Modelo de asignación dinámica de roles y distribución de tareas en organizaciones virtuales

Tesis Doctoral

VNiVERSiDAD D SALAMANCA

Autora

Ana de Luis Reboredo

Departamento de Informática y Automática

Directores

Dr. D. Juan Manuel Corchado Rodríguez

Dr. D. Juan Francisco de Paz Santana

Enero 2014

La memoria titulada “Modelo de asignación dinámica de roles y distribución de tareas en organizaciones virtuales” que presenta Dña. Ana de Luis Reboredo para optar al Grado de Doctor en Informática y Automática por la Universidad de Salamanca ha sido realizada bajo la dirección del profesor Dr. Juan Manuel Corchado Rodríguez, Catedrático del Departamento de Informática y Automática de la Universidad de Salamanca, y el profesor Dr. Juan Francisco De Paz Santana, Profesor Ayudante Doctor del Departamento de Informática y Automática de la Universidad de Salamanca.

Salamanca, Enero de 2014

Los directores

Fdo: Dr. Juan Manuel Corchado Rodríguez
Catedrático de Universidad
Universidad de Salamanca

Fdo: Dr. Juan Francisco De Paz Santana
Profesor Ayudante Doctor
Universidad de Salamanca

El doctorando

Fdo: Ana de Luis Reboredo
Profesora Titular de Escuela Universitaria
Universidad de Salamanca

Resumen

Las organizaciones virtuales suponen un enfoque en el modelado de sistemas multiagente en el que se tienen en cuenta los aspectos organizativos más allá de las simples capacidades de comunicación de los agentes. En las organizaciones virtuales están implícitos una serie de objetivos globales de la organización que se superponen a los objetivos específicos de cada agente y que existen de manera independiente de estos. La estructura de la organización se define en base a grupos y roles.

Este trabajo se propone definir un modelo de planificación de tareas en organizaciones virtuales que garantice la consecución de los objetivos de la organización de manera óptima, ajustando la operación de la organización a los recursos disponibles. Para ello, se diseñará un modelo que permita la adaptación dinámica de la estructura de la organización a las necesidades y medios existentes en cada caso. Esta adaptación consistirá en la determinación del número de agentes que deben incorporarse a la organización y en la asignación de roles a dichos agentes. Posteriormente se llevará a cabo una distribución de tareas entre dichos agentes de manera que se maximicen los beneficios de la organización. El sistema incorporará un modelo de razonamiento basado en casos en el que se aplican algoritmos genéticos para realizar la fase de adaptación.

Se estudiarán las características de los agentes y sistemas multiagente, así como de las organizaciones virtuales. A partir de ahí se realizará el diseño del modelo propuesto que se validará mediante un caso de estudio.

Abstract

Virtual organizations involve a focusing of multi-agent systems which has into account organization aspects, beyond the bare communication capacities of the agents. A series of organization global goals are implicit in virtual organizations, which overlap specific goals of each agent, and exist independently of them. Organization structure is defined on the base of groups and roles.

This study aims to define a task planning model in virtual organizations, which guarantee reaching the organization goals in an optimal way, suiting the organization operation to the resources available. In order to get these purposes, we will design a model allowing a dynamic organization structure adaptation to the needs and media eventually available. This adaptation consists of determining the number of agents that shall join the organization, and assigning roles to these agents. Then, tasks will be distributed to these agents so organization benefits are maximized. The systems will include a case-based reasoning model which applies genetic algorithms, for fulfilling the adaptation phase.

The agents and multi-agent systems features will be studied, and so will be the virtual organizations. Thereafter, we will carry out the design of the proposed model, which will be validated through a study case.

Agradecimientos

Quisiera hacer público mi agradecimiento a los directores de este estudio, sin cuya guía y estímulo no hubiera sido posible llevarlo a cabo. Al profesor Juan Manuel Corchado Rodríguez, de quien partió la idea inicial para la elaboración de este trabajo, por su constante apoyo, y también por su paciencia a lo largo del prolongado proceso de elaboración del mismo. Al profesor Juan Francisco de Paz Santana por su decisivo papel en el desarrollo del estudio y en la elaboración y análisis del manuscrito.

A Juan Luis, por su incansable apoyo y su inagotable insistencia.

A Ana Belén, por su constante estímulo y apoyo en los momentos en los que el desarrollo de trabajo avanzaba más despacio de lo que todos hubiéramos deseado y, sobre todo, porque es muy tranquilizador saber que ella está siempre en la puerta de al lado.

A los miembros del grupo BISITE, en cuyo seno se ha desarrollado este trabajo, por todas las facilidades recibidas de ellos en las diferentes fases del estudio. En especial, quiero agradecer a Carolina Zato su inestimable ayuda.

A todos los miembros del Departamento de Informática y Automática de la Universidad de Salamanca, por los ya largos años de amistosa colaboración en las tareas docentes e investigadoras.

Contenido

Lista de figuras

Capítulo I

Introducción

VNiVERSiDAD
D SALAMANCA

Introducción

El vertiginoso desarrollo que, desde hace décadas, están experimentando los avances tecnológicos ha propiciado que cada vez sea más amplia su implicación en los diversos procesos y ámbitos de acción que afectan a las actividades humanas. Así, para dar una respuesta eficaz a los distintos problemas, se crean sistemas cada vez más complejos, en los que pueden llegar a intervenir gran cantidad de componentes que deben acoplarse de modo que el funcionamiento de todo el conjunto esté perfectamente coordinado y garantice la consecución de los objetivos marcados. Por otra parte, son múltiples los ámbitos en los que los sistemas deben ser capaces de responder adecuadamente en entornos que, por su propia naturaleza, son inestables o cambiantes [Bajo *et al.,* 2009a], [Corchado *et al.,* 2008a]. Por todo ello, resulta interesante enfocar esfuerzos hacia la investigación de sistemas que permitan determinar la estructura óptima de un sistema complejo y su adaptación a las variaciones del entorno.

La tecnología de agentes y sistemas multiagente se ha revelado como una excelente forma de abordar un amplio rango de problemas de computación que implican la toma de decisiones en entornos cambiantes. Los agentes, son entidades que se caracterizan por capacidades como autonomía, reactividad, pro-actividad, comunicación o razonamiento, que les hacen especialmente adecuados para enfrentarse a multitud de situaciones en diversos campos. Podemos describir un agente como un sistema computacional situado en un entorno con el que interactúa de forma autónoma para alcanzar unos objetivos establecidos [Wooldridge, 2009]. Por otra parte, desde hace unos años, ha tomado auge una línea de trabajo que trata de incorporar las técnicas y métodos del diseño de organizaciones humanas al diseño de organizaciones de agentes, facilitando el diseño de sistemas multiagente abiertos [Argente *et al.,* 2008a]. Las organizaciones virtuales pueden adaptar su estructura y funcionamiento a las necesidades derivadas de las evoluciones del entorno. Para ello se requiere contar con un sistema de planificación que permita un uso óptimo de los recursos manteniendo la garantía de prestación de los servicios objeto de la organización virtual.

Los diversos trabajos referenciados en la literatura sobre planificación en sistemas multiagente se concentran especialmente en el establecimiento de sistemas de ordenación de las acciones a realizar para alcanzar un estado final deseado. Sin embargo, esto no permite dotar al sistema de mecanismos de autoajuste dinámico de su estructura al volumen y tipo de los servicios requeridos a lo largo tiempo. El presente trabajo se plantea abordar el modelado de un sistema que permita tratar de forma integrada tanto los aspectos estructurales como los funcionales de una organización virtual.

1.1 Hipótesis y Objetivos

Este trabajo consiste en el diseño de un nuevo modelo de planificación para organizaciones virtuales que, por una parte permita definir la estructura de la organización mediante la asignación dinámica de roles a los componentes de la misma y, por otra, incorpore mecanismos de distribución de tareas de manera que se realice un uso óptimo de los recursos disponibles y la satisfacción de los objetivos de la organización.

Para la consecución de este fin se pueden establecer los siguientes objetivos parciales:

• Identificación y análisis de las distintas tecnologías utilizadas para abordar los problemas de planificación en sistemas multiagente y estudio comparativo de las mismas con el fin de establecer un punto de partida en el trabajo de investigación.

• Formalización del problema de la asignación dinámica de roles y tareas a las entidades de una organización virtual.

• Creación de un nuevo modelo de planificación que permita el ajuste dinámico de la estructura y funcionalidad de una organización virtual a las evoluciones de su entorno.

• Evaluación de las soluciones propuestas mediante su aplicación a casos de estudio seleccionados para entornos con comportamientos altamente dinámicos.

Este trabajo defiende la hipótesis de que es posible definir una arquitectura de sistema multiagente basada en organizaciones virtuales que incorpore un sistema de planificación de tareas en el que se contemple la adaptación de la estructura de la propia organización a los recursos disponibles y a las necesidades existentes de modo que se logre maximizar la eficiencia del sistema.

Para ello, se buscará establecer un diseño de organización virtual que incorpore la definición de una serie de roles de agentes que se encarguen de, a partir de una lista de tareas pendientes y de unos recursos disponibles, establecer el número de agentes, en función de los roles necesarios y realizar la distribución de tareas entre los mismos optimizando la eficiencia del sistema y maximizando los beneficios. El sistema deberá incorporar mecanismos de control que verifiquen que los planes establecidos se mantienen viables a lo largo de su ejecución y, en caso contrario, actúen para realizar las replanificaciones que sean necesarias.

1.2 Motivación

El presente trabajo se sitúa en un campo de investigación en auge en lo que se refiere a sistemas de computación y, en particular, a los sistemas inteligentes. Pretende realizar una aportación al diseño de sistemas multiagente basados en organizaciones virtuales. El planteamiento de un sistema de asignación dinámica de roles y de distribución de tareas que permita la adaptación de la estructura de la organización a las evoluciones del entorno constituye una visión innovadora en el modelado de organizaciones virtuales.

La implicación, cada vez más amplia, de sistemas computacionales en la mayoría de los procesos de cualquier organización, lleva a la necesidad de buscar sistemas que permitan una óptima integración de las unidades de computación en las estructuras organizativas, de modo que el aprovechamiento de los recursos sea máximo y se establezcan estrategias de distribución de las tareas orientadas a lograr los mayores beneficios a partir de dichos recursos.

Un punto fundamental de esta investigación se basa en el estudio de las organizaciones virtuales como soporte a sistemas de gestión de la coordinación de agentes en sistemas multiagente. En este trabajo se estudiará la adaptación de los mecanismos de coordinación de las organizaciones virtuales a los sistemas multiagentes y las posibilidades que ofrecen para gestionar el ajuste dinámico de la estructura de la organización.

1.3 Metodología y plan de trabajo

Para el desarrollo del presente trabajo se aplicará una metodología que permita la realización de una investigación empírica.

Partiendo del estudio de la situación actual se identifica un problema real. A partir de ahí, se formula una hipótesis sobre la que se basará el desarrollo de una propuesta enfocada a la obtención de una solución. A continuación, se ejecutan las acciones necesarias para poder realizar la comprobación de la hipótesis seleccionada mediante la aplicación de la solución propuesta de uno o varios casos de uso que se consideren representativos. Finalmente, se analizan los resultados obtenidos y se formulan las conclusiones de la investigación sobre dicha propuesta.

En aplicación de la metodología de investigación anteriormente descrita, se propone un plan de trabajo compuesto por las siguientes acciones:

• Estudio inicial. En esta fase se realizará una revisión del estado del arte de los distintos campos científicos implicados en el trabajo. En concreto se realizará un estudio sobre características, arquitecturas y metodologías de diseño de agentes y sistemas multiagente, sistemas de planificación y organizaciones virtuales.

• Formalización del problema. Una vez delimitado el alcance del problema que se va a abordar, se formulará este situándolo en el contexto científico que le corresponda y mediante las herramientas formales propias de dicho contexto.

• Diseño de la arquitectura y el modelo de planificación propuesto. Mediante un proceso de diseño y desarrollo incremental, se generará una solución que se refinará de forma iterativa hasta desembocar en una propuesta que satisfaga los objetivos planteados.

• Experimentación mediante la aplicación de la solución propuesta a un caso de estudio. Se verificará la idoneidad de la propuesta mediante la definición de un prototipo que permita simular su aplicación a un caso de estudio representativo del ámbito de la problemática. Se recogerá un conjunto de datos como resultado de esta aplicación que servirán para realizar una evaluación de la solución propuesta.

• Análisis de los resultados y formulación de conclusiones. Se realizará un análisis de los resultados obtenidos en la experimentación con el caso de estudio y se contrastarán estos resultados con la hipótesis formulada, estableciendo las conclusiones del trabajo.

• Difusión de resultados y conclusiones. Las experiencias, conclusiones y avances de la investigación se darán a conocer a la comunidad científica mediante publicaciones en revistas, comunicaciones a congresos o en otros foros que se consideren de interés.

1.4 Estructura de la memoria

El presente documento constituye la memoria del trabajo de investigación y se ha estructurado en capítulos según se describe en este punto.

Este primer capítulo contiene una introducción al trabajo de investigación realizado, describiendo el problema de la adaptación dinámica de la estructura de una organización virtual y su integración con un sistema de planificación de tareas. A continuación se presentan los objetivos del trabajo, se plantea la hipótesis y se argumenta la motivación que ha llevado a la realización del mismo. El capítulo incluye también la descripción de la metodología de trabajo empleada y finaliza con la descripción de la estructura de todo el documento.

Los capítulos 2 y 3 realizan una revisión del estado del arte de los agentes y sistemas multiagente y de las organizaciones virtuales respectivamente, haciendo especial hincapié en aquellos aspectos más relacionados con el desarrollo de este trabajo. En el capítulo 2 se revisan las aportaciones científicas más significativas del ámbito de los agentes poniendo el mayor interés en el análisis de las distintas arquitecturas de agentes propuestas y en el estudio de su aplicación en el ámbito de nuestra investigación. A continuación se incluye un estudio sobre los distintos sistemas de planificación de tareas que se concluye con la revisión de los sistemas de planificación en los sistemas multiagente. El capítulo 3 repasa los avances alcanzados en el terreno de las organizaciones virtuales estudiando especialmente los aspectos estructurales de las mismas y sus implicaciones en la adaptación dinámica de su estructura.

El capítulo 4 contiene la descripción del modelo propuesto con el que se pretende dar respuesta al problema planteado. El modelo incluye una descripción de la arquitectura del sistema y la descripción de los distintos roles que se proponen implementar en la organización virtual. A continuación se describe el modelo de planificación aplicable.

El capítulo 5 presenta los resultados de la aplicación del modelo propuesto a un caso de estudio relacionado con el ámbito de la administración electrónica, así como las conclusiones que se extraen del análisis de dichos resultados.

Las conclusiones de todo el trabajo se recogen en el capítulo 6, junto con las líneas de trabajo que se proponen seguir en el futuro en relación con esta investigación.

Finalmente, se incluye un capítulo recopilatorio redactado en inglés y la relación de referencias bibliográficas.

Capítulo II

Agentes y Sistemas Multiagente

VNiVERSiDAD
D SALAMANCA

Introducción

La necesidad de desarrollar entidades software que hicieran frente a retos cada vez más complejos dio lugar a la búsqueda, de nuevos enfoques de sistemas computacionales. Entre otros, se desarrollaron los conceptos de agente y sistema multiagente que permiten emular características que se consideran propias del ser humano, como la percepción del entorno, la inteligencia, la capacidad de aprender o la capacidad de cooperar con otros entes para alcanzar un objetivo común.

La tecnología de agentes es uno de los pilares en los que se apoya esta investigación. Uno de nuestros objetivos es la obtención de un sistema de planificación de tareas en organizaciones virtuales. Para ello, es necesario partir de un estudio de las características de los sistemas multiagente como constitutivos de organizaciones virtuales y, en particular, de las alternativas disponibles para el establecimiento de sistemas de planificación.

En este capítulo se revisarán los conceptos fundamentales relacionados con la tecnología de agentes y se estudiarán las arquitecturas clásicas que se han aplicado a los mismos. En particular, nos centraremos en el estudio de los mecanismos de razonamiento empleados en las arquitecturas deliberativas, por ser estas las que más se ajustan a los objetivos que nos hemos planteado. A continuación se planteará el problema de la planificación en los agentes y se revisarán aquellas alternativas aplicables en este ámbito que guardan una relación más estrecha con el modelo de planificación que proponemos en este trabajo. Por último, centraremos la atención en estudiar las características de los sistemas multiagente y los sistemas multiagente abiertos, que constituyen el punto central de este trabajo.

2.1 Agentes

Se pueden encontrar múltiples definiciones del concepto de agente. No obstante, no es fácil obtener una definición que permita recoger los distintos puntos de vista del entorno multidisciplinar en el que se ha desarrollado la teoría de agentes y que incluye campos como la Inteligencia Artificial, la Ingeniería de Software, los Sistemas Distribuidos, la Psicología o la Sociología.

Una de las definiciones propuestas, que aparece ampliamente referenciada en la literatura sobre agentes y que se adapta perfectamente al enfoque que se va a dar en este trabajo, es la que define a un agente como un sistema computacional situado en un

entorno y que es capaz de actuar de forma autónoma en ese entorno para alcanzar los objetivos para los que se ha diseñado [Wooldridge, 2009].

Esta definición pretende condensar las principales características que se exigen a un sistema computacional para que pueda ser considerado un agente y para las que sí que existe un consenso en la comunidad científica.

[Wooldridge y Jennings, 1995] enumera y describe las propiedades básicas que acompañan al concepto de agente:

- Autonomía: un agente es capaz de operar sin intervención directa de un humano o de otros agentes y tiene un cierto control sobre sus acciones y estado interno.
- Capacidades sociales: un agente es capaz de interactuar con otros agentes mediante algún lenguaje de comunicación de agentes.
- Reactividad: los agentes perciben su entorno, y responden a los cambios que se producen en él.
- Proactividad: los agentes son capaces tomar iniciativas y de manifestar comportamientos dirigidos por metas.

Otra definición de agentes es la proporcionada por [Labidi y Lejouad, 1993] que define un agente como: entidad física o abstracta que puede percibir su ambiente a través de sensores y que es capaz de evaluar tales percepciones y tomar decisiones por medio de mecanismos de razonamiento sencillos o complejos, comunicarse con otros agentes para obtener información y actuar sobre el medio en el que se desenvuelve a través de ejecutores.

En cualquier caso, los puntos comunes que aparecen en las distintas definiciones manejadas se centran en hacer hincapié en los siguientes puntos: en la capacidad de los agentes para relacionarse con su entorno, recibiendo de él información y pudiendo modificarlo mediante las acciones realizadas por el agente; en la autonomía del agente para decidir sus actuaciones en función de sus mecanismos internos de razonamiento y en la capacidad de comunicación con otros agentes.

A las propiedades anteriores se pueden añadir otras que se suelen considerar necesarias para que los agentes puedan alcanzar sus objetivos de forma satisfactoria:

- Inteligencia: Rodearse de conocimiento (creencias, deseos, intenciones y metas).
- Organización. Organizarse dentro de sociedades que siguen unas estructuras similares a las definidas en sociedades humanas o ecológicas.
- Aprendizaje. Habilidad de adaptarse progresivamente a cambios en entornos dinámicos, mediante técnicas de aprendizaje.

