3. Un conjunto de factores o estados de la naturaleza que definen el contexto en el que se plantea el problema de decisión.

4. Un conjunto de valores de utilidad o consecuencias asociados a los pares formados por cada alternativa y estado de la naturaleza.

Dado que en la vida real se puede presentar una enorme variedad de problemas de decisión, la Teoría de la Decisión ha establecido una serie de criterios para clasificarlos bajo diferentes puntos de vista:

1. Según el número de criterios o atributos que se han de valorar para cada alternativa. Tenemos problemas de toma de decisión de un solo criterio y multi-criterio [15, 80].

2. Según el ambiente de decisión en el que se han de tomar las decisiones. Se definen problemas de toma de decisión en ambiente de certidumbre, riesgo e incertidumbre [58].

3. Según el número de expertos. Tenemos problemas de toma de decisión individual y toma de decisión en grupo (TDG) [54, 57].

En este trabajo de investigación nos centramos en problemas de decisión bajo incertidumbre donde participan varios expertos, más concretamente, en problemas de TDG. Tomar decisiones en grupo implica la participación de varios decisores que han de tomar decisiones de forma colectiva, de cara a alcanzar una solución común a un problema. Un proceso de toma de decisión en el que participen varios individuos o expertos, donde cada uno de ellos aporta sus propios conocimientos y experiencia, dará como resultado, en ciertos ambientes, una decisión de mayor calidad que aquella aportada por un único experto.

La solución a un problema de TDG se puede obtener aplicando un *enfoque directo* o bien un *enfoque indirecto* [36, 37]. En un enfoque directo, la solución se obtiene a partir de las preferencias individuales de los expertos (sin obtener una opinión social o general antes de

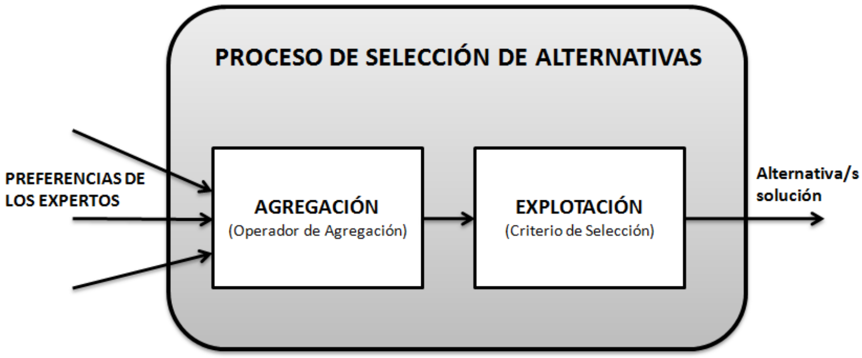

Figura 2.1: Fases del proceso de selección en problemas de TDG

la resolución del problema), mientras que en un enfoque indirecto dicha solución se consigue determinando *a priori* una opinión social, y empleando dicha opinión para la obtención de la solución. Tal y como se observa en la Figura 2.1, en ambos enfoques el proceso general para alcanzar una solución al problema de TDG se compone de dos fases [72]:

(1) *Fase de Agregación*: Se combinan las preferencias de los expertos.

(2) *Fase de Explotación*: Consiste en obtener una alternativa o un subconjunto de alternativas que den solución al problema de decisión.

Formalmente, un problema de TDG se caracteriza por:

- La existencia de un problema o cuestión común a resolver.
- Un conjunto de posibles alternativas entre las que escoger.

$$X = \{x_1, x_2, \ldots, x_n\}(n \geq 2) \tag{2.1}$$

- Un conjunto de individuos (expertos) que expresan sus juicios, opiniones o preferencias sobre el conjunto de alternativas y que tienen la intención de alcanzar una solución en común al problema planteado.

$$E = \{e_1, e_2, \ldots, e_m\}(m \geq 2) \tag{2.2}$$

Cada experto debe utilizar una estructura de preferencia para representar su opinión sobre un conjunto de alternativas. Una de las estructuras más habituales en problemas de TDG es la *relación de preferencia difusa* [41], que será la que utilizaremos en nuestro modelo de consenso. Dado un conjunto finito de alternativas X, una relación de preferencia difusa P_i asociada al experto e_i es una matriz cuadrada de dimensión n:

$$P_i = \begin{pmatrix} p_i^{11} & \ldots & p_i^{1n} \\ \vdots & \ddots & \vdots \\ p_i^{n1} & \ldots & p_i^{nn} \end{pmatrix}$$

donde cada elemento $p_i^{lk} = \mu_{P_i}(x_l, x_k) \in [0,1]$ representa la preferencia de la alternativa x_l sobre la alternativa x_k del experto e_i, cumpliéndose:

- $p_i^{lk} > 0.5$ indica que el experto e_i prefiere la alternativa x_l respecto de x_k.
- $p_i^{lk} < 0.5$ indica que el experto e_i prefiere la alternativa x_k respecto de x_l.
- $p_i^{lk} = 0.5$ indica indiferencia del experto e_i entre las alternativas x_l y x_k.

Los elementos situados en la diagonal de esta matriz carecen de relevancia durante la resolución de un problema de TDG, ya que no es adecuado hablar de la preferencia de una alternativa sobre sí misma.

Dependiendo del problema al que nos enfrentemos, existen situaciones en las que todos los individuos participan para tomar la decisión, y otras en las que el proceso de decisión sólo debe concernir a un decisor o un pequeño grupo de estos, de ahí que puedan surgir inconvenientes cuando un grupo de individuos participa en un proceso de decisión. Así, podemos encontrarnos en situaciones de colaboración entre expertos, de competitividad entre expertos, propuestas compatibles e incompatibles con uno o más expertos, e incluso propuestas que involucren a diferentes entornos (por ejemplo, entre compañías, gobiernos, etc.). Por esta razón, existen diferentes criterios clásicos que ayudan a resolver problemas de toma de decisión en grupo, basados en diferentes reglas para obtener la solución [16]:

- *Regla de la Mayoría*: Se toma la decisión teniendo en cuenta la opinión de la mayoría de individuos que componen el grupo envuelto en el problema de decisión. Una vez adoptada la decisión de la mayoría, ésta debe ser respetada por las minorías del grupo, por lo que éstas no deben oponerse a la misma, ya que se asume que todos aceptan el uso de la regla. La noción de mayoría admite dos grandes modalidades de aplicación de la regla:

  1) *Mayoría absoluta*, cuando la opinión mayoritaria ha sido tenida en cuenta por más de la mitad del total de expertos.

  2) *Mayoría relativa o simple*, cuando solamente se requiere que la opinión mayoritaria haya sido la más numerosa en cuanto a expertos se refiere, aunque la suma del resto de expertos la supere.

- *Regla de la Minoría*: Se delega la toma de la decisión en un subgrupo de personas, ya que el problema requiere un nivel de experiencia que solamente presentan dichas personas. Es necesario que todos los expertos participantes acepten la regla y, por consiguiente, estén de acuerdo con delegar la toma de la decisión al subgrupo acordado.

- *Individual*: Esta situación se presenta cuando el grupo recurre a un experto para tomar la decisión o cuando existe un líder en el grupo.

- *Unanimidad*: Todos los miembros deben estar de acuerdo con la decisión tomada.

En la mayoría de estas situaciones se presenta el problema de que algunos expertos consideren que sus opiniones no han sido tenidas en cuenta suficientemente [3]. Además, existen situaciones en las que es necesario un alto nivel de acuerdo entre los expertos participantes. Por esta razón, surge la necesidad de emplear enfoques basados en *consenso*, que añaden una nueva fase al proceso de TDG con el objetivo de alcanzar un acuerdo global entre todos los expertos antes de tomar la decisión. En la siguiente sección abordaremos el concepto de consenso y revisaremos algunos enfoques y modelos de consenso propuestos por diferentes autores.

2.2. Modelos de Consenso

En esta sección introducimos el concepto de consenso y presentamos las diferentes interpretaciones existentes del mismo, destacando el enfoque conocido como *soft consensus*. Seguidamente, presentaremos el esquema general de los procesos de consenso y daremos una breve revisión de los principales modelos de consenso propuestos en la literatura por diferentes autores.

2.2.1. Concepto e Interpretaciones de Consenso

Según la Real Academia de la Lengua Española [27], se define el término consenso como el *acuerdo producido por consentimiento mutuo entre todos los miembros de un grupo o entre varios grupos*. Por su parte, Saint define en [73] el consenso como *un estado de acuerdo mutuo entre los miembros de un grupo, donde todas las opiniones e inquietudes de cada uno de los individuos han sido tenidas en cuenta para conseguir la satisfacción del grupo*.

Estas definiciones asumen la idea de un proceso de TDG en el que ningún experto está en desacuerdo sobre las decisiones tomadas, aunque algunos expertos pueden seguir opinando que su solución individual fuera mejor que la finalmente tomada. Para conseguir el acuerdo es necesario, pues, que *todos* los expertos cambien sus opiniones iniciales, tendiendo a aproximarlas hacia una opinión colectiva que consideren satisfactoria.

A pesar de que la fase de consenso se introduce en los procesos de TDG para incrementar el nivel de acuerdo entre los expertos que participan en el problema de decisión, en ocasiones el concepto de consenso causa cierta controversia, ya que puede ser interpretado de distintas formas, desde un total acuerdo (unanimidad) a una interpretación más flexible.

El enfoque tradicional de consenso supone una visión *rígida* del mismo, donde se considera que existe consenso solamente cuando el acuerdo entre los expertos es total y unánime [17]. Este enfoque presenta el inconveniente de que el acuerdo total suele ser difícil o imposible de alcanzar en la práctica. Además, el consenso por unanimidad a veces puede haberse alcanzado mediante intimidación u otras circunstancias en las que el acuerdo alcanzado no es verdadero (*consenso normativo*) [61], razón por la cual se ha mostrado a menudo como un método poco realista para tomar decisiones reales. El consenso normativo implica en la práctica una imposición *a priori* desde el exterior del grupo que los miembros del mismo asumen sin discutir.

El consenso no debería entenderse como un acuerdo unánime, sino como un proceso en el que la decisión finalmente tomada posiblemente no coincida totalmente con las posiciones iniciales de los expertos. Esta concepción del consenso se conoce como *Consenso Cognitivo* [61], e implica que los expertos modifican sus opiniones iniciales tras una serie de rondas de discusión y negociación. En la literatura podemos encontrar diferentes ideas de *consenso* que intentan "suavizar" la versión rígida de consenso como unanimidad. Algunas de ellas son [46, 47, 73]:

- Todos los miembros del grupo apoyan la decisión tomada.
- Ningún miembro del grupo se opone a la decisión tomada.
- Todos los miembros del grupo asumen la decisión tomada, aunque no la apoyen.
- Estado de mutuo acuerdo entre los miembros del grupo, y donde las opiniones cruciales de los individuos han sido consideradas para satisfacción de estos.

Uno de los enfoques más aceptados para "suavizar" la noción rígida de consenso como unanimidad es el de *soft consensus*. Este enfoque, propuesto por Kacprzyk [46, 47, 48, 49], profundiza en el estudio del concepto de mayoría en problemas de TDG, proponiendo suavizarlo mediante otro más flexible, denominado "mayoría difusa", que introduce el uso de cuantificadores lingüísticos difusos para medir el nivel de acuerdo [46, 52], y donde el concepto de *soft consensus* se entiende por: *La mayor parte de los expertos están de acuerdo en las alternativas importantes*. Nótese que la definición de *soft consensus* está basada en la teoría de conjuntos difusos [55]. Este enfoque ha proporcionado resultados satisfactorios en diferentes problemas de TDG [28, 38, 84].

En definitiva, el consenso es un área de investigación de gran importancia en el campo de la toma de decisión en grupo, y son muchos los autores que han propuesto modelos que justifican la necesidad de llevar a cabo un proceso de consenso previo a la selección de alternativas en este tipo de problemas [8, 9, 11, 28, 38, 43, 51, 63, 73].

2.2.2. Esquema General de los Procesos de Consenso

El principal propósito de los procesos de consenso consiste en alcanzar un nivel de acuerdo mínimo antes de iniciar el proceso de selección de alternativas, mediante discusión de preferencias, durante una o varias rondas [73]. Este proceso suele estar coordinado o dirigido por una figura humana: el *moderador*. El moderador es una figura clave en procesos de consenso, y sus funciones fundamentales son:

- Evaluar el nivel de acuerdo alcanzado en cada ronda de consenso.
- Identificar las alternativas que impiden alcanzar el consenso deseado.
- Informar a los expertos sobre los cambios que estos deben considerar sobre las preferencias en dichas alternativas.

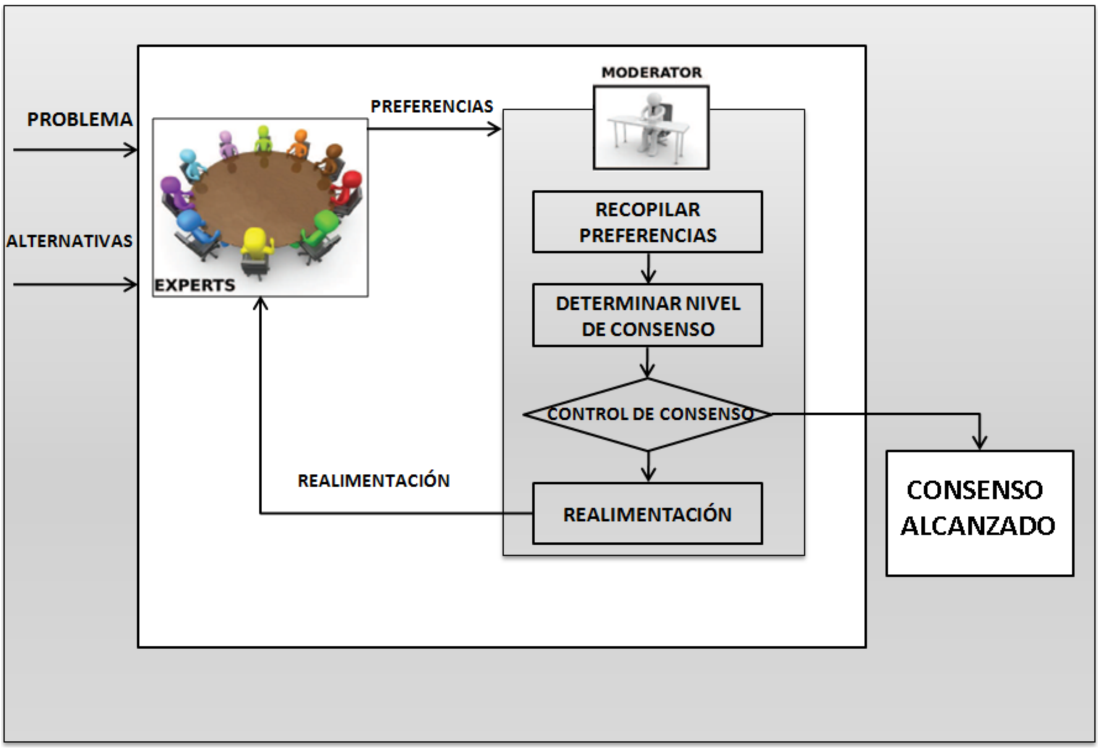

Figura 2.2: Esquema general del proceso de consenso para TDG

A continuación, pasamos a describir el esquema general de actuación para procesos de consenso seguido por la mayoría de autores, así como en este trabajo, en el cuál se considera un enfoque suavizado de medición del consenso. El esquema de este proceso se muestra en la figura 2.2, y las etapas que lo componen son las siguientes:

(1) Descripción del problema de toma de decisión, incluyendo las posibles alternativas que lo componen.

(2) Identificación del formato empleado para representar las preferencias, junto con las medidas para calcular el consenso a partir de dichas preferencias [11, 21, 24, 35, 38, 64, 81].

(3) Cada experto proporciona sus preferencias individuales al moderador.

(4) Comprobación del nivel de consenso actual, de acuerdo a las medidas de consenso elegidas. Las *medidas de consenso* son un indicador para evaluar cómo de lejos se encuentra

el grupo de expertos de un acuerdo unánime, utilizando para ello diferentes *medidas de similitud* y *operadores de agregación* [8, 31, 56], como veremos en la Sección 3.1 al presentar el modelo de consenso utilizado en nuestro sistema. Si el nivel de consenso es suficiente, el proceso finaliza y el grupo pasa a la fase de selección de alternativas; de lo contrario, continuar con el paso (5).

(5) Realimentación por parte del moderador a los expertos. Cuando existe una cierta discrepancia entre las preferencias de los expertos, el moderador identifica a los expertos y/o preferencias que impiden alcanzar el nivel de consenso deseado. A continuación, el moderador guiará a los expertos para que modifiquen dichas preferencias, con el objeto de acercar sus opiniones y mejorar el grado de consenso en la siguiente ronda. Como puede verse, el moderador juega un papel clave en esta etapa, siendo responsable de guiar, controlar y finalizar el proceso de consenso.

(6) Volver al paso (3). Dado que el número de rondas de consenso debe estar limitado, en los casos en que este número se haya sobrepasado sin llegar a un acuerdo, deben buscarse estrategias alternativas o finalizar el proceso sin éxito [73].

2.2.3. Revisión de Modelos de Consenso

En esta sección revisamos brevemente algunos modelos de consenso presentes en la literatura [42, 53, 63, 73, 84]. La estructura de estos modelos es similar a la del esquema general mostrado en la anterior sección, aunque cada uno de ellos se adapta al contexto específico para el que fue diseñado.

2.2.3.1. Modelo Teórico del Proceso de Consenso

Este primer modelo propuesto por S. Saint y J.R. Lawson [73] describe en detalle las diferentes fases llevadas a cabo en los procesos de consenso, tal y como se desarrollan en situaciones reales dentro de una empresa u organización. En la Figura 2.3 se muestra una representación gráfica de este modelo, el cual se divide en tres fases principales:

1. *Asimilación de la propuesta.* Se lleva a cabo la presentación y aclaración de las posibles dudas iniciales acerca de la propuesta que se pretende aprobar por consenso. También se consulta si existe algún inconveniente a que la propuesta inicial sea considerada como la solución ideal al problema planteado.

2. *Resolución de desacuerdos.* Se compone de varias rondas de consenso donde los expertos expresan sus preferencias y discuten para resolver los inconvenientes individuales. Al final de cada ronda se comprueba si se ha alcanzado el acuerdo deseado.

3. *Cierre del proceso.* Se llega a esta fase si no se ha podido alcanzar un consenso entre los expertos en el período de tiempo inicialmente fijado. Las posibles medidas a tomar en el cierre de la sesión incluyen: retirar la propuesta, prorrogar el tiempo establecido, tomar como decisión final la opinión de la mayoría y/o excluir a los expertos que impiden alcanzar el consenso.

Además, este modelo propone una serie de roles participantes además del propio moderador.

a) **Moderador**. Es el encargado de poner en marcha todo el proceso y de la correcta ejecución del mismo. Es responsable de aconsejar a los expertos participantes para acercar su opinión a la de la mayoría.

b) **Redactor del acta**. Es el encargado de redactar el acta de cada sesión de consenso, donde se recogen por escrito las conclusiones y/o decisiones alcanzadas. Resulta aconsejable la lectura del acta antes de abandonar la reunión para evitar que algún experto tenga ideas diferentes sobre las decisiones tomadas.

c) **Controlador del tiempo**. El controlador de tiempo trabaja junto al moderador para asegurar que no se exceda el tiempo asignado a cada una de las fases. Entre sus funciones está la de notificar periódicamente a los participantes el tiempo restante de cada fase.

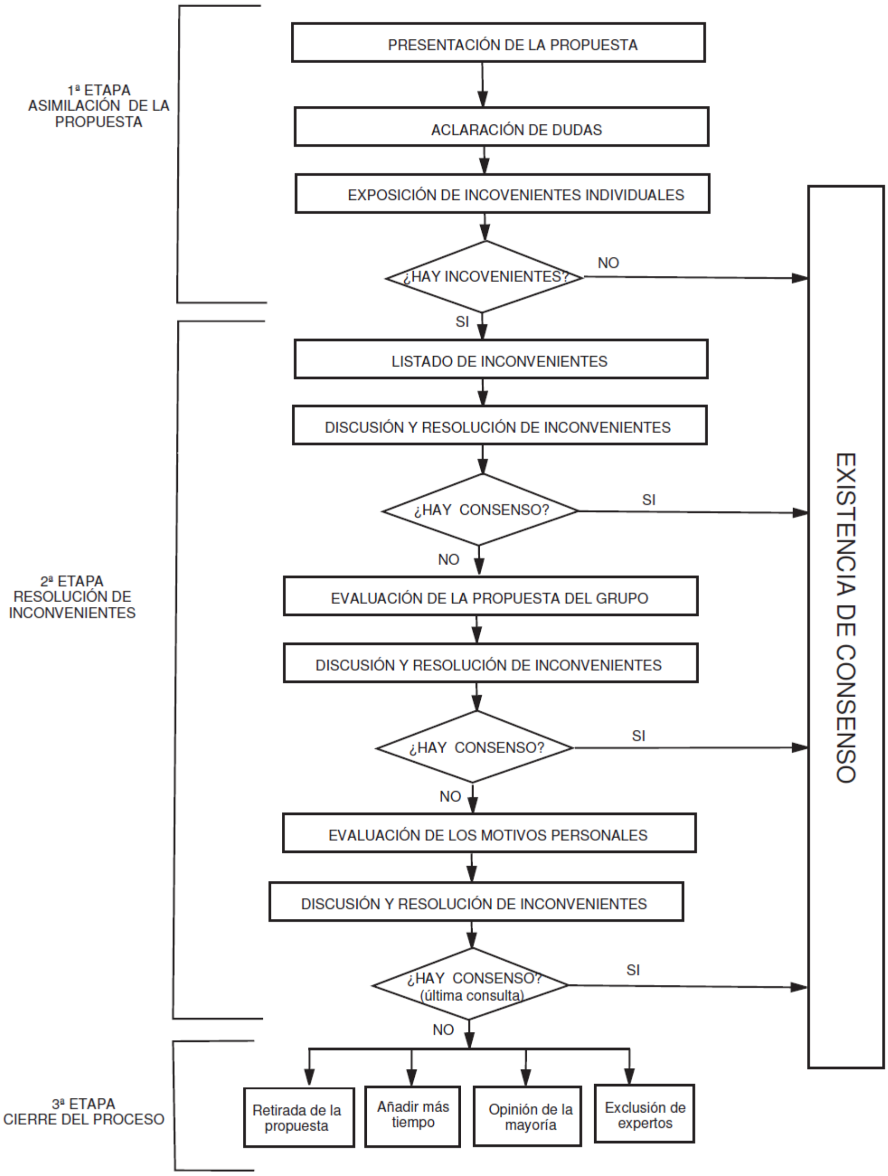

Figura 2.3: Modelo de consenso propuesto por Saint y Lawson

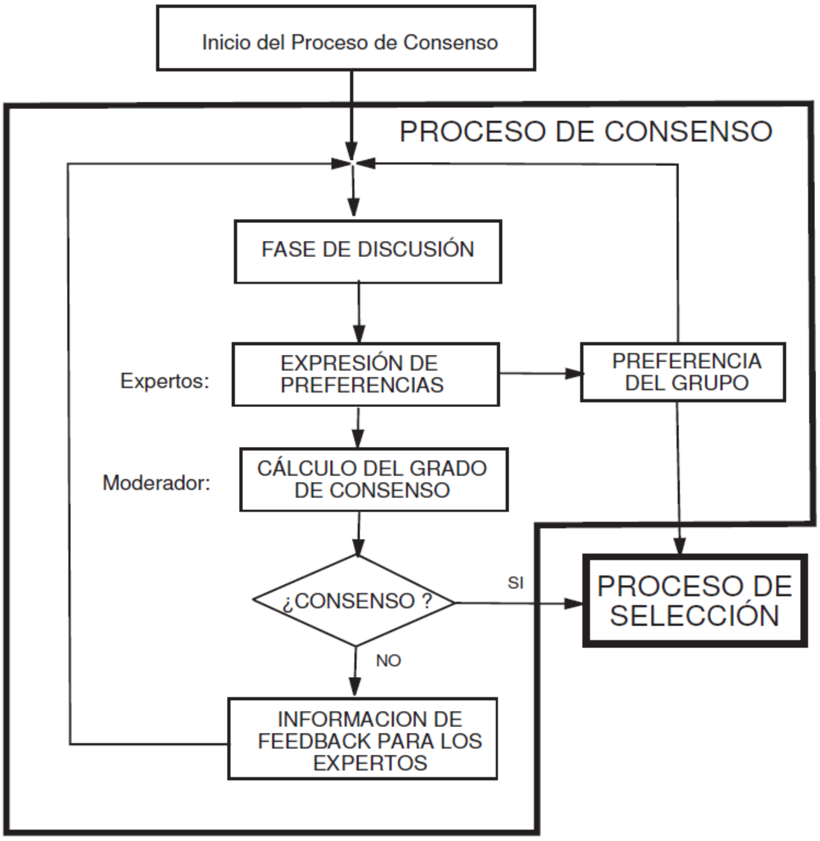

Figura 2.4: Modelo de consenso propuesto por Zadrozny

2.2.3.2. Modelos basados en Soft Consensus

Kacprzyk et al. propusieron varios modelos de consenso inspirados en la idea de *soft consensus* propuesta por dicho autor en [46]. Así, Kacprzyk y Fedrizzi establecieron en [48] una medida del grado de consenso entre opiniones de expertos expresadas como relaciones de preferencia difusas, en la que mediante el uso de cuantificadores lingüísticos [82] es posible aplicar el concepto de mayoría difusa y, en consecuencia, medir el nivel de acuerdo de forma consistente y similar a como es percibido por el ser humano.

Zadrozny y Kacprzyk propusieron en [84] un modelo de consenso cuyo esquema se muestra en la Figura 2.4, siendo sus principales características las siguientes:

a) Cada experto expresa sus preferencias mediante relaciones de preferencias difusas.

b) Se agrupa el conjunto de expertos en diferentes subgrupos en función de la coincidencia en sus preferencias.

c) Utiliza el concepto de consenso como mayoría difusa, con el cálculo de proposiciones cuantificadas lingüísticamente propuesto por Zadeh [83].

d) Además de medir el grado de consenso entre los expertos, introduce nuevos indicadores de consenso, que permiten conocer la situación de las preferencias de cada experto dentro del espacio de preferencias del grupo, e incluyen indicadores de contribución al consenso y grado de consenso personal. Los indicadores de consenso a nivel de grupo proporcionan al moderador información adicional sobre el estado actual del consenso dentro del grupo.

Este modelo cuenta además con una plataforma Web dotada de interfaz de usuario y basada en la arquitectura cliente-servidor. Recientemente, Kacprzyk y Zadrozny integraron en [53] el modelo de consenso con técnicas adicionales basadas *soft computing* para manejar conocimiento, tales como el uso de ontologías que contienen conocimiento relativo a los problemas de GDM y los procesos de consenso.

2.2.3.3. Modelo de Consenso con diferentes Estructuras de Preferencia

Este modelo, propuesto por E. Herrera-Viedma et al. [42], se basa en una visión suavizada de consenso, y se caracteriza por el uso de diferentes estructuras de preferencia por parte de los expertos. El proceso seguido en este modelo se muestra en la figura 2.5.

La principal característica de este modelo, consiste en que cada experto $e_i \in E$ expresa sus opiniones sobre un conjunto de alternativas X mediante una de las cuatro posibles estructuras de preferencia que describimos a continuación:

1. *Orden de preferencia*: El experto proporciona un orden de las preferencias $O_i = \{o_i(1), \ldots, o_i(n)\}$, donde $o_i(j) \in \{1, \ldots, n\}$ es el índice de una de las alternativas

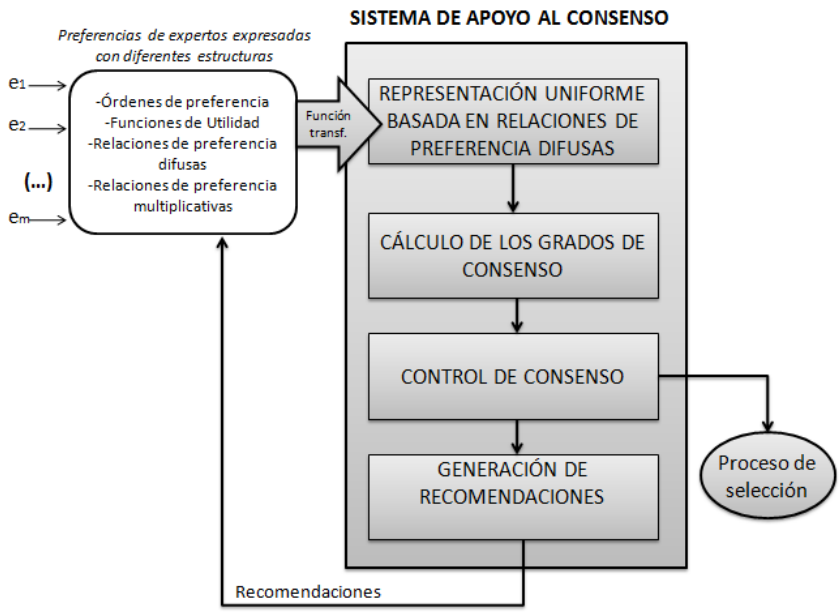

Figura 2.5: Modelo de consenso con diferentes estructuras de preferencia

existentes. Esta estructura consiste, por tanto, es un vector de alternativas ordenadas de mejor a peor, según la opinión individual del experto.

2. *Relación de preferencia difusa*: Las opiniones se representan con relaciones de preferencia P_i, donde cada $p_i^{lk} \in [0,1]$ denota el grado de preferencia de x_l sobre x_k.

3. *Relación de preferencia multiplicativa*: Se trata de relaciones de preferencia donde cada valor p_i^{lk} se interpreta como: "la alternativa x_l es p_i^{lk} veces tan buena como lo es x_k". Cada valoración se realiza dentro de una escala numérica, siendo común emplear una escala de 1 a 9, donde un valor de 1 indica indiferencia entre alternativas y un valor de 9 indica total preferencia de x_l sobre x_k. Se asumen además valoraciones recíprocas, tales que $p_i^{lk} \cdot p_i^{kl} = 1$.

4. *Vector de utilidad*: El experto expresa sus preferencias mediante un vector de valoraciones numéricas dentro del intervalo unitario.

Otras características importantes de este modelo son:

a) Representación uniforme de las preferencias: Para poder trabajar con diferentes estructuras de preferencia de forma conjunta, se aplica sobre ellas una función de transformación. Herrera-Viedma et al., proponen en [42] el uso de *relaciones de preferencia difusas* como la estructura para representar las opiniones de manera uniforme, ya que han demostrado ser de gran utilidad en problemas de TDG en la práctica, especialmente para la obtención de la opinión colectiva mediante agregación.

b) El modelo se basa en dos criterios de consenso: *medidas de consenso* flexibles y *medidas de proximidad* entre las opiniones de cada experto y la opinión colectiva.

c) Un mecanismo de realimentación, basado en sugerencias de cambio sobre las opiniones, permite reemplazar al moderador humano en esta tarea.

2.2.3.4. Modelo de Consenso Adaptativo

F. Mata et al. presentaron en [63] un modelo de consenso adaptativo, caracterizado por adaptar su comportamiento según el nivel de acuerdo alcanzado en cada ronda de discusión, con el objetivo de reducir el número total de rondas necesarias para alcanzar el consenso, en comparación con otros modelos similares no adaptativos. Las principales características de este modelo son las siguientes:

- Representación de preferencias mediante un dominio lingüístico difuso multi-granular [43], donde los expertos utilizan diferentes escalas lingüísticas para expresar sus preferencias, las cuales deberán ser unificadas en un dominio uniforme común antes de calcular el grado de consenso.

- Cálculo del grado de consenso de forma flexible, mediante la obtención de un grado de consenso global cr, que debe ser comparado con un umbral o nivel de acuerdo mínimo requerido para decidir si el equipo pasa o no a la fase de selección.

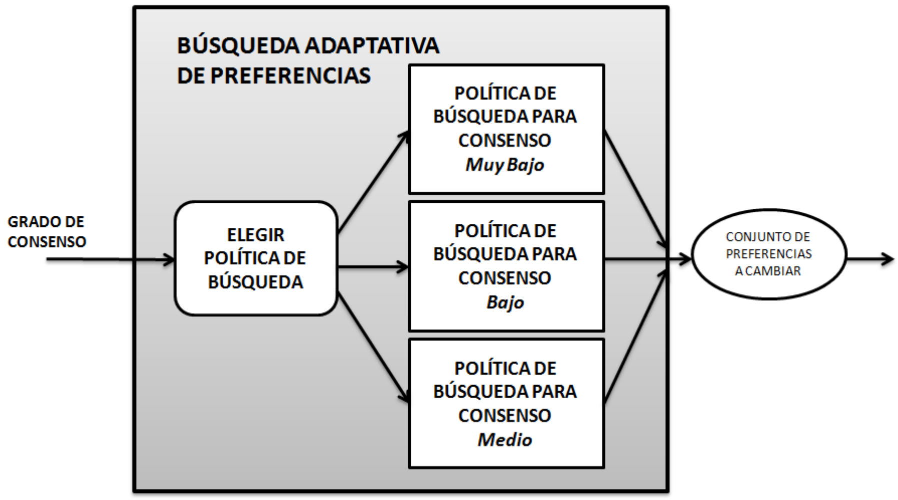

Figura 2.6: Búsqueda adaptativa de preferencias

- Búsqueda adaptativa de preferencias: Si el nivel de acuerdo obtenido es bajo, entonces existe un gran número de preferencias alejadas del consenso, por lo que el número de cambios sugeridos por el modelo debe ser alto. Sin embargo, si el nivel es alto y cercano al umbral γ, la mayoría de preferencias está cercana al consenso, por lo que el número de cambios sugeridos debe ser menor. En base a esta idea, el modelo distingue, con la ayuda de dos parámetros θ_1 y θ_2, tres niveles de consenso: *muy bajo*, *bajo* y *medio*. Cada nivel implica una política de búsqueda diferente para identificar el conjunto las preferencias alejadas del consenso, como puede verse en la Figura 2.6.

2.3. Sistemas Multi-Agente

En esta sección abordamos una introducción y estudio del estado del arte en Sistemas Multi-Agente. A lo largo de los siguientes apartados, mostraremos los principales conceptos relativos a Agentes Software y daremos una clasificación de Agentes bajo diferentes puntos de vista, veremos en qué consisten los Sistemas Multi-Agente (SMA), las arquitecturas existentes y estándares de desarrollo, haciendo hincapié en el estándar FIPA. Por último, conoceremos los mecanismos y protocolos de comunicación entre agentes.

2.3.1. Concepto y Tipos de Agentes Software

No existe una definición precisa para el término agente pero, debido a que estamos situados en un contexto de Ingeniería, es posible distinguir lo que es un agente de lo que no lo es. Así, podemos decir que el concepto de *agente* caracteriza a una entidad software con una arquitectura robusta y adaptable capaz de funcionar en diversos entornos o plataformas y capaz de cumplir objetivos de forma autónoma e "inteligente", intercambiando información con el entorno o con otros agentes humanos o software [33]. Un *agente inteligente* es una entidad capaz de *percibir*, *razonar* y *actuar* [68].

Las principales características que definen el comportamiento de un agente son [62]:

- Funcionamiento continuo y autónomo.
- Comunicación en su entorno y con otros agentes existentes en él, a través de un lenguaje o formalismo de comunicación.
- Robustez: capacidad de reaccionar de forma apropiada ante situaciones excepcionales o fallos.
- Adaptabilidad, entendida como la capacidad de cumplir objetivos y tareas en diferentes contextos de forma flexible.

Además, algunos agentes cumplen también las siguientes características:

- Razonamiento y Aprendizaje, requisito indispensable en agentes inteligentes.
- Movilidad: Un agente móvil es aquel capaz de desplazarse entre nodos de una red y ejecutarse en distintas plataformas.

A partir de las principales propiedades de los agentes, podemos dar una primera clasificación de los mismos en agentes reactivos, proactivos y sociales.

- **Agente reactivo**: Se caracteriza por ser capaz de interactuar con el entorno que le rodea de forma dinámica, y de responder ante eventos no esperados (comportamiento no determinista) [67].
- **Agente proactivo**: Su comportamiento va un paso más allá respecto al del agente reactivo. Un agente proactivo genera y trata de alcanzar sus propios objetivos, es capaz de reconocer diferentes oportunidades o alternativas de actuación y, sobretodo, se caracteriza por tomar la iniciativa al tratar con una tarea particular [76].
- **Agente social**: Es el resultado de añadir al agente proactivo la habilidad de comunicarse y colaborar con otros agentes [2]. Diferentes habilidades sociales pueden ser definidas para este tipo de agentes:
  1) *Comunicación*: Esta habilidad se basa en la capacidad de recepción y envío de mensajes por parte del agente para alcanzar sus objetivos. Una buena comunicación dependerá del nivel conversacional del agente, el número de mensajes necesarios para alcanzar un objetivo y del tamaño y complejidad de dichos mensajes [75].
  2) *Cooperación*: Indica la habilidad del agente para responder a servicios solicitados por otros agentes, así como ofrecer servicios a estos [78].
  3) *Negociación*: Se refiere a la habilidad de un agente para comprometerse, resolver conflictos y alcanzar acuerdos con otros agentes, de cara a cumplir sus objetivos. La capacidad de negociación depende en gran medida del número de mensajes que el agente requiere para alcanzar un objetivo que implica comunicarse o colaborar con uno o más agentes [75, 78].

Esta clasificación es una visión tradicional de los tipos de agentes inteligentes existentes. No obstante, posteriormente han surgido nuevas formas de clasificarlos. A continuación, exponemos brevemente los tipos de agentes existentes según diferentes criterios.

Se puede hacer una clasificación de los agentes desde varias perspectivas: según sus características individuales, según el entorno en el que trabajan, según su modo de interacción, según el modo de organización y según su utilidad [26].

Tipos de agentes según sus características individuales

Se distingue entre los agentes reactivos y agentes cognitivos.

- **Agentes reactivos**: Son agentes que realizan tareas sencillas, bajo un ciclo de *percepción/acción* [67]. El agente reactivo recibe una percepción de su entorno, y en base a ella realiza una acción, que puede consistir en cambiar su estado interno o modificar el entorno. Los agentes reactivos no realizan procesos de razonamiento, y carecen de mecanismos de representación de conocimiento.

- **Agentes Cognitivos**: Estos agentes realizan tareas más complejas, utilizando algún tipo de representación simbólica del conocimiento [19, 85]. Para cumplir su objetivo, deben realizar procesos de razonamiento, planificación y aprendizaje. Su modelo computacional se basa en un ciclo de *percepción, asimilación, razonamiento y actuación* [44].

Tipos de agentes según el entorno en el que funcionan

Se entiende por entorno del agente toda la infraestructura computacional que le rodea, y que le proporciona los medios necesarios para desarrollar su actividad. Bajo este punto de vista, podemos considerar dos tipos de agentes:

- **Agentes que requieren un entorno especial**: Precisan de una plataforma software específica para su funcionamiento. Ejemplos de ello son los agentes móviles, ya que cada plataforma en la que pueden encontrarse proporciona los mecanismos para gestionar su ciclo de vida y para facilitar su desplazamiento entre nodos de una red; y los agentes propios del estándar FIPA (véase sección 2.3.3).

- **Agentes que se se ejecutan en las plataformas computacionales existentes**: Son agentes que se crean mediante los recursos del sistema operativo y siguen el ciclo de vida de cualquier aplicación. Se suelen implementar en lenguajes independientes de la plataforma empleada, como Java.

Tipos de agentes según el modo de interacción

La interacción se entiende como la comunicación e intercambio de información entre un agente y otras entidades. Estas interacciones son de tres tipos: *agente-agente*, mediante lenguajes estándar de comunicación entre agentes como ACL; *agente-persona*, utilizando los medios adecuados para que las personas puedan comunicarse con los agentes; y *agente-entorno*, que consiste en intercambiar información con elementos como bases de datos, sistema operativo, etc.

Tipos de agentes según el modo de organización

Los agentes se pueden clasificar también según el tipo de estructura organizacional y sus capacidades. La clasificación más común desde este punto de vista distingue entre agentes individuales y agentes que cooperan [26, 85]:

- **Agentes individuales**: Son agentes sin capacidad de cooperación, que realizan sus tareas de forma individual y sin precisar de la colaboración de otros agentes.
- **Agentes cooperativos**: Pueden realizar tareas individualmente o en colaboración con otros agentes. Suelen formar parte de organizaciones de agentes.
- **Agentes competitivos**: Son agentes que para alcanzar su objetivo normalmente deben competir con otros agentes de su entorno por el uso un recurso común limitado.

Tipos de agentes según su utilidad

Los agentes se clasifican bajo este criterio según la finalidad o propósito con el que han sido creados. Las áreas donde se han aplicado las Tecnologías de Agentes han ido creciendo de forma progresiva. Actualmente existen Sistemas Multi-Agente (SMA) en áreas como el comercio electrónico, telecomunicaciones, economía, administración, procesos industriales, ocio y entretenimiento, etc. [1, 5, 6, 25, 71]. Las tareas que llevan a cabo los agentes comprenden actividades como la monitorización, diagnóstico, control de sistemas, búsqueda y recuperación de información, clasificación, tareas de mediación, etc.

2.3.2. Arquitecturas de Diseño de Agentes

Tras revisar brevemente el concepto de agente y dar una clasificación de agentes bajo diferentes criterios, en esta sección presentamos las arquitecturas existentes más comunes para construir agentes, que se clasifican como: reactivas, deliberativas e híbridas.

2.3.2.1. Arquitecturas Deliberativas. La Arquitectura BDI

Las arquitecturas deliberativas son aquellas que utilizan modelos de representación simbólica del conocimiento [59]. Los agentes que siguen esta arquitectura parten de un estado inicial y son capaces de deliberar, es decir, generar planes para alcanzar sus propios objetivos.

Un agente deliberativo debe disponer de un modelo simbólico del mundo, representado explícitamente, para realizar un razonamiento lógico a partir de él y tomar las decisiones oportunas. La principal ventaja de esta arquitectura es que el conocimiento es más fácil de entender y codificar por el ser humano. Las desventajas que presenta son la dificultad para traducir el mundo real a un modelo simbólico adecuado y preciso, y el elevado coste temporal, necesario a veces para trabajar con este modelo [7].

Sin duda, la arquitectura deliberativa BDI (*Belief, Desire, Intention*) es la más estudiada y posiblemente la arquitectura deliberativa más extendida [69].

Arquitectura BDI

La arquitectura BDI, cuyos acrónimos en español significan *Creencia, Deseo e Intención*, combina un sólido modelo filosófico del razonamiento humano y una semántica abstracta, lógica y elegante [12, 13, 20, 22, 34]. Desde su establecimiento en la década de los 80, prácticamente ha permanecido inalterable. Los tres componentes básicos de esta arquitectura son las creencias, los deseos u objetivos, y las intenciones o planes.

Las **creencias** representan conocimiento de los agentes sobre el mundo. En términos computacionales, son una forma de representar el estado del mundo, ya sea mediante el valor de una variable, una BD relacional o expresiones simbólicas. Las creencias son esenciales debido al dinamismo y la visión local del mundo. Dado que las creencias pueden representar información imperfecta del mundo, su semántica subyacente debe obedecer a una lógica de creencias, aunque la representación computacional no necesite ser puramente lógica o simbólica.

Un **deseo** (objetivo) representa un estado final deseado. En términos informáticos, un objetivo para un agente será alcanzar cierto valor en una variable, un registro, o cumplir una expresión simbólica representada en alguna formalización lógica.

Con las creencias y los objetivos no es suficiente para poder modelar un sistema capaz de operar en entornos dinámicos e inciertos: si hemos decidido sobre un curso de acción (plan), y el mundo cambia ligeramente, deberíamos cuestionarnos si seguir con el plan o replantearlo. La respuesta es que el sistema necesita acometer los planes y sub-objetivos que planifica, pero además debe ser capaz de reconsiderarlos en momentos cruciales. Estos planes acometidos o formas de proceder se llaman **intenciones**, y representan el tercer componente la arquitectura BDI. Son un subconjunto de los deseos: aquellos que el agente se ha propuesto alcanzar en un momento dado, y que caracterizan el “estado mental” actual de dicho agente.

Para determinar el grado de persistencia de una intención (su resistencia a ser cambiada ante percepciones de cambio en el mundo), se han definido tres estrategias de dedicación:

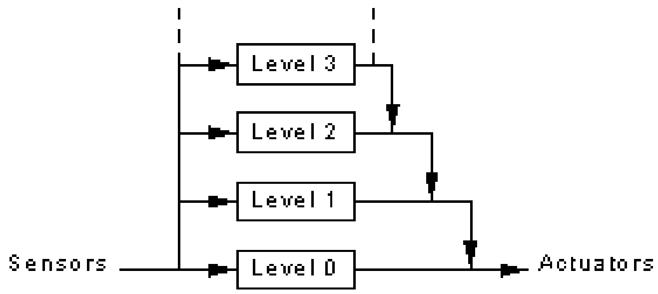

Figura 2.7: Esquema básico de la arquitectura de subsunción

- Dedicación total (a ciegas): El agente seguirá con su intención actual hasta que crea haberla alcanzado.
- Dedicación firme: El agente seguirá con su intención actual hasta que crea haberla alcanzado, o bien crea que no puede ser lograda.
- Dedicación abierta: Se mantendrá la intención solamente mientras el agente la considere viable.