Al igual que ocurre con la definición del concepto de agente, resulta complicado establecer una clasificación que permita diferenciar los tipos de agentes existentes. Distintos autores [Russel y Norvig, 1995], [Nwana, 1995], [Brenner *et al.,* 1998], han identificado clases de agentes estableciendo criterios que se basan en características comunes de los agentes o de los entornos de aplicación. A continuación se muestran dos

de las principales clasificaciones propuestas por dichos autores, aunque sería posible ampliar este punto atendiendo a otros criterios de clasificación como la arquitectura o el diseño conceptual.

1) Clasificación atendiendo a los atributos propios de los agentes

Nwana [1995] indica una serie de criterios a partir de los cuales pueden definirse tipos o clases de agentes, destacando como criterios importantes la movilidad y el tipo de razonamiento utilizado por el agente y considerando el resto de atributos característicos de un agente para distinguir nuevos tipos. De esta forma es posible obtener atributos que determinan clases de agentes.

Movilidad: Habilidad de los agentes para moverse a través de una red de comunicaciones. Atendiendo a este criterio Nwana [1995] distingue:

- Agentes estáticos: permanecen instalados en un sistema red y no tienen capacidad para migrar a otros sistemas.

- Agentes móviles: son capaces de autotrasladarse de una plataforma de agentes a otra o bien de un host a otro host dentro una red [Braun y Rossak, 2005]. El propio agente determina su destino, así como el momento de la migración. Mediante esta capacidad de movilidad logran evitar una sobrecarga de la comunicación, para lo cual el agente se traslada a la máquina en la que reside directamente la información y regresa a su lugar de origen cuando haya finalizado su tarea.

Modelo de razonamiento interno: Representación del conocimiento del agente y forma en que el agente utiliza ese conocimiento para resolver problemas.

- Agentes deliberativos: incorporan el paradigma del pensamiento deliberativo como modelo de razonamiento. Estos agentes realizan una representación simbólica del conocimiento y poseen un modelo de razonamiento sobre ese conocimiento que les permite realizar planificaciones o negociaciones y trabajar de forma coordinada con otros agentes.

- Agentes reactivos: reaccionan ante estímulos de su entorno. Para dar respuesta a un estímulo se basan en el conocimiento actual disponible en el momento en que éste se produce. No disponen de modelos de representación simbólica del conocimiento, con lo que carecen de conocimiento del entorno.

Otros atributos característicos de los agentes: Autonomía, Cooperación y Aprendizaje.

Nwana [1995] considera que pueden definirse otros tipos de agentes atendiendo a los atributos principales de los agentes. Concretamente considera que los tres atributos más importantes a tener en cuenta para completar su clasificación son la autonomía, la cooperación y el aprendizaje. Atendiendo a estos tres atributos se obtienen los cuatro tipos de agentes:

- Agentes colaborativos: Son agentes autónomos que cooperan con otros agentes.

- Agentes interfaz: Son agentes autónomos con capacidad de aprendizaje.
- Agentes de aprendizaje colaborativo: Son agentes que con capacidad de aprendizaje y que trabajan de forma coordinada.
- Agentes smart: Agentes autónomos, con capacidad de aprendizaje y que cooperan con otros agentes tanto para alcanzar sus metas personales como las metas finales del sistema.

2) Clasificación atendiendo al tipo de comportamiento utilizado para implementar las funcionalidades del agente, estableciendo el estado intermedio entre la reactividad y la proactividad. En realidad, esta clasificación simplemente define con mayor nivel de detalle los tipos de agentes según el modelo de razonamiento que utilicen.

[Russell y Norving 2014] proponen una clasificación basada en el tipo de capacidades que implementa el agente. De esta forma se distinguen cuatro tipos de agentes:

- **Agente de reflejo simple**: son agentes que reaccionan ante estímulos, proporcionando una respuesta. Se trata, por tanto, de agentes que no disponen de capacidad de memoria. Este tipo de agentes utiliza reglas de condición-acción para establecer la conexión entre percepciones y acciones. El agente actúa encontrando una regla cuya condición coincida con la situación actual, definida por la percepción, y efectuando la acción que corresponda a tal regla.
- **Agente reflejo con estado interno**: En este caso el agente incorpora, con respecto al agente de reflejo simple, un estado de memoria interno. La capacidad de memoria permite al agente recordar experiencias pasadas cercanas en el tiempo. De esta forma el agente es capaz de elaborar respuestas más adecuadas. El estado interno se actualiza con información de cómo evoluciona el mundo independientemente del agente y de cómo las acciones del agente afectan al mundo [Carrascosa *et al.,* 2008].
- **Agentes basados en metas**: Las incorporación de metas permite dotar a este tipo de agente de capacidades deliberativas. Se trata de un agente que persigue alcanzar objetivos concretos. Para alcanzar cada objetivo el agente necesita información que le permita decidir qué hacer, información que detalle las situaciones deseables. El agente ha de elegir las posibles acciones que le permitan alcanzar su meta, para ello puede utilizar técnicas de búsqueda y planificación [Pokahr *et al.,* 2003].
- **Agentes basados en utilidad**: Un proceso deliberativo dirigido por metas simplemente, no permite que el agente adquiera características propias de la conducta humana tales como la satisfacción o las preferencias personales. La utilidad es una función que a cada estado le asocia su grado de utilidad. La completa especificación de la función de utilidad permite la toma de decisiones racionales cuando satisfacer algunas metas implica un conflicto o

cuando el agente puede desear obtener varias metas y no existe la certeza de lograr ninguna de ellas [Kephart y Walsh, 2004]

Los distintos agentes que se integran en una organización virtual pueden corresponderse con cualquiera de las diversas categorías establecidas en las clasificaciones anteriores. En el modelo que se propone tendrán cabida agentes con distintos modelos de razonamiento interno y de distintas categorías según sus atributos de autonomía, cooperación y aprendizaje. Sin embargo, en este trabajo no se ha tenido en cuanta la consideración de los agentes en lo que a su capacidad de movimiento se refiere. Todo el modelo está planteado suponiendo que los agentes que lo integran son agentes estáticos. No obstante, sería posible plantear una ampliación del modelo que considerase la movilidad de los agentes.

2.2 Arquitecturas de agentes

Como en cualquier sistema computacional, las arquitecturas para la construcción de agentes especifican cómo se descomponen los agentes en un conjunto de módulos que interactúan entre sí para lograr la funcionalidad requerida. Sin embargo, clásicamente, se han diferenciado tres tipos de arquitecturas caracterizadas por el modelo de razonamiento que utilizan [Mas 2005] en lugar de por la representación de los módulos que las componen.

- **Reactivas**. Carecen de razonamiento simbólico complejo y de conocimiento o representación de su entorno. Sus mecanismos de comunicación con otros agentes son muy básicos. Los agentes que utilizan este tipo de arquitectura reciben estímulos de su entorno y reaccionan ante ellos modificando sus comportamientos y el mismo entorno [Maes, 1991].
- **Deliberativas**. Utilizan modelos de representación simbólica del conocimiento, emplean mecanismos de comunicación complejos y contienen un modelo simbólico del entorno. Toman decisiones utilizando razonamiento lógico basado en la manipulación simbólica, partiendo de un estado inicial y un conjunto de planes con un objetivo a satisfacer [Jennings, 1993].
- **Híbridas**. Son arquitecturas que combinan las dos anteriores. Los agentes de este tipo incluyen comportamientos reactivos y deliberativos, generando un ciclo percepción-decisión-acción. El comportamiento reactivo se utiliza para reaccionar ante eventos que no requieran decisiones complejas sobre ciertas acciones.

A continuación se tratarán las características principales de cada una de estas arquitecturas, profundizando especialmente en las arquitecturas deliberativas por ser las que están más implicadas en el desarrollo del presente trabajo.

2.2.1 Reactivas

Las arquitecturas reactivas siguen un enfoque conductivista, con un modelo estímulo-respuesta [Decker *et al.,* 1997]. Las arquitecturas reactivas no mantienen una representación simbólica del mundo como elemento de razonamiento y no utilizan razonamiento simbólico complejo. Utilizan un procesamiento ascendente (bottom-up), para lo cual mantienen una serie de reglas que se activan con ciertas condiciones de los sensores y tienen un efecto directo en los actuadores. Las principales arquitecturas reactivas son:

- Reglas situadas: la implementación más sencilla de reactividad consiste en definir el comportamiento con reglas del tipo si situación-percibida entonces acciones-específicas.
- Arquitecturas de subsunción (subsumption) [Brooks, 1991] y autómatas de estado finito: permiten gestionar problemas de mayor complejidad que las reglas. Las arquitecturas de subsunción están compuestas por capas que ejecutan una determinada conducta (p. ej. explorar, evitar un obstáculo, etc.). La estructura de cada capa es la de una red de topología fija de máquinas de estados finitos. Las capas mantienen una relación de inhibición sobre las capas inferiores (inhibir entradas de los sensores y acciones en los actuadores). El control no es central, sino dirigido por los datos en cada capa.
- Tareas competitivas: un agente debe decidir qué tarea debe realizar de entre varias posibles, seleccionando la que proporciona un nivel de activación mayor.

2.2.2 Deliberativas

Esta arquitectura utiliza modelos de representación simbólica del conocimiento. Tanto el problema como el entorno se describen mediante modelos de representación simbólica. Las acciones del agente vienen determinadas por la aplicación de razonamiento lógico basado en la manipulación simbólica. Un agente deliberativo podría definirse tomando como propuesta la definición de [McCarthy, 1978] de sistema planificador clásico: aquel sistema que, a partir de una base de datos donde se almacenan formulas en algún lenguaje simbólico que representan su conocimiento, opera siguiendo el ciclo “Percepción - Deliberación - Acción”, con un mecanismo de razonamiento y decisión de las acciones que se van a ejecutar basado esencialmente en alguna forma de inferencia lógica.

En lo relativo a la planificación, se basan en la teoría clásica de planificación, partiendo de un estado inicial en el que existen una serie de planes y un estado final al que se quiere llegar. La transición del estado inicial al final se hace siguiendo un sistema

de planificación que permite determinar los planes a ejecutar para así conseguir los objetivos [Maes, 1991].

2.2.2.1 Arquitecturas Intencionales. Modelo BDI

Se utiliza el término sistema intencional para describir entidades cuyo comportamiento puede ser predicho mediante la atribución de creencias, deseos y comportamiento racional al sistema.

Los agentes intencionales se pueden implementar utilizando una arquitectura deliberativa. Estos agentes están basados en una serie de creencias e intenciones que utilizan para razonar [Jennings, 1993]. Entre las arquitecturas intencionales la más ampliamente extendida es la que basa su implementación en el modelo BDI (*Beliefs*, *Desires*, *Intentions*) [Rao, Georgeff, 1995] [Bajo *et al.,* 2006b]. El modelo BDI posee una descripción simbólica del problema mediante la especificación de los siguientes elementos: creencias la información que posee el agente del entorno, deseos las metas y objetivos que quiere alcanzar y finalmente las intenciones las acciones que los agentes pueden llevar a cabo.

Creencias *(Beliefs)***.** El módulo de creencias es considerado como la representación del mundo de la que el agente dispone. La incompletitud de la representación del mundo, a causa de la dinámica propia del mismo, obliga a adjuntar un submódulo de revisión y actualización a esta base, desde el que se ajustan las variaciones que ocurren en el mundo.

Deseos *(Desires)***.** Se pueden entender los deseos como objetivos. Un agente se construye para que desempeñe unas tareas concretas que quedan explicitadas previamente. Así pues, es razonable asociar aquello que el agente pueda desear con aquello para lo que fue construido que puede ser identificado con los objetivos del sistema. Llamaremos objetivos a aquellas situaciones del entorno que se pretende, o bien mantener, o bien lograr. Los objetos que representan los deseos son descripciones del mundo (representaciones análogas a las creencias) cuyo funcionamiento básico es: contrastar si la descripción objetivo se contradice con la descripción actual del mundo; si es así, generar la intención que ponga en marcha un proceso de planificación por el que se logre el estado deseado. Este módulo no debe confundirse con las intenciones, pues con él distinguimos entre lo que se pretende hacer, lograr, alcanzar -los objetivos- y lo que se puede hacer -actitudes intencionales para alcanzar objetivos- dado que los agentes no están dotados de todas las capacidades necesarias para satisfacer por sí mismos todas las tareas.

Intenciones *(Intentions)*. Son un conjunto de caminos de ejecución que pueden requerir ser recalculados

La modularización del diseño de agente realizada, en la que se diferencia el modulo de intenciones del proceso de planificación y acción por un lado, y por el otro, de los módulos de creencia y objetivos, divide claramente la estructura del agente en dos partes diferenciadas. Así, el módulo intencional sería una especie de mecanismo que seleccionaría las intenciones en base a las preferencias del agente, ordenándolas, contrastándolas con las creencias. A partir de aquí, comienza el proceso de planificación que, recibiendo como inputs intenciones, explicita acciones.

Los agentes con la arquitectura BDI tienen sus orígenes en el razonamiento filosófico. Los agentes son capaces de decidir en cada momento la acción a ejecutar en función a sus objetivos. El proceso de razonamiento se descompone en dos fases, en una inicial, se establecen las metas y en una segunda se define cómo alcanzarlas. Una representación basada en acciones requiere una arquitectura que permita adquirir conocimiento del entorno. [Kolodner, 1983a] [Kolodner, 1983b] [Joh, 1997] [Corchado *et al.,* 2008b] describen procedimientos para adquirir este conocimiento a través del modelo BDI.

2.2.3 Híbridas

Para algunos investigadores, la construcción de agentes utilizando una arquitectura totalmente deliberativa, o totalmente reactiva no constituye un enfoque acertado. En consecuencia, se han propuesto sistemas híbridos que pretenden combinar aspectos de ambos modelos. Estas arquitecturas combinan módulos reactivos con módulos deliberativos. Los primeros se encargan de procesar los estímulos que no necesitan deliberación, mientras que los módulos deliberativos determinan qué acciones deben realizarse para satisfacer los objetivos locales y cooperativos de los agentes.

Una primera aproximación consiste en construir un agente con dos subsistemas: uno deliberativo, conteniendo un modelo simbólico del mundo, que desarrolla planes y toma decisiones, y uno reactivo, que es capaz de reaccionar a sucesos que ocurren en su entorno sin comprometerse en el razonamiento complejo. Frecuentemente, al componente reactivo se le da algún tipo de precedencia sobre el deliberativo, para que dar una respuesta rápida a sucesos importantes. Este tipo de estructura conduce a la idea de una arquitectura de capas [Ferguson, 1992] [Müller, 1996]. En las arquitecturas híbridas, los subsistemas de control del agente se organizan en una jerarquía, con capas más altas que barajan información a niveles crecientes de abstracción. Así, por ejemplo, la capa más baja podría enviar datos directamente del sensor a las salidas efectoras, mientras que la capa superior trata con metas a largo plazo. Un problema clave en estas arquitecturas es qué clase de estructura de control se incrusta en los subsistemas del agente para dirigir las interacciones entre las diversas capas.

2.3 Agentes en planificación

El núcleo fundamental de este trabajo consiste en el diseño de un sistema de planificación aplicable en una organización virtual basada en un sistema multiagente. Establecer un plan para un agente implica definir la secuencia de acciones que debe realizar el agente para, partiendo de un determinado estado inicial, alcanzar un estado final en el que se hayan logrado los objetivos marcados. Nuestro objetivo es aplicar distintas técnicas de manera coordinada para conseguir predecir la carga de trabajo y en función de la carga hacer que un conjunto de tareas sean distribuidas entre los agentes incluidos en una organización virtual de manera que los objetivos globales de la organización sean satisfechos de manera eficiente.

En esta sección se revisan los distintos métodos aplicables para establecer sistemas de planificación en sistemas multiagentes.

2.3.1 Algoritmos metaheurísticos para la planificación

En la búsqueda por resolver problemas de planificación se opta entre dos alternativas (métodos exactos y métodos aproximados). La primera consiste en intentar resolverlos en forma exacta, con el riesgo de incurrir en dilatados tiempos de computación, posiblemente impracticables. La segunda implica usar métodos aproximados con un alto grado de aproximación al óptimo, invirtiendo menos tiempo en encontrarlas.

Los métodos exactos [Blum y Furst, 1997] [Kumara *et al*., 1998] [Monostori *et al.,* 2006] pretenden hallar un plan jerárquico único analizando todos los posibles ordenamientos de las tareas o procesos involucrados (exploración exhaustiva). Sin embargo, una estrategia de búsqueda y ordenamiento que analice todas las combinaciones posibles es computacionalmente cara y sólo funciona para algunos tipos (tamaños) de casos.

Los métodos aproximados [Pinedo, 2012] [Feo y Resende, 1995] buscan resolver las variantes más complejas en las que interviene el comportamiento de tareas y recursos. Dichos métodos no analizan exhaustivamente todas las posibles combinaciones de patrones del problema, sino que más bien eligen los que cumplan determinados criterios. Obtienen finalmente soluciones lo suficientemente buenas para las instancias que resuelven, lo que justifica su uso.

Los sistemas existentes que pueden ser considerados como planificadores de tareas no buscan la optimización del ordenamiento y la distribución de las tareas a ejecutar, sino más bien dan énfasis a encontrar planes factibles de ser ejecutados [Tupia y Mauricio, 2004]. Debido a esto, tampoco existen sistemas que replanifiquen eficazmente y son sistemas muy cerrados y dependientes del entorno.

La naturaleza de los problemas de programación de tareas es combinatoria y, por ello, el tiempo necesario para encontrar la solución óptima crece exponencialmente con el número de tareas consideradas. Estos problemas se engloban dentro del área conocida como optimización combinatoria y, en el mejor de los casos, se puede determinar que son problemas de tipo NP (polinómico no determinístico) [Garey y Johnson, 1979], que no tiene una solución óptima (no existe un algoritmo con complejidad polinomial que los resuelva) pero sí se ha logrado obtener una solución (no óptima ni tan eficiente), muchas veces consumiendo un amplio periodo de tiempo, para pequeñas instancias [Ríos y Bard, 2000]. Sin embargo, la mayoría de los problemas de optimización combinatoria pertenecen a la familia de los problemas NP-hard (subclase de los NP) [Pinedo, 2012]. Esta situación justifica la aplicación de algoritmos aproximados en la búsqueda de solución bajo tiempos razonables.

Dentro de los procedimientos aproximados se deben nombrar los heurísticos [Adams *et al.,* 1998]. Se habla de heurística para referirse a una técnica o método inteligente, para realizar una tarea que no es producto de un riguroso análisis formal, sino del conocimiento experto sobre un tema a solucionar. Los métodos heurísticos aportan soluciones a problemas combinacionales con un buen rendimiento en lo referente a calidad de soluciones y a los recursos empleados, procurando cierto grado de confianza al encontrar soluciones de alta calidad con un costo computacional razonable, aunque no garantice su óptimo rendimiento o factibilidad, e incluso en algunos casos, sin lograr establecer lo cerca que se está de dicha situación [Melián *et al.,* 2003]. Por tanto, los procedimientos heurísticos se basan en reglas extraídas de la experiencia que permiten obtener soluciones válidas, generalmente bastantes buenas, en tiempos razonables aunque no se puede garantizar la obtención del óptimo para el problema considerado. Por otra parte, al tratarse de soluciones diseñadas para un problema en base a la experiencia acumulada sobre el mismo, la mayoría de estos algoritmos son difícilmente adaptables a problemas ligeramente diferentes a aquéllos para los que fueron concebidos [Sánchez y López, 2005].