2.3.2.2. Arquitecturas Reactivas

Las arquitecturas reactivas implementan la forma de actuar de los agentes como un mecanismo de correspondencia directa *percepción-acción*, y se basan en mecanismos de respuesta ante estímulos. A diferencia de las arquitecturas deliberativas, carecen de modelo simbólico y por tanto no emplean ningún mecanismo de razonamiento. Su principal ventaja es la mayor efectividad, al prescindir de un mecanismo de razonamiento complejo. Por contra, presentan el inconveniente de que el comportamiento del agente es fijo, y este no puede aprender y razonar a partir de la información que recibe.

La arquitectura reactiva más conocida es la llamada *Arquitectura de Subsunción* [14]. Esta arquitectura define una serie de capas conectadas a sensores que transmiten información en tiempo real, de forma que las capas componen una jerarquía de tareas en la que los niveles inferiores tienen menos control sobre los niveles superiores. La figura 2.7 muestra el esquema

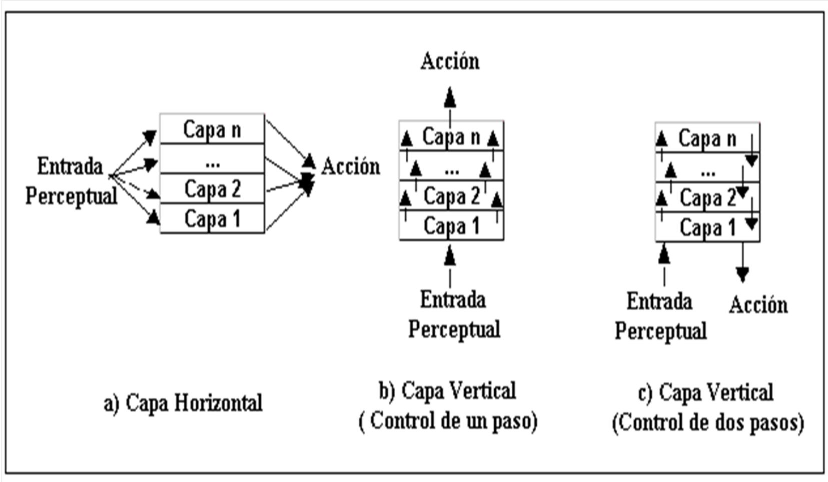

Figura 2.8: Diferentes esquemas de arquitectura híbrida

básico de la arquitectura de subsunción.

2.3.2.3. Arquitecturas Híbridas

Para intentar solventar las limitaciones de las arquitecturas deliberativas y reactivas, se han propuesto arquitecturas híbridas, que combinan aspectos de ambos modelos. Los agentes en arquitecturas híbridas presentan dos subsistemas: uno deliberativo, que utiliza un modelo simbólico para generar planes, y otro reactivo, centrado en reaccionar ante eventos en el entorno.

La Figura 2.8 muestra el esquema básico de las arquitecturas híbridas, las cuales se suelen estructurar por capas. Así, la forma de estructurarse puede ser [29, 66]:

- *Vertical*: Una única capa tiene acceso a los sensores y actuadores.
- *Horizontal*: Todas las capas tienen acceso a los sensores y actuadores.

Al igual que en las arquitecturas de subsunción, las capas se disponen jerárquicamente, brindando información sobre el entorno a diferentes niveles de abstracción. Los tres niveles básicos presentes en la mayoría de arquitecturas son:

- *Nivel reactivo*: Es el nivel más bajo, y en él se toman decisiones acerca de cómo responder ante los estímulos recibidos en tiempo real. Normalmente se basa en arquitecturas de subsunción.

- *Nivel de conocimiento*: Basado en el conocimiento que el agente posee sobre el medio. En este nivel suele emplearse una representación simbólica del mundo.

- *Nivel social*: Es la capa de más alto nivel, y se encarga de aspectos sociales del entorno, incluyendo información de otros agentes, deseos, intenciones, etc.

2.3.3. Arquitecturas Multi-Agente: El Estándar FIPA

La necesidad de desarrollar aplicaciones complejas, compuestas de multitud de subsistemas que interactúen entre sí hace necesaria la utilización de *Sistemas Multi-Agente* (SMA), que son sistemas compuestos por un número más o menos grande de agentes que trabajan de forma organizada y coordinada para la gestión inteligente de un sistema complejo, integrando los objetivos particulares de cada uno de los subsistemas que lo componen en un objetivo común.

Los SMA aportan un alto nivel de abstracción, en comparación con otras arquitecturas clásicas de computación distribuida [6]. Además, facilitan el análisis y diseño del problema a tratar en términos de agentes inteligentes, lo cual otorga una mayor flexibilidad para incorporar comportamientos humanos en estas arquitecturas. Evidentemente, el diseño de SMA es complejo y, por consiguiente, es necesario un análisis detallado del problema a tratar y de la arquitectura más adecuada para resolverlo.

La tecnología de SMA hace posible cubrir una amplia gama de problemas, por lo que resultan apropiados en problemas físicamente distribuidos, cuando la complejidad de la solución requiere de experiencia heterogénea o cuando el problema está definido sobre redes de computadores. En los últimos años, la creciente complejidad de los problemas hace que cada vez sea más necesario el uso de arquitecturas basadas en SMA [62], y son muchas las aplicaciones basadas en arquitecturas multi-agente que diferentes autores han desarrollado y propuesto en la literatura [5, 6, 19, 25, 32, 71, 85]

Como ocurre con el uso creciente de toda nueva tecnología, surgen dos cuestiones a resolver en la aplicabilidad de SMA: la *interoperabilidad*, que es la facilidad de conexión e integración de SMA, y la *apertura*, o posibilidad de extensión de los mismos. Por esta razón, es importante disponer de estándares de desarrollo, y en el ámbito de los SMA el estándar adoptado por la mayoría de entornos de desarrollo en la actualidad es **FIPA** (*Foundation for Intelligent Physical Agents*) [30].

FIPA nació en 1996 como una asociación para el desarrollo de estándares relativos a tecnología de agentes software. Su principal característica, y la que quizá lo ha convertido en el estándar más extendido, es que únicamente define el comportamiento externo (interfaz) del sistema, dejando a cargo del equipo de desarrollo toda decisión de diseño.

Algunos de los principales logros alcanzados con FIPA son los siguientes [7]:

- Un conjunto de 25 especificaciones estándares, entre las que destacan aquellas que apoyan la comunicación entre agentes, y una serie de servicios intermedios clave (*middleware*). Algunas de las especificaciones más relevantes son:
  - *FIPA Abstract Architecture Specification (SC00001).*
  - *FIPA-ACL Message Structure Specification (SC00061).*
  - *FIPA-ACL Communicative Act Library Specification (SC00037).*
  - *FIPA-SL Content Language Specification (SC00008).*
- Una arquitectura abstracta con una vista completa de los estándares FIPA2000.
- Un lenguaje de comunicación entre agentes (FIPA-ACL), además de una selección de lenguajes de contenido, como FIPA-SL.
- Un conjunto de protocolos de interacción, que abarcan desde un simple paso de mensajes hasta transacciones más complejas.

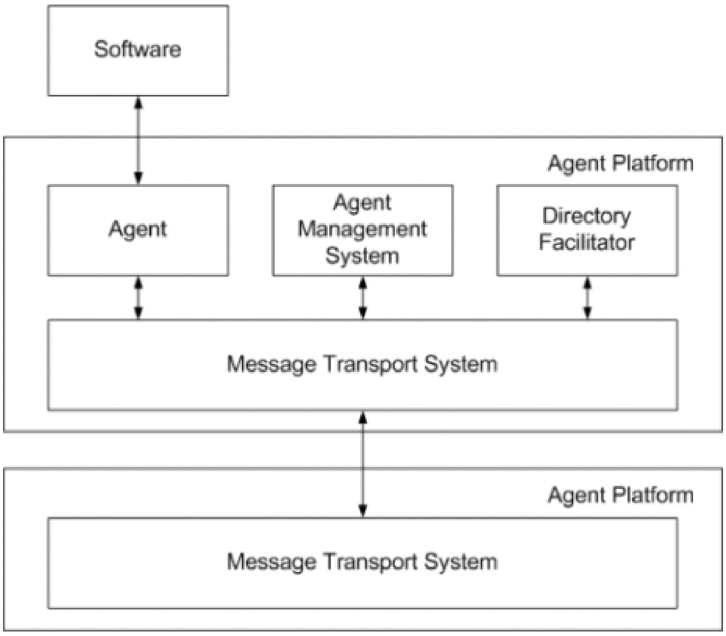

Figura 2.9: Modelo de referencia del estándar FIPA

- Varias herramientas y librerías para el desarrollo de SMA bajo este estándar, siendo JADE una de las más utilizadas y la elegida para el desarrollo de nuestro sistema.

- Especificación UML extendida, específica para el desarrollo de agentes: AUML[1].

La arquitectura seguida en FIPA define un modelo para la administración de agentes en un contexto en el que los agentes FIPA pueden existir, operar y ser gestionados, estableciendo así una referencia lógica para la creación, registro, localización, comunicación, migración y operaciones entre agentes. El modelo de referencia FIPA consta de una serie de componentes, mostrados en la Figura 2.9, y que pasamos a describir a continuación:

- **Plataforma de Agentes**: Es la infraestructura en la que se establecen y utilizan los agentes. El diseño interno de esta plataforma se deja en manos de los desarrolladores y no forma parte del estándar FIPA más allá de lo aquí expuesto. Dado que una misma

[1] http://www.auml.org/

plataforma de agentes puede encontrarse distribuida en varios ordenadores, los agentes que residen en la misma no tienen por qué encontrarse en el mismo equipo.

- **Agente**: Un agente en FIPA se considera como un proceso computacional que reside en la plataforma de agentes y normalmente ofrece uno o más servicios, cada uno de los cuales se publica junto a una descripción del mismo.

- **Facilitador de Directorio (DF)**: Este componente opcional de la plataforma de agentes proporciona un servicio de "páginas amarillas" al resto de agentes. Este componente mantiene una lista completa y precisa de los agentes existentes y proporciona información actualizada sobre los servicios que estos ofrecen.

- **Sistema de Gestión de Agentes (AMS)**: Es el elemento de gestión principal, que conoce en todo momento el estado de la plataforma y de los agentes que pertenecen a ella. Algunos de los servicios que ofrece son la creación, eliminación y control de los cambios de estado entre agentes, supervisión para el registro de nuevos agentes en la plataforma y gestión de los recursos y canales de comunicación.

- **Sistema de Transporte de Mensajes (MTS)**: Servicio proporcionado por la plataforma de agentes para transportar mensajes FIPA-ACL entre agentes, ya sea dentro de la misma plataforma o entre plataformas diferentes.

Dado que el AMS es el componente encargado de gestionar a los agentes durante el ciclo de vida de estos, conviene estudiar los diferentes estados y transiciones que componen el ciclo de vida de los agentes bajo el estándar FIPA.

2.3.3.1. Ciclo de vida de los Agentes en FIPA

El ciclo de vida de un agente define los estados en los cuales se puede encontrar en cada momento, así como las transiciones que este puede realizar entre dichos estados. El modelo de ciclo de vida propuesto por FIPA consta de los siguientes estados:

- *Iniciado*: El agente ha sido creado, pero aún no se ha registrado en el AMS, no tiene nombre ni dirección y tampoco se puede comunicar con otros agentes.
- *Activo*: El agente está registrado en el AMS, tiene un nombre, una dirección y puede acceder a los diferentes servicios ofrecidos.
- *Suspendido*: El agente está parado, su hilo de ejecución está detenido y no ejecuta ninguna tarea.
- *En espera*: El agente está bloqueado, a la espera de un mensaje, recurso u otro tipo de evento. Se desbloqueará cuando se cumpla una determinada condición.
- *Desconocido*: El agente ha sido eliminado, su hilo de ejecución ha terminado y ha sido borrado del AMS.
- *En tránsito*: Un agente móvil entra en este estado cuando está migrando de una localización a otra. Vuelve a estar activo cuando llega a su destino.

Un agente puede cambiar de un estado a otro a través de transiciones. Las diferentes acciones que puede llevar a cabo un agente para ello son:

- *Crear*: Creación o instalación de un nuevo agente.
- *Invocar*: Invocación de un nuevo agente. El agente pasa a estar activo.
- *Suspender*: Pone a un agente en estado suspendido. Puede ser iniciado por el propio agente o por el AMS.
- *Reanudar*: Continúa con la ejecución de un agente que se encontraba en estado suspendido. Sólo puede ser iniciado por el AMS.
- *Esperar*: Pone a un agente en estado de espera. Sólo puede ser iniciado por el propio agente.

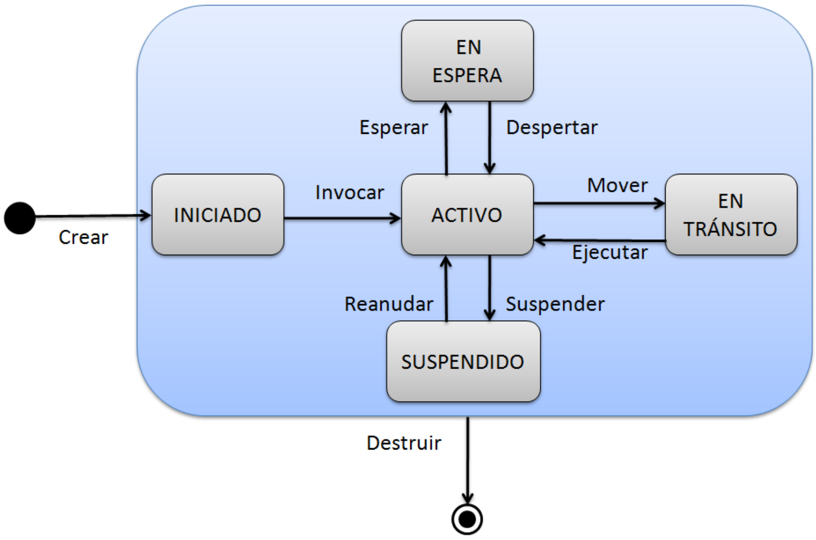

Figura 2.10: Esquema del ciclo de vida de un agente en FIPA

- *Despertar*: Continúa con la ejecución de un agente que se encontraba en estado de espera. Sólo puede ser iniciado por el AMS.
- *Mover*: Envía al agente a otra plataforma, asignándole el estado de tránsito. Sólo puede ser iniciado por el propio agente.
- *Ejecutar*: Continúa con la ejecución de un agente que se encontraba en estado de tránsito. Sólo puede ser iniciado por el AMS.
- *Destruir*: Terminación normal o forzosa de un agente. Sólo puede ser iniciado por el AMS y no puede ser ignorado por el agente.

La figura 2.10 muestra los estados y transiciones que componen el ciclo de vida estudiado.

2.3.4. Comunicación entre Agentes

Una de las características clave en los agentes software es, tal y como hemos visto, la capacidad de estos para comunicarse entre sí. Los agentes pertenecientes a un SMA se comunican entre sí por intercambio de mensajes. En el caso del estándar FIPA, esta comunicación

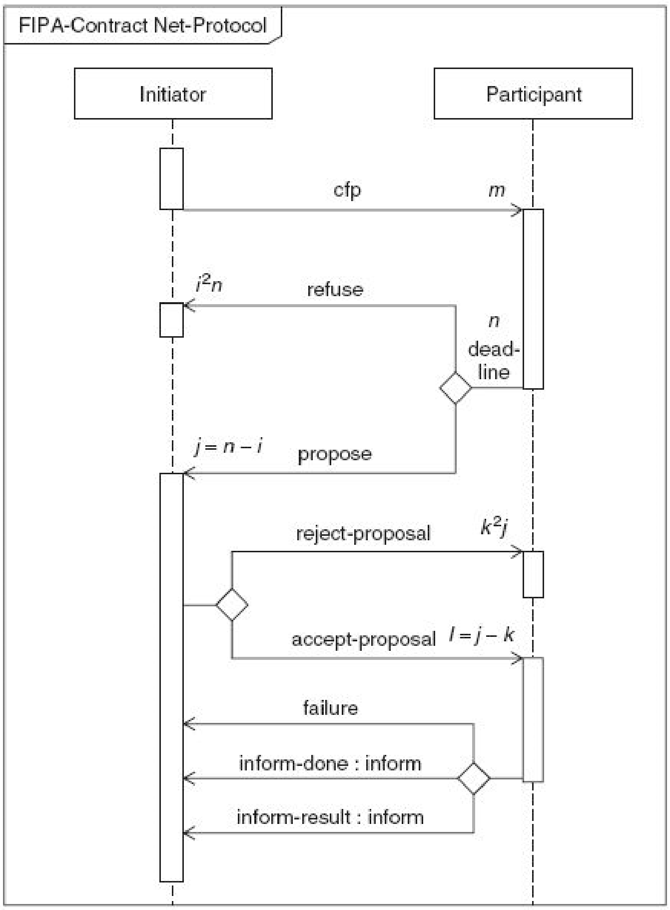

Figura 2.11: Protocolo de Interacción FIPA Contract Net

se lleva a cabo mediante el empleo de mensajes FIPA-ACL. El formato de mensajes FIPA-ACL define la comunicación en términos de una función o acción, llamada *acto comunicativo*. Existen 22 actos comunicativos diferentes en ACL, basados por lo general en la Teoría del Discurso de J. Searle [74]. Estos actos comunicativos son implementados mediante diferentes *protocolos de comunicación*, que aparecen en detalle en las especificaciones de FIPA [30], siendo algunos de los más empleados los siguientes:

- *Call for Proposal (cfp)*: Forma parte de un protocolo de comunicación completo lladado *Contract Net*, cuya estructura se muestra en el diagrama de secuencia UML de la Figura 2.11. El agente envía un mensaje (cfp), en el que propone a uno o varios receptores participar o llevar a cabo una acción. La respuesta por parte de un agente receptor

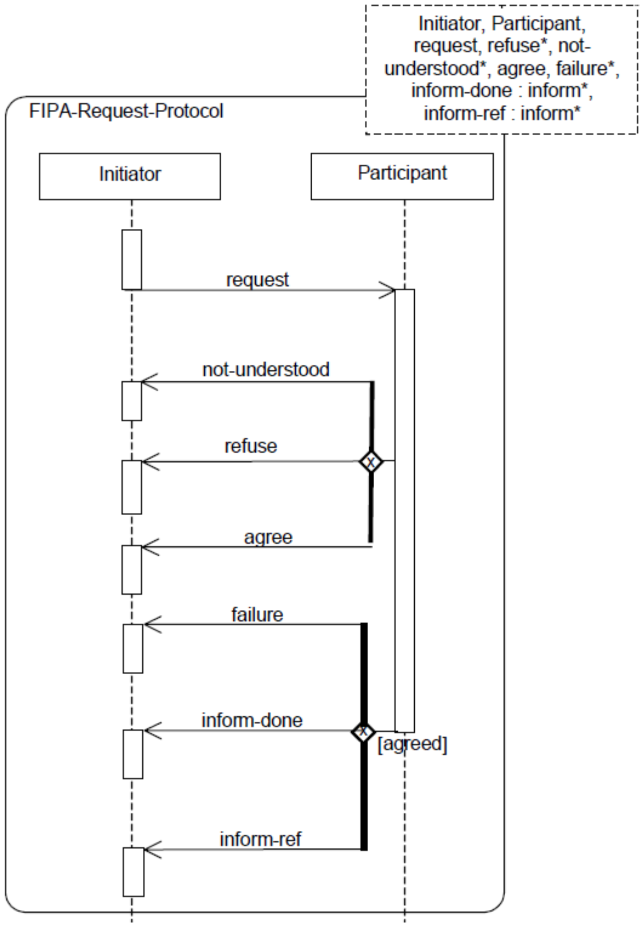

Figura 2.12: Protocolo de Interacción FIPA Request

puede ser un mensaje de tipo *Accept Proposal* si este acepta la propuesta, o de tipo *Reject Proposal* si este decide rechazarla.

- *Request*: El emisor pide a el/los receptor/es la realización de una acción. Se trata de un protocolo de comunicación muy utilizado, en el que el receptor responde en primer lugar indicando si acepta la petición (*Agree*) o la rechaza (*Refuse*). En caso de haberla aceptado, enviará además el resultado de la acción que se le ha pedido realizar mediante un mensaje *Inform*, o bien un mensaje *Failure* en caso de haber ocurrido algún error. En la Figura 2.12 aparece un diagrama de secuencia UML para representar el protocolo de interacción *Request*.

- *Query*: Consiste en la consulta de un agente a otro. Puede ser de dos tipos: consulta sobre la veracidad o falsedad de una proposición (*Query-If*), o bien sobre el objeto al que se refiere una expresión referencia dada (*Query-Ref*).

Otro aspecto importante a tener en cuenta a la hora de diseñar un SMA es el uso de *ontologías* para hacer más efectiva la comunicación entre agentes. Por defecto, un mensaje ACL de comunicación entre agentes consiste en una serie de campos, siendo uno de ellos el contenido del mismo. Este contenido es, por defecto, una simple cadena o secuencia de bytes. Sin embargo, en aplicaciones reales, los agentes necesitan a menudo transmitir o recibir información más compleja [32, 53]. Para ayudar a realizar esta tarea, diferentes entornos de desarrollo de SMA brindan la posibilidad de emplear ontologías y lenguajes de contenido. Como veremos en el capítulo 3, nuestro sistema contará con dos ontologías utilizadas por los agentes para el intercambio de información sobre el proceso de consenso en el que participan los agentes y sobre el problema de decisión a resolver.

Para hacer posible el uso de ontologías en la comunicación entre agentes, el contenido del mensaje ACL debe obedecer a una cierta sintaxis, conocida como *lenguaje de contenido*. FIPA no obliga a emplear ningún lenguaje de contenido en particular, pero recomienda el lenguaje SL (*Semantic Language*). Para que la comunicación entre agentes sea exitosa, todos ellos deben "conocer" el lenguaje de contenido utilizado para la codificación de los mensajes.

Para crear mensajes complejos bajo un lenguaje de contenido, es necesario haber definido previamente un vocabulario o conjunto de términos, y una semántica asociada a dicho vocabulario, de manera que el contenido de dichos mensajes tenga un significado claro para cualquiera de los agentes participantes en el acto comunicativo y no sean meros datos sin significado. Esto se consigue mediante el diseño y establecimiento de una ontología para el sistema [77]. En términos informáticos, una ontología es una entidad computacional y artificial, que se crea con el objeto de constituir una forma común y compartida de conocimiento de un dominio [60].

Una ontología en SMA supone la especificación de una conceptualización, la descripción de los conceptos y relaciones entre ellos, que pueden formar parte del conocimiento de un agente o una sociedad de agentes. En la actualidad, son muchas las propuestas y aplicaciones realizadas sobre SMA en las que se utilizan ontologías. Algunos ejemplos son:

- García-Sánchez et al. [32] han desarrollado en la Universidad de Murcia el sistema basado en ontologías SEMMAS, capaz de integrar agentes inteligentes y servicios Web semánticos. Gracias a la ontología de SEMMAS, se consigue explotar el potencial de ambas tecnologías, así como superar las limitaciones de comunicación existentes en ambas. La ontología desarrollada permite una comunicación transparente entre agente y servicio Web, de manera que ninguno de estos componentes precisa de cambios en su implementación y especificación originales. Algunos de los objetos representables por la ontología de SEMMAS son las tareas y responsabilidades asumidas por un agente, así como la descripción y prestaciones que ofrece un servicio Web concreto.

- El sistema desarrollado por Middleton et al. [65], consiste en un sistema de recomendación híbrido de artículos académicos on-line, provisto de una ontología para modelar perfiles de usuario. Un conjunto de agentes Web utiliza dicha ontología para determinar el perfil de un usuario en función de sus preferencias al utilizar el sistema, así como para generar las recomendaciones en base a dichas preferencias.

- Jung et al. [45] proponen en su trabajo un SMA compuesto por agentes mediadores, para estimar la semántica asociada a espacios Web desconocidos, aprendiendo a partir de los fragmentos leídos por los usuarios durante sus búsquedas. Este sistema de recuperación de información emplea una ontología para recopilar información de sitios Web nuevos así como de otros ya conocidos, y reorganizar dicha información de cara a proporcionar información semántica relativa a los nuevos sitios visitados.

En el siguiente capítulo profundizaremos sobre la estructura y modelo de contenido de las ontologías, bajo el punto de vista del problema y herramientas de desarrollo utilizadas en este trabajo.

Capítulo 3

Sistema Multi-Agente con Soporte a Procesos de Consenso

En este capítulo de la memoria de investigación presentamos el sistema multi-agente para dar soporte a procesos de consenso que hemos desarrollado (COMAS). COMAS es un Sistema de Apoyo al Consenso (SAC) basado en un Sistema Multi-Agente (SMA), desarrollado con el propósito de facilitar, guiar y automatizar procesos de búsqueda de consenso en problemas de TDG bajo incertidumbre, cada vez más frecuentes en la mayoría de organizaciones y entornos sociales y empresariales, y dotado de un grado de autonomía que permite llevar a cabo dichos procesos de forma semi-supervisada.

El sistema se compone de un conjunto de agentes inteligentes, cada uno de ellos con un rol y responsabilidades determinados, encargados de guiar, supervisar y controlar los procesos de consenso. COMAS nos permite además evaluar diferentes modelos de consenso, lo cual servirá como base para futuros desarrollos en este campo de investigación, así como para llevar a cabo simulaciones de procesos de consenso para resolver problemas de TDG definidos en diferentes contextos.

Este capítulo se estructura de la siguiente forma: comenzamos estudiando el modelo teórico de consenso utilizado e implementado en nuestro sistema, prestando especial atención al grado de autonomía y mecanismo de semi-supervisión que nuestro sistema ofrece para llevar a cabo las distintas tareas que componen dicho modelo. A continuación, presentaremos los principales aspectos relativos a la arquitectura multi-agente del sistema de apoyo al consenso, incluyendo una descripción de los diferentes tipos de agentes existentes, los procesos de comunicación entre agentes y las herramientas de desarrollo utilizadas. Seguidamente, mostraremos el diseño de la ontología para el manejo e intercambio de información compleja por parte de nuestros agentes. Finalmente, mostraremos un breve ejemplo mediante simulación para ilustrar el funcionamiento del sistema, mediante la resolución de un problema de TDG bajo incertidumbre en el que participan expertos con diferentes tipos de comportamiento durante el proceso de discusión.

3.1. Modelo de Consenso

En esta sección explicaremos detalladamente el modelo de consenso para problemas de TDG bajo incertidumbre utilizado en nuestro SMA, el cual se caracteriza por el empleo de *medidas suavizadas* de consenso para determinar el nivel de acuerdo en el grupo.

Dado que en este trabajo de investigación nos planteamos desarrollar un SAC con un cierto grado de *automatización* (abordando así uno de los principales retos presentes en los procesos de consenso), resulta fundamental adoptar un modelo de consenso que nos permita conseguir dicho grado de automatización. Muchos de los modelos presentes en la literatura no están diseñados para aplicar una automatización directa sobre los mismos. El modelo de consenso que hemos considerado, el cual se basa en ideas expuestas en [42, 43, 63], sí nos permite un cierto grado de automatización, especialmente en las tareas llevadas a cabo por el moderador. Además, para alcanzar uno de los principales objetivos de este trabajo (conseguir un SAC semi-supervisado), nos proponemos incorporar en dicho modelo un mecanismo que permita la automatización semi-supervisada de las tareas llevadas a cabo por los expertos durante el proceso.

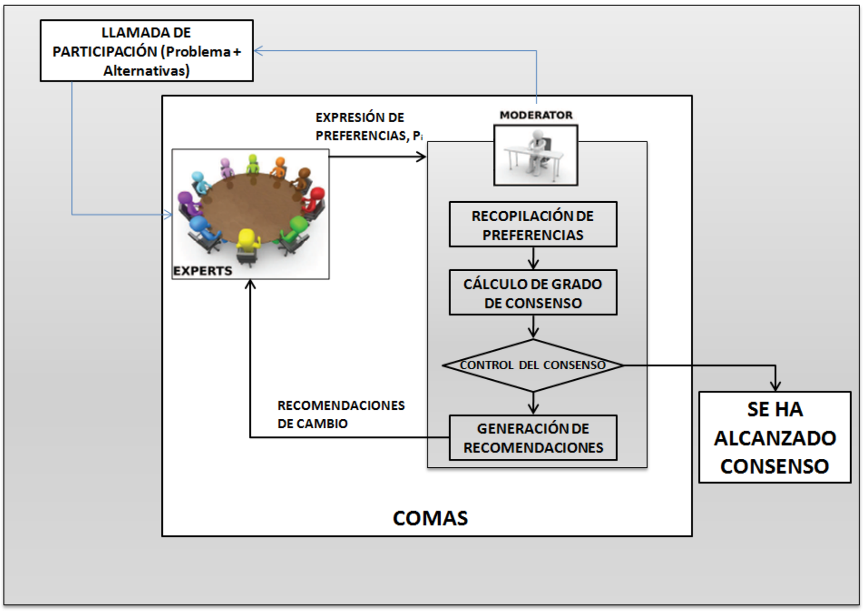

Figura 3.1: Modelo de consenso de COMAS

3.1.1. Modelo Básico

Para estudiar las fases de las que se compone nuestro modelo de consenso, es necesario conocer en primer lugar la definición y contexto en el que se situarán los problemas de TDG a abordar. Formalmente, el problema de TDG estará definido de esta forma: dado un conjunto de expertos E, cada experto e_i debe expresar sus preferencias sobre un conjunto de alternativas X mediante una relación de preferencia difusa P_i, donde $p_i^{lk} \in [0, 1]$.

A continuación describimos en detalle las principales etapas seguidas en nuestro modelo, representado en la Figura 3.1.

Llamada de participación en el Proceso de Consenso

En esta primera etapa, el moderador informa a un conjunto de expertos E sobre un nuevo problema de decisión, proporcionándoles el conjunto de alternativas X que lo componen e invitándoles a participar en el mismo. Cada experto decide si participa o no en el problema. Una vez transcurrido un período de tiempo previamente fijado por el moderador, dará comienzo el problema, (siempre que al menos hayan aceptado participar 2 expertos en él). Antes de comenzar el proceso de consenso, el moderador fija los parámetros del problema, incluyendo el umbral de consenso μ y el número máximo de rondas permitido *Maxrounds* (durante las siguientes etapas explicaremos en qué consisten dichos parámetros).

Expresión de Preferencias de los Expertos

Los expertos proporcionan al moderador sus opiniones sobre las alternativas por medio de relaciones de preferencia difusas, cuyos elementos notamos como p_i^{lk}, siendo valoraciones comprendidas en el intervalo unitario, como se definió en el capítulo 2. Así, $p_i^{lk} = 1$ implica que el experto i tiene absoluta preferencia de la alternativa l sobre la alternativa k, y $p_i^{lk} = 0$ implica que dicho experto muestra un total rechazo de la alternativa l frente a la alternativa k.

Es deseable que las valoraciones cumplan la propiedad de reciprocidad, de manera que si $p_i^{lk} = x$ ($x \in [0,1]$), entonces $p_i^{kl} = 1 - x$. De esta forma se mantendrá la consistencia entre valoraciones sobre pares de alternativas recíprocos. Además, resulta irrelevante considerar la valoración de una alternativa sobre sí misma p_i^{ll}, por lo que los elementos en la diagonal de las relaciones de preferencia no son tenidos en cuenta durante el proceso.

Cálculo del Grado de Consenso

Una vez recibidas las preferencias de los expertos, el moderador procede al cálculo del nivel de acuerdo entre estos. Para ello, se realizan los siguientes pasos:

- En primer lugar, para cada par de expertos $e_i, e_j (i < j)$, se calcula una matriz de similitud $SM_{ij} = (sm_{ij}^{lk})$ utilizando la medida de similitud propuesta en [43] para calcular cada elemento de dicha matriz,

$$sm_{ij}^{lk} = 1 - |(p_i^{lk} - p_j^{lk})| \tag{3.1}$$

  donde $sm_{ij}^{lk} \in [0,1]$ es la similitud entre los expertos e_i y e_j sobre el par de alternativas (x_l, x_k).

- Se calcula una matriz de consenso, $CM = (cm^{lk})$, agregando a nivel de pares todas las matrices de similitud previamente calculadas. Cada elemento de esta matriz $cm^{lk} \in [0,1]$ se calcula como,

$$cm^{lk} = \phi(sm_{12}^{lk}, sm_{13}^{lk}, \ldots, sm_{1m}^{lk}, sm_{23}^{lk}, \ldots, sm_{2m}^{lk}, \ldots, sm_{(m-1)m}^{lk}) \tag{3.2}$$

  donde ϕ es el operador de agregación empleado, y $l, k \in \{1, \ldots, n\}$.

  Dado que el cálculo del grado de consenso se realiza según un enfoque de consenso suavizado, nuestro modelo permite implementar y utilizar diferentes operadores de agregación para la medición del mismo. Por ejemplo, algunos operadores de agregación, como es el caso del operador OWA basado en cuantificadores lingüísticos [79, 82] permiten aplicar el enfoque de *soft consensus* basado en el concepto de mayoría difusa durante la agregación de valores para el cálculo del grado de consenso.

- Por último, se procede a calcular el nivel de acuerdo. Esto se hace a tres niveles:

  1. Consenso a nivel de pares de alternativas. Se obtuvo en el paso anterior como,

  $$cp^{lk} = cm^{lk}, \forall l, k = 1, \ldots, n \wedge l \neq k, \tag{3.3}$$

     donde cp^{lk} representa el acuerdo alcanzado sobre el par de alternativas (x_l, x_k), obtenido directamente a partir de la matriz de consenso CM.

  2. Consenso a nivel de alternativas,

  $$ca^l = \phi(cp^{l1}, \ldots, cp^{l(l-1)}, cp^{l(l+1)}, \ldots, cp^{ln}), \tag{3.4}$$

     donde ca^l representa el acuerdo sobre la alternativa x_l.

  3. Consenso a nivel de relación de preferencia,

  $$cr = \phi(ca^1, \ldots, ca^n), \tag{3.5}$$

     donde cr representa el grado de consenso global alcanzado entre los expertos en la ronda actual.

Control del Consenso

Se comprueba el nivel de consenso cr obtenido en la anterior fase. Si cr es mayor o igual que un umbral de consenso inicialmente fijado, μ, se entiende que se ha alcanzado el nivel de acuerdo deseado y el proceso finaliza; de lo contrario, el proceso requiere mayor discusión y debe continuar, a menos que se haya alcanzado el número máximo de rondas permitido *Maxrounds* (parámetro previamente fijado por el moderador), en cuyo caso el proceso finaliza sin haber llegado a un acuerdo. La estrategia a seguir por el grupo en estos casos suele ser aplicar un criterio clásico de resolución de problemas de TDG, por ejemplo la regla de la mayoría.

Generación de Recomendaciones

Esta fase se da si en la etapa de control de consenso el grado obtenido era menor que el umbral μ. Se trata de una fase crucial en el modelo por dos razones: (i) en ella se identifica aquellas opiniones de los expertos en las que existe un mayor desacuerdo, y (ii) se sugiere a los expertos nuevos valores en dichas opiniones para ser tenidos en cuenta en las siguientes rondas del proceso, de cara a incrementar el nivel de acuerdo en el grupo. El resultado de esta fase es una lista de *recomendaciones de cambio*, obtenida tras llevar a cabo las siguientes tareas:

- *Calcular la preferencia colectiva y las matrices de proximidad de los expertos.* La preferencia colectiva $P_c = (p_c^{lk})$ se calcula agregando todas las preferencias de los expertos $\{P_1, \ldots, P_m\}$ a nivel de pares de alternativas:

$$p_c^{lk} = \phi(p_1^{lk}, \ldots, p_m^{lk}) \tag{3.6}$$

  A continuación, se determinan las matrices de proximidad, que indican cuán próximas están las preferencias de cada experto a la preferencia colectiva. Cada experto e_i tiene una matriz de proximidad $PP_i = (pp_i^{lk})$, y cada elemento de dicha matriz se calcula midiendo la similitud entre la preferencia de cada experto y la preferencia colectiva,

$$pp_i^{lk} = 1 - |(p_i^{lk} - p_c^{lk})| \tag{3.7}$$

Con estos valores de proximidad, es posible identificar las preferencias más alejadas de la preferencia colectiva, y por consiguiente, los expertos que deberían cambiar sus opiniones.

- *Identificación de las preferencias que deben ser cambiadas.* Existen diferentes criterios para elegir las preferencias que deben modificarse, considerando tanto el grado de consenso como los valores de proximidad [18, 63]. Nuestro modelo propone hacer los cambios en aquellos pares de alternativas cuyo grado de consenso a nivel de alternativas ca^l y a nivel de par cp^{lk} no sean suficientes, obteniendo el conjunto de cambios, CC,

$$CC = \{(x_l, x_k) | ca^l < cr \wedge cp^{lk} < cr\} \forall l, k \in \{1, \ldots, n\} \tag{3.8}$$

  Una vez identificados los pares a cambiar, nuestro modelo debe identificar a los expertos que deberían hacer cambios en cada uno de estos pares. De este modo, los expertos cuya preferencia esté más alejada de la preferencia colectiva para el par $(x_l, x_k) \in CC$, deberán de modificar su valoración a dicho par. Para poder identificar a estos expertos, definimos un umbral de proximidad o *proximidad media* $\overline{pp}^{lk}$ para cada par, que se calcula agregando todos los valores o matrices de proximidad de los expertos a nivel de pares:

$$\overline{pp}^{lk} = \phi(pp_1^{lk}, \ldots, pp_m^{lk}) \tag{3.9}$$

  Nuestro modelo aconsejará cambiar cada par $(x_l, x_k) \in CC$ a todos aquellos expertos e_i cuyo $pp_i^{lk} < \overline{pp}^{lk}$.

- *Establecer las direcciones de cambio.* El modelo utiliza una serie de reglas de dirección para sugerir la dirección correcta en las recomendaciones de cambio, y mejorar el acuerdo en siguientes rondas. Para cada recomendación de cambio $((x_l, x_k), e_i)$ asociada a un experto y par de alternativas determinados, se sugerirá incrementar o decrementar la valoración del experto sobre dicho par. Para ello se tendrá en cuenta además la preferencia colectiva P_c. La forma de realizar los cambios consiste en considerar la recomendación, y aumentar o disminuir el valor anterior, siempre dentro del intervalo [0,1].

- DIR.1: Si $(p_i^{lk} - p_c^{lk}) < 0$, entonces se recomienda al experto e_i incrementar la valoración asociada al par (x_l, x_k).
- DIR.2: Si $(p_i^{lk} - p_c^{lk}) > 0$, entonces se recomienda al experto e_i decrementar la valoración asociada al par (x_l, x_k).
- DIR.3: Si $(p_i^{lk} - p_c^{lk}) = 0$, entonces el experto e_i no tiene que modificar la valoración asociada al par (x_l, x_k).

Una vez obtenidas todas las recomendaciones de cambio, el moderador se las proporciona a los expertos. La siguiente ronda comienza con una nueva expresión de preferencias por parte de los expertos, basándose en las recomendaciones de cambio recibidas.

3.1.2. Modelo de Autonomía Semi-Supervisada de los Agentes

Una vez estudiadas las fases que componen el modelo de consenso utilizado en COMAS, y dado que será un modelo utilizado por agentes inteligentes, nos proponemos dotar a dichos agentes de la mayor autonomía posible al utilizarlo. Así, se añadirá al modelo un mecanismo que permita a los expertos realizar las tareas de modificar y proporcionar sus opiniones de forma semi-supervisada, con el objetivo de que la necesidad de supervisión por parte del experto humano sea lo menor posible durante el proceso.

Por ello, surgen dos cuestiones que hay que estudiar:

1. Establecer el grado de cambio, es decir, el valor del incremento/decremento que un experto realiza sobre una valoración que debe modificar.
2. Fijar el grado de autonomía que los agentes de nuestro sistema tendrán para realizar los cambios sin necesidad de una supervisión directa por parte del experto humano.

En todo proceso de consenso real suele ocurrir que los expertos participantes sigan diferentes estrategias para alcanzar el acuerdo: mientras que algunos expertos pueden preferir cambiar sus valoraciones de forma significativa en las primeras rondas del proceso para alcanzar un acuerdo rápido, otros pueden optar por mantener una actitud más conservadora para que sus preferencias iniciales cambien lo menos posible, o incluso aplicar los cambios sugeridos de manera uniforme durante todo el proceso.

De cara a abordar esta idea, en este apartado vamos a definir un conjunto de *perfiles de cambio* para los expertos, de manera que cada experto seleccione el perfil que mejor se ajuste a su comportamiento y/o necesidades antes de proporcionar sus preferencias iniciales al moderador. Estos perfiles de cambio son modelados mediante una serie de funciones matemáticas que representan las diferentes estrategias seguidas por el experto a lo largo del proceso de consenso. Así, proponemos tres perfiles de cambio diferentes:

- *Perfil Seguro*: Representa a aquellos expertos que están muy seguros de su opinión inicial y presentan por tanto una mayor dificultad para realizar cambios al comienzo del proceso, pero se van convenciendo de la necesidad de llegar al consenso a medida que el proceso avanza. Estos expertos presentan un grado de cambio bajo en las primeras rondas, que aumentará cuando el número de rondas del proceso se acerque al límite de rondas permitido.

- *Perfil Inseguro*: Representa a aquellos expertos que presentan una mayor facilidad para modificar sus preferencias al comienzo del proceso, debido a que están poco seguros de su opinión inicial, aunque tenderán a realizar cambios menores en las mismas a medida que el proceso avanza. Por ello, estos expertos están dispuestos a hacer cambios considerables en las primeras rondas del proceso, aunque dichos cambios serán menores cuando el proceso de acerque al número máximo de rondas permitido.

- *Perfil Indiferente*: Representa a aquellos expertos más o menos seguros de sus opiniones iniciales, pero que están dispuestos a realizar cambios en dichas opiniones durante todo el proceso, por lo que su grado de cambio (incremento o decremento) se mantiene constante durante todo el proceso.

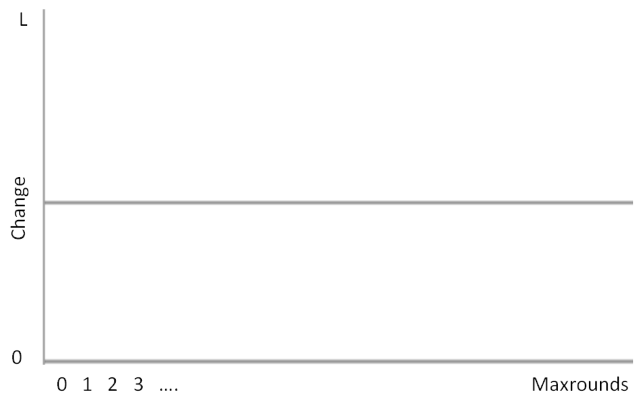

Figura 3.2: Función de cambio para perfil indiferente

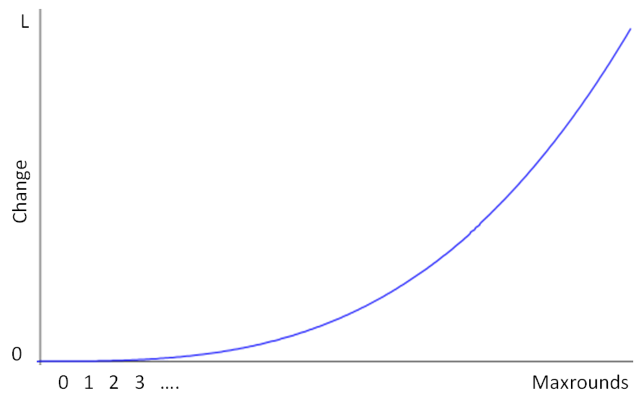

Figura 3.3: Función de cambio para perfil seguro

Para representar gráficamente los perfiles de cambio indiferente, seguro e inseguro, utilizamos tres *funciones de cambio*: una función constante, una función creciente y una función decreciente, respectivamente. Estas funciones forman parte de los componentes configurables en nuestro sistema y, aunque pueden modificarse según los requerimientos de cada problema a tratar, normalmente tendrán una forma similar a la mostrada en las Figuras 3.2, 3.3 y 3.4.

Nótese que el valor devuelto por una función de cambio, pese a ser positivo, no debe interpretarse como un incremento, sino como una variación en una valoración, ya sea para incrementar o para disminuir su valor. Una función de cambio asociada al experto e_i, y definida a partir de dos parámetros $Maxrounds$, L, se denota formalmente como

$$\Delta_i^{lk} : [0, Maxrounds] \rightarrow [0, L] \tag{3.10}$$

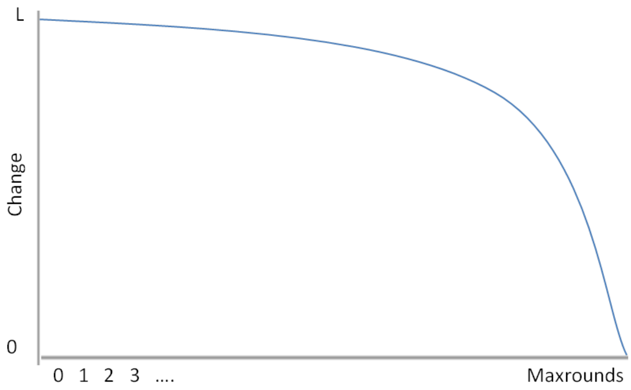

Figura 3.4: Función de cambio para perfil inseguro

La variación que e_i aplica en sus cambios a realizar tras la ronda r se denota como $\Delta_i^{lk}(r) \in [0, L]$, $r \in \{1, \ldots, (Maxrounds - 1)\}$. L es un parámetro establecido por el moderador antes del comienzo del problema, y representa una cota superior de la variación que el experto puede llevar a cabo al aplicar los cambios en cada ronda.