Un ejemplo de algoritmo heurístico lo encontramos en el denominado voraz. Un algoritmo voraz (AVM) es aquel que, para resolver un determinado problema, elige la opción óptima en cada paso, con la esperanza de llegar a una solución general óptima. También son conocidos como algoritmos golosos-miopes (del inglés greedy-myopic) [Zamudio *et al.,* 2006]: golosos porque siempre escogen el mejor candidato para formar parte de la solución [Cormen *et al.,* 2001] y miopes porque esta elección es única e inmodificable dado que no analiza más allá los efectos de haber seleccionado un elemento como parte de la solución.

Los AVM producen soluciones rápidas y satisfactorias debido a que utilizan el conocimiento acerca del problema a solucionar mediante las restricciones establecidas para dicho problema pero tiene dos deficiencias principales [Ramírez, 2006]:

- Dependen mucho de las características de las instancias que intentan resolver, es decir, pueden arrojar muy buenos resultados para determinadas instancias del problema, pero no para otras.

- Se limitan a los óptimos locales de las funciones que pretenden optimizar y, probablemente, no analizan vecindades más allá del criterio voraz, por lo cual pueden no considerar al óptimo global [Tupia, 2004].

Por ejemplo, en un espacio de dos dimensiones los algoritmos voraces no recorren el total de espacio de posibles soluciones, sino vecindades (subconjuntos) dejando pasar por tanto, valores que originarían soluciones globales [Ramírez, 2006].

Algunos ejemplos de aplicaciones de algoritmos voraces para la planificación de tareas pueden encontrarse en [Campello y Maculan, 2002] [Tupia y Mauricio, 2004].

Los algoritmos heurísticos que son fácilmente generalizables a varios tipos de problemas, como los algoritmos genéticos o la búsqueda tabú, son denominados algoritmos metaheurísticos. De estos se habla a continuación, referenciando algunos trabajos donde se pueden encontrar ejemplos de utilización.

El término metaheurística apareció por primera vez a mediados de los años 80 y desde entonces, han surgido numerosas propuestas y pautas (combinando técnicas de Inteligencia Artificial, evolución biológica y métodos estadísticos) para diseñar buenos procedimientos en la resolución de problemas de optimización combinatoria [Marti y Moreno, 1996], como es el problema que se afronta en este trabajo. Las metaheurísticas son estrategias inteligentes para diseñar, mejorar y optimizar procedimientos, heurísticas muy generales con un alto rendimiento respecto a las heurísticas tradicionales. De ahí que se le agrega el sufijo meta, que significa más allá o a un nivel superior. A continuación se muestran algunos algoritmos que hacen uso de metaheurísticas.

2.3.1.1 Algoritmos GRASP

GRASP es un procedimiento iterativo en donde cada paso consiste en una fase constructiva y una de mejora. En la fase constructiva se aplica un procedimiento heurístico para obtener una buena solución inicial. Esta solución se mejora en la segunda fase mediante un algoritmo de búsqueda local. La mejor de todas las soluciones examinadas se almacena como resultado final [Coral *et al*., 2007].

Su nombre proviene de:

- Greedy (voraz): porque el criterio de escoger el mejor valor después de cada evaluación prima para seleccionar los candidatos.
- Randomized (aleatorio): porque después de tener la lista de candidatos escoge a uno de ellos al azar.
- Adaptive (adaptable): porque es capaz de adaptarse a diferentes contextos de aplicaciones o modificaciones relevantes al modelo.

- Search Procedure: porque realiza una búsqueda dentro de un espacio o vecindad, evaluando aleatoriamente a cierto número de integrantes.

Mientras el argumento de los AVM se centra en escoger al que posee en un determinado instante el mejor valor, el criterio GRASP se relaja y amplía dicha restricción, seleccionando no al mejor, sino a un elemento de dentro de un conjunto de valores (previamente evaluados por la función objetivo) garantizando la superación del carácter voraz. Dichos valores no son necesariamente los óptimos locales, pero serán los principales gestores para encontrarlos mediante búsquedas locales [Resende y Ribeiro, 2003].

Los algoritmos GRASP son recomendables cuando el conjunto de datos a trabajar es grande. Si bien es cierto que el algoritmo no obtiene la solución óptima, sí llega a una solución aproximada con un ahorro en tiempo de ejecución en el proceso y disminución de costo al tener un menor desperdicio [Pitsoulis y Resende, 2002].

Pueden encontrarse aplicaciones de este algoritmo a la planificación en [Kim y Park, 2004] [Andrés *et al.,* 2008] [Binato *et al.,* 2000].

2.3.1.2 Búsqueda tabú

La búsqueda tabú (TS, Taboo search) fue diseñada inicialmente por [Glover, 1989] [Glover, 1990] y ha sido aplicada a varios problemas de optimización combinatoria.

Al igual que el procedimiento de recocido simulado, que veremos a continuación, se trata de un algoritmo de búsqueda local, basado en la idea de que una determinada solución puede ser mejorada realizando cambios pequeños. Para cualquier solución se genera un conjunto de desplazamientos que definen su vecindario, y se selecciona el mejor de ellos para la siguiente iteración. Para evitar que el procedimiento iterativo entre en un bucle y que se quede estancado en óptimos locales, se mantiene en memoria el histórico de la búsqueda. Se suele distinguir dos tipos de memoria: a corto plazo para los movimientos más recientes y a largo plazo para la parte de la búsqueda más distante. La memoria a corto plazo está formada por una lista tabú que contiene información sobre las últimas soluciones visitadas, de modo que se prohíben los desplazamientos hacia soluciones vecinas que se correspondan con la lista tabú. Típicamente, la lista tabú consiste en una lista FIFO (First In First Out) de longitud definida que contiene los últimos desplazamientos unitarios prohibidos. Así, en cada iteración se introduce el desplazamiento inverso al realizado al principio de la lista y se elimina el desplazamiento que se encuentra al final [Bonrostro, 1998].

En [Porto *et al.,* 2000] [Taillard, 1994] [Chambers y Barnes, 1999] se consideran algunos casos de planificación de tareas donde la aplicación de estos algoritmos ha resultado efectiva.

2.3.1.3 Recocido simulado

Los algoritmos de recocido Simulado ("Simulated Annealing", SA, en inglés) se basan en la analogía entre los procesos físicos termodinámicos y los elementos de un problema de optimización combinatoria. El recocido denota un proceso de calentamiento de un sólido a una temperatura en la que sus granos deformados recristalizan para producir nuevos granos. La temperatura de recocido o de recristalización, depende del tipo de material, del grado de deformación del mismo, además de su uso futuro. Después de la fase de calentamiento, se aplica un proceso de enfriamiento en el que la temperatura se baja poco a poco. De esta manera, las partículas se reacomodan en estados de más baja energía hasta que se obtiene un sólido con sus partículas acomodadas conforme a una estructura de cristal. Se comienza con un valor máximo de la temperatura y, en la fase de enfriamiento, para cada valor de la temperatura se debe permitir que se alcance su equilibrio térmico. Sin embargo, si el proceso de enfriamiento es demasiado rápido y no se alcanza en cada etapa el equilibrio térmico, el sólido congelará en un estado cuya estructura será amorfa en lugar de la estructura cristalina de más baja energía [Gutiérrez, 2005].

La simulación del proceso de recocido puede usarse para describir un proceso de generación de una sucesión de soluciones de un problema de optimización combinatoria, en donde se vayan obteniendo, conforme el proceso avanza, mejores soluciones al mismo. Para este propósito, se puede observar una analogía entre el sistema físico y un problema de optimización combinatoria en donde cada solución del problema puede verse como un estado del sólido y el valor de la función objetivo para la el nivel de energía del sólido. En resumen, se puede pensar en las siguientes equivalencias [Restrepo *et al.,* 2004]:

EN METALURGIA	EN OPTIMIZACIÓN
Energía de la configuración	Costo de la solución
Temperatura	Parámetro
Configuración fundamental	Solución óptima
Mínima energía	Función objetivo
Configuración	Solución factible

Figura 1 Tabla equivalencias Recocido Simulado

Los algoritmos tradicionales de búsqueda local parten de una solución inicial que de modo paulatino es transformada en otras que a su vez son mejoradas al introducirles pequeñas perturbaciones o cambios. Si el cambio da lugar a una solución mejor que la actual, se sustituye ésta por la nueva continuando el proceso hasta que no es posible ninguna nueva mejora. Esto significa que la búsqueda finaliza en un óptimo local, que no tiene que ser precisamente el global.

Un modo de evitar este problema es permitir que algunos movimientos sean hacia soluciones peores. Pero por si la búsqueda está realmente yendo hacia una buena solución, estos movimientos de escape deben realizarse de un modo controlado [Dowsland y Adenso, 2003]. En el caso del recocido simulado, esto se realiza mediante una función de probabilidad que hará disminuir la probabilidad de esos movimientos hacia soluciones peores conforme avanza la búsqueda (y por tanto, se supone que previsiblemente se está más cerca del óptimo global).

La idea es movernos de los extremos locales mediante sacudidas (simulan la temperatura) que irán decreciendo en intensidad. Se selecciona aleatoriamente un sucesor del estado actual y se pasa a él de forma condicional

- Si su valoración es mejor, se pasa a ese nuevo estado
- Si la valoración del sucesor no es mejor, pasamos con probabilidad dependiente de la valoración y de la metáfora de la temperatura.

Si T disminuye suficientemente despacio, el algoritmo encontrará un óptimo global con probabilidad cerca de uno.

Algunos trabajos relacionados con la utilización de este método a la planificación de tareas pueden encontrarse en [Kolonco, 2009] [van Laarhoven *et al.,* 2002] [Steinhofel *et al.,* 1999].

2.3.2 Teoría de Colas

La teoría de colas se puede definir como el estudio matemático del comportamiento de las colas que se generan por los usuarios para el acceso a unos determinados servicios. Para realizar dicho estudio, se crean una serie de modelos matemáticos que tratan de describir el comportamiento de las colas generadas por los usuarios. En estos estudios, se tienen en cuenta diversas variables y se busca alcanzar un equilibrio entre el coste y el tiempo de espera [Tadj, 1995].

Esta teoría es aplicable a multitud de campos. Originalmente [Erlang, 1909] se aplicó al estudio de la congestión de las comunicaciones de las líneas telefónicas en Copenhaghe bajo una demanda incierta, pero los campos de aplicación son muy variados. Así, por ejemplo, más recientemente se ha utilizado para la fabricación de semiconductores [Shanthikumar, *et al.,* 2007] o en campos más próximos a nuestro trabajo, como es el caso de su aplicación en el balanceo de carga de servidores web [Menasce, 2002]. En este último trabajo, se dispone de diversos servidores web montados en clúster y se pretende decidir el número adecuado de servidores teniendo en cuenta los costes y la demanda. Otras aplicaciones en entornos reales se pueden ver en el trabajo de [Tadj, 1995]. En nuestro caso, los servicios de los que se habla en la teoría de colas son equivalentes a los agentes que deben estar disponibles dentro de la organización.

En un sistema de colas se pueden distinguir diferentes parámetros que determinan el funcionamiento, entre ellos tenemos [Antoniol, 2004]:

- *Tasa de llegada*: La distribución de llegada que sigue la entrada de tareas en el sistema. Normalmente la función de distribución seguida es una de Poisson.
- *Tiempo de servicio*: Tiempo empleado en llevar a cabo la tarea una vez el agente correspondiente se hace cargo de ella. La distribución seguida normalmente es una exponencial aunque existen otras como la deterministica, Erlang o incluso hay estudios que se basan en el uso de una distribución genérica [Tadj, 1995].
- *Servidores*: Uno o varios agentes con la posibilidad de ausencia.
- *Disciplina de la cola*: FIFO (first in first out), LIFO (last in first out), RR (Round Robin), RSS (random selection of service), with priority.
- *Capacidad de la cola*: Longitud de la cola finita o infinita.
- *Tamaño de la fuente de entrada al sistema*

Para la especificación de los parámetros se suele seguir la denominada notación de Kendall. Esta notación consiste en una séxtupla que almacena los datos indicados anteriormente. Se representa mediante la siguiente sintaxis 1/2/3/4/5/6. Los valores de cada uno de los términos de la expresión varían de la forma siguiente para cada uno de los términos:

1. M: Poisson, D tiempos determinísticos, E_k distribución de Erlang, G distribución general
2. M: Exponencial, D tiempos determinísticos, E_k distribución de Erlang, G distribución general
3. 1 para un sólo servidor, S para varios.
4. En la disciplina se ponen las siglas del método seleccionado
5. Capacidad de la cola un número ó ∞.
6. Número máximo de consumidores en el sistema, un número ó ∞.

Los modelos de colas más usuales son los que siguen las nomenclaturas M/M/1/FIFO/∞/∞ y M/M/s/FIFO/∞/∞. Las M indican una tasa de llegada de Poisson y la de servicio exponencial, el 1 indica un servidor, s servidores en lugar de uno sólo en el segundo caso, FIFO indica la política de servicio First Input First Output y los signos ∞ indican una capacidad de la cola y tamaño de la fuente ilimitados.

2.3.3 Algoritmos genéticos

Los algoritmos genéticos, [Holland, 1992], constituyen el paradigma más completo de la computación evolutiva ya que incorporan, de modo natural, todas las ideas fundamentales de dicho enfoque.

Son muy flexibles, es decir, pueden adoptar con facilidad nuevas ideas, generales o específicas, que surjan dentro del campo de la computación evolutiva. Además, se pueden combinar fácilmente con otros paradigmas y enfoques, aunque no tengan ninguna relación con la computación evolutiva.

Los algoritmos genéticos son el paradigma con mayor base teórica de entre los de la computación evolutiva. Además, dicha base teórica es sencilla en su desarrollo y con grandes posibilidades de ampliación.

De entre todos los paradigmas de la computación evolutiva son los que menos conocimiento específico necesitan para su funcionamiento y, en consecuencia, los más versátiles. Además pueden incorporar conocimiento específico con poco esfuerzo adicional.

Son fáciles de implementar en computadores con capacidades medias, proporcionando resultados aceptables, en cuanto a precisión y recursos empleados, para una gran cantidad de problemas difícilmente resolubles por otros métodos.

Existe una gran cantidad de ensayos empíricos que proporcionan operadores, parámetros e implementaciones específicas para una amplia gama de problemas.

En definitiva, los algoritmos genéticos no son más que métodos estocásticos de búsqueda de soluciones cuasióptimas. En ellos se mantiene una población que representa a un conjunto de posibles soluciones la cual es sometida a ciertas transformaciones con las que se trata de obtener nuevos candidatos mediante un proceso de selección sesgado a favor de los mejores candidatos.

A grandes rasgos un algoritmo genético consiste en una población de soluciones codificadas de forma similar a cromosomas. Cada uno de estos cromosomas, tendrá asociado un ajuste o valor de bondad que cuantifica su validez como solución al problema. En función de este valor se le darán más o menos oportunidades de reproducción. Además, con cierta probabilidad se realizarán mutaciones de estos cromosomas.

La aplicación más común de los algoritmos genéticos es la de resolver problemas de optimización, dónde se han mostrado ser bastante eficientes. Sin embargo, no todos los problemas son apropiados para esta técnica y sólo se debería de aplicar cuando se dan las siguientes condiciones:

- El espacio de búsqueda está limitado dentro de un cierto rango
- Se puede definir una función de aptitud que indique lo buena que es una determinada solución. La función de aptitud no es más que la función objetivo del problema.

- La codificación de la información debe ser sencilla a nivel computacional

La información del problema se va a almacenar en lo que se denomina cromosomas, que no son más que una secuencia de genes en la que cada gen representa una determinada información del problema. Los módulos en los que se va a descomponer el algoritmo son los siguientes:

- *Módulo evolutivo*. Mecanismo de codificación y función de aptitud.
- *Módulo poblacional.* Tiene una representación poblacional y técnicas para manipularla: técnica de representación, de arranque, criterio de selección y de reemplazo. También se define el tamaño poblacional y la condición de terminación.
- *Módulo reproductivo*. Operadores genéticos.

El ciclo evolutivo de un algoritmo genético se puede esquematizar en los siguientes pasos:

1. Iniciar la población.
2. Evaluar la aptitud de los individuos.
3. Seleccionar progenitores y crear descendientes para la nueva generación hasta completar cupo de individuos nuevos por cruce.
4. Añadir a la nueva generación los cromosomas elitistas.
5. Añadir a la nueva generación individuos de la vieja hasta completar el tamaño poblacional. Aplicar un método de selección para escoger los cromosomas.
6. Ir a paso 2 si no se cumpla la condición de parada.

En la implementación de un algoritmo genético es necesario determinar una serie de factores como son la codificación de los cromosomas, la aptitud, el número de cromosomas, procedimiento de selección, criterios de iniciación, parada y reemplazo y una serie de operadores de cruce y mutación.

Codificación: Como se dijo anteriormente, la información se almacena en cromosomas que contienen una serie de genes que se corresponden con parámetros del problema. Los genes se tienen que codificar de forma que sea fácilmente interpretable de manera algorítmica. La codificación más habitual es la binaria, en la que cada uno de los genes puede tomar una secuencia de 0 ó 1, no obstante, se puede almacenar cualquier otro tipo de dato según las necesidades del problema. Cada uno de los bits perteneciente a un gen recibe el nombre de alelo.

La codificación es quizás el aspecto más importante y más complicado a la hora de intentar resolver un problema. Una mala elección de la codificación conllevará a unos malos resultados.

Función de aptitud (fitness): Función que proporciona un valor sobre la adecuación de un cromosoma como solución de un determinado problema. Se corresponde con la función objetivo. El rango de valores que proporciona debe de ser lo suficientemente amplio como para que los cromosomas que no se adecuen al problema obtengan valores bajos para que no pasen a la siguiente generación. No obstante, si el rango fuera demasiado amplio, se podría producir una convergencia prematura.

Tamaño de la población: Si se elige un tamaño de población pequeño, se producirá una rápida convergencia de los cromosomas y por tanto se perderá la diversidad. Para evitar esta situación, se pueden sustituir periódicamente cromosomas por otros nuevos creados: sería lo que se conoce como inmigración. Si se eligen tamaños poblacionales grandes, los recursos computacionales aumentarán considerablemente, por eso es mejor considerar tamaños no muy grandes e introducir nuevos elementos mediante la inmigración.

Criterio de selección: Los individuos se copian de acuerdo a su aptitud. Los cromosomas con mayor valor en la función de aptitud tendrán mayor probabilidad de ser elegidos para la siguiente generación. Existen diferentes implementaciones, pero la más conocida es el método de la ruleta. En la selección de la ruleta, a cada uno de los individuos de la población se le asigna una parte de una ruleta proporcional a su aptitud, de tal forma que la suma de todos los porcentajes sea la unidad. Los mejores individuos recibirán una porción de la ruleta mayor que la recibida por los peores. Para seleccionar un individuo basta con generar un numero aleatorio del intervalo [0..1] y devolver el individuo situado en esa posición de la ruleta. Esta posición se suele obtener recorriendo los individuos de la población y acumulando sus proporciones de ruleta hasta que la suma exceda el valor obtenido.

Criterio de iniciación: El criterio de iniciación consiste básicamente en generar una serie de cromosomas válidos a partir de los cuales se va a evolucionar. La generación de los cromosomas será aleatoria pero todos ellos tienen que ser válidos para la codificación que se esté usando.

Criterio de parada: Los criterios de parada pueden ser diversos aunque normalmente es un número máximo de iteraciones. En otras ocasiones puede coincidir con una medida de convergencia de la población. Una vez se haya alcanzado, se procede a detener la evolución de la población.

Criterios de reemplazo: Las dos estrategias de evolución más empleadas son la (λ +λ)-ES y la (λ, λ)-ES. En la primera de ellas un total de λ producen λ descendientes reduciéndose nuevamente la población a λ individuos (los padres de la siguiente generación) por selección de los mejores individuos. De esta manera, los padres sobreviven hasta que son reemplazados por hijos mejores que ellos. En la (λ, λ)-ES la descendencia reemplaza directamente a los padres, sin hacer ningún tipo de comprobación.

Operadores: Los operadores definen el procedimiento de evolución de la población. Entre los principales operadores vamos a tener los siguientes:

- ***Cruce***: Es el operador que permite generar nuevos individuos a partir de unos determinados padres seleccionados. Para ello recombina genéticamente la información de los progenitores. Tiene una probabilidad alta de ser utilizado y es el operador más importante responsable de la búsqueda en profundidad. Hay que tener en cuenta que los cromosomas generados por este operador tienen que ser válidos, por lo que su implementación dependerá del problema.
- ***Mutación***: Tiene poca probabilidad de ser utilizado y es el responsable de la búsqueda en amplitud. Básicamente, consiste en alterar un cromosoma modificando sus genes para crear un nuevo cromosoma válido. De esta forma, se pretende evitar la convergencia de la población.
- ***Elitismo***: Garantiza que un determinado número de cromosomas que mejor están adaptados pasen a la siguiente generación.

2.3.4 Razonamiento basado en casos (CBR)

El desarrollo de agentes basados en modelos deliberativos implica la dotación de mecanismos de razonamiento que, en muchos casos, deben complementarse con capacidades de adaptación y aprendizaje. El razonamiento basado en casos (CBR: *Case Based Reasoning*) [Laza y Corchado, 2001] plantea un sistema de resolución de nuevos problemas a partir de las soluciones previamente experimentadas en la resolución de otros problemas similares. El razonamiento basado en casos (CBR) es un tipo de razonamiento basado en experiencias pasadas [Kolodner, 1993].

Los sistemas CBR, almacenan en una base de casos las soluciones aplicadas a los problemas tratados e incorporan un ciclo de razonamiento que permite extraer y adaptar estas soluciones para construir la solución a un nuevo problema. El término principal cuando se trabaja con CBR es el concepto de caso. Cada caso almacenado en la base de casos contiene información sobre una solución experimentada con un problema anterior. La información que compone cada caso incluye la descripción del problema (situación inicial), la solución aplicada (secuencia de acciones que se pueden llevar a cabo con el objetivo de resolver el problema) y el resultado obtenido (estado final alcanzado una vez que se ha aplicado la solución). Los casos se gestionan mediante un ciclo de razonamiento CBR que se compone de cuatro etapas: recuperación, reutilización, revisión y retención o aprendizaje.

Recuperación: comienza cuando el sistema recibe un nuevo problema a resolver. Consiste en extraer de la base de casos aquellos que se consideran aplicables al nuevo problema. En esta fase, tiene lugar el acceso a los casos almacenados que

cuentan con una descripción de problema más similar a la del problema actual. La similitud entre casos se establece en base a una métrica que es preciso determinar.

Reutilización: recibe como entradas los casos más similares recuperados durante la etapa anterior. La reutilización o adaptación consiste en trabajar con las soluciones correspondientes a los casos más similares recuperados para obtener una solución al problema actual. Trabajar con las soluciones significa modificarlas y combinarlas, o simplemente decidir cuál de ellas es la más óptima y reutilizarla.

Revisión: determina si la solución obtenida a partir de la ejecución de la etapa anterior es válida para la resolución del caso actual.

Aprendizaje: incorpora la nueva experiencia a la base de conocimientos. Una vez aplicada la solución al nuevo caso se comprueba si el resultado ha sido satisfactorio y se almacena la información del nuevo caso.

2.3.4.1 Planificación basada en casos (CBP)

El modelo de razonamiento basado en casos se puede extender para su aplicación a la resolución de problemas de planificación [Hammond, 1989] [Bajo *et al.,* 2007b]. El problema a tratar en el caso de la planificación es el establecimiento de la secuencia de acciones que, a partir de un estado inicial dado, conducen a alcanzar un estado final que se corresponde con el objetivo establecido. La planificación basada en casos (CBP) es una especialización del razonamiento basado en casos (CBR). En CBP, la solución propuesta es un plan que se genera a partir de los planes aplicados en el pasado para resolver problemas similares. Los problemas y sus correspondientes planes se almacenan en una memoria de planes. En este caso, la descripción del problema (estado inicial) y la solución (situación cuando se alcanza un estado final) son representadas por medio de creencias, el estado final alcanzado y la secuencia de acciones se representan como planes.

Así, se puede asimilar un plan a un caso que se definiría por los siguientes datos: conjunto de operaciones, secuencia en que se aplican las acciones, conjunto de acciones permitidas bajo diferentes estados y conjunto de aplicaciones que permiten cambiar entre conjuntos de acciones. Por tanto, un plan P es una tupla *<S,B,O,L>*:

- S es un conjunto de acciones
- O es una relación de orden sobre S que permiten establecer un orden entre las acciones en un plan
- B es un conjunto que permite describir conexiones permitidas y prohibidas entre las acciones de un plan
- L conjunto de enlaces que permite cambiar de un conjunto de acciones S a otro S'

A partir de la definición del planificador basado en casos, se puede establecer la relación con el modelo CBR para generar el modelo de planificación híbrido que integra los BDI [Corchado *et al.,* 2008b] a modo de especialización del CBR-BDI visto anteriormente. De este modo, es posible dotar a los agentes deliberativos BDI, con mecanismos de planificación basados en casos, derivados de los sistemas CBR, diseñados específicamente para la elaboración de planes.

Los agentes con capacidades de planificación basados en el modelo BDI parten del modelo BDI y establecen una correspondencia entre los elementos del modelo BDI y de los sistemas de planificación basado en casos. El modelo BDI se ajusta a los requisitos del sistema puesto que permite definir una serie de metas a alcanzar a partir del conocimiento registrado acerca del mundo. La fusión de los agentes con capacidades de planificación junto con el modelo BDI, generan agentes con capacidades de planificación basados en el modelo BDI que permiten formalizar tanto la información disponible, la definición de las metas y acciones disponibles para resolver el problema así como el procedimiento para resolver nuevos problemas mediante la adopción del ciclo de razonamiento.

Este tipo de sistemas, comienzan identificando los roles y las metas de los agentes, siguiendo la pauta de los sistemas CBR en el diseño y la implementación de la arquitectura de los agentes, facilitando el aprendizaje y la adaptación, lo que conlleva a un mayor grado de autonomía del que se encuentra en la arquitectura BDI. Los agentes planificadores con el modelo BDI, al utilizar CBR como mecanismo de razonamiento, son capaces de aprender a partir de un conocimiento inicial, interactuar de forma independiente con el entorno, usuarios y otros agentes dentro del sistema, y adaptarse a las necesidades del entorno. La estructura básica de un agente deliberativo con capacidades de planificación incluye el mecanismo de razonamiento de un agente CBR, en la fase de recuperación se accede a la base de planes y se recuperan los planes similares, una vez recuperados se seleccionan los planes que mejor se ajustan al problema para, posteriormente, en la fase de reutilización adaptarlos al agente BDI con las creencias y los objetivos y generar el nuevo plan. Finalmente, se pasa por la fase de revisión en la que se determina la adecuación del plan generado y en caso de ser una solución satisfactoria se almacena en la base de planes.

Nuestra hipótesis de trabajo se basa en la creencia de que se pueden integrar varios de los sistemas de planificación en un modelo que permita aprovechar en cada caso las ventajas que presentan cada uno de ellos. Nuestro modelo incorpora un sistema de planificación basada en casos que se sustenta sobre método de recuperación que aplica la teoría de colas y una fase de adaptación que utiliza algoritmos genéticos.

2.4 Sistemas Multiagente

El uso generalizado de las redes de ordenadores ha propiciado que los problemas que tradicionalmente se habían abordado de manera aislada pasasen a afrontarse mediante

la utilización de sistemas distribuidos. Dentro de estos se encuentran los sistemas multiagente, en los que un conjunto de agentes con capacidades parciales deben coordinarse para, a través de sus objetivos particulares, colaborar para satisfacer unas metas globales.

De modo muy general, se habla de sistemas multiagente (SMA, MAS, Multi-Agent System) para referise a cualquier sistema compuesto de múltiples componentes autónomos que presente las siguientes características [Jennings *et al.,* 1998]:

- Cada agente tiene capacidades incompletas para resolver un problema.
- Carencia de un sistema de control global.
- Descentralización de los datos.
- Computación asíncrona.

Un sistema multiagente es básicamente una red de entidades enfocadas a resolver problemas, y que trabajan de manera conjunta para encontrar respuestas a los problemas que están más allá de las capacidades o del conocimiento individual de cada entidad [Durfee *et al.,* 1989].

Cuando hablamos de sistema multiagente, extendemos la idea de un agente en solitario completándola con una infraestructura para la interacción y comunicación. Los SMA idealmente tienen las siguientes características [Huhns y Stephens, 1999]:

- Son típicamente abiertos y tienen un diseño no centralizado.
- Contienen agentes autónomos, heterogéneos y distribuidos
- Proporcionan una infraestructura para especificar comunicación y protocolos de interacción.

Los SMA abiertos deben permitir la participación de agentes heterogéneos, con arquitecturas e incluso lenguajes distintos [Zambonelli *et al.,* 2003]. Por esta razón, no se puede confiar en el comportamiento de los agentes, siendo necesario establecer controles en base a normas o reglas sociales. Para ello, los desarrolladores se han centrado en los aspectos organizativos de la sociedad de agentes, guiando el proceso de desarrollo de los sistemas mediante los conceptos de organización, normas, roles, etc.

Los agentes en un SMA basado en conceptos organizacionales se coordinan e intercambian servicios e información; son capaces de negociar y llegar a acuerdos; y pueden llevar a cabo otras acciones sociales más complejas. Coordinación y comunicación son temas muy importantes dentro de los SMA. En un SMA, los agentes tienen que encontrarse, anunciar sus capacidades y las tareas que pueden llevar a cabo, además de pedir o solicitar tareas a otros agentes.

En las interacciones de un SMA se distinguen cuatro conceptos que hacen referencia a características diferentes de un SMA: comunicación, coordinación, cooperación y negociación.

- **Comunicación.** Es la habilidad de los agentes para comunicarse entre sí, intercambiar información y conocimiento de forma comprensible. Permite a los

agentes obtener el conocimiento necesario para decidir la secuencia de acciones que debe ejecutar en función de sus objetivos.

Para que la comunicación pueda producirse es necesario establecer un lenguaje común y asegurar la capacidad de los agentes para intercambiar conocimiento así como para entenderlo e interpretarlo.

Los lenguajes de comunicación de agentes se denominan genéricamente ACL (Agent Communication Language) [Labrou *et al.,* 1999]. El uso de un ACL normalizado es esencial para permitir la comunicación de agentes pertenecientes a diferentes sistemas multiagente. FIPA ACL es un estándar desarrollado por la organización FIPA, basado en KQML (Knowledge Query and Manipulation Language) [FIPA, 2002].

- **Coordinación.** La coordinación es una característica clave en el desarrollo de SMA. Malone [1988] describe la coordinación de acciones como un conjunto de acciones suplementarias que pueden realizarse en un entorno multiagente para alcanzar un objetivo y que un agente, con los mismos objetivos, no podría alcanzar por sí solo.

  [Ferber *et al.,* 2003] define 4 razones fundamentales para realizar acciones coordinadas:

  1. Los agentes necesitan información y resultados que sólo pueden ser suministrados por otros agentes.
  2. Los recursos son limitados. Los agentes pueden compartir recursos para poder desarrollar las acciones.
  3. La coordinación permite optimizar costes, ya que elimina el desarrollo de acciones injustificadas o redundantes.
  4. Los agentes tienen objetivos distintos pero interdependientes, así pues, los agentes pueden alcanzar sus objetivos beneficiándose de esta interdependencia.
- **Cooperación y negociación.** La cooperación es el mecanismo por el cual los agentes, que trabajan juntos para lograr un objetivo común, definen una estrategia para alcanzar este objetivo.

  Por el contrario, la negociación modela la coordinación entre agentes auto-interesados capaces de llegar a acuerdos vinculantes. La negociación permite alcanzar decisiones de coordinación conjuntas mediante la comunicación explícita [Muller, 1996]. Estos mecanismos están inspirados en modelos tomados de las ciencias sociales y especialmente de la economía, donde prestan especial atención en la negociación estratégica y la teoría de juegos.

  Los SMA se preocupan por las interacciones de los agentes que lo forman. Estos agentes forman parte de una colección y pueden coordinar su conocimiento, objetivos, habilidades y planes juntamente para tomar una acción o resolver una meta global. En todos los sistemas debe haber un proceso de racionalización para la coordinación del conjunto de agentes. Por lo general en estos sistemas, los agentes, con sus creencias, deseos e intenciones construyen el problema y el

plan o secuencia de acciones para solucionarlo. La coordinación es un punto clave en el desarrollo de esta investigación.

2.4.1 Sistemas Multiagente abiertos

Las aplicaciones del paradigma de agentes pueden ser clasificadas en tres clases [Dignum, 2004]: sistemas abiertos, sistemas complejos y sistemas ubicuos.

- Los **sistemas abiertos** son sistemas en los cuales la estructura es capaz de cambiar dinámicamente. Sus componentes no son conocidos a priori, cambian en el tiempo y pueden ser heterogéneos. Un ejemplo de un sistema abierto es Internet: cualquier sistema informático que debe trabajar en internet tiene que ser capaz de operar con organizaciones de muy distinta índole y además sin guiado constante a los usuarios. Este tipo de funcionalidad requiere de técnicas de negociación y cooperación que encontramos también en el dominio de los sistemas multiagente [Rodríguez, 2010].
- Los **sistemas complejos** están relacionados con dominios grandes, impredecibles, en definitiva, complejos. Las herramientas más poderosas para hacer frente a la complejidad de los sistemas son la modularidad y la abstracción. Un problema a resolver con agentes puede dividirse en un número de sub-problemas de menor complejidad, que son más fáciles de manejar. Esta descomposición permite a los agentes emplear la solución posible más apropiada para resolver un sub-problema
- Los **sistemas ubicuos** tienen el objetivo de mejorar el uso de un sistema informático mediante la utilización de ordenadores disponibles en un entorno físico, normalmente distribuido, pero haciendo todo de manera invisible al usuario. Se puede considerar este tipo de sistemas como lo contrario de la realidad virtual. Si la realidad virtual sitúa a las personas dentro de un mundo generado por ordenador, la computación ubicua obliga al ordenador a "vivir" en el mundo con la gente [Weiser, 1993]. El sistema tiene que cooperar con el usuario para alcanzar su objetivo. Las aplicaciones tienen que comportarse como un agente inteligente.

Los SMA abiertos deben permitir la participación de agentes heterogéneos, con arquitecturas e incluso lenguajes distintos [Zambonelli *et al.,* 2003]. Por esta razón, no se puede confiar en el comportamiento de los agentes, siendo necesario establecer controles en base a normas o reglas sociales. Para ello, los desarrolladores se han centrado en los aspectos organizativos de la sociedad de agentes, guiando el proceso de desarrollo de los sistemas mediante los conceptos de organización, normas, roles, etc. Los agentes en un SMA basado en conceptos organizacionales se coordinan e intercambian servicios e información; son capaces de negociar y llegar a acuerdos; y

pueden llevar a cabo otras acciones sociales más complejas. Coordinación y comunicación son temas muy importantes dentro de los SMA. En un SMA, los agentes tienen que encontrarse, anunciar sus capacidades y las tareas que pueden llevar a cabo, además de pedir o solicitar tareas a otros agentes [Rodríguez, 2010].

La gran mayoría de las implementaciones de sistemas multiagente se corresponden con sistemas cerrados, donde ningún agente externo está habilitado para entrar a participar en la sociedad o agrupación. En estos sistemas, se conoce a priori el conjunto de agentes durante la fase de diseño. Además se asume que los agentes son benevolentes y confían en sus interacciones [Zambonelli *et al.,* 2000].

Los sistemas abiertos son especialmente recomendables en entornos operativos dinámicos, en los que se integran nuevos componentes, o bien componentes existentes abandonan el sistema de forma continua y donde las propias condiciones de operación cambian de forma impredecible [Zambonelli *et al.,* 2003]. Por tanto, los participantes entran o salen del sistema de forma dinámica, de modo que no se conoce de antemano (en la fase de diseño) el número total de participantes que existe en cada momento. Por ello, al diseñar el sistema se debe tener en cuenta la llegada dinámica de agentes desconocidos y la posibilidad de comportamientos auto-interesados en el transcurso de las interacciones [Argente, 2008a].

En resumen, los sistemas abiertos se caracterizan por la heterogeneidad de sus participantes, la confianza limitada, la existencia de objetivos individuales en conflicto y una gran probabilidad de no conformidad a las especificaciones [Dastani *et al.,* 2003]. En este tipo de sistemas la complejidad principal reside en las comunicaciones, pues el diseñador del agente no siempre va a conocer qué otros tipos de agente existirán en el sistema ni cómo se comunicará con ellos [Argente *et al.,* 2008a].

2.5 Conclusiones

Existen distintos sistemas de planificación de tareas que son aplicables en sistemas multiagente. A su vez los agentes pueden responder a diferentes arquitecturas de entre las cuales tienen especial interés aquellas que manejan una representación simbólica del entorno e incorporan modelos de razonamiento.

La hipótesis de este estudio trabaja con la idea de que es posible definir una arquitectura de sistema multiagente basada en organizaciones virtuales que incorpore un sistema de planificación de tareas en el que se contemple la adaptación de la estructura de la propia organización a los recursos disponibles y a las necesidades existentes con el fin de maximizar la eficiencia del sistema. Para ello aprovechará las ventajas que presentan los distintos sistemas de planificación, integrando en un sistema de planificación basado en casos, algunas de las técnicas habituales en planificación, como la teoría de colas y los algoritmos genéticos.