Por tanto, la nueva valoración del experto e_i sobre el par (x_l, x_k) tras sugerirle un cambio al finalizar la ronda r, la cual denotamos como $p_{i,r}^{lk}$, viene dada por,

$$p_{i,r}^{lk} = p_{i,r-1}^{lk} \pm \Delta_i^{lk}(r) \tag{3.11}$$

Otra cuestión de vital importancia surgió al preguntarnos si los agentes del sistema debían realizar toda la gestión de cambios de forma totalmente autónoma (sin supervisión del experto humano), o por el contrario todas las sugerencias de cambio debían ser supervisadas por el experto humano correspondiente, para que este decida aceptarlas o rechazarlas.

Dado que por un lado perseguimos dotar a los agentes de nuestro sistema con el mayor grado de autonomía posible, y por otro, existen situaciones en las que los expertos están muy seguros de su preferencia hacia una alternativa y no desearían cambiarla, hemos optado por un enfoque semi-supervisado en el que los agentes aplican los cambios en preferencias de forma autónoma, a menos que estos impliquen un cambio en la preferencia de p_i^{lk}. Así, dado un experto e_i cuya valoración inicial es $p_{i,0}^{lk} \geq 0.5$, el modelo realizará cambios sobre dicha valoración de forma autónoma, a menos que este suponga un nuevo valor $p_{i,r}^{lk} < 0.5$, y viceversa. En estos casos, el sistema solicitará la supervisión y aprobación del experto humano antes de hacer efectivo el cambio.

De esta forma, el grado de autonomía conseguido en nuestro sistema para la resolución de procesos de consenso es el siguiente:

- Bajo el punto de vista del moderador, se consigue una total autonomía, ya que el sistema únicamente precisa de supervisión humana para configurar los parámetros del problema antes de comenzar el proceso, y acceder a los resultados del mismo tras su finalización.
- Bajo el punto de vista del experto, se consigue un alto grado de autonomía gracias al modelo de autonomía semi-supervisada que proponemos, siendo necesaria la supervisión humana solamente para proporcionar las preferencias iniciales antes de comenzar el proceso, y aprobar o rechazar una recomendación de cambio, solamente cuando esta implique un cambio en la alternativa preferida.

3.2. Arquitectura del Sistema

En esta sección presentamos la arquitectura del SMA utilizado para guiar procesos de consenso. Dicha arquitectura está basada en el estándar FIPA[1] (*Foundation for Intelligent and Physical Agents*) que estudiamos en el capítulo anterior.

COMAS es un sistema multi-agente cooperativo, en el que los agentes deben de colaborar entre sí para alcanzar un objetivo común: la solución a un problema de TDG mediante consenso. La Figura 3.5 muestra de forma gráfica la arquitectura multi-agente diseñada, con cada uno de los componentes necesarios y la comunicación entre estos.

Dado que en todo modelo de consenso, y en particular en el modelo implementado en nuestro sistema, se distinguen diferentes roles participantes en el problema de TDG, en

[1] http://www.fipa.org

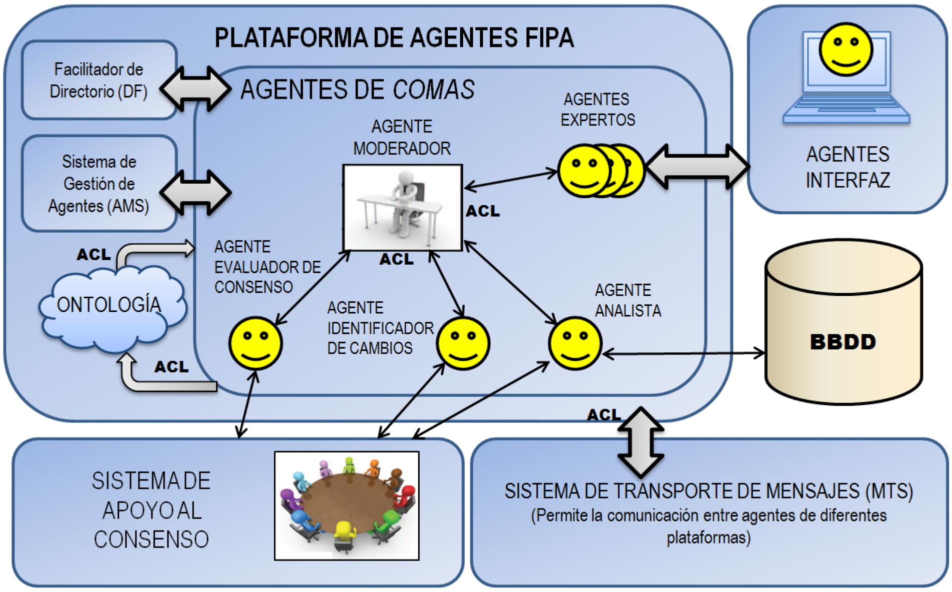

Figura 3.5: Arquitectura de COMAS

nuestro sistema multi-agente existirán diferentes agentes con roles determinados, definiéndose para cada uno de estos roles una serie de responsabilidades y líneas de actuación claras. Así, los primeros tipos o roles de agentes en COMAS surgieron directamente del estudio sobre modelos de consenso llevado a cabo en el capítulo anterior, obteniendo así las figuras de *Agente Moderador* y *Agente Experto*.

- *Agente Moderador*: Este agente asume el rol del moderador humano en el proceso de consenso, y es el principal responsable de garantizar un correcto desarrollo en dicho proceso de forma global. Existe un agente moderador por cada problema que se esté abordando en un momento dado.

- *Agente Experto*: Un agente experto representa a un experto humano en el sistema, actuando como tal de forma autónoma. El número de agentes expertos existentes en el sistema es variable, y depende del número de expertos que participen en el problema de TDG en un momento dado.

Debido a que las responsabilidades inherentes a un moderador humano resultaban demasiado complejas para ser asumidas únicamente por un solo agente, decidimos añadir una serie de agentes específicos y propios de COMAS, con el objetivo de dar soporte al agente moderador con algunas de las tareas de las que es responsable durante el proceso completo de consenso. Estos agentes son los siguientes:

- *Agente Evaluador de Consenso*: Este agente se encarga de la obtención del grado de consenso alcanzado en cada ronda, así como de notificar al agente moderador sobre ello.

- *Agente Identificador de Cambios*: Su responsabilidad se centra en llevar a cabo las operaciones necesarias para sugerir a los agentes expertos una serie de cambios en sus opiniones, con el objeto de hacerlas más cercanas entre sí.

- *Agente Analista*: Este agente adicional asume la labor de almacenar y recuperar la información relativa a cada problema resuelto de forma persistente.

Otros agentes y componentes fundamentales en la arquitectura del sistema son los siguientes:

- *Agente Interfaz*: Es un agente intermediario entre los agentes de COMAS y la interfaz del usuario. Proporciona al agente moderador los parámetros de entrada introducidos por el moderador mediante la interfaz, así como las preferencias iniciales de los expertos humanos. También recoge los resultados de la ejecución del proceso de consenso para mostrarlos en la interfaz de todos los usuarios participantes. Existe una instancia de este agente para cada usuario (experto o moderador) participando en el problema.

- *Agentes y componentes FIPA*: Nuestra arquitectura, al estar basada en el estándar FIPA, incluye una serie de agentes y componentes de utilidad para nuestra arquitectura, como son el agente *DF (Directory Facilitator)* que registra los servicios ofrecidos por cada agente de COMAS, el *AMS (Agent Management System)* para administrar y controlar a todos los agentes del sistema durante su ciclo de vida, y el *MTS (Message*

Transport System), que permite una ejecución distribuida de los procesos de consenso, con agentes provenientes de diferentes plataformas.

- *Ontología y mensajes ACL*: Nuestros agentes se comunican mutuamente por intercambio de mensajes FIPA ACL. Dado que el sistema cuenta con dos ontologías diseñadas para que los agentes compartan el mismo vocabulario y semántica asociados a los problemas a tratar, los mensajes que estos intercambien tendrán como contenido expresiones con predicados o acciones de agente pertenecientes a estas ontologías.

- *Sistema de Apoyo al Consenso*: Contiene todo el software necesario para realizar todos los cálculos y operaciones propios del modelo de consenso que presentaremos más adelante en este capítulo. Destacar que este sistema ha sido realizado con el objeto de utilizar medidas "suavizadas" del consenso, por lo que resulta apto para aplicar diferentes enfoques suavizados de consenso, como el de *soft consensus* de Kacprzyk.

- *Base de Datos*: Contiene información sobre los procesos de alcance de consenso llevados a cabo por los agentes.

3.2.1. Descripción de los Agentes

Dada la gran importancia de los agentes específicos de COMAS diseñados en este trabajo investigador, en los siguientes sub-apartados analizaremos en mayor detalle aquellos agentes dedicados a realizar tareas relacionadas con el propio proceso de alcance de consenso: agente moderador, agente experto, agente evaluador de consenso y agente identificador de cambios.

3.2.1.1. Agente Moderador

El agente moderador supone el eje central de nuestro sistema multi-agente, ya que además de emular a la figura del moderador humano en procesos de consenso reales, es el encargado de mediar todos los actos comunicativos entre el resto de agentes que componen el sistema.

Al igual que en los procesos de consenso reales, solamente existe un agente moderador en un proceso de alcance de consenso en COMAS. Sus principales funciones son las siguientes:

- *Recibir la información relativa a un nuevo problema a resolver.*
- *Localización de agentes para participar en un proceso de consenso*: El agente moderador accede al servicio de páginas amarillas proporcionado por el agente *DF* para encontrar a todos los agentes expertos, evaluadores de consenso e identificadores de cambios existentes en el sistema.
- *Llamada a la Participación en el Consenso*: Una vez detectados todos los agentes disponibles para iniciar un proceso de consenso, el agente moderador envía una propuesta de participación a cada uno de ellos, indicándoles el problema a tratar y las alternativas disponibles. Es imprescindible la participación de un agente evaluador de consenso, un agente identificador de cambios y al menos dos agentes expertos para iniciar el proceso. Cada agente experto es responsable de decidir si participa o no en el problema.
- *Petición de preferencias*: Al inicio de cada ronda, el agente moderador debe de solicitar a cada uno de los agentes expertos las preferencias sobre el conjunto de alternativas. En la segunda y sucesivas rondas, esta petición puede ir acompañada por una serie recomendaciones de cambio para cada agente experto.
- *Solicitar el cálculo del grado de consenso*: El agente moderador recopila las preferencias de los agentes expertos y se las proporciona al agente evaluador de consenso, junto con los datos y parámetros que este último necesitará para calcular el nivel de acuerdo.
- *Notificar el alcance del consenso*: Si el agente evaluador de consenso obtiene un grado de consenso igual o superior al umbral, el agente moderador informa al resto de agentes de que se ha alcanzado el nivel de acuerdo deseado.
- *Petición de recomendaciones de cambio*: En caso de no haber alcanzado un acuerdo, el agente moderador comunica al agente identificador de cambios la intención de obtener

una serie de sugerencias de cambio para los expertos cuyas opiniones están más alejadas del consenso global.

- *Comunicar recomendaciones*: Tras obtener las sugerencias de cambio por parte del agente identificador de cambios, el agente moderador proporciona a cada agente experto aquellas sugerencias que impliquen cambios en sus preferencias.

3.2.1.2. Agente Experto

La figura del agente experto tiene como objetivo automatizar en el mayor grado posible las tareas llevadas a cabo por un experto humano en un problema real de TDG, como son la expresión de preferencias y la aceptación o rechazo de las sugerencias de cambio sobre las mismas. Existe un agente experto para cada experto humano capaz de participar en un problema de TDG, aunque es posible que no todos participen un mismo problema, debido a su capacidad de decidir si aceptar o no la propuesta de participación del agente moderador.

A continuación explicamos las principales responsabilidades asumidas por un agente experto:

- *Decidir participación en el problema*: Tras recibir una propuesta del agente moderador con la descripción de un problema y las alternativas consideradas, el agente experto informa a su experto humano correspondiente para que este decida si participa o no en el problema.
- *Expresión de Preferencias*: Al inicio de cada ronda, tras recibir la petición por parte del agente moderador, el agente experto recopila su opinión sobre las alternativas en forma de una relación de preferencia, y se las proporciona al agente moderador. Al inicio de la primera ronda, es el experto humano quien, mediante un agente interfaz, proporciona sus preferencias iniciales al agente experto.
- *Realizar cambios en las valoraciones*: Ocasionalmente, un agente experto puede recibir una o varias sugerencias de cambio sobre alguna de sus valoraciones, indicándole

cuál de ellas debe modificar, así como la dirección de cambio (incrementar o decrementar). Nuestra meta es la de brindar al agente experto la mayor autonomía posible para hacer esta tarea de forma semi-supervisada. Para ello, proponemos modelar el comportamiento de cada experto [23], de forma que los agentes presenten diferentes perfiles de cambio, al igual que sucedería en procesos de consenso reales (véase sección 3.3.1).

3.2.1.3. Agente Evaluador de Consenso

Este agente se encarga de parte de las tareas que en principio debería asumir el agente moderador, más concretamente, aquellas relativas a la obtención del grado de consenso en cada ronda. El agente evaluador de consenso accede al modelo de SAC implementado para realizar las operaciones conducentes a la obtención del nivel de acuerdo alcanzado por los expertos, a partir de las opiniones proporcionadas por estos a través del agente moderador. Los principales pasos a seguir por este agente para cumplir su objetivo incluyen la obtención de los valores de similitud para cada par de expertos, el cálculo de grado de consenso sobre cada alternativa, a partir de dichas similitudes, y el cálculo del grado de consenso global en cada ronda, el cuál deberá comparar con el umbral mínimo establecido para decidir sobre la existencia o no de consenso en el grupo.

3.2.1.4. Agente Identificador de Cambios

El agente identificador de cambios también asume parte de las responsabilidades de las que un moderador humano tendría que encargarse en procesos reales. En este caso, su labor consiste en obtener un conjunto de recomendaciones o sugerencias de cambio para los expertos. Por ello, el agente identificador de cambios únicamente debe llevar a cabo su labor cuando reciba una petición del agente moderador tras haber obtenido un grado de consenso insuficiente. Al igual que el agente evaluador de consenso, este agente también tiene acceso al modelo de SAC implementado para realizar todas las operaciones necesarias sobre él.

Para cumplir su objetivo, el agente identificador de cambios realiza las siguientes operaciones sobre el modelo:

- Identificar las valoraciones (pares de alternativas) en las que no existe un nivel de acuerdo suficiente.
- Para cada valoración identificada en el paso anterior, determinar aquellos agentes expertos más alejados de la opinión de la mayoría, es decir, los agentes expertos que contribuyen en menor grado a lograr un consenso sobre la valoración.
- Generar una recomendación de cambio para cada experto y par de alternativas identificados, y asignarle una dirección de cambio (incrementar valoración o decrementar valoración).
- Proporcionar todas las recomendaciones generadas al agente moderador.

Una vez estudiadas las responsabilidades de los agentes, es el momento de estudiar cómo se comunican entre ellos durante todo el proceso de consenso.

3.2.2. Procesos de Comunicación

Los agentes en COMAS emplean los protocolos de comunicación de FIPA (basados en actos comunicativos) para intercambiarse mensajes ACL con información relevante sobre el problema. En particular, nuestros agentes utilizan los protocolos *Propose* y *Request*, cuyo funcionamiento fue estudiado en la Sección 2.3.4.

En los siguientes sub-apartados conoceremos los principales flujos de comunicación entre los agentes desarrollados, prestando especial atención a las tareas de agente (*behaviors*), utilizadas para llevar a cabo dicha comunicación. Un *behavior* es el nombre que recibe cada uno de los procesos o tareas que puede llevar a cabo un agente particular. La implementación de estos procesos es responsabilidad del programador, no obstante FIPA proporciona modelos genéricos o tipos de comportamiento [7], a partir de los cuales definiremos normalmente nuestros comportamientos particulares.

3.2.2.1. Agente Moderador - Agente Experto

1. El agente moderador comienza su interacción con un agente experto enviándole un mensaje *ACL-Propose* invitándole a participar en el problema de consenso. Esto se hace mediante una tarea de tipo *ProposeInitiator*, llamado *CallForConsensus*.

2. A través del comportamiento *CallForConsensusAnswer*, de tipo *ProposeResponder*, el agente experto recibe la propuesta enviada por el agente moderador, y le responde aceptando (mensaje *ACL-AcceptProposal*) o rechazando (mensaje *ACL-RejectProposal*) dicha propuesta.

3. Si el agente experto aceptó la propuesta, el agente moderador le envía un mensaje *ACL-Request* para solicitarle las preferencias sobre el problema. La implementación necesaria para esta petición se realiza mediante las tareas *AsssessmentRequest* y *AssessmentRequestHandler*, de tipo *AchieveREInitiator*.

4. El agente experto debe atender la petición de valoraciones y, en primer lugar, responder al agente moderador indicándole si acepta (mensaje *ACL-Agree*), rechaza (mensaje *ACL-Refuse*) o no entiende la petición (mensaje *ACL-NotUnderstood*). En caso de haber aceptado, el agente experto envía un segundo mensaje de respuesta, de tipo *ACL-Inform*, que contiene sus preferencias sobre el problema.

5. Se repiten los pasos 3 y 4 para cada ronda de consenso. En la segunda y siguientes rondas, el mensaje de petición de valoraciones por parte del agente moderador incluirá las recomendaciones de cambio sugeridas al agente experto.

6. Una vez se ha llegado al final del proceso, el agente moderador notifica de haber alcanzado consenso, o bien de haberse superado el límite de rondas, en su caso.

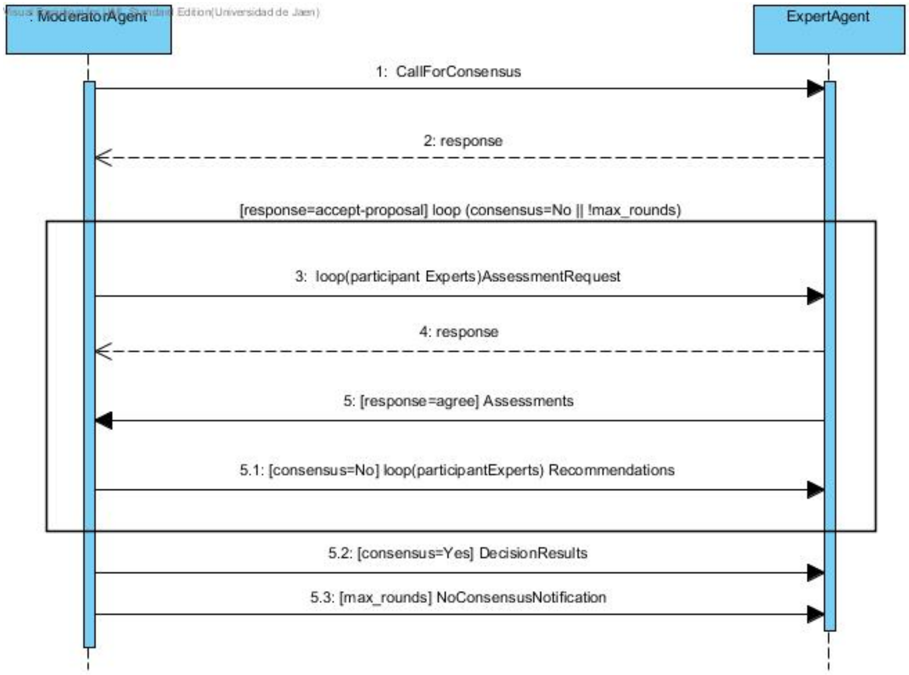

Figura 3.6: Comunicación entre agente moderador y agente experto

La Figura 3.6 muestra un diagrama de secuencia con las principales etapas del proceso comunicativo entre estos dos agentes.

3.2.2.2. Moderador-Evaluador de Consenso

1. El agente moderador comienza su interacción con el agente evaluador de consenso enviándole un mensaje *ACL-Propose* invitándole a participar en el problema de consenso, mediante una tarea de tipo *ProposeInitiator*, llamada *CallForConsensus*.

2. A través de la tarea *CallForConsensusAnswer*, de tipo *ProposeResponder*, el agente evaluador de consenso recibe y analiza la propuesta enviada por el agente moderador, y le responde aceptando (*ACL-AcceptProposal*) o rechazando (*ACL-RejectProposal*) la propuesta.

3. El agente moderador envía un mensaje *ACL-Request* para solicitar al agente evaluador de consenso el nivel de acuerdo en la ronda actual según las preferencias de los ex-

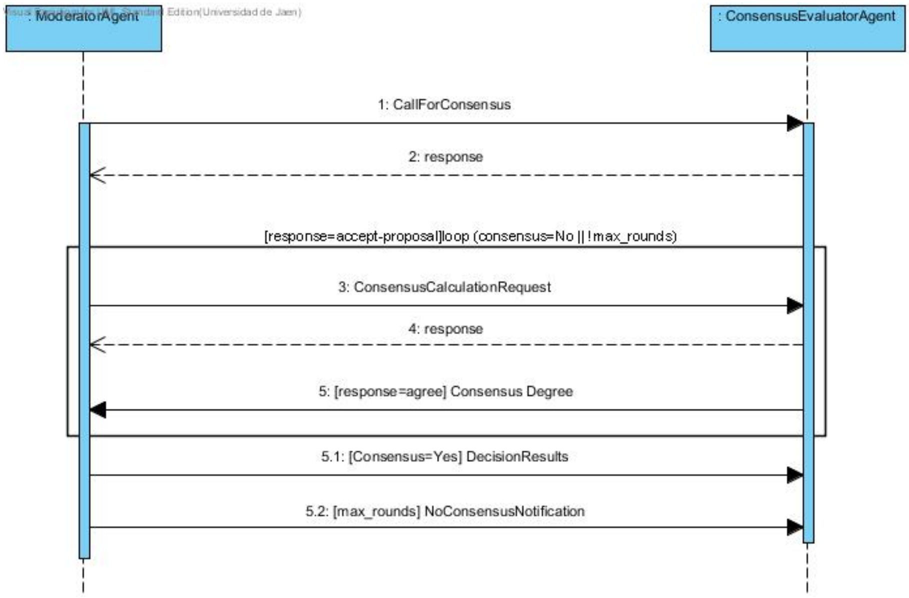

Figura 3.7: Comunicación entre agente moderador y agente evaluador de consenso

pertos. La implementación necesaria para esta petición se realiza mediante las tareas *ConsensusCalcRequest* y *ConsensusCalcRequestHandler*, de tipo *AchieveREInitiator*.

4. El agente evaluador de consenso debe atender la petición de valoraciones y, en primer lugar, responder al agente moderador indicándole si acepta (mensaje *ACL-Agree*), rechaza (mensaje *ACL-Refuse*) o no entiende la petición (mensaje *ACL-NotUnderstood*). En caso de haber aceptado, el agente evaluador de consenso envía un segundo mensaje de respuesta, de tipo *ACL-Inform*, con los resultados de determinar el nivel de acuerdo en la ronda actual.

5. Se repiten los pasos 3 y 4 para cada ronda de consenso.

6. Una vez se ha llegado al final del proceso, el agente moderador notifica de haber alcanzado consenso, o bien de haberse superado el límite de rondas, en su caso.

En la Figura 3.7 podemos apreciar los pasos del proceso comunicativo llevado a cabo entre estos dos agentes.

3.2.2.3. Moderador-Identificador de Cambios

1. El agente moderador comienza su interacción con el agente identificador de cambios enviándole un mensaje *ACL-Propose* invitándole a participar en el problema de consenso, mediante una tarea de tipo *ProposeInitiator*, llamado *CallForConsensus*.

2. A través de la tarea *CallForConsensusAnswer*, de tipo *ProposeResponder*, el agente identificador de cambios recibe y analiza la propuesta enviada por el agente moderador, y le responde aceptando (*ACL-AcceptProposal*) o rechazando (*ACL-RejectProposal*) la propuesta.

3. Si en la ronda actual no se ha alcanzado consenso, el agente moderador envía un mensaje *ACL-Request* para solicitar al agente identificador de cambios una serie de recomendaciones de cambio para los expertos y pares de alternativas más alejados del consenso. La implementación necesaria para esta petición se realiza mediante las tareas *RecommendationRequest* y *RecommendationRequestHandler*, de tipo *AchieveREInitiator*.

4. El agente identificador de cambios debe atender la petición de valoraciones y, en primer lugar, responder al agente moderador indicándole si acepta (mensaje *ACL-Agree*), rechaza (mensaje *ACL-Refuse*) o no entiende la petición (mensaje *ACL-NotUnderstood*). En caso de haber aceptado, el agente identificador de cambios envía un segundo mensaje de respuesta, de tipo *ACL-Inform*, que contiene las recomendaciones de cambio sugeridas.

5. Se repiten los pasos 3 y 4 para cada ronda de consenso.

6. Una vez se ha llegado al final del proceso, el agente moderador notifica de haber alcanzado consenso, o bien de haberse superado el límite de rondas, en su caso.

En la Figura 3.8 podemos apreciar las etapas del acto comunicativo llevado a cabo entre estos dos agentes.

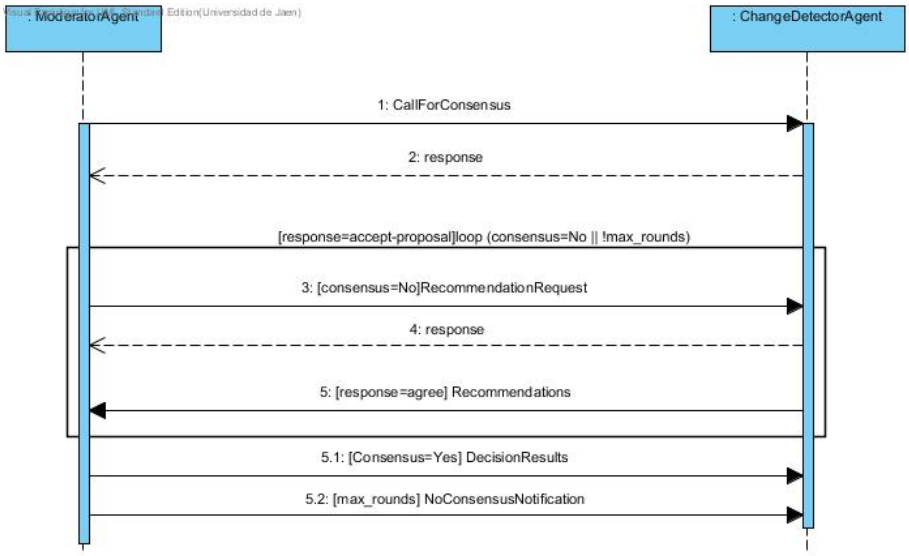

Figura 3.8: Comunicación entre agente moderador y agente identificador de cambios

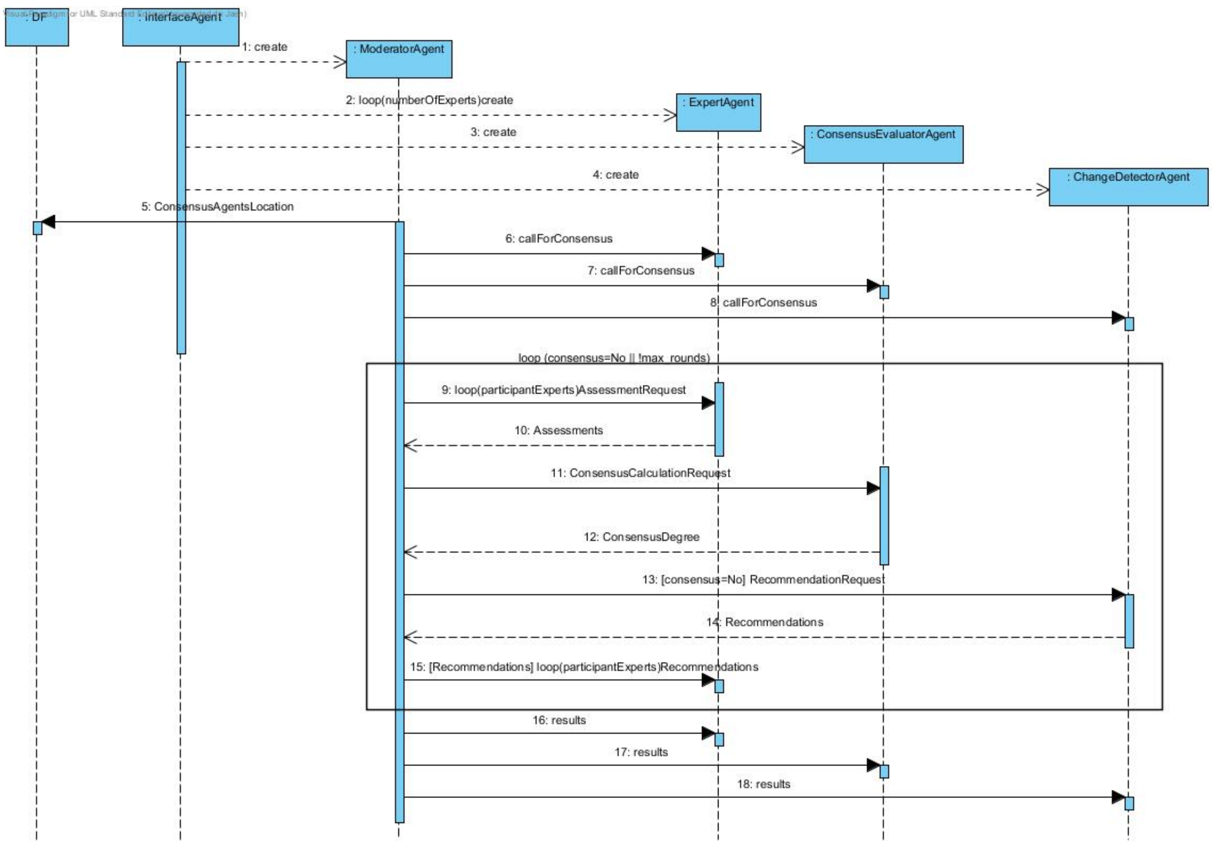

Figura 3.9: Proceso general de comunicación entre agentes

3.2.2.4. Proceso General de Comunicación entre Agentes

El diagrama de secuencia representado en la Figura 3.9 muestra, de forma resumida, el proceso completo de comunicación entre todos los agentes propios de COMAS durante un proceso de consenso. En este diagrama se puede apreciar además cómo los agentes interfaz de los usuarios se encargan de instanciar cada uno de los agentes de nuestro sistema, además del uso del *DF* por parte del agente moderador para localizar al resto de agentes en la plataforma.

3.2.3. Herramientas de Desarrollo utilizadas

Durante el desarrollo del sistema, no menos importante ha sido la fase de implementación del mismo, para la cual hemos utilizado una serie de aplicaciones de software libre que nos han ayudado a implementar tanto el modelo de SAC como el comportamiento de los agentes y las ontologías empleadas. En este apartado describiremos brevemente las principales herramientas de desarrollo utilizadas: *Netbeans*, *JADE* y *Protégé*.

NETBEANS

Netbeans[2] es un entorno de desarrollo para todo tipo de aplicaciones en lenguaje Java, que permite que las aplicaciones sean desarrolladas de forma modular. Se trata de un proyecto de código abierto de gran éxito y una enorme cantidad de usuarios en la actualidad, a la vez que una comunidad en continuo crecimiento con más de 100 socios importantes en todo el mundo. Fue fundado como proyecto de código abierto por Sun Microsystems en junio de 2000 y continua siendo el principal patrocinador de los proyectos Java.

Hemos utilizado Netbeans para implementar toda la lógica del modelo de consenso y de los agentes que componen el sistema, para los cuales es además necesaria una plataforma de gestión de agentes, como JADE.

[2] http://www.netbeans.org

JADE

JADE[3] es una conocida plataforma de desarrollo de agentes y SMA, basada en el estándar FIPA, y ha sido la principal herramienta que ha hecho posible la puesta en marcha de nuestro sistema. La plataforma JADE (*Java Agent DEvelopment Framework*) es un entorno software, implementado en Java, para simplificar la implementación de SMA a través de un *middleware* acorde con las especificaciones FIPA y dotado de un conjunto de herramientas gráficas para ayudar al desarrollador con tareas de depuración y puesta en marcha de la aplicación [7].

Algunas de las funcionalidades más atractivas que JADE proporciona al programador son:

- Un sistema en el que cada agente se ejecuta como un hilo separado de los demás, capaz de ejecutarse en máquinas remotas y de comunicarse con otros agentes de forma totalmente transparente.
- Transporte eficiente de mensajes asíncronos, con una API transparente.
- Implementación de servicios de *Páginas amarillas* y *Páginas blancas*.
- Administración sencilla y eficiente del ciclo de vida de los agentes. Los agentes reciben un identificador único (AID, *Agent Identifier*) y una dirección automáticamente durante su creación. Se proporcionan APIs para crear, suspender, reanudar, bloquear, despertar, migrar, clonar y destruir agentes.
- Soporte para movilidad de agentes, entre procesos y entre distintas máquinas, de forma transparente.
- Herramientas para interceptar y analizar mensajes de comunicación entre agentes.
- Soporte para ontologías y lenguajes de contenido.

[3] http://jade.tilab.com

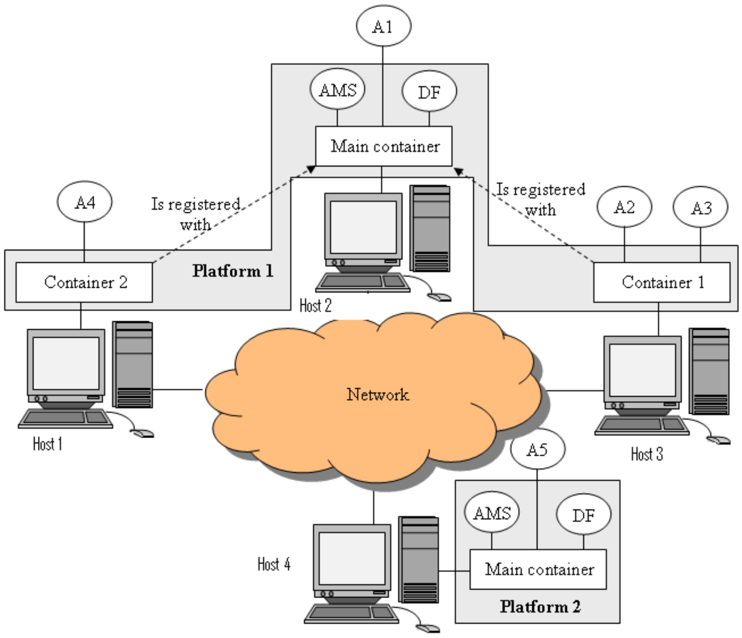

Figura 3.10: Arquitectura de JADE

- Integración con tecnologías Web como JSP, servlets, applets y servicios Web.

- Soporte para la plataforma J2ME para dispositivos móviles.

La arquitectura de JADE está basada en los componentes de la arquitectura FIPA estudiada, tal y como podemos apreciar en la Figura 3.10, ya que como hemos dicho JADE sigue las especificaciones de dicho estándar.

La principal particularidad añadida de esta arquitectura es que una plataforma de agentes se compone de uno o varios contenedores de agentes (*containers*), donde residen los agentes. Estos contenedores, al igual que la plataforma que los contiene, pueden encontrarse en un mismo equipo o estar distribuidos en una red. Existe un contenedor especial, el contenedor principal (*main container*), que es el núcleo de la plataforma y el primer contenedor en ponerse en funcionamiento, además de ser el responsable de albergar el AMS y DF. El resto de contenedores debe unirse al contenedor principal, registrándose en él por medio de un registro o tabla de contenedores (CT), para formar parte del sistema.

PROTÉGÉ

Protégé[4] es un editor de código abierto y escrito en Java para la adquisición y administración de conocimiento. Permite principalmente la construcción de aplicaciones basadas en conocimiento mediante ontologías. Protégé incluye un amplio repertorio de acciones y estructuras de modelado de conocimiento, para dar soporte a la creación, visualización y manipulación de ontologías en diversos formatos de representación. La plataforma soporta dos posibles formas de modelar ontologías:

- *Protégé-Frames*: Este editor permite a los usuarios construir y publicar ontologías de acuerdo al protocolo OKBC (*Open Knowledge Base Connectivity protocol*).
- *Protégé-OWL*: Permite a los usuarios construir ontologías para la Web Semántica, siguiendo los estándares de la W3C. Una ontología en OWL (*Ontology Web Language*) incluye la descripción de las clases, sus propiedades e instancias, pertenecientes al dominio de conocimiento en el que nos encontremos.

Gracias a Protégé, ha sido posible diseñar las dos ontologías utilizadas por nuestros agentes, las cuales han sido convertidas en paquetes de clases Java con la ayuda del plugin *BeanGenerator*.

3.3. Diseño de la Ontología

Un aspecto importante en el diseño de COMAS ha sido la definición de la ontología apropiada para representar el conocimiento relativo al problema que abordamos, y facilitar una comunicación entre agentes efectiva y comprensible, bajo un lenguaje y semántica comunes [77].

Es necesario diseñar una ontología que defina de forma apropiada los actos comunicativos llevados a cabo entre los agentes durante todo el proceso de consenso. Por ello, vamos

[4] http://protege.stanford.edu/

a considerar el enfoque propuesto por Kacprzyk y Zadrozny en [53], donde se definen *dos* ontologías para llevar a cabo procesos de consenso: (i) una ontología para representar conocimiento general acerca del proceso de consenso en sí, y (ii) una ontología para representar el conocimiento particular sobre cada problema de decisión a resolver. El diseño de nuestra ontología se basa en esta idea, mediante la definición de las siguientes dos ontologías:

1. La primera ontología incluye los componentes necesarios para representar el *dominio de la aplicación* (véase Figura 3.12), es decir, el conocimiento general relativo a los procesos de alcance de consenso que se llevarán a cabo en nuestro sistema. Un ejemplo del conocimiento representado en esta ontología son los roles de los agentes de COMAS previamente definidos y las acciones llevadas a cabo por el moderador durante el proceso.

2. La segunda ontología contiene los componentes necesarios para representar el *dominio del problema* (véase Figura 3.13), es decir, el conocimiento particular para definir cada problema de TDG resuelto mediante COMAS. Un ejemplo del conocimiento representado en esta ontología es el nombre de las alternativas que componen un problema o el grado de consenso alcanzado en cada ronda al resolver dicho problema.

En los siguientes apartados describimos el modelo de contenido de JADE, en el cual nos hemos basado para definir los componentes de nuestras ontologías, así como la estructura de las mismas.

3.3.1. Modelo de Contenido

Las ontologías constan de una serie de componentes, que forman un *modelo de contenido*. En el caso de JADE, el modelo de contenido proporcionado se compone de los siguientes elementos [7]:

- **Conceptos**: Expresiones que representan objetos, cuya información se estructura en

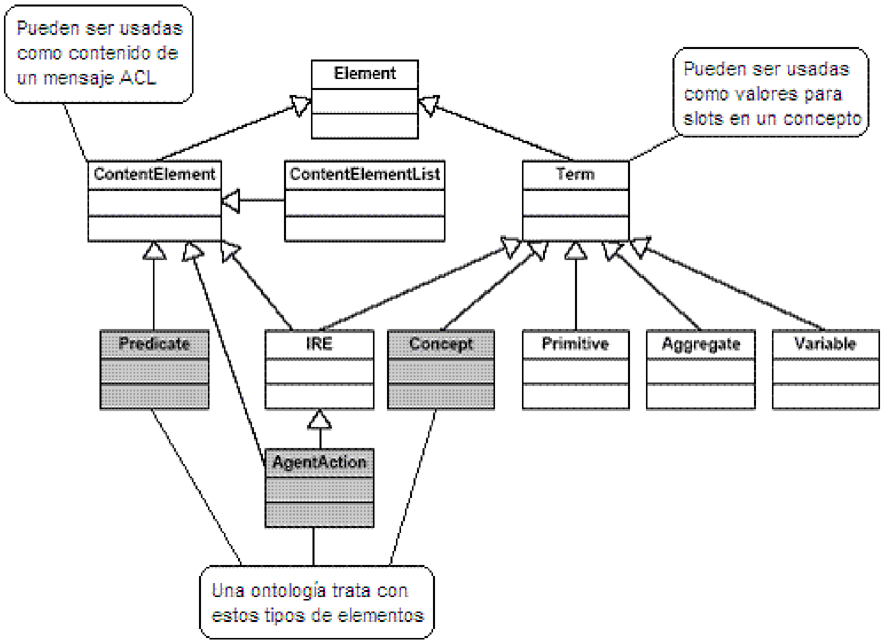

Figura 3.11: Modelo de contenido para ontologías en JADE

varios atributos. No aparecen aislados en los mensajes sino incluidos en predicados y acciones de agentes. Sus atributos pueden ser de tipos de datos simples (primitivas) o pueden ser instancias de otros conceptos.

- **Predicados**: Son expresiones sobre el estado del mundo, que pueden ser verdaderas o falsas. Se suelen emplear en mensajes *Query-If* y en mensajes de respuesta *Inform*.
- **Acciones de los agentes**: Son expresiones que indican acciones que pueden realizar los agentes. Normalmente constituyen el contenido de mensajes *Request*.
- **Otros elementos**: primitivas (elementos atómicos como números o cadenas de caracteres), agregaciones (atributos compuestos de múltiples elementos), expresiones (identifican las entidades para las que se cumple un predicado), variables.

El modelo de contenido de JADE [7] aparece representado en la Figura 3.11. Como podemos observar, este modelo consta de un amplio conjunto de componentes estructurados de forma jerárquica. Como veremos en el siguiente apartado, las ontologías de nuestro sistema (y en

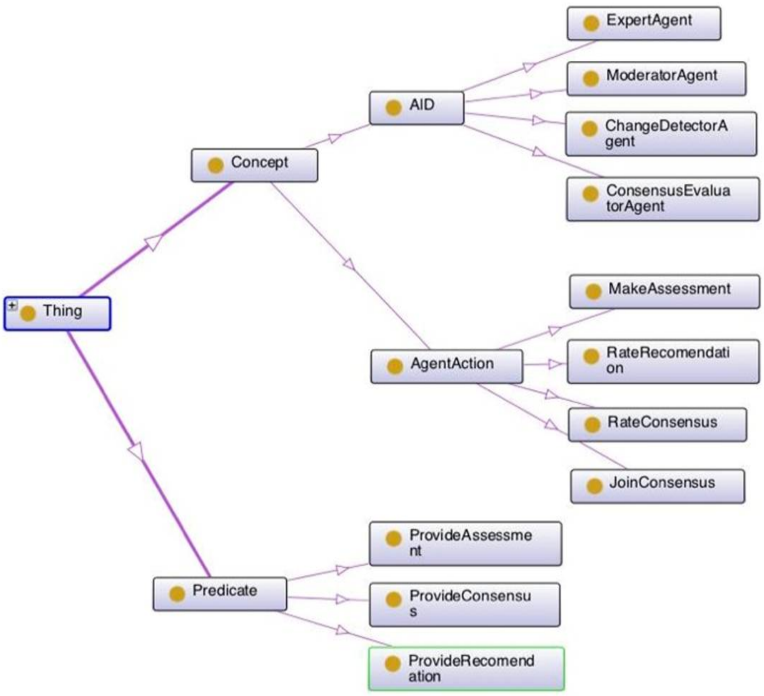

Figura 3.12: Ontología sobre el dominio de la aplicación.

general la mayoría de ontologías desarrolladas en JADE) utilizan únicamente predicados y acciones de agente de forma directa. Estos a su vez se componen de uno o varios *términos*, los cuales pueden ser conceptos (explícitamente definidos por el desarrollador de la ontología), primitivas o agregaciones, entre otros. Además, un agente debe encapsular todo predicado o acción de agente como un objeto de la clase *ContentElement* para poder enviarlo a otros agentes como el contenido de un mensaje ACL.

3.3.2. Componentes de la Ontología

La estructura y componentes de las dos ontologías definidas para COMAS aparecen representados en las Figuras 3.12 y 3.13. A continuación explicaremos con mayor detalle cada uno de estos componentes.

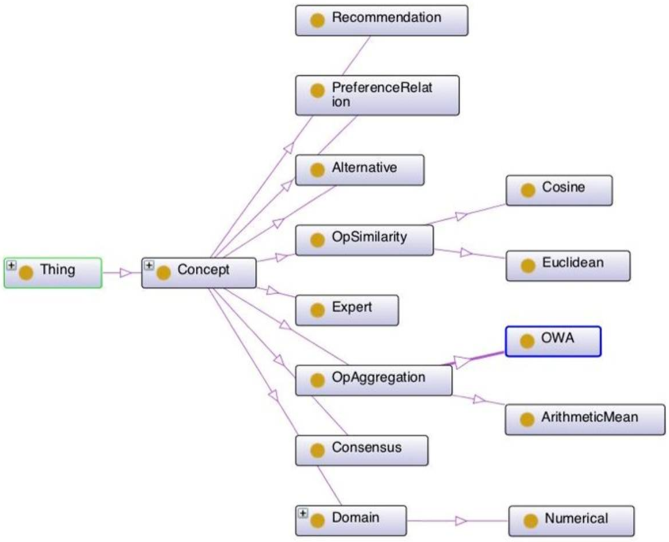

Figura 3.13: Ontología sobre el dominio del problema.