Capítulo III

Organizaciones Virtuales

VNiVERSiDAD
D SALAMANCA

Introducción

El concepto de organización se presenta como una solución prometedora para gestionar la coordinación de los agentes y controlar sus comportamientos y acciones, especialmente en los sistemas multiagente abiertos [Argente *et al.,* 2008a]. La Teoría de la Organización [Daft, 2010] [Fox, 1981] determina cuáles son los conceptos básicos, relaciones y características intrínsecas de cada tipo de organización existente. Las organizaciones humanas resultan muy eficientes para la gestión de la coordinación de sus miembros y la consecución de sus objetivos. Muchos de sus mecanismos de coordinación se han empleado también en distintas aplicaciones de sistemas multiagente [Horling y Lesser, 2004]. Por ello, en este trabajo, que pretende diseñar un sistema integrable en el comportamiento de una entidad, donde la distribución y la coordinación son aspectos claves, resulta especialmente atractivo el uso de organizaciones virtuales de agentes.

Tal y como se ha explicado en la introducción a esta tesis, el objetivo es presentar una arquitectura basada en la utilización de organizaciones virtuales de agentes que permita sugerir una planificación de tareas eficiente para la organización virtual de agentes. Con este modelo se persigue obtener un reparto del trabajo eficiente, tanto a corto como a largo plazo que permita la ejecución con éxito de todas las tareas pendientes. Para ello, el primer paso consiste en exponer qué entendemos como una organización y sus características clave. Posteriormente, este capítulo se centrará en detallar las características de las organizaciones humanas y empresariales para poder compararlas con las organizaciones de agentes.

Una vez extraídas las principales propiedades que determinan el concepto de organización se verá cómo éstas pueden ser representadas con éxito por organizaciones virtuales de agentes y cómo surge la analogía entre agente y humano.

3.1 Organizaciones

Uno de los puntos clave en el estudio y análisis de las organizaciones son las entidades que la componen y sus relaciones. Pero, a su vez, éstas entidades se ven claramente afectadas por los objetivos que persiguen, la funcionalidad del sistema en el que conviven, el entorno con el que se relaciona y las reglas que rigen el comportamiento de

todos los miembros. Por tanto, existen diversos factores a considerar en el modelado de una organización [Argente *et al.,* 2008a]:

- **Estructura**: comprende todos aquellos elementos que persisten en la organización, independientemente de cuáles sean los individuos finales de la misma en cada momento. Vendrá definida en base a los roles, sus agrupaciones, dependencias y sus patrones de interrelación.
- **Funcionalidad**: especifica cuáles son los objetivos globales de la organización; los servicios y funcionalidades que ofrece; los objetivos que persiguen los distintos componentes de la organización y qué tareas y planes deben seguir para alcanzarlos.
- **Normatividad**: determina el conjunto de normas y acciones definidas para controlar los comportamientos de los miembros de la organización. Se incluyen aquí tanto las normas sobre la actuación (obligaciones, permisos y prohibiciones), como las sanciones y recompensas a efectuar sobre sus miembros.
- **Dinamicidad**: especifica cómo evoluciona la organización a lo largo del tiempo, indicando el modo en que los agentes entran y salen de la organización de forma dinámica; adoptan determinados roles en función de sus capacidades y habilidades; y participan en aquellas unidades o agrupaciones de la organización en las que sean admitidos.
- **Entorno**: comprende tanto los recursos de los que depende la organización, como los proveedores de dichos recursos y los clientes o beneficiarios de la existencia de la organización.

3.1.1 Organizaciones humanas

Con el fin de entender correctamente la relación de los sistemas multiagente y cómo los conceptos de la Teoría de la Organización pueden mejorar su modelado, se debe comenzar extrayendo lo que se entiende por organización desde el punto de vista humano. Según el diccionario de la Real Academia de la Lengua Española, una organización se define como una "asociación de personas regulada por un conjunto de normas en función de determinados fines". También encontramos otras definiciones como:

"Formación o entidad social con un número de miembros que puede ser precisado y una diferenciación interna de las funciones que son desempeñadas por dichos miembros" [Peiró,1992].

"La organización es la coordinación de las actividades de todos los individuos que integran una empresa con el propósito de obtener el máximo aprovechamiento posible de elementos materiales, técnicos y humanos, en la realización de los fines que la propia empresa persigue" [Guzmán, 1983].

"Grupo cooperativo de seres humanos que asigna las tareas entre los miembros, identifica las relaciones e integra sus actividades hacia objetivos comunes, de manera estructurada" [Massie, 1973].

Por tanto, una organización consiste en una serie de individuos que realizan unas actividades o funciones específicas y diferenciadas. Además, estos individuos se estructuran siguiendo unas pautas y reglas determinadas que les permitan alcanzar los fines u objetivos de la organización. Los fines deben ser objetos de conocimiento común, que cumplan principalmente las funciones de guiar los esfuerzos de los miembros en vistas a su consecución [Peiró, 1991]. Asimismo, deben proporcionar una fuente de legitimidad que determine las conductas y acciones adecuadas en el contexto de la organización y establezcan los niveles mínimos o estándares que han de conseguirse [Argente, 2008a].

De forma esquemática, la organización humana se caracteriza por [Hodge *et al.,* 2003]:

- Está compuesta por personas.
- Persigue unos fines determinados, que guían las actividades de sus miembros, a través de mecanismos de coordinación y control de la acción.
- Existe una subdivisión del trabajo entre los individuos, mediante la especialización y la división de las tareas.
- Precisa de una estructura formal, con unos roles definidos (independientes de la persona que realice dicho rol); unas responsabilidades asociadas a esos roles; y ciertas relaciones preestablecidas entre los miembros de la organización.
- Todas las actividades establecidas deben relacionarse con los objetivos globales de la organización, pues la existencia de un rol determinado sólo es justificable si sirve para alcanzar realmente esos objetivos.
- Posee unos límites definidos, que establecen quiénes son los miembros de la organización (bien por la especificación directa de cada miembro; o bien por el lugar donde se realiza la actividad).

Una especialización dentro de las organizaciones humanas sería la aplicable al ámbito empresarial. En este tipo de organizaciones toma especial relevancia el concepto de organigrama (coincidente con lo que se ha nombrado como estructura). Según la RAE organigrama se define como "sinopsis o esquema de la organización de una entidad, de

una empresa o de una tarea". En una empresa se traduce como las relaciones jerárquicas existentes entre los diferentes miembros de la misma. Los esquemas organizativos varían en función de las variables tenidas en cuenta. Así, existen clasificaciones atendiendo a la naturaleza de las variables, finalidad, ámbito, contenido y representación [De Zuani, 2005]. Si en lugar de estudiar la naturaleza se estudia el nivel organizativo seguido en la descomposición de las responsabilidades se tienen diferentes estructuras asociadas a la organización. Las estructuras de las organizaciones no son fijas, van variando a medida que la empresa se consolida pasando por diferentes tipos de organizaciones. Aquí es donde cobran importancia la dinamicidad y la adaptación. Cualquier organización debe permitir que sus miembros puedan entrar y salir de ella de forma dinámica. La organización incorpora a sus miembros en función de unas determinadas habilidades, conocimientos o aptitudes que se crean útiles para la consecución de sus fines [Peiró, 1991].

Los diferentes tipos de organización son [Bueno, 2009]: estructura simple, funcional, divisional, matricial y en red.

La **estructura simple** se corresponde con inicios de la organización asociados a empresas pequeñas sin muchos empleados en las que la dirección de la empresa está a cargo de los propios propietarios. La propia dirección se encarga de cohesionar los diferentes puestos organizativos llevando a cabo una supervisión directa de las tareas realizadas en los diferentes puestos de trabajo habiendo poca especialización de los puestos de trabajo. Estas empresas se caracterizan por una toma de decisiones muy ágil y que se adaptan de manera sencilla a los cambios del entorno. Sin embargo, no es aplicable a grandes empresas por la sobrecarga de trabajo que le supondría a la dirección y está muy supeditada a las capacidades de la dirección de la empresa.

La **estructura funcional** se puede descomponer en estructura funcional centralizada y descentralizada**.** La **estructura funcional centralizada** es adecuada para organizaciones que desempeñen tareas no muy variadas en las que es posible realizar una descomposición jerárquica de las responsabilidades sin necesidad de comunicaciones entre niveles homólogos de modo directo. En este tipo de organización se hace una departamentalización por funciones siguiendo una agrupación lógica según las actividades en la que se fomenta la especialización del personal. Esta estructura tiene como principal problema el elevado coste de formalización y centralización siendo pocos adaptables a entornos cambiantes. La **estructura funcional descentralizada** al igual que la centralizada es adecuada para organizaciones que desempeñen tareas no muy variadas. Normalmente tienen una estructura más plana en la descomposición jerárquica lo que provoca la necesidad de tener un mayor nivel de preparación por parte de los miembros.

La **estructura divisional** está asociada con organizaciones que siguen estrategias de diversificación. Se lleva a cabo una departamentalización o divisiones en base a los al mercado lo que incluye productos, áreas geográficas, cliente. Las partes superiores de

las jerarquías de mando realizan actividades de control, coordinación y apoyo a las divisiones. Tienen como principal ventaja que cada división se orienta a actividades concretas asociadas con su división aunque la división implica replicación de actividades y recursos.

En la **estructura matricial** se da una mezcla de la estructura funcional y la estructura divisional. Se realiza una departamentalización por mercados y a su vez una departamentalización funcional por las diferentes divisiones. La división por mercados proporciona la facilidad de adaptación y flexibilidad de la organización permitiendo la inclusión de nuevas divisiones. La división funcional dota por otro lado de estabilidad y de personas especializadas a la organización proporcionando una mayor eficiencia, sin embargo, causan conflictos de autoridad y la comunicación es compleja.

Finalmente se tiene la **estructura en red**, que se caracteriza por:

- *Fronteras permeables*. Las fronteras internas (por ejemplo, entre departamentos o unidades de negocio) y externas (con empresas proveedoras, competidoras...) se difuminan o permeabilizan.
- *Aplanamiento jerárquico*. La gestión se vuelve menos jerárquica y el trabajo colaborativo crece, a medida que la jerarquía se desdibuja. Aparecen nuevos métodos y canales para estimular acciones, disminuyen las diferencias entre los distintos niveles (las relaciones son más horizontales y menos verticales) y aumenta la importancia de las relaciones externas como fuente de poder e influencia.
- *Compromiso y confianza*. La pertenencia a esta organización en red implica necesariamente un alto grado de compromiso y confianza entre las diferentes partes involucradas, sabiendo que, por lo general, los posibles conflictos que se puedan producir se solucionarán mediante protocolos de negociación previamente establecidos entre los miembros participantes.
- *Comunicación directa*. En un entorno en red, gana importancia la comunicación directa, más “punto a punto”. Las organizaciones red se conciben como entornos ricos en comunicación, con flujos de información que distorsionan las fronteras tradicionales.
- *Procesamiento de información masiva,* que se traduce en la capacidad de generar conocimiento diferencial

Otros aspectos importantes que caracterizan a una organización empresarial desde el punto de vista estructural son:

- La **especialización o división del trabajo** indica el grado en el que las tareas de la organización se dividen en cometidos separados. Cuanta mayor especialización exista, más rutinarias serán las tareas.

- La **departamentalización** consiste en agrupar los trabajos de manera que se puedan coordinar las tareas comunes. Dicha agrupación se realiza de distintos modos: (i) por funciones, que consiste en reunir a los especialistas en los mismos departamentos, para así lograr economías de escala al colocar a gente con habilidades y orientaciones similares en una misma unidad; (ii) por producto, que agrupa tareas en departamentos según el producto o servicio que la organización genera, aumentando así la responsabilidad por el desempeño del producto; (iii) por geografía, organizando los departamentos por regiones o territorios; (iv) por procesos, de modo que cada departamento se especializa en una fase de la producción; o (v) por tipo de cliente, de forma que se puedan satisfacer mejor los problemas y necesidades de cada cliente.
- La **cadena de mando** representa la línea continua de autoridad que se extiende desde la parte superior de la empresa al último peldaño y aclara quién informa a quién. La autoridad se refiere al derecho para dar órdenes y esperar que sean obedecidas. El tramo de control, por contra, determina el número de empleados que un gerente dirige con eficacia y eficiencia, estableciendo así el número de niveles y gerentes de las empresas.
- El **grado de centralización** de las empresas es también un aspecto importante de su estructura, pues indica dónde se toman las decisiones. Así, el término centralización se refiere al grado en que la toma de decisiones se concentra en un único punto en la organización. Por su parte, la descentralización implica que las decisiones se delegan a los gerentes que están más cerca de la acción.
- La **formalización** indica el grado en que los puestos de la organización están estandarizados, es decir, la cantidad de reglas organizativas, procedimientos de los procesos de trabajo y descripciones explícitas de los puestos de trabajo que se tiene. En concreto, describe los patrones de interrelación establecidos entre los miembros, que permiten coordinar sus tareas.

En cuanto a la finalidad, en una organización humana y especialmente en una empresarial, la misión describe el motivo de la existencia especificando cuáles son los resultados que proporciona, quiénes son los grupos de interés a los que se dirige y qué beneficios globales se espera obtener. Por tanto, determina: (i) los objetivos globales del sistema; (ii) los servicios que se ofrecen o requieren, así como los productos asociados a dichos servicios; y (iii) los clientes, usuarios, etc. que se verán afectados por el sistema.

Los fines de la organización deben ser objeto de conocimiento común y deben guiar los esfuerzos de los miembros en vistas a su consecución [Peiró, 1991].

La normatividad necesaria para conseguir una buena coordinación se consigue principalmente a través de tres mecanismos: adaptación mutua, supervisión directa y normalización [Wagner y Hollenbeck, 2004]. Con la adaptación mutua, los miembros comparten la información relacionada con su trabajo y deciden entre sí sobre el modo en que se debería realizar una tarea y quién debería realizarla. En la supervisión directa, una persona asume la responsabilidad del trabajo de un grupo, adquiriendo autoridad para decidir qué tareas han de realizarse, quién las realizará y cómo se vincularán para producir el resultado final. Por su parte, la normalización proporciona los estándares y procedimientos para ayudar a los miembros de la organización a determinar cómo realizar sus tareas. Existen cuatro tipos de normalización: de tareas, de resultados, de habilidades y de comportamientos [Wagner y Hollenbeck, 2004]. La normalización de tareas implica la especificación de las tareas concretas y los procedimientos de trabajo que los empleados deben seguir para cumplir con sus responsabilidades. En la normalización de resultados, se definen formalmente los objetivos de los resultados o las metas de rendimiento esperadas, por lo que los miembros de la organización tienen mayor flexibilidad sobre la funcionalidad de las tareas a realizar, siempre que sus resultados sean acordes a las metas especificadas. En la normalización de habilidades, se indican las cualificaciones, conocimientos y habilidades necesarias para desarrollar las tareas de forma competente. Finalmente, en la normalización de los comportamientos, los miembros de un grupo u organización comparten un conjunto de creencias sobre la aceptabilidad de determinados tipos de comportamiento, como por ejemplo la calidad en los procesos de trabajo [Argente, 2008b].

En una organización empresarial, el entorno abarca todo aquello que se encuentra fuera de la misma: sus proveedores, clientes, la competencia, los organismos gubernamentales que regulan sus operaciones, las instituciones financieras e inversores que le suministran fondos y el mercado laboral que aporta a sus empleados. También son parte del entorno las condiciones económicas, geográficas y políticas [Wagner y Hollenbeck, 2004]. Por tanto, el entorno es la fuente de recursos necesarios para sobrevivir [Hodge *et al.,* 2003], pues le proporciona los materiales, tecnología e incluso los miembros requeridos para desarrollar sus productos y servicios, así como los clientes suficientes que consuman dichos productos y ofrezcan beneficios a la organización. Por tanto, todas las oportunidades de éxito y amenazas para su existencia provienen del entorno [Hodge *et al.,* 2003].

3.1.2 Organizaciones de agentes

A partir de la similitud entre las denominaciones, se plantea el estudio de la similitud entre las organizaciones formadas por humanos y las formadas por agentes. Existe un amplio rango de investigaciones enfocadas a la construcción de organizaciones virtuales

que simulen comportamientos de comunidades ya sea de agentes o humanos [Sansores y Pavón, 2005] [Iglesias, 2010] y que ponen de manifiesto la analogía entre organizaciones humanas y organizaciones de agentes. Tras haber iniciado este apartado con definiciones del término organización desde el punto de vista humano, se presentan a continuación algunas definiciones desde el punto de vista de agentes:

"Una organización proporciona un marco de trabajo para la actividad e interacción de los agentes a través de la definición de roles, expectativas de comportamiento y relaciones de autoridad, como el control." [Gasser y Ishida 1991].

"Una organización es una colección de roles, que mantienen ciertas relaciones entre sí y que toman parte en patrones de interacción con otros roles de forma institucionalizada y sistemática" [Zambonelli *et al.,* 2003].

"Una organización se entiende como un conjunto de restricciones de comportamiento que un grupo de agentes adopta para controlar la autonomía de los agentes y conseguir alcanzar sus objetivos globales fácilmente" [Hübner *et al.,* 2006].

"La organización de agentes proporciona una forma para dividir el sistema, separándolo en grupos o unidades que constituyen el contexto de interacción de los agentes. Además, la organización se basa en dos aspectos: estructural y dinámico. La estructura de la organización representa todo aquello que persiste cuando los componentes e individuos entran o salen de la organización, es decir, el conjunto de relaciones que permiten ver un agregado de elementos como un todo. Mientras que la dinámica organizativa se centra en los patrones de interacción institucionalizados que se definen para los roles, describiendo la manera de entrar y salir de la organización, las obligaciones y permisos requeridos a los roles, así como el modo en el que se realiza la asignación de roles a los agentes." [Ferber *et al.,* 2003].

En definitiva, las organizaciones de agentes asumen la existencia de objetivos globales, aparte de los objetivos individuales de los agentes y además existen de manera independiente a los agentes, sin asumir ninguna característica interna específica de los mismos [Dignum y Dignum, 2006]. Los roles representan posiciones organizativas responsables de alcanzar parte de esos objetivos globales, de acuerdo a ciertas reglas de interacción predeterminadas. Los agentes pueden tener sus propios objetivos y decidir si adoptan o no unos roles específicos del sistema, así como determinar cuáles de los protocolos disponibles resultan más apropiados para alcanzar los objetivos marcados por esas posiciones o roles que han adoptado. Podemos decir que los puntos clave de una organización de agentes son [Argente, 2008b]:

- Está compuesta por agentes (software, físicos y/o humanos), con independencia de sus características internas y objetivos individuales.
- Persigue un objetivo común global, que no depende directamente de los objetivos individuales de los agentes particulares que participen en ella en cada momento.

- Existe una subdivisión de las tareas de los agentes, mediante su asignación a roles, los cuales describen las actividades y funcionalidad propia de la organización.
- Proporciona una partición del sistema en grupos o unidades, en los cuales tienen lugar la interacción de los agentes.
- Posee unos límites bien definidos, determinados por: el entorno de la organización; los agentes internos y externos; así como por la funcionalidad de la organización y sus servicios ofrecidos.