Conceptos

- **AID**: Concepto abstracto que contiene la información básica para identificar a un agente. Contiene entre otros elementos de información el nombre de un agente y su dirección. Actualmente se utiliza como un atributo del concepto *ExpertAgent*, aunque el diseño de la ontología queda abierto para su uso por otros tipos de agente.
- **Alternative**: Representación de una de las alternativas para el problema de TDG. Estos elementos forman parte de la descripción general del problema que el agente moderador debe proponer al resto de agentes. Se compone de los siguientes atributos:
  - *description*: Atributo de tipo cadena. Contiene la descripción de la alternativa.
- **Consensus**: Contiene información sobre el nivel de consenso alcanzado en un momento dado. Consta de los atributos siguientes:
  - *currentConsensus*: Atributo de tipo real, que contiene el grado de consenso actual.
  - *consensusMatrix*: Atributo múltiple (agregado) de tipo real. Contiene los elementos de la matriz de consenso obtenida en la ronda actual.

- *consensusThreshold*: Atributo de tipo real. Contiene el umbral de consenso.

- **ExpertAgent**: Este concepto se refiere a uno de los agentes expertos participantes en el consenso. Se compone del siguiente atributo:
  - *aid*: Atributo de tipo concepto *AID*. Identificador del agente experto.

- **OpAggregation**: Operador de agregación que se utilizará durante el proceso de consenso. Se compone del siguiente atributo:
  - *aggOperator*: Atributo de tipo cadena, que indica el nombre del operador de agregación a utilizar.

- **OpSimilarity**: Especifica el operador de similitud con el que se trabajará durante el proceso de consenso. Se compone de un atributo:
  - *simOperator*: Atributo de tipo cadena. Contiene el nombre del operador de similitud.

- **PreferenceRelation**: Este concepto tiene como objetivo representar una relación de preferencia completa, que como sabemos es la estructura empleada por un experto para representar sus opiniones sobre un conjunto de alternativas. Contiene el siguiente atributo:
  - *assessments*: Atributo agregado de tipo real. Contiene el conjunto de valoraciones dadas por el experto a cada par de alternativas.

- **Recommendation**: Sirve para representar una recomendación de cambio individual sugerida a un experto para que este modifique sus preferencias en vista a alcanzar un acuerdo. Consta de los siguientes atributos:
  - *aid*: Atributo de tipo concepto *AID*. Identificador del agente experto a quien es destinada la recomendación.
  - *numAlt1*: Atributo de tipo entero. Número de la primera alternativa existente en el par a modificar.

- *numAlt2*: Atributo de tipo entero. Número de la segunda alternativa existente en el par a modificar.
- *direction*: Atributo de tipo cadena. Dirección en la que el experto debe modificar la valoración, puede tomar los valores “Increase” o “Decrease”.

Acciones de Agente

- **JoinConsensus**: Contiene toda la información necesaria para elaborar una propuesta de participación en un problema de TDG a resolver mediante consenso. Los atributos de que se compone son:
    - *maxRounds*: Atributo de tipo entero. Indicador del número máximo de rondas permitido en el proceso, para dar una idea aproximada de la duración del mismo.
    - *setOfAlternatives*: Atributo agregado de tipo concepto *Alternative*. Conjunto de alternativas existentes en el problema.
    - *problemDescription*: Atributo de tipo cadena. Descripción del problema propuesto.
- **MakeAssessment**: Acción utilizada para solicitar las preferencias a los expertos. Se compone de los atributos siguientes:
    - *setOfRecommendations*: Atributo agregado de tipo concepto *Recommendation*. Conjunto de recomendaciones de cambio para los expertos. Este atributo se utiliza únicamente a partir de la segunda ronda, ya que en la primera se solicitan las valoraciones iniciales.
    - *round*: Atributo de tipo entero. Número de ronda actual.
- **RateConsensus**: Acción de vital importancia para el buen funcionamiento del sistema. Contiene toda la información de entrada necesaria para determinar el grado de consenso actual, mediante el sistema de apoyo al consenso desarrollado. A continuación describimos cada uno de los atributos que contiene:

- *maxRounds*: Atributo de tipo entero. Indica el número máximo de rondas permitido.
- *numAlt*: Atributo de tipo entero. Indica el número de alternativas existentes en el problema.
- *opSim*: Atributo de tipo concepto *OpSimilarity*. Operador de similitud empleado.
- *consensus*: Atributo de tipo concepto *consensus*. Contiene la estructura necesaria para incluir el grado de consenso obtenido, una vez calculado.
- *opAgg*: Atributo de tipo concepto *opAggregation*. Operador de agregación empleado.
- *setOfExperts*: Atributo agregado de tipo concepto *ExpertAgent*. Conjunto de expertos que participan en el problema, identificados a través de su agente experto correspondiente.
- *round*: Atributo de tipo entero. Número de ronda actual.
- *setOfPreferences*: Atributo agregado de tipo concepto *PreferenceRelation*. Contiene el conjunto de relaciones de preferencia, con las opiniones actuales de todos los expertos.

▪ **RateRecommendation**: Esta acción contiene la información necesaria para solicitar el cálculo de las recomendaciones de cambio tras un control de consenso con resultado negativo. Contiene los siguientes atributos:

- *numAlt*: Atributo de tipo entero. Número de alternativas existentes.
- *opSim*: Atributo de tipo concepto *OpSimilarity*. Operador de similitud empleado, de utilidad también en el cálculo de las matrices de proximidad.
- *consensus*: Atributo de tipo concepto *consensus*. Contiene el grado de consenso a diferentes niveles.
- *setOfExperts*: Atributo agregado de tipo concepto *ExpertAgent*. Conjunto de expertos que participan en el problema, identificados a través de su agente experto correspondiente.

- *setOfPreferences*: Atributo agregado de tipo concepto *PreferenceRelation*. Contiene el conjunto de relaciones de preferencia, con las opiniones actuales de todos los expertos.

Predicados

- **ProvideAssessment**: Este predicado es utilizado por un agente experto como respuesta al recibir una acción de agente del tipo *MakeAssessment*. Consta de un atributo:
  - *preference*: Atributo de tipo concepto *PreferenceRelation*. Consiste en la relación de preferencia actual proporcionada por el experto.
- **ProvideConsensus**: Predicado que proporciona el grado de consenso recién obtenido en la ronda actual. Incluye el siguiente atributo:
  - *consensus*: Atributo de tipo concepto *Consensus*. Grado de consenso obtenido en la ronda actual.
- **ProvideRecommendation**: Este predicado proporciona el conjunto de recomendaciones de cambio generadas en cada ronda. Los atributos de que se compone son:
  - *setOfExperts*: Atributo agregado de tipo concepto *ExpertAgent*. Contiene el conjunto de expertos a los que se les debe sugerir recomendaciones.
  - *setOfRecommendations*: Atributo agregado de tipo concepto *Recommendation*. Contiene las recomendaciones generadas en la actual ronda, ordenadas según el experto a quien van dirigidas.
  - *numRecommendationsByExpert*: Atributo agregado de tipo entero. Indica cuántas de las recomendaciones en el atributo anterior pertenecen a cada experto. Dado que tanto este atributo como *setOfExperts* mantienen una relación de orden entre sí, se consigue de esta manera enviar a cada experto sus recomendaciones correspondientes.

Por último, tras analizar en profundidad la estructura de nuestra ontología, veremos en qué etapas del proceso son utilizados cada predicado y acción de agente.

1. Acción *JoinConsensus*: El agente moderador utiliza esta acción como el contenido de un mensaje ACL-Propose para invitar al resto de agentes a participar en el problema, en el comportamiento *CallForConsensus*. Un ejemplo de instancia de esta acción es el siguiente:

```
(JoinConsensus :maxRounds 10 :setOfAlternatives (sequence
(Alternative :description Marqués de Cáceres)
(Alternative :description Los Molinos)
(Alternative :description Viña Mayor)
(Alternative :description René Barbier))
:problemDescription "Spanish Wine Selection Problem")
```

2. Acción *MakeAssessment*: Esta acción es utilizada como contenido de un mensaje ACL-Request en el comportamiento *AssessmentRequest*, del agente moderador. Sus destinatarios son una serie de agentes expertos que deben decidir si aceptar o no una petición de sus preferencias para el agente moderador. Recordemos que esta acción de agente es utilizada en la segunda y siguientes rondas para proporcionar a los agentes expertos las recomendaciones de cambio, por lo que en la primera ronda utilizamos una recomendación "vacía" (referida a un par de alternativas ficticio, (0,0)) para indicar que no existen aún recomendaciones. A continuación mostramos un ejemplo de instancia de este predicado:

```
(MakeAssessment :setOfRecommendations (sequence
(Recommendation :numAtl2 0 :numAlt1 0 :aid (agent-identifier :name "")
:direction Increase)) :round 1)
```

3. Predicado *ProvideAssessment*: Este predicado supone la respuesta de un agente experto a la acción *MakeAssessment*, cuando este acepta proporcionar sus preferencias al

moderador. Se encapsula como el contenido de un mensaje ACL-Inform en el comportamiento *MakeAssessmentHandler*. Su estructura es similar a la del siguiente ejemplo, en el que un agente experto asociado al experto e_i expresa sus preferencias p_i^{lk} sobre 4 alternativas (nótese que el predicado consta de 12 elementos en lugar de 16, al excluir los elementos vacíos de la diagonal principal en la correspondiente relación de preferencia):

```
((ProvideAssessment (PreferenceRelation :assessments (sequence
0.6725 0.1725 0.9375
0.32875 0.5 0.32875
0.6725 0.5 0.32875
0.0625 0.82875 0.5))))
```

4. Acción *RateConsensus*: Tras recibir las preferencias por parte de los agentes expertos, el agente moderador envía un mensaje ACL-Request al agente evaluador de consenso, cuyo contenido es una instancia de esta acción de agente. Esto se hace en el comportamiento *ConsensusCalcRequest* del agente moderador. En el siguiente ejemplo mostramos parte del contenido de esta acción.

```
(RateConsensus :maxRounds 10 :numAlt 4 :opSim (OpSimilarity
:simOperator Euclidean) :consensus (Consensus :currentConsensus
0.7089 :consensusMatrix (sequence 0.0) :consensusThreshold 0.75)
:opAgg (OpAggregation :aggOperator WeightedMean) :setOfExperts
(sequence (ExpertAgent :aid <IDENTIFICADORES DE LOS AGENTES EXPERTOS>
)) :round 3 :setOfPreferences (sequence (PreferenceRelation
:assessments (sequence 0.18375000000000002 0.18375000000000002 0.5025
0.7175 0.5525 0.7175 0.7525 0.28375000000000006 0.2525
0.5025 0.35250000000000004 0.6525))
<RESTO DE PREFERENCIAS DE EXPERTOS>))
```

5. Predicado *ProvideConsensus*: Se utiliza como contenido del mensaje ACL-Inform con el

que el agente evaluador de consenso responde al agente moderador, una vez calculado el grado de consenso en la actual ronda. Este predicado es creado en el comportamiento *ConsensusCalcRequestHandler*, y su contenido sigue la estructura mostrada a continuación:

```
((ProvideConsensus (Consensus :currentConsensus 0.7631399
:consensusMatrix (sequence
-1.0 0.728705357142857 0.7678571428571429 0.758169642857143
0.7615624999999999 -1.0 0.7011160714285712 0.8450892857142857
0.7582589285714286 0.7386160714285717 -1.0 0.7760267857142856
0.7270982142857145 0.8029017857142858 0.7922767857142857 -1.0)
:consensusThreshold 0.75)))
```

6. Acción *RateRecommendation*: Si durante la etapa de control del grado de consenso, el agente moderador determina que este no es suficiente, deberá preparar un mensaje ACL-Request destinado al agente identificador de cambios con la información necesaria para que este le devuelva un conjunto de sugerencias de cambio. El contenido de dicho mensaje es una instancia de esta acción de agente, la cual se crea y prepara para su envío en el comportamiento *RecommendationRequest*. Veamos un ejemplo resumido de esta acción de agente:

```
(RateRecommendation :numAlt 4 :opSim (OpSimilarity
:simOperator EuclideanDistance) :consensus (Consensus
:currentConsensus 0.7089658 :consensusMatrix (sequence
<VALORES DE LA MATRIZ DE CONSENSO> )
:consensusThreshold 0.75) :setOfExperts
<IDENTIFICADORES DE LOS AGENTES EXPERTOS>
 :setOfPreferences <PREFERENCIAS DE LOS EXPERTOS> )
```

7. Predicado *ProvideRecommendation*: Este último predicado supone el contenido del mensaje ACL-Inform que el agente identificador de cambios da como respuesta al

agente moderador, e incluye las recomendaciones de cambio sugeridas a los expertos. Es creado en el comportamiento *RecommendationRequestHandler* y su estructura es análoga a la del siguiente ejemplo, el cual aparece resumido debido a que este predicado puede llegar a tener una gran extensión:

```
((ProvideRecommendation (sequence 7 4 3 5 6 4 2 3)
(sequence (Recommendation :numAtl2 1 :numAlt1 0
:aid <IDENTIFICADOR DEL AGENTE EXPERTO>
 :direction Increase) <RESTO DE RECOMENDACIONES>)
(sequence (ExpertAgent <...>) <AGENTES EXPERTOS>)))
```

3.4. Ejemplo Ilustrativo

Para finalizar el tercer capítulo de la memoria de investigación, en esta sección mostraremos de forma breve un ejemplo de aplicación del sistema que hemos presentado para ilustrar el funcionamiento del mismo, mediante la simulación de un proceso de consenso por parte de un grupo de expertos que presentan diferentes perfiles de comportamiento.

Hemos considerado el siguiente problema de TDG: *Un comité de congreso compuesto por 10 científicos desea conceder un premio al mejor paper presentado, entre cuatro posibles candidatos (x_1: paper de John, x_2: paper de Li, x_3: paper de Vladimir, y x_4: paper de Iván). Los miembros del comité desean alcanzar un alto nivel de acuerdo antes de tomar la decisión.*

A continuación enumeramos los datos y parámetros utilizados en el problema de TDG y el proceso de consenso:

- Un conjunto E formado por 10 expertos, $E = \{e_1, \ldots, e_{10}\}$.
- Un conjunto X formado por 4 alternativas, $X = \{x_1, x_2, x_3, x_4\}$.
- Perfiles de cambio adoptados por los expertos: 4 indiferentes, 3 seguros y 3 inseguros.

- Umbral de consenso que se desea alcanzar, $\mu = 0.85$.
- Número máximo de rondas permitido, $Maxrounds = 10$.
- Límite superior de cambio por ronda, $L = 0.2$.
- Elección de la *media aritmética* como operador de agregación durante todo el proceso.

Por otra parte, para cada perfil de cambio hemos establecido las siguientes funciones de cambio, a partir de los parámetros L, $Maxrounds$ y el número de ronda de consenso en la que se encuentre el grupo, n:

- Función de cambio para perfil indiferente:

$$\Delta_i^{lk}(n) = \frac{L}{2} = 0.1 \tag{3.12}$$

- Función de cambio para perfil seguro:

$$\Delta_i^{lk}(n) = L\left(\frac{n}{Maxrounds}\right)^3 = 0.2\left(\frac{n}{10}\right)^3 \tag{3.13}$$

- Función de cambio para perfil inseguro:

$$\Delta_i^{lk}(n) = L\left(1 - \left(\frac{n}{Maxrounds}\right)^3\right) = 0.2\left(1 - \left(\frac{n}{10}\right)^3\right) \tag{3.14}$$

Una vez introducidos los anteriores parámetros y funciones de cambio en nuestro SMA, hemos llevado a cabo la simulación del proceso de consenso, obteniendo los resultados que se muestran en la Tabla 3.1 y que pasamos a describir a continuación:

- cr: Grado de consenso global alcanzado en cada ronda.
- *Total cambios*: Número total de recomendaciones de cambio sugeridos por el agente moderador en una ronda, cada uno de ellos sobre un experto y valoración determinados.
- *C. realizados*: Número total de recomendaciones de cambio llevadas a cabo por los agentes expertos, ya sea de forma autónoma o supervisada.

- *Num. sup.*: Número de veces que un agente experto ha solicitado la supervisión del experto humano correspondiente para modificar una valoración, debido a que dicha modificación supone un cambio en la alternativa preferida.
- *Aceptadas*: Número de recomendaciones supervisadas aceptadas por el experto humano.
- *Rechazadas*: Número de recomendaciones supervisadas rechazadas por el experto humano.

Tabla 3.1: Resultados obtenidos al realizar un proceso de consenso en COMAS

Ronda	*cr*	Total cambios	C. realizados	Num. sup.	Aceptadas	Rechazadas
1	0,6422	54	54	0	0	0
2	0,7194	56	53	4	1	3
3	0,7715	54	51	6	3	3
4	0,8147	38	35	8	5	3
5	0,8385	22	18	8	4	4
6	**0,8521**					

Los resultados muestran cómo el grado de consenso global en el grupo aumenta progresivamente hasta alcanzar el umbral en la sexta ronda. Además, el número de recomendaciones de cambio sugeridas por el agente moderador tiende a disminuir a medida que las opiniones de los expertos están más cercanas entre sí. La mayor parte de las recomendaciones sobre una valoración no suponen un cambio en la alternativa que se prefiere, por lo que solamente es necesario un número muy reducido de supervisiones humanas durante el proceso, que suele ser menor al comienzo del mismo y aumentar levemente en las últimas rondas, cuando ya se han realizado más cambios sobre las valoraciones.

Una vez mostrado el anterior ejemplo de aplicación del sistema, nos planteamos la hipótesis que el perfil de cambio adoptado por los expertos afecta al proceso de consenso, más concretamente, a la convergencia hacia el nivel de acuerdo necesario y el número de rondas necesario para alcanzarlo. Por ello, decidimos realizar otras tres simulaciones con la misma configuración anteriormente descrita, pero variando los perfiles de cambio de los expertos, de manera que en cada simulación todos los expertos siguen el mismo perfil.

Así, para un grupo compuesto por 10 expertos con un perfil de cambio indiferente se obtuvieron los resultados mostrados en la Tabla 3.2, para un grupo de 10 expertos con perfil seguro obtuvimos los resultados mostrados en la Tabla 3.3, y para 10 expertos con un perfil inseguro obtuvimos los resultados mostrados en la Tabla 3.4. Por último, los gráficos representados en las Figuras 3.14 y 3.15 muestran, respectivamente, el número de rondas de discusión necesarias y la evolución del número de cambios sugeridos en cada ronda, para cada una de las cuatro resoluciones del problema llevadas a cabo.

A partir de estos resultados, se concluye que los expertos con un perfil de cambio seguro propician una menor convergencia hacia el consenso, siendo necesario un mayor número de rondas de discusión, debido a que realizan cambios muy pequeños en sus valoraciones durante las primeras rondas. Por el contrario, los expertos con un perfil inseguro propician mayor convergencia hacia el consenso, que se alcanzará tras un número de rondas menor, ya que dichos expertos realizan grandes cambios en sus valoraciones al comienzo del proceso. Por último, en el grupo de expertos con perfil indiferente, observamos una convergencia hacia el consenso moderada, que vendrá determinada por la función de cambio utilizada.

Tabla 3.2: Resultados obtenidos para un grupo de expertos con perfil indiferente

Ronda	cr	Total cambios	C. realizados	Num. sup.	Aceptadas	Rechazadas
1	0,6099	60	60	0	0	0
2	0,6878	56	56	0	0	0
3	0,758	62	57	8	3	5
4	0,8214	38	38	2	2	0
5	**0,8595**					

Tabla 3.3: Resultados obtenidos para un grupo de expertos con perfil seguro

Ronda	*cr*	Total cambios	C. realizados	Num. sup.	Aceptadas	Rechazadas
1	0,6752	64	64	0	0	0
2	0,6767	64	63	2	1	1
3	0,6811	62	62	2	2	0
4	0,6918	61	61	1	1	0
5	0,712	55	55	1	1	0
6	0,7448	61	61	0	0	0
7	0,7973	50	48	4	2	2
8	**0,858**					

Tabla 3.4: Resultados obtenidos para un grupo de expertos con perfil inseguro

Ronda	*cr*	Total cambios	C. realizados	Num. sup.	Aceptadas	Rechazadas
1	0,6865	58	58	4	4	0
2	0,8245	30	24	10	4	6
3	**0,8768**					

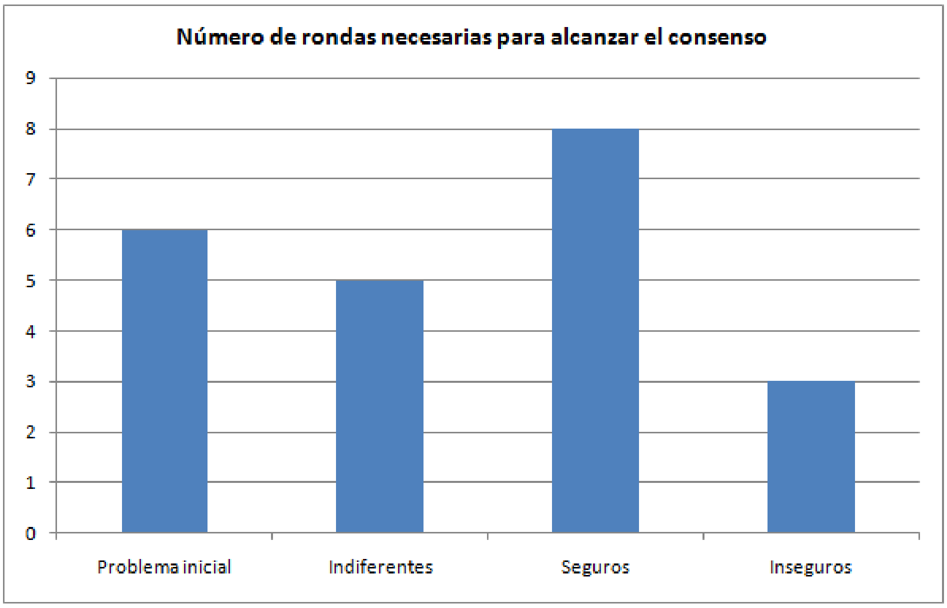

Figura 3.14: Número de rondas necesarias para llegar al consenso

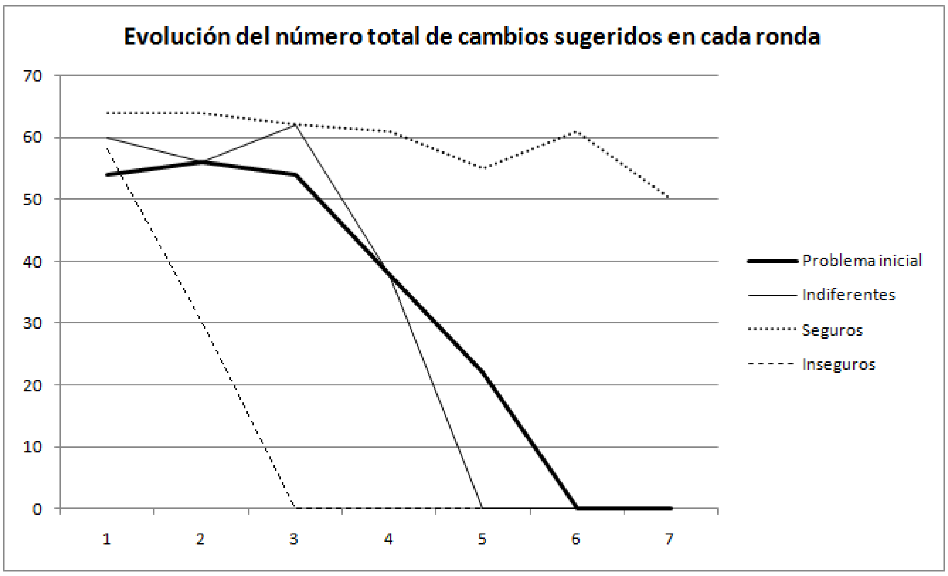

Figura 3.15: Evolución del número de cambios sugeridos durante el proceso

Capítulo 4

Conclusiones y Trabajos Futuros

Este capítulo cierra la memoria del Trabajo Fin de Master con las principales conclusiones extraídas a partir de la misma, revisando la principal propuesta del trabajo y los resultados obtenidos. Además, finalizaremos presentando las líneas de investigación y trabajos futuros que nos hemos planteado abordar a partir de estos resultados, algunos sobre los cuales ya estamos trabajando en la actualidad.

Propuesta y resultados obtenidos

Los procesos de consenso reales en entornos organizativos y/o empresariales implican la participación de múltiples decisores o expertos, así como de una persona encargada de guiarlos, llamada moderador.

Dado que llevar a cabo un proceso de consenso real no suele resultar sencillo, debido al elevado coste temporal en muchas ocasiones, así como la necesidad de llevar a cabo una reunión y la asistencia a la misma por parte de todos los expertos, al comienzo de esta memoria nos planteamos el objetivo de desarrollar un sistema destinado a la automatización de estos procesos. Esta automatización implica un coste computacional tanto mayor cuantos más expertos participen en el problema, razón por la cual decidimos utilizar la tecnología de *Sistemas Multi-Agente*, capaz de dar soporte a procesos de consenso mediante un elevado número de agentes inteligentes de forma distribuida.

Teniendo en cuenta la elección de la tecnología de sistemas multi-agente para llevar a cabo esta automatización, nuestra meta ha sido la de construir un SAC (Sistema de Apoyo al Consenso) combinado con un Sistema Multi-Agente (SMA) en el que un conjunto de agentes inteligentes con roles determinados se encarguen de realizar un proceso de consenso completo, desde la expresión de las preferencias iniciales por parte del conjunto de expertos hasta el momento en que se alcance un acuerdo entre estos. Estos agentes se comunican, expresan y analizan la información utilizada durante el proceso con la ayuda de una *ontología*, diseñada para el manejo de información en procesos de TDG mediante consenso. Todo ello ha sido realizado con vistas a utilizar nuestro sistema basado en agentes en problemas de consenso con gran demanda computacional. La versión actual es un prototipo diseñado con el principal objetivo de estudiar la viabilidad en este tipo de sistemas.

Los resultados obtenidos a partir del objetivo planteado se pueden resumir en los siguientes puntos:

- El sistema propuesto consigue automatizar los procesos de consenso, reemplazando así la figura del moderador humano. La única responsabilidad de los expertos humanos es la de proporcionar sus preferencias al sistema y, de forma ocasional, supervisar aquellas recomendaciones de cambio que impliquen un cambio importante en sus preferencias.

- La arquitectura multi-agente permite la existencia de un entorno, centralizado o distribuido, en el que un grupo de agentes cooperan y se comunican para lograr un fin común: alcanzar un acuerdo para todos los expertos.

- La ontología diseñada proporciona además un dominio de conocimiento común para todos los agentes, compartiendo así todos ellos el mismo lenguaje y semántica durante todos los actos comunicativos que llevan a cabo entre sí.

- La implementación de diversos perfiles de cambio para los agentes expertos permite dotar al sistema de un alto grado de autonomía mediante una gestión semi-supervisada de los cambios, de forma que la necesidad de supervisión directa del experto humano durante el proceso es mínima.

Trabajos Futuros

Este trabajo ha presentado una primera versión de COMAS, un SMA para dar soporte a procesos de consenso. Teniendo en cuenta la importancia actual del consenso en el área de TDG, así como las múltiples líneas de ampliación y mejora de nuestro sistema, nuestros trabajos futuros pretenden abordar las siguientes líneas de investigación (algunas de las cuáles actualmente ya se encuentran en desarrollo):

1. De cara al uso on-line del sistema por parte de diferentes expertos humanos, pretendemos desarrollar una Interfaz Web distribuida, basada en la tecnología de Servicios Web, de manera que diferentes expertos humanos situados en distintas localizaciones en un momento dado pueda expresar sus preferencias iniciales a traves de Internet, y así poder llevarse a cabo procesos automáticos de consenso "a distancia".

2. Profundizar en los actuales desafíos y debilidades encontrados en los procesos de consenso, especialmente aquellos problemas relacionados con el comportamiento social de algunos expertos en el grupo. Así, además de los diferentes perfiles de cambio, pretendemos abordar el estudio e integración en el sistema de diferentes mecanismos de negociación entre agentes, que ayuden a dotar al proceso de discusión de un mayor realismo.

3. Incorporar diferentes modelados de preferencias de los expertos, en problemas definidos en entornos de incertidumbre, así como nuevas estructuras de preferencia que faciliten la expresión de opiniones entre los expertos de un mismo grupo.

4. Integrar la idea de *actitud* del grupo hacia el consenso, mediante la cual el grupo decidirá, en función de las necesidades de los expertos y/o de las características del problema de TDG a resolver, de qué forma desean considerar las diferentes posiciones de acuerdo entre ellos a la hora de medir el consenso.

5. De forma general, nuestra principal meta de cara al futuro consiste en aplicar COMAS a entornos de elevada demanda computacional, donde el consenso sobre información o toma de decisiones sea un factor crítico. Un posible ejemplo de aplicación son los procesos de democracia electrónica (*e*-Democracia).

Bibliografía

[1] M. Adellinde and D.W. Uhrmacher. *Multi-Agent Systems: Simulations and Applications*. CRC Press, 2009.

[2] F. Alonso, J.L. Fuertes, L. Martínez, and H. Soza. Measuring the social ability of software agents. *6th International Conference on Software Engineering Research, Management and Applications*, 3-10, 2008.

[3] S. Alonso, E. Herrera-Viedma, F. Chiclana, and F. Herrera. Individual and social strategies to deal with ignorance situations in multi-person decision making. *International Journal of Information Technology and Decision Making*, 8(2):313–333, 2009.

[4] S. Alonso, I.J. Pérez, F.J. Cabrerizo, and E. Herrera-Viedma. A fuzzy group decision making model for large groups of individuals. *Proceedings of the IEEE International Conference on Fuzzy Systems (Fuzz-IEEE09)*, pages 643–648, 2009.

[5] J. Bajo, J.M. Corchado, C. Pinzón, Y. Paz, and B. Pérez-Lancho. Scmas: A distributed hierarchical multi-agent architecture for blocking attacks to databases. *International Journal of Innovative Computing, Information and Control*, 6(9):3787–3817, 2010.

[6] J. Bajo, J.A. Fraile, B. Pérez-Lancho, and J.M. Corchado. The thomas architecture in home care scenarios: A case study. *Expert Systems with Applications*, 37(5):3986–3999, 2010.

[7] F. Bellifemine, G. Caire, and D. Greenwood. *Developing Multi-agent systems with JADE*. Wiley, 2007.

[8] D. Ben-Arieh and Z. Chen. Linguistic labels aggregation and consensus measure for automatic decision-making using group recommendations. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 36(1):558–568, 2006.

[9] D. Ben-Arieh, T. Easton, and B. Evans. Minimum cost consensus with quadratic cost functions. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 39(1):210–217, 2009.

[10] J.C. Bezdek, B. Spillman, and R. Spillman. A fuzzy relation space for group decision theory. *Fuzzy sets and Systems*, 1(4):255–268, 1978.

[11] G. Bordogna, M. Fedrizzi, and G. Pasi. A linguistic modeling of consensus in group decision making based on owa operators. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 27(1):126–133, 1997.

[12] M.E. Bratman, D.J. Israel, and M.E. Pollack. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4(4):1–22, 1988.

[13] M.E. Bratman, D.J. Israel, and M.E. Pollack. Intention, practical rationality and self-governance. *Ethics*, 119(1):411–443, 2009.

[14] R. Brooks. Intelligence without representation. *Artificial Intelligence*, 47(1):139–159, 2001.

[15] T.X. Bui. *A group decision support system for cooperative multiple criteria group decision making*. Springer-Verlag, 1987.

[16] C.T.L. Butler and A. Rothstein. *On Conflict and Consensus: A Handbook on Formal Consensus Decision Making*. Takoma Park, 2006.

[17] F.J. Cabrerizo, S. Alonso, I.J. Pérez, and E. Herrera-Viedma. On consensus measures in fuzzy group decision making. *Lecture Notes in Computer Science*, 5285:86–97, 2008.

[18] F.J. Cabrerizo, I.J. Pérez, and E. Herrera-Viedma. Managing the consensus in group decision making in an unbalanced fuzzy linguistic context incomplete information. *Knowledge-Based Systems*, 23(2):169–181, 2010.

[19] F.J. Cantú and H.G. Ceballos. A multiagent knowledge and information network approach for managing research assets. *Expert Systems with Applications*, 37(7):5272–5284, 2010.

[20] J. Carbó, J.M. Molina, and J. Davila. A bdi agent architecture for reasoning about reputation. *IEEE Systems, Man and Cybernetics Conference*, 2(1):817–822, 2001.

[21] C. Carlsson, D. Ehrenberg, P. Eklund, M. Fedrizzi, P. Gustafsson, P. Lindholm, G. Merkuryeva, T. Riissanen, and A.G.S. Ventre. Consensus in distributed soft environments. *European Journal of Operational Research*, 61(1-2):165–185, 1992.

[22] F. Castanedo, J.M. Molina, J. García, and J.M. Molina. Extending surveillance systems capabilities using bdi cooperative sensor agents. *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, 1(1):131–138, 2006.

[23] A. Chávez and P. Maes. Kasbah: An agent marketplace for buying and selling goods. *Proceedings of the 1st International Conference on the Practical Appication of Intelligent Agents and Multi-Agent Technology*, pages 75–90, 1996.

[24] W. Cook and L. Seiford. Priority ranking and consensus formation. *Management Science*, 24(16):1721–1732, 1978.

[25] J.M. Corchado, J.A. Fraile, J. Bajo, and B. Pérez-Lancho. Hoca: Multiagent architecture for developing intelligent home care environments. *International Symposium on Distributed Computing and Artificial Intelligence*, 50(1):52–61, 2009.

[26] Y. Demazeau. *La méthode VOYELLES dans Systèmes Multi-Agents: Des Theories Organisationnelles aux Applications Industrielles*. Hermès, Oslo, 2001.

[27] Real Academia Española. Diccionario de la lengua española. vigésimo segunda edición, 2001.

[28] M. Fedrizzi, M. Fedrizzi, and R.A.M. Pereira. Soft consensus and network dynamics in group decision making. *International Journal of Intelligent Systems*, 14(1):63–77, 1999.

[29] I.A. Ferguson. Towards an architecture for adaptive, rational, mobile agents. *Proceedings of the Third European Workshop on Modelling Autonomous Agents and Multi-Agent World*, pages 249–262, 1991.

[30] FIPA. Fundation for intelligent physical agents, website: http://www.fipa.org.

[31] J.L. García-Lapresta. Some consensus measures and their applications in group decision making. *FLINS 2008. The 8th International FLINS Conference on Computational Intelligence in Decision and Control*, pages 611–616, 2008.

[32] F. García-Sánchez, R. Valencia, R. Martínez, and J.T. Fernández. An ontology, intelligent agent-based framework for the provision of semantic web services. *Expert Systems with Applications*, 36(2):3167–3187, 2009.

[33] F.J. Garijo. Tecnología de agentes: Experiencias y perspectivas para el desarrollo de nuevos servicios y aplicaciones. *Bole.tic*, 24:1–9, 2002.

[34] M. Georgeff, B. Pell, M.E. Pollack, M. Tambe, and M. Wooldridge. The belief-desire-intention model of agency. *Lecture Notes in Computer Science*, 1555:1–10, 1998.

[35] L. Hamer, Y. Hemeryck, G. Herweyers, M. Janssen, H. Keters, R. Rousseau, and A. Vanhoutte. Similarity measures in scientometric research: The jaccard index versus salton's cosine formula. *Information Processing and Management*, 25(3):315–318, 1989.

[36] F. Herrera and E. Herrera-Viedma. Linguistic decision analysis: Steps for solving decision problems under linguistic information. *Fuzzy Sets and Systems*, 115(2000):67–82, 2000.

[37] F. Herrera, E. Herrera-Viedma, and J. Verdegay. A sequential selection process in group decision making with linguistic assessments. *Information Sciences*, 85(1995):223–239, 1995.

[38] F. Herrera, E. Herrera-Viedma, and J. Verdegay. A model of consensus in group decision making under linguistic assessments. *Fuzzy sets and Systems*, 78(1):73–87, 1996.

[39] F. Herrera, E. Herrera-Viedma, and J.L. Verdegay. A rational consensus model in group decision making using linguistic assessments. *Fuzzy Sets and Systems*, 88(1):31–49, 1997.

[40] E. Herrera-Viedma, S. Alonso, F. Chiclana, and F. Herrera. A consensus model for group decision making with incomplete fuzzy preference relations. *IEEE Transactions on Fuzzy Systems*, 15(5):863–877, 2007.

[41] E. Herrera-Viedma, F. Chiclana, F. Herrera, and S. Alonso. Group decision-making model with incomplete fuzzy preference relations based on additive consistency. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 37(1):176–189, 2007.

[42] E. Herrera-Viedma, F. Herrera, and F. Chiclana. A consensus model for multiperson decision making with different preference structures. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 32(3):394–402, 2002.

[43] E. Herrera-Viedma, L. Martínez, F. Mata, and F. Chiclana. A consensus support system model for group decision making problems with multigranular linguistic preference relations. *IEEE Transactions on Fuzzy Systems*, 13(5):644–658, 2005.

[44] M.N. Huhns and M.P. Singh. Cognitive agents. *Internet Computing*, 2(6):87–89, 1998.

[45] J. Jung. Ontological framework based on contextual mediation for collaborative information retrieval. *Information Retrieval*, 10(1):85–109, 2006.

[46] J. Kacprzyk. Group decision making with a fuzzy linguistic majority. *Fuzzy Sets and Systems*, 18(2):105–118, 1986.

[47] J. Kacprzyk. On some fuzzy cores and 'soft' consensus measures in group decision making. *The Analysis of Fuzzy Information*, pages 119–130, J.C. Bezdek, 1987.

[48] J. Kacprzyk and M. Fedrizzi. A "soft" measure of consensus in the setting of partial (fuzzy) preferences. *European Journal on Operational Research*, 34(1):316–325, 1988.

[49] J. Kacprzyk, M. Fedrizzi, and H. Nurmi. Group decision making and consensus under fuzzy preferences and fuzzy majority. *Fuzzy Sets and Systems*, 49(1):21–31, 1992.

[50] J. Kacprzyk, M. Fedrizzi, and H. Nurmi. *"Soft" Degrees of Consensus Under Fuzzy Preferences and Fuzzy Majorities.* Kluwer Academic Publishers, 1997.

[51] J. Kacprzyk and S. Zadrozny. On a concept of a consensus reaching process support system based on the use of soft computing and web techniques. *Proceedings of the 8th International FLINS Conference*, pages 859–864, 2008.

[52] J. Kacprzyk and S. Zadrozny. Towards a general and unified characterization of individual and collective choice functions under fuzzy and nonfuzzy preferences and majority via the ordered weighted average operators. *International Journal of Intelligent Systems*, 24(1):4–26, January 2009.

[53] J. Kacprzyk and S. Zadrozny. Soft computing and web intelligence for supporting consensus reaching. *Soft Computing*, 14(8):833–846, 2010.

[54] J.K. Kim, S.H. Choi, C.H. Han, and S.H. Kim. An interactive procedure for multiple criteria group decision making with incomplete information. *Computers and Industrial Engineering*, 35(1-2):295–298, 1998.

[55] G.J. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications.* Prentice Hall, 1995.

[56] L.I. Kuncheva and R. Krishnapuram. A fuzzy consensus aggregation operator. *Fuzzt Sets and Systems*, 79(3):347–356, 1995.

[57] H.M. Lee. Generalization of the group decision making using fuzzy sets theory for evaluating the rate of aggregative risk in software development. *Information Sciences*, 113(3-4):301–311, FEB 1999.

[58] R.D. Luce and H. Raiffa. *Games and Decisions: Introduction and Critical Survey*. Dover Publications, 1989.

[59] P. Maes. Situated agents can have goals. *Robotics and Autonomous Systems*, 6(1-2):49–70, 1990.

[60] K. Mahesh and S. Nirenburg. Principles of ontological engineering for natural language processing. *European Conference on Artificial Intelligence, ECAI-96*, 1996.

[61] L. Martínez and J. Montero. Challenges for improving consensus reaching process in collective decisions. *New Mathematics and Natural Computation*, 3(2):203–217, 2007.

[62] A. Mas. *Agentes Software y sistemas multi-agente: Conceptos, arquitecturas y aplicaciones*. Pearson Education, 2004.

[63] F. Mata, L. Martínez, and E. Herrera-Viedma. An adaptive consensus support model for group decision-making problems in a multigranular fuzzy linguistic context. *IEEE Transactions on Fuzzy Systems*, 17(2):279–290, 2009.

[64] M.Delgado, J.L. Verdegay, and M.A. Vila. A model for linguistic partial information in decision making problem. *International Journal of Intelligent Systems*, 9(5):365–378, 1994.

[65] S.E. Middleton, N.R. Shadbolt, and D.C. de Roure. Ontological user profiling in recommender systems. *ACM Transactions on Information Systems*, 22(1):54–88, 2004.

[66] J.P. Muller, M. Pischel, and M. Thiel. Modelling reactive behaviour in vertically layered agent architectures. *Intelligent Agents: Theories, Architectures and Languages (LNAI 890)*, pages 261–276, 1995.

[67] S. Nolfi. Power and the limits of reactive agents. *Neurocomputing*, 42(1):119–145, 2002.

[68] D.L. Poole and A.K. Mackworth. *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press, 2010.

[69] A.S. Rao and M.P. Georgeff. Bdi agents: From theory to practice. *Proceedings of the First International Conference on Multi-Agent Systems, ICMAS-95*, 1995.

[70] S. Rios, C. Bielza, and A. Mateos. *Fundamentos de los sistemas de ayuda a la decisión*. RA-MA, 2001.

[71] D. Rosaci and G.M.L. Sarné. Masha: A multi-agent system handling user and device adaptivity of web sites. *User Model User-Adap Inter*, 16(5):435–462, 2006.

[72] M. Roubens. Fuzzy sets and decision analysis. *Fuzzy Sets and Systems*, 90(2):199–206, 1997.

[73] S. Saint and J. R. Lawson. *Rules for Reaching Consensus. A Modern Approach to Decision Making*. Jossey-Bass, 1994.

[74] J. Searle. *Speech Acts*. Cambridge University Press, 1969.

[75] K. Shin. Software agents metrics. a preliminary study and development of a metric analyzer. *Project Report No. H98010, Dept. Computer Science, National University of Singapore*, 2003/2004.

[76] R. So and L. Sonenberg. Situation awareness in intelligent agents: Foundations for a theory of proactive agent behavior. *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, 86-92, 2004.

[77] S. Staab and R. Studer. *Handbook on Ontologies (International Handbooks on Information Systems)*. Springer, 2009.

[78] M. Wooldridge. *An Introduction to Multi-Agent Systems*. John Wiley & Sons, Inc., 2002.

[79] R.R. Yager. On orderer weighted averaging aggregation operators in multi-criteria decision making. *IEEE Transactions on Systems, Man a Cybernetics*, 18(1):183–190, 1988.

[80] R.R. Yager. Non-numeric multi-criteria multi-person decision making. *Group Decision and Negotiation*, 2(1):81–93, 1993.

[81] R.R. Yager. Fusion of ordinal information using weighted mean aggregation. *International Journal of Approximate Reasoning*, 12(1-2):35–52, 1998.

[82] L. Zadeh. A computational approach to fuzzy quantifiers in natural languages. *Computing and Mathematics with Applications*, 9:149–184, 1983.

[83] L.A. Zadeh. Fuzzy logic equals computing with words. *IEEE Transactions on Fuzzy Systems*, 4(2):103–111, 1996.

[84] S. Zadrozny and J. Kacprzyk. An internet-based group decision and consensus reaching support system. *Applied Decision Support with Soft Computing (Studies in Fuzziness and Soft Computing)*, 124:263–275, Springer, 2003.

[85] X.F. Zha, S.Y.E. Lim, and W.F. Lu. A knowledge intensive multi-agent framework for cooperative/colaborative design modeling and decision support of assemblies. *Journal of Integrated Desifn and Process Science*, 7(1):99–122, 2002.

Publicaciones Relacionadas

- *F. Mata, P.J. Sánchez, I. Palomares, F.J. Quesada, L. Martínez*, **COMAS: A Consensus Multi-Agent based System**, Proceedings of the 10th International Conference on Intelligent Systems Design and Applications (ISDA), Cairo, Egypt, pp. 457-462, 2010.

- *I. Palomares, P.J. Sánchez, F.J. Quesada, F. Mata, L. Martínez*, **COMAS: A Multiagent System for Performing Consensus Processes**, in A. Abraham et al. (Eds.), International Symposium on Distributed Computing and Artificial Intelligence. Advances in Intelligent and Soft Computing, 91, pp. 125-132, Springer, 2011.