En las organizaciones de agentes la estructura de la organización se define normalmente en base a roles y grupos:

- Los **roles** representan las funcionalidades o actividades de los agentes. Se suelen especificar distintos tipos de dependencias entre los roles: de herencia, de compatibilidad, de comunicación y coordinación, de autoridad o poder, de control, etc. Estas dependencias determinan las relaciones a seguir entre los roles, a través de las cuales se coordinan las acciones de los agentes. Un rol conlleva conjunto de restricciones que un agente debe seguir cuando acepta entrar en un grupo jugando dicho rol. Dichas restricciones afectan a su relación con otros roles, así como a los objetivos y planes que debe seguir.
- Los **grupos** especifican el contexto o localidad para las actividades de los agentes, de modo que la comunicación se lleva a cabo dentro de los grupos [Dignum y Dignum, 2006]. Un grupo consiste en un conjunto de relaciones y roles, determinando restricciones de cardinalidad (mínimo y máximo número de agentes jugando un determinado rol dentro de un grupo), así como relaciones de herencia y compatibilidad.

También se puede tener en cuenta para definir la estructura organizativa la topología y régimen de control [Zambonelli *et al.,* 2000]. La topología determina los patrones de interacción entre los miembros, mientras que el régimen de control especifica dónde recae la autoridad y cómo se produce la supervisión y monitorización de las tareas.

En cuanto a la funcionalidad, en las organizaciones de agentes se definen también, al igual que en las humanas, unos objetivos globales que especifican el comportamiento general deseado por el sistema y unos objetivos particulares de los roles y grupos, para los cuales se establecen el conjunto de tareas y acciones que permiten alcanzaros. Otro aspecto a tener en cuenta en las organizaciones es el concepto de servicio, que se define como un bloque coherente de funcionalidad que se realiza sirviendo a otra entidad.

La normatividad necesaria para la coordinación entre agentes se consigue mediante el uso de normas sociales. Éstas describen el comportamiento esperado de los

miembros, es decir, qué acciones son permitidas, requeridas o necesarias y cuáles se deben evitar. También indican las sanciones que se deben aplicar en el caso de acciones no deseables, así como las recompensas o reconocimientos a ofrecer para aquellas acciones cumplidas según lo establecido por la normas. Las normas resultan indispensables para solucionar problemas de coordinación en sistemas grandes, complejos y heterogéneos, en los que el control social directo y total no puede ejercerse [López *et al.,* 2006].

Para fomentar el dinamismo en estas organizaciones se deben establecer mecanismos que controlen cuándo los agentes pueden entrar a formar parte de la organización y cuál será su posición en ella (qué roles adoptarán y en qué grupos se integrarán). También se deben considerar los procesos de expulsión de los agentes con comportamientos anómalos y el proceso de creación y eliminación de las distintas agrupaciones y unidades.

Finalmente, el entorno de las organizaciones de agentes viene determinado principalmente por los recursos y aplicaciones de los que hacen uso los agentes. También pertenecen al entorno los denominados stakeholders o entidades externas a la organización, pero que se benefician, dependen o facilitan la existencia de la misma. Se incluyen en este grupo a los proveedores, clientes y demás usuarios de la organización.

3.2 Organizaciones Virtuales

Una vez establecido el concepto de organización podemos definir qué es una organización virtual (OV). De manera general podemos encontrar definiciones como esta:

"Una Organización Virtual es un conjunto de individuos e instituciones que necesitan coordinar sus recursos y servicios dentro de unos límites institucionales" [Foster *et al.,* 2001] [Boella *et al.,* 2005].

Si lo centramos en el ámbito empresarial, una organización virtual consiste en la contratación por parte de una empresa, de otras empresas que le proporcionen las principales funciones comerciales que necesita [Robbins, 2009]. Así, se crean redes de contactos que les permiten contratar la manufactura, la distribución, la mercadotecnia y otras funciones comerciales que la gerencia cree que otros pueden realizar mejor o a menor costo. Esta estructura ofrece flexibilidad pero reduce el control de la gerencia sobre partes fundamentales de la organización, ya que dicho control lo ejercerá cada una de las empresas subcontratadas. Se distinguen tres tipos de organización virtual [Fox, 1981]:

- **Mercado simple**: consiste en varias organizaciones disjuntas que negocian la producción o contratación de servicios, estableciendo contratos. De este modo, una empresa u organización no necesita crear una nueva unidad para cada nueva función que tenga, sino que contrata dicha función en el mercado.
- **Organización colectiva**: consiste en varias organizaciones separadas que cooperan para alcanzar objetivos comunes, estableciendo contratos a largo plazo. Las organizaciones deciden juntas cómo adaptarse a las situaciones. Además, la información se comparte por el grupo y no aparece el oportunismo.
- **Mercado general**: varias organizaciones contratan entre sí servicios para periodos de tiempo cortos o largos, estableciéndose una gran competitividad. Además, cada organización debe examinar si sus propios objetivos se corresponden con las necesidades de otras organizaciones.

Una organización virtual ofrece una gran flexibilidad y capacidad de adaptación a cambios del entorno y de los objetivos de la propia organización. Sin embargo, resulta más complicada de controlar y mantener que las estructuras anteriores, pues el control está totalmente distribuido y además muchas de las funciones de la organización se delegan o subcontratan a otras organizaciones [Argente, 2008b].

Por lo tanto, una OV es un sistema abierto formado por la agrupación y la colaboración de entidades heterogéneas y donde hay una separación entre la forma y la función que define su comportamiento [Rodríguez, 2010].

La tecnología multiagente, la cual permite la formación dinámica de organizaciones de agentes, es particularmente adecuada para el desarrollo de este tipo de sistemas. El modelado de organizaciones basadas en SMA abiertos no sólo hacen posible describir la composición estructural del sistema (por ejemplo roles, agentes, grupos, tareas, planes o servicios), sino también las normas y reglas para el control del comportamiento de los agentes, la entrada/salida dinámica de los componentes y la formación, también dinámica, de grupos de agentes.

El desarrollo de SMA abiertos es un campo de investigación en el paradigma de agentes y su desarrollo está permitiendo aplicar esta tecnología en una variaedad dominios complejos [Hernández *et al* 2006] [Jiao y Mei, 2004]. Ello pone de manifiesto el interés de investigar nuevos métodos para lograr la perfecta coordinación de todos los miembros y el máximo aprovechamiento de los recursos de una organización. Por ello, este trabajo presenta un sistema de planificación que permitirá resolver las tareas que se le presenten a una organización de una manera eficiente.

3.2.1 Organizaciones Virtuales de Agentes

Las organizaciones virtuales poseen todas las características de los sistemas abiertos de los que se habló con respecto a los sistemas multiagente. Algunas características básicas en las que las organizaciones son vistas como sistemas abiertos son:

- En el crecimiento.
- En el hecho de volverse más complejas a medida que crecen.
- En el hecho de que haciéndose más complejas, sus partes exigen una creciente interdependencia.
- Su vida tiene mayor extensión comparada con la vida de sus unidades componentes.
- En ambos casos existe creciente integración acompañada por creciente heterogeneidad.

Schein [2010] propone una relación de aspectos que deberían considerarse en la creación de una organización y que definen en su conjunto el concepto de "modelo social" desde una perspectiva general:

- La organización debe ser considerada como un sistema abierto.
- La organización debe ser concebida como un sistema con objetivos o funciones múltiples.
- La organización debe ser visualizada como una constitución de muchos subsistemas que están en interacción dinámica unos con otros.
- Al ser los subsistemas mutuamente dependientes, un cambio en uno de ellos, afectará a los demás.
- La organización existe en un ambiente dinámico que comprende otros sistemas.
- Los múltiples eslabones entre la organización y su medio ambiente hacen difícil definir las fronteras de cualquier organización.

Las organizaciones virtuales de agentes se modelan teniendo en cuenta los siguientes conceptos clave [Rodríguez, 2010]: entidad social, estructura, funcionalidad, normatividad, entorno, dinamicidad, adaptación social y aprendizaje social. En los siguientes puntos se definen cada uno de estos conceptos.

3.2.1.1 Entidad social

Las organizaciones están formadas por componentes o entidades sociales que pueden estar compuestos a su vez por un número específico de miembros o agentes. Estas entidades, según [Pattison *et al.,* 1987]:

- tienen unas responsabilidades, es decir, un conjunto de sub-tareas que llevar a cabo, incluidas dentro de los objetivos de la organización,
- tienen y consumen recursos. Los componentes poseen ciertos recursos con los cuales llevan a cabo sus tareas. Los recursos requeridos por un componente dependerán del rol que esté jugando en ese momento en la organización,
- están estructurados siguiendo determinados patrones de comunicación,
- tratan de alcanzar los objetivos globales de la organización y
- están reguladas por normas y restricciones.

3.2.1.2 Estructura

Las entidades en una organización no son independientes unas de otras. Interactúan entre sí. Comandos, información, etc. son unidades pasadas entre ellas. Estas interacciones son expresadas como relaciones entre los componentes. En general, estas relaciones no vienen dadas de manera individual dentro de una organización, sino que se requiere una conjunción de relaciones entre grupos de entidades.

La estructura puede definirse como la distribución, orden e interrelación de las diferentes partes que componen la organización. En esa estructura los agentes se ordenarán y comunicarán dependiendo de la topología que defina esta estructura. Existen diferentes topologías de organización, entre ellas jerarquías, holarquías, coaliciones, grupos, congregaciones, federaciones. etc.

3.2.1.3 Funcionalidad

La funcionalidad de una organización está determinada por su misión, es decir, por los objetivos globales que describen la finalidad de su propia existencia. La misión define la

estrategia, los requerimientos funcionales (qué hace la organización) y de interacción (cómo lo hace).

Los objetivos se pueden clasificar en:

- *Funcionales*: de cada grupo o unidad organizativa.
- *Operativos*: de los agentes, sus planes (tareas que llevarán a cabo).

3.2.1.4 Normatividad

Las normas sociales definen las consecuencias de las acciones de los agentes:

- Restricciones sobre la organización.
- Obligaciones, sanciones a aplicar.
- Control de accesos externos.
- Descomposición:
  - Acciones que provocan la activación de la norma.
  - Conjunto de obligaciones que adquiere el agente.
  - Acciones que se deben llevar a cabo para eliminar la obligación.

3.2.1.5 Entorno

El entorno define qué existe alrededor del sistema: recursos, aplicaciones, objetos, supuestos, restricciones, *stakeholders* (proveedores, clientes, beneficiarios). Definiendo el entorno se establece la relación de los roles respecto a los elementos del entorno: modo de acceso (lectura, interacción, extracción de información), permisos de acceso, etc.

3.2.1.6 Dinamicidad

La dinámica organizativa está relacionada con la entrada/salida de agentes, con la adopción de roles por parte de los mismos, la creación de grupos y con el control del comportamiento.

En la definición de la dinámica de una organización habrá que especificar:

- *Respecto a la entrada de agentes*: cuándo permitir a los agentes entrar en la organización; cuál será su posición en la organización; los procesos de expulsión de agentes con comportamientos anómalos. Por ejemplo: en el modelo definido por [Esteva *et al.,* 2002] existe un agente *Institution Manager* que autoriza a agentes externos a entrar en la institución.
- *Respecto a la adopción de roles*: cómo los agentes adoptan un rol concreto; la asociación de agentes a uno o más roles. Por ejemplo, en [Esteva *et al.,* 2002], se describen las transiciones entre escenas en función de roles; cambios de roles.
- *Respecto a la creación dinámica de grupos*: la definición de federaciones, coaliciones, congregaciones, etc.
- Por último, *respecto al control del comportamiento*: cómo controlar la conformidad del comportamiento de los agentes a las normas de la sociedad. Por ejemplo, en [Esteva *et al.,* 2002] existe una capa social que garantiza que las interacciones se producen de acuerdo a las normas.

3.2.1.7 Adaptación social

La adaptación para una sociedad es su capacidad para involucrase con el entorno y hacer parte de éste una simbiosis que permita a ambos disponer uno del otro. Y no sólo puede considerarse como una capacidad, sino también como una necesidad de involucrarse al entorno para aprovechar al máximo las necesidades de aprendizaje de cada individuo y que llevará al sistema a adquirir un aprendizaje significativo.

3.2.1.8 Aprendizaje social

El aprendizaje social se percibe como un proceso en el cual, gracias a un ambiente o entorno en común (proveído por una sociedad artificial), diferentes entes pueden interactuar y evaluar sus experiencias e información, teniendo en cuenta que no es un proceso crítico de un ente hacia otro en donde se determina si éste tiene información correcta o no [Duong y Grefenstette, 2005], sino que cada miembro de la sociedad artificial observa al otro como una simple fuente de datos, en donde según la función de utilidad o el objetivo propuesto para el ente que está aprendiendo se define la pertinencia de los datos encontrados y, de esta manera, decide qué aprender y qué no [Conte y Paolucci, 2001]. Un ente que aprende de otro ente está ahorrando esfuerzo en experimentación y tiempo al captar información obtenida mediante el esfuerzo de otros.

3.2.2 Planificación y adaptación

Cada organización necesita de un soporte de coordinación que determine explícitamente cómo deben organizarse y llevar cabo las acciones y tareas dentro de la misma. Un *mecanismo de coordinación* determina el modo en el que uno o varios agentes realizan las tareas [Ossowski, 1999].

Se han propuesto mecanismos para manejar la coordinación en sistemas multiagente de modos muy diferentes [Wooldridge y Jennings 1995]. Los más utilizados han sido los métodos "mediados". Al igual que ocurre en muchos ámbitos de las sociedades humanas, los agentes intermediarios juegan un papel importante en las sociedades artificiales. Estos agentes proporcionan medios para facilitar la coordinación entre agentes en entonos abiertos, mediante servicios de localización y comunicación entre agentes que proporcionan servicios, y agentes que los solicitan [Klusch y Sycara, 2001].

La utilización de THOMAS en este trabajo impone que las organizaciones estén compuestas por Unidades Organizativas, que proporcionan limitaciones comunicativas y de visibilidad a los agentes. Estas Unidades Organizativas pueden ser principalmente de tres tipos: *jerarquía*, *equipo* o *plana*. En las jerarquías, un agente supervisor tiene el control sobre el resto de miembros, coordina las tareas y centraliza la planificación y toma de decisiones.

La realización de las tareas por parte de los agentes, puede ser organizada de varias formas, por ejemplo: cada agente ejecuta una de las tareas, o bien, las tareas son

divididas en sub-tareas, por medio de algún mecanismo de descomposición de problemas, y éstas son las realizadas por los agentes. En una organización, las tareas que debe realizar un agente dependen, entre otros factores, del rol que este agente asume en el sistema. Por ejemplo, en un sistema "oficina", un agente "persona" asume el rol de "secretaria" y realiza las labores relacionadas con ese rol; este mismo agente podría asumir el rol de "jefe", si estuviera capacitado, y realizaría labores muy diferentes, relacionadas con su nuevo rol. Para la realización de tareas, un agente puede necesitar recursos del sistema, en este caso tiene que coordinarse con los otros agentes del sistema que deseen usar el mismo recurso.

La coordinación en sistemas multiagente ha sido, y es, un reto para los investigadores relacionados con la tecnología de agentes. Existen numerosos modelos de coordinación [Tambe, 1997]. Todos los enfoques están basados en las observaciones de equipos de trabajo humanos. El enfoque de coordinación propuesto por la teoría de los planes compartidos [Grosz y Kraus, 1996] es el que más se asemeja al modelo de planificación propuesto en este trabajo. La formalización de los planes compartidos enfatiza la necesidad de un modelo de equipo común y de alto nivel que permita a los agentes comprender todos los requisitos de los planes que podrían llevarse a cabo en el sistema, como objetivo de grupo. Esto permite a los miembros del equipo poner a disposición sus capacidades para llevar a cabo el plan y conseguir el objetivo global.

Existen muchos *modelos de coordinación*, los dos principales se mencionan a continuación y se completan con gran cantidad de modelos intermedios.

a) **Coordinación Global**. El sistema multiagente determina y planifica globalmente las acciones de los diferentes agentes.

b) **Coordinación Individual**. Los agentes tienen completa autonomía. Cada agente decide qué hacer y resuelve localmente los conflictos que detecte con otros agentes.

En este trabajo se presenta una aproximación para llevar a cabo una coordinación global dentro de una organización desarrollada mediante THOMAS que, como se verá a continuación, es una arquitectura basada en conceptos organizacionales.

Para que una organización pueda adaptarse rápidamente a los cambios de su entorno, los agentes deben poder coordinarse cuando sea necesario realizar cambios en sus metas o los roles que tienen asignados. Es decir, es necesario que *la organización se adapte*. Esta afirmación, transforma la tarea de coordinación de los agentes en un aspecto fundamental de cualquier sistema multiagente [Doran *et al.,* 1997].

En el caso de un sistema cooperativo (coordinación orientada por cooperación), la coordinación consistirá en la planificación de la distribución de las tareas, mientras que en el caso competitivo (coordinación orientada por problema) la coordinación consistirá de una negociación entre los agentes [Jennings *et al.,* 1998] [Rueda *et al.,* 2002].

Por lo tanto, una organización virtual puede verse como un sistema cooperativo, en el que la coordinación se basa en una planificación y distribución de tareas. La coordinación de tareas compartidas o tareas que se combinan para resolver un problema común, requerirá o bien de una planificación centralizada, o bien de una planificación distribuida llevada a cabo por los propios agentes del sistema. Esto constituye un problema abierto, ya que los sistemas clásicos de planificación [McAllester y Rosenblatt,1991] [Penberthy y Weld, 1992] no son apropiados por varios motivos:

- Asumen que los agentes tienen un conocimiento completo de su entorno.
- Asumen que las acciones no van a fallar.
- Asumen que el entorno cambiará únicamente por la ejecución de las acciones del agente.

Ninguna de estas tres suposiciones puede asumirse de forma realista en un sistema multiagente, por lo tanto se requiere de una planificación que pueda adaptarse a estas circunstancias.

El modelo de adaptación de la organización propuesto utiliza mecanismos basados en interacciones directas para reorganizar los agentes. Además, la utilización de la plataforma THOMAS [Carrascosa *et al.,* 2009] [Giret *et al.,* 2010], nos acerca a nuevos métodos de adaptación basados en arquitecturas [Razavi *et al.,* 2005] en los que es posible modificar la estructura de la organización para adaptarse a los cambios.

En este trabajo la coordinación consistirá en la planificación de la distribución de las tareas de los agentes miembro de la organización. Además, esta coordinación permitirá la adaptación de la organización frente a nuevos objetivos.

3.2.3 Arquitectura THOMAS

Las organizaciones virtuales pueden considerarse sistemas abiertos formados por la agrupación y colaboración de entidades heterogéneas y donde existe una separación clara entre estructura y funcionalidad [Foster *et al.,* 2001][Boella *et al.,* 2005].

Aunque algunas de las plataformas hacen frente a los conceptos de organización mediante patrones de diseño y técnicas similares, la mayoría no pueden ser aplicadas directamente en el desarrollo de sistemas multiagente abiertos donde las estructuras organizacionales pueden cambiar dinámicamente en tiempo de ejecución. Las abstracciones y herramientas actualmente disponibles todavía no son lo suficientemente potentes para desarrollar sistemas abiertos que traten con problemas

dinámicos reales. Así, el principal problema para la implementación de una Organización Virtual es la falta de plataformas que dan soporte a este tipo de sistemas.

THOMAS [Thomas-Tin, 2013] surge por la necesidad de dar soporte a la necesidad de desarrollar arquitecturas con las características expuestas en la sección anterior para el desarrollo de sistemas multiagente abiertos desde el punto de vista organizativo [GTI-IA, 2009]. La arquitectura presenta la infraestructura necesaria para emplear los conceptos de la tecnología de agentes en el proceso de desarrollo, aplicando las técnicas de descomposición, abstracción y organización y teniendo en cuenta todos estos requerimientos. Es sobre THOMAS donde se propone el modelo de este estudio para permitir la evaluación del comportamiento de una organización virtual de tal manera que los agentes puedan adaptarse y reorganizarse dinámicamente.

La arquitectura está formada básicamente por un conjunto de servicios estructurados modularmente. THOMAS toma como base la arquitectura FIPA [O'Brien y Nicol, 1998], expandiendo sus capacidades respecto al diseño de organizaciones e impulsando la capacidad de los servicios. En THOMAS, existe un módulo con el único objetivo de manejar las organizaciones que hayan sido introducidas en la arquitectura y es introducida una redefinición del *FIPA Directory Facilitator*, capaz de hacer frente a los servicios de un modo más elaborado.

Los agentes acceden a la infraestructura ofertada por THOMAS por medio de una serie de servicios incluidos en lo que se denomina OMS (*Organization Manager Service*). Los componentes principales de THOMAS podemos verlos en la Figura 2 (adaptada de [GTI-IA, 2009]):

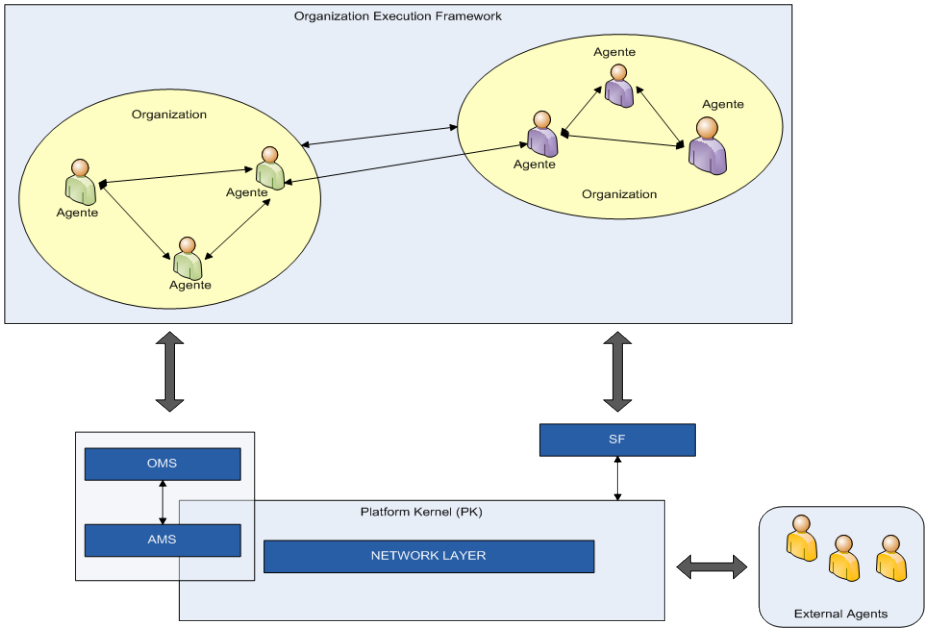

Figura 2 Arquitectura THOMAS (adaptada de [GTI-IA, 2009])

A continuación se explicarán los componentes principales de la arquitectura THOMAS: *Service Facilitator (SF), Organization Manager Service (OMS), Platform Kernel (PK)* y *Execution Framework*:

- ***Service Facilitator*** **(SF)**: este componente ofrece servicios simples y complejos a los agentes y organizaciones activas. Básicamente su funcionalidad se resume en ofrecer un directorio de páginas amarillas y verdes sobre los servicios disponibles. En general, un servicio ofrece unas capacidades, cada una de ellas encaminadas a cumplir un objetivo, que necesita que se verifiquen ciertas condiciones antes de su ejecución y que intercambian uno o más mensajes de entrada y salida y que tras su ejecución con éxito tiene una serie de efectos en el entorno
- ***Organization Manager Service*** **(OMS):** principalmente se encarga de la gestión de organizaciones y de las entidades incluidas en ellas. Permite por tanto la creación y gestión del ciclo de vida de una organización, incluyendo la especificación y administración de sus componentes estructurales (roles, unidades y normas) y sus componentes de ejecución (agentes participantes y roles que juegan, y unidades activas en cada momento).
    - Un *rol* representa una posición dentro de la unidad donde se define. Lleva asociado unas reglas de interacción, impuestas por la estructura de su unidad y la posición concreta que ocupa en ella, y unas reglas de comportamiento, que especifican su funcionalidad (tipos de servicios que ofrece y requiere), limitan sus acciones (prohibiciones, obligaciones y permisos) y determinan las consecuencias de las mismas (sanciones y recompensas). Los agentes pueden dinámicamente adoptar roles dentro de las unidades, por lo que el OMS controla este proceso de adopción de roles y controla qué entidades juegan qué rol en cada momento.
    - Una *norma* indica las obligaciones, permisos y prohibiciones de los roles respecto al registro, solicitud y desempeño de servicios, su composición, o bien la calidad de sus resultados. Permite definir aquellas restricciones que no se pueden expresar en las precondiciones (o postcondiciones) de un servicio determinado.
    - Una *unidad* representa a agrupaciones de agentes y admite la recursión (unidades dentro de otras). Establece así la estructura topológica del sistema. Por ejemplo, facilita la representación de estructuras de tipo jerárquico, matricial, coalición, etc.

- ***Platform Kernel* (PK):** permite mantener los servicios básicos de gestión de una plataforma de agentes. Los servicios de PK necesarios en la arquitectura THOMAS son clasificados en cuatro clases:
  - *Registration*: servicios necesarios para añadir, modificar y eliminar agentes nativos de la plataforma.
  - *Discovery*: servicios que proporcionan la funcionalidad de obtención de información sobre los agentes nativos en la plataforma.
  - *Management*: servicios para controlar el estado de activación de los agentes nativos de la plataforma.
  - *Communication*: servicios para la comunicación de agentes dentro de la plataforma y fuera de ella.
- ***Execution Framework***: permite a cualquier agente (desde cualquier tipo de plataforma) crear su propia organización virtual con una estructura y unas normas que manejar, junto con la oferta y la demanda de servicios necesarios. El *framework* manejará la estructura, las normas y el ciclo de vida de la organización, junto con la visibilidad de la oferta y demanda de servicios y el cumplimiento de las condiciones de uso.

3.2.4 GORMAS

GORMAS es una guía metodológica para el diseño de sistemas multiagentes abiertos desde la perspectiva de las organizaciones humanas. Esta guía se basa en la Teoría de Organización y en el Modelo de Organización [Argente *et al.,* 2008b]. Esta teoría aborda las propiedades de las organizaciones humanas y sus aspectos de diseño, mientras que el Modelo de Organización describe los principales aspectos de las organizaciones: estructura, funcionalidad, normalización, dinamicidad y entorno. Este modelo consta de un conjunto de metamodelos que emplean fundamentalmente los conceptos de unidad organizativa, servicio, norma y entorno. Además, contiene un conjunto de patrones de diseño con los que facilitar el modelado de la estructura de la organización.

GORMAS consta de un conjunto de fases que cubren el análisis, el diseño de la estructura organizativa y el diseño de la dinámica de la organización. Con estas fases se especifica cuáles son los servicios que ofrece la organización, cuál es su estructura interna y qué normas rigen su comportamiento. Las fases son las siguientes:

- **Fase A. Misión**: se realiza un análisis de la motivación que se persigue al definir la organización o sistema, es decir, del porqué de dicha

organización o para qué se crea; de los resultados que, en conjunto, se esperan conseguir; y del entorno en el que existe, detallando los productos y/o servicios a ofrecer, los grupos de interés y su localización.

- **Fase B. Tareas y procesos**: se analizan con mayor detalle los servicios a ofrecer en el sistema, sus requisitos y los procesos que conllevan. Se detallan también las tareas y objetivos asociados a dichos servicios.
- **Fase C. Dimensiones organizativas**: se analizan las dimensiones de la organización (departamentalización, especialización, sistema decisor, formalización, coordinación), que imponen ciertos requisitos sobre los tipos de trabajo, así como sobre la diversidad e interdependencia de las tareas a realizar.
- **Fase D. Estructura organizativa**: se determina y selecciona la estructura organizativa más adecuada para la organización, en función de sus dimensiones. Se hace uso de modelos organizativos para especificar los roles, interacciones y normas relacionados con la propia estructura.
- **Fase E. Procesos de información-decisión**: para cada servicio identificado, se detallan las interacciones (flujos de información y de adopción de decisiones) necesarios para llevar a cabo el servicio. Además, se definen los contratos de calidad de servicio a los que se comprometen los proveedores y consumidores cuando éste se lleva a cabo.
- **Fase F. Dinamicidad del sistema abierto**: se establece la funcionalidad ofrecida como sistema abierto, que incluye los servicios que se deben publicitar y las políticas de adquisición y liberación de roles. Además, se diseñan los agentes propios del sistema.
- **Fase G. Sistemas de medición, evaluación y control**: se cuantifican o evalúan las tareas y actividades y se establecen mecanismos para determinar si los objetivos del sistema se cumplen. Asimismo, se revisan las normas de la organización para especificar quiénes se encargan de ellas y las supervisan.
- **Fase H. Sistemas de recompensas**: se determina el sistema de incentivos, para recompensar a los miembros que avancen en dirección de los intereses de la organización. También se analizan los sistemas de sanción para aquellos miembros que no cumplan con las normas dadas.

La secuencia-guía propuesta permite ser integrada en un proceso de desarrollo de software completo, que comprende las fases de análisis, diseño, implementación, instalación y mantenimiento del sistema multiagente.

3.3 Conclusiones

Actualmente, investigaciones centradas en el diseño de sistemas multiagente desde el punto de vista organizacional están ganando terreno. Cada vez es más fuerte la idea de que modelar las interacciones de un sistema multiagente no puede estar relacionado solamente con el propio agente y sus capacidades de comunicación, sino que es necesario una ingeniería organizacional. Los conceptos de reglas [Zambonelli, 2002], normas e instituciones [Esteva *et al.,* 2001] y estructuras sociales [Parunak y Odell, 2002] nacen de la idea de que es necesario un nivel de abstracción mayor, independiente del agente y que defina explícitamente la organización en la que los agentes viven.

Los factores sociales, que permiten alcanzar acuerdos en las organizaciones de sistemas multiagente también son cada vez más importantes para estructurar las interacciones en mundos abiertos, dinámicos y cercanos a la realidad.

En definitiva, debido a las características de estos entornos abiertos, en especial a su dinamismo, son necesarios enfoques que permitan soportar la evolución de los sistemas y facilitar su crecimiento y actualización en tiempo de ejecución.

La utilización de organizaciones de agentes para el diseño y modelado de sistemas abiertos permitirá establecer qué estructura de roles y agrupaciones tendrá el sistema, qué mecanismos se adoptarán para la entrada y salida dinámica de sus componentes y cómo gestionar y controlar las normas que rigen el comportamiento de los miembros del sistema. Por ello, las organizaciones virtuales se perfilan como el mecanismo idóneo para reflejar el funcionamiento altamente dinámico de cualquier empresa o entidad donde la replanificación y la adaptación toman especial relevancia.

Capítulo IV

Arquitectura Propuesta

VNiVERSiDAD
D SALAMANCA

Introducción

En los capítulos anteriores se ha introducido una revisión del estado del arte de agentes, sistemas multiagente y organizaciones virtuales junto con las alternativas existentes en la inteligencia artificial para la creación de sistemas autoadaptables con capacidades de aprendizaje. En este capítulo se introducirá la arquitectura genérica propuesta y el modelo de planificación seguido para la generación del mecanismo de planificación y distribución de tareas de modo automático. La utilización de la plataforma THOMAS nos permitirá aprovechar métodos de adaptación en los que es posible modificar la estructura de la organización para adaptarse a los cambios

4.1 Modelo Propuesto

Los sistemas multiagentes abiertos son entidades que permiten el acceso de nuevos agentes que juegan determinados roles en organizaciones virtuales de agentes. Las organizaciones virtuales de agentes están formadas por un conjunto de agentes que evolucionan. La evolución incluye modificaciones en los roles que llevan a cabo los agentes durante el transcurso de su existencia en la organización. En este contexto de comportamiento dinámico de los agentes, es necesario establecer unos mecanismos que permitan controlar la evolución de los agentes y las acciones que realizan a medida que éstos varían su comportamiento. En este capítulo se va a exponer un modelo de arquitectura multiagente abierta que controla la conexión de agentes y asigna roles en función de las necesidades de la organización.

La arquitectura propuesta es un modelo organizativo orientado a entornos dinámicos que incorpora agentes con capacidades de generación de planes de trabajo y realiza la distribución de tareas de modo automático. El núcleo del sistema es un novedoso mecanismo de planificación basado en planes que interacciona con el modelo BDI a través de la implementación de servicios web que proporciona una capacidad autoadaptativa en diferentes entornos. Además, el sistema proporciona mecanismos de comunicación para facilitar la integración con arquitecturas SOA (Service Oriented Architecture) lo que permite separar la implementación de la funcionalidad de la implementación de los agentes.

Un ejemplo de aplicación del sistema propuesto, que exploraremos en el caso de estudio que se verá en el próximo capítulo, lo constituyen los entornos de administración electrónica e-Administración o *e-Goverment*. En los entornos de e-Administración se tienen que realizar diferentes actividades por el personal de la administración. El objetivo es asignar actividades a los diferentes agentes que están asociados a personal de la entidad administrativa. Una actividad típica podría ser gestionar un impuesto, una sanción o tareas similares, cada impuesto o sanción puede requerir diferentes trámites hasta que se finaliza el proceso. La terminología usada para cada una de esos trámites es la de tarea, es decir, cada vez que se haga referencia a tarea se refiere a uno de los trámites necesarios para la resolución de un determinado expediente.

El modelo propuesto se ha diseñado para desarrollar un mecanismo de planificación para coordinar a los agentes que se encuentran dentro de la OV. Para ello, se define una serie de roles y servicios adecuados a estas necesidades. Los agentes bajo determinados roles actúan como coordinadores y controladores de los servicios, mientras que los servicios son los responsables de implementar el comportamiento de los agentes y llevar a cabo el procesamiento de la información facilitando características como la replicación y la modularidad. En el sistema propuesto los roles se organizan de modo jerárquico de modo que la capa organizativa se integra por los roles propios de la plataforma, responsables de la distribución de las tareas, y en las capas inferiores se sitúan los roles tramitadores responsables de realizar las actividades. Los roles asociados a los agentes son los siguientes:

- **Tramitador**: estos agentes son los encargados de llevar a cabo las actividades que requiere cada tarea concreta. Las tareas que puede llevar a cabo el sistema son de diferente naturaleza y por lo tanto requieren diferentes tratamientos. Por esta razón, los agentes Tramitador se especializarán según los tipos de tareas que el sistema deba resolver.

- **Planificador**: es el encargado de diseñar el plan global que llevará a cabo la organización. Dependiendo del modelo de replanificación escogido, el planificador genera un plan que permite establecer el número de agentes Tramitadores (con sus respectivas especializaciones) que deben existir en la organización para cumplir con las tareas. Además, asigna cada tarea a un determinado agente para que el plan global sea lo más eficiente posible.

- **Distribuidor**: una vez el agente Planificador ha generado el plan, el agente Distribuidor se encarga de que todos los miembros de la organización se adapten a él. Para ello, cada vez que un agente Tramitador finaliza su tarea asignada, el Distribuidor se encarga de proporcionarle una nueva. También controla que cada tarea se esté realizando bajo los límites de tiempo establecidos para cumplir el plan, y en caso contrario, se encarga de avisar al agente Planificador para llevar a cabo una replanificación.

- **Coordinador**: realiza el control general del sistema. Se encarga de controlar la conexión, desconexión, fallos u otras excepciones de los agentes existentes. Asimismo, bajo petición del agente Planificador de encarga de introducir nuevos agentes o eliminar existentes cuando la organización necesite adaptarse por cambios en la cola de tareas con el fin de mantener el número adecuado de agentes con sus respectivos roles. Se comunica con los elementos de la plataforma THOMAS con el fin de llevar a cabo todas estas acciones.
- **Gestor**: este agente se encarga de la comunicación con el exterior. Contiene toda la información de las tarea a realizar y comunica al usuario final los resultados una vez el tramitador correspondiente ha finalizado su tarea.

La capa de servicios es la encargada de realizar la funcionalidad de los diferentes agentes, realizando de este modo una separación de la funcionalidad de la lógica de negocio. El comportamiento de los agentes se separa de los agentes ya que depende de los roles que están desempeñando. Para conseguir esta independencia, se implementa en servicios web que integran el comportamiento correspondiente a modelos CBP-BDI y modelos reactivos asociados a cada rol que son asignados por el rol distribuidor. Los modelos reactivos se basan en la definición de reglas de modo dinámico que se analizan y ejecutan mediante los denominados BRMS (Business Rule Management System). Los servicios disponibles asociados a roles de los agentes son planificador y reactivo. El servicio planificador integra las tareas del modelo CBP-BDI asociadas al rol planificador, mientras que el servicio reactivo, incorpora un motor de reglas para interpretar la lógica de negocio de la aplicación.

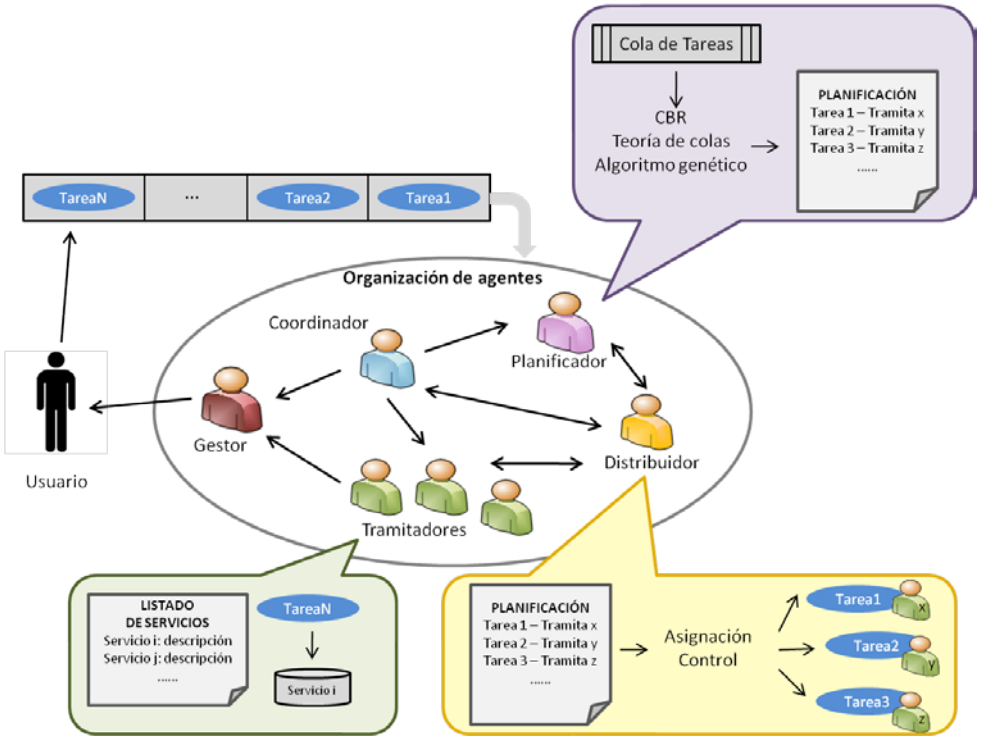

Figura 3 Esquema del modelo

Dentro de la capa de servicios, se tiene un servicio Directory Facilitator que proporciona la información sobre los diferentes servicios disponibles y gestiona el fichero XML de UDDI (Universal Discovery Description and Integration). Para facilitar comunicación entre agentes y servicios la arquitectura integra una capa de comunicación que proporciona compatibilidad FIPA-ACL y SOAP (Simple Object Access Protocol) [Bauer y Huget, 2003].

Los roles de la capa de organización son agentes con capacidad de planificación basada en planes bajo el modelo BDI por lo que todos ellos, disponen de la capacidad de generar planes de modo automático en base a los planes previos existentes en el sistema. Además, los agentes reactivos ajustan las reglas de negocio en función de los datos de planificaciones pasadas. El conjunto de los roles comparten una memoria de planes que es accedida y actualizada por ellos. La memoria de planes es actualizada al final de la jornada laboral indicada.

Todos los roles pueden ser desempeñados simultáneamente por varios agentes excepto el rol de Planificador. Es indispensable que el sistema cuente con un único agente que desempeñe el rol de Planificador puesto que no se contempla la opción de disponer de diferentes planes globales que se distribuyan entre los agentes. Sólo se contempla replicación del agente para garantizar la disponibilidad del sistema.

4.2 Modelado de la arquitectura

Uno de los principales problemas en el desarrollo de una arquitectura basada en sistemas multiagentes es que actualmente no existe un estándar claro o metodologías desarrolladas para definir los pasos de análisis y diseño que es necesario llevar a cabo para el desarrollo de los sistemas. Se han descrito distintas metodologías como Gaia [Zambonelli *et al* 2003], INGENIAS [Pavón *et al.,* 2007], TROPOS [Giorginia *et al.,* 2005] o MESSAGE [EURESCOM, 2001]. Para este trabajo, se analizó el uso de Gaia en combinación con el lenguaje de modelado AUML (Agent UML) AUML [Corchado y Laza, 2003] [Odell *et al.,* 2004] y de este modo tener las ventajas de ambos [Bajo *et al.,* 2009a] y la alternativa de manejo de la metodología GORMAS. Finalmente se optó por el uso de GORMAS debido que incorpora la especificación asociada a organizaciones virtuales de agentes que no se encuentra presente en Gaia.

4.3 Roles de la plataforma

Los roles de agentes descritos a continuación definen las actividades y el comportamiento que pueden realizar los agentes de la plataforma. Determinados roles sólo se pueden realizar por agentes determinados, como es el caso del rol de planificación que está fijado a un determinado agente y solo puede haber un agente con esta funcionalidad en un momento dado. Otros roles asociados con la organización se asignan de modo dinámico y se pueden replicar para mejorar el rendimiento del sistema. Finalmente, los roles tramitadores se pueden asignar y desasignar a los agentes de modo dinámico a medida que la cola de tareas varíe. La asignación de roles se realiza de modo que se maximice el rendimiento empleando el mínimo personal necesario. Para realizar la asignación se realizan previsiones de flujo de tareas.

En este apartado, se explican los roles seguidos por los agentes de la plataforma propuesta. Junto con los roles se explican los modelos de planificación seguidos por los agentes.

4.3.1 Rol Distribuidor

El rol distribuidor, como se indicaba anteriormente, es el encargado de realizar la asignación de tareas a los agentes con el rol tramitador. El agente parte de la lista de

tareas global y asigna por orden cada una de las tareas existentes al agente designado. El agente distribuidor también realiza tareas de control sobre las tareas asignadas para determinar si se producen retrasos relevantes.

Para cada una de las tareas asignadas a un determinado agente tramitador, el agente distribuidor controla su estado, almacenando la información de cada una de ellas:

$$C_d = \{t_i / t_i = (ids, idt, f, p, d, t, a), i = 1...n\} \quad (1)$$

Donde t_i presenta la tarea i, ids es el identificador del tipo de tarea, idt el trámite, f fecha de entrada, p plazo, d duración, t umbral de alarma para el que se estima que algo extraño está pasando. El umbral se define en función de la desviación obtenida para el tipo de tarea y agente en concreto, cuando se supera dos veces la desviación, se considera que hay que realizar un seguimiento de la tarea y hay que sondear al agente para comprobar el estado de cumplimiento de la misma y avisar al agente planificador en caso de detectar que se está produciendo una anomalía. También avisa al agente planificador en caso de que determine que es necesario realizar una replanificación por la llegada de nuevas tareas, incumplimiento de la planificación o llegada de una tarea urgente.

El comportamiento del agente para el rol es **reactivo**, a medida que se cumplen determinadas condiciones se ejecutan las acciones asignadas. El conjunto de reglas condición/acción se generan en base a la información de planificaciones anteriores realizadas por el agente.

4.3.1.1 Modelo reactivo

El modelo reactivo seguido por roles como el distribuidor o coordinador se fundamenta en el uso de reglas mediante la inclusión de un motor de reglas BRMS (*Business Rules Management System*). Los BRMS incluyen un conjunto de reglas que definen la lógica de negocio de la aplicación que se pueden externalizar, un conjunto de herramientas que permiten manejar y definir las reglas de modo sencillo y un motor de reglas que analice la información. Los motores de reglas permiten interpretar el comportamiento de los sistemas de modo dinámico sin ser necesario recompilar el código de las aplicaciones a partir de la definición de reglas de negocio que son interpretadas por motores en tiempo de ejecución. Las reglas de negocio tienen la apariencia mostrada en la Figura 4.

```
rule "nombre"
    atributos
    when
        condición de la regla
    then
        código a ser ejecutado
end
```

Figura 4 Definición de reglas

Dentro de la arquitectura se definen servicios que se encargan de realizar la interpretación de las reglas y su ejecución mediante la incorporación de un motor en su funcionamiento. De este modo también se desacopla el motor de reglas usado del comportamiento del agente pudiendo variarlo o incorporar nuevos motores sin necesidad de modificar el agente.

4.3.2 Rol Coordinador

El rol coordinador es desarrollado por el agente encargado de controlar los agentes conectados al sistema. Cuando detecta la conexión o desconexión de un agente determina la necesidad de realizar una replanificación y en tal caso avisa al rol planificador para que la realice. Este comportamiento es evidentemente **reactivo** y sólo necesita la definición de reglas básicas para su funcionamiento.

El rol coordinador además de controlar los agentes conectados al sistema determina los agentes más capacitados para realizar las tareas que son necesarias desarrollar. El rol recibe el número de agentes necesarios para realizar cada trámite y selecciona los agentes entre los disponibles. La selección de los agentes **no es un comportamiento reactivo** y se realiza en función de su eficiencia. A partir de los agentes disponibles, se realiza una selección de aquellos que son más eficientes para cumplir la previsión de tareas de la jornada. Esta tarea se puede resolver sin aplicar procedimientos heurísticos puesto que existen algoritmos no heurísticos que permiten resolver el problema de modo eficiente. Así, el problema se puede plantear según la ecuación (2).

$$\begin{aligned} \min_{st} \quad & \sum_{i,j} \bar{d}_i^{\,j} x_i^j \\ & \sum_i x_i^j = n_i \\ & \sum_j x_i^j = 1 \\ & x_i^j \ \text{int}\, eger \end{aligned} \tag{2}$$

Donde x_i^j representa el agente j y la tarea i, $\bar{d}_i^{\,j}$ representa el tiempo medio para realizar la tarea i por el agente j, n_i el número de agentes necesarios para realizar la tarea i.

El problema planteado en (2) se puede resolver mediante programación lineal entera (PLE). Los algoritmos aplicados para resolver PLE se basan en los algoritmos de programación lineal, como el simplex o el algoritmo húngaro. Estos algoritmos se combinan con otros procedimientos para limitar el conjunto de resultados posibles a valores enteros como es el caso del procedimiento de ramificar y acotar. De modo básico, el algoritmo de ramificar y acotar consiste en añadir restricciones para intentar restringir las soluciones a valores enteros, cada vez que se añade una nueva restricción se resuelve de nuevo el problema de PL, este proceso se repite hasta que las soluciones obtenidas son enteras.

Resolviendo el problema mediante (2), se obtiene como resultado un conjunto de $x_i^j = 1$ que indica cuáles son los agentes que hay que seleccionar para llevar a cabo la planificación de modo que la eficiencia en cuanto a tiempo de ejecución sea máxima.

Este sistema se puede adaptar fácilmente si se desea lograr una optimización basada en un parámetro distinto del tiempo de ejecución. Así, si se quisieran usar costes en lugar de tiempos de ejecución bastaría con modificar el término $\bar{d}_i^{\,j}$ para que refleje el coste que supone que el agente *j* realice la tarea *i*. En nuestro caso, se ha estimado que el coste de los agentes homogéneo y por eso el término sólo depende del tiempo.

4.3.3 Rol tramitador

El rol tramitador lo desempeñan los agentes responsables de realizar las tareas. Cada agente con el rol tramitador se encarga de realizar las tareas que le asigna el agente distribuidor. Al finalizar cada una de las tareas el agente tramitador informa al agente

distribuidor y gestor que ha finalizado la tarea y solicita posteriormente una nueva para continuar con el trabajo. El agente tramitador, realiza tareas de seguimiento sobre sus propias actividades avisando al agente distribuidor sobre retrasos en la actividad que está realizando. A medida que se alcanza el tiempo de duración estimado indicado por el agente distribuidor realiza sondeos sobre el cumplimiento de la actividad.

El comportamiento de los agentes para este rol es básico y su implementación no requiere de modelos simbólicos. El comportamiento del agente es **reactivo**, a medida que se cumplen determinadas condiciones se ejecutan una serie de acciones preestablecidas. El conjunto de reglas condición/acción se las proporciona el agente distribuidor en el momento de realizar la asignación de las tareas. De este modo el comportamiento del agente se adecúa en cada instante a la tarea que se está realizando. Para la interpretación de las reglas se hace uso del servicio que integra un motor de reglas y que es indicado por el agente distribuidor en el momento de la asignación de las tareas.

4.3.4 Rol Gestor

El agente con el rol gestor es el encargado de controlar el acceso a la cola de tareas desde el exterior del sistema. El agente con el rol gestor es el encargado de añadir tareas a la cola y avisar de los resultados una vez que los trámites han finalizado. De igual modo, el agente gestor también incluye tareas de modo automático como consecuencia de la finalización de una tarea en caso de que lo estime oportuno.

El comportamiento del agente en función de lo indicado sigue el modelo **reactivo**. Se definen reglas para que el agente, según recibe las tareas e inserta las mismas en la cola, avise al rol distribuidor sobre su llegada. De igual modo, se definen las reglas para determinar la acción a realizar cada vez que se finaliza una tarea de un trámite y establecer el comportamiento del agente. El comportamiento definido en estas reglas se hace *offline*, para cada una de las tareas de los diferentes trámites, y establece el *workflow* y las alternativas por las que puede pasar un trámite. El *workflow* debe ser definido por personal experto para el correcto funcionamiento del sistema.

4.3.5 Rol Planificador

El rol planificador es el encargado de realizar la planificación a partir de las tareas recibidas. El agente responsable de este rol es el encargado de recibir la lista de tareas a

realizar, establece el número de agentes necesarios para atender los servicios y ordena las tareas de modo que los retrasos afecten lo menos posible a las planificaciones. En la Figura 5 se puede ver la información almacenada en la lista de tareas. Cada tarea de la lista contiene la siguiente información:

- **IdTarea**: identificador de la tarea global/expediente que la representa de manera unívoca dentro del sistema. Si un expediente requiere de varias tareas a realizar, todas ellas tendrían este mismo id.
- **TipoTarea**: Cada tarea requiere un conjunto de actividades que se pueden realizar de modo secuencial o paralelo, se puede decir que son subtareas o pasos dentro de la tarea. Este campo permite conocer en qué estado se encuentra la tarea global y por lo tanto, la siguiente actividad o paso a realizar. De ahora en adelante, se hablará de tarea para referirse a una de las actividades o trámites que se realizan para resolver la tarea final/expediente. Teniendo esto cuenta, una tarea viene definida a su vez por el IdTarea y el TipoTarea.
- **Beneficio**: Rentabilidad total de la ejecución de la tarea para el caso en particular. Este campo es importante a la hora de determinar el número de agentes necesarios para minimizar los gastos frente a las pérdidas por la no realización de ciertas tareas.
- **Beneficio acumulado**: Valor acumulado que se perdería si la tarea no se finaliza en sus plazos. Es el resultante del trabajo realizado con anterioridad. Está asociado con la suma acumulada de los beneficios de los pasos previos para la misma tarea.
- **Fecha de entrada**: La fecha de entrada contiene información sobre la fecha de llegada de la tarea. Este campo es importante para poder determinar la tasa de llegada de las actividades de un tipo concreto para una fecha dada.
- **Plazo**: Tiempo máximo para la resolución del expediente. Este valor se fija para cada tipo de tarea.
- **Duración**: La duración es el tiempo para realizar la tarea.
- **Agente:** Agente responsable de realizar la tarea.

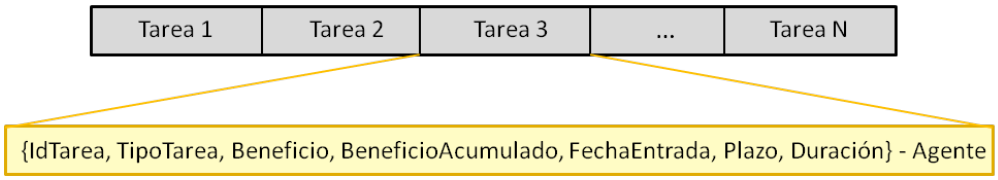

Figura 5 Información lista de tareas

El agente con el rol planificador, a partir de la lista de expedientes a realizar, selecciona el número de agentes necesarios para atender la demanda minimizando los gastos y además distribuye las tareas entre agentes. La selección de los agentes se lleva a cabo por el coordinador. La distribución de las tareas se realiza de modo se minimicen las replanificaciones necesarias para completar las tareas en el tiempo previsto. Este proceso se resume en la Figura 6. Según se puede observar, se dispone de un conjunto de tareas a realizar. El agente Planificador recibe el conjunto de tareas, decide el número de agentes Tramitadores adecuados y las distribuye.

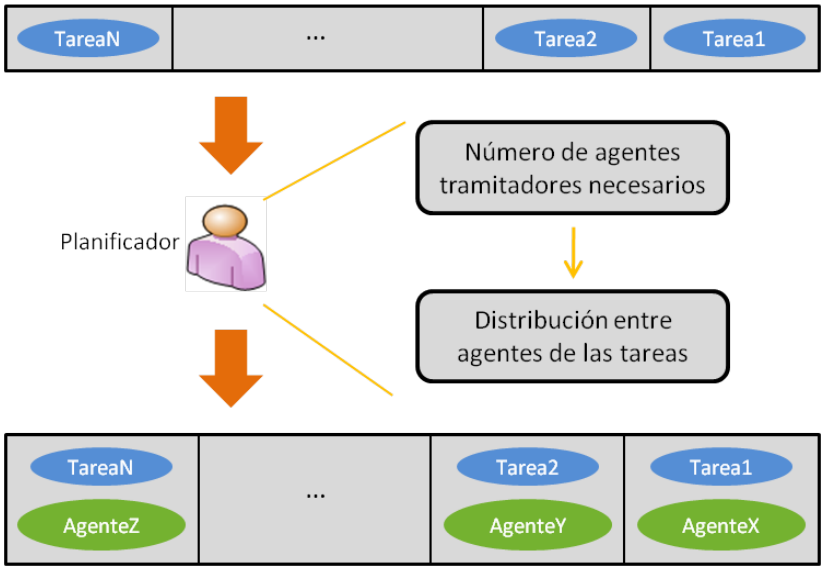

Figura 6 Proceso de planificación

En caso de que el sistema no disponga de agentes necesarios para realizar los trámites se ordenan de modo que se minimicen las pérdidas. No obstante, el sistema debería de funcionar siempre que fuera posible en condiciones de estabilidad, es decir el factor de utilización del sistema se debería mantener por debajo de 1.

La replanificación se puede realizar atendiendo a diferentes razonamientos:

- *Con la llegada de una nueva tarea*: cada vez que el sistema recibe una tarea en la cola, se replanifica. Este mecanismo no es válido si la frecuencia de llegada de las tareas es alta puesto que el planificador emplearía demasiado tiempo en replanificaciones.
- *Siguiendo un periodo de tiempo determinado*: se replanifica cada vez que pasa un determinado número de minutos. Surge el problema de calcular cuál es el mejor tiempo entre planificaciones y debido a la irregularidad en la frecuencia de la llegada de tareas tampoco resulta muy aceptable. Puede

ocurrir que se replanifique con sólo una tarea nueva o que existan demasiadas nuevas a la espera.

- *Cuando se acumulan en la cola de entrada un número determinado de tareas sin asignar*: en vez de replanificar cada vez que llega una tarea se espera a que se acumulen un número determinado para la replanificación.
- *Cuando el plan establecido ya no puede llevarse a cabo*: se replanifica cuando el último plan establecido no puede llevarse a cabo por retrasos de algún Tramitador.

Las dos primeras opciones de replanificación pueden considerarse en sistemas sencillos y con pocas tareas. En este caso, para construir un sistema que permita su adaptación en todos los escenarios, se lleva a cabo un replanificación teniendo en cuenta el número de tareas a la espera de asignación y cuando el plan previsto no puede completarse con éxito. En cuanto al número de tareas que deben llegar al sistema para realizar una nueva planificación se establece que deben ser el 5% de las tareas existentes en la cola previa. También se realizan replanificaciones en el caso de la llegada de tareas urgentes aquellas en las que el plazo de realización es similar a la duración media de las mismas.

El modelo de planificación seguido en este agente es de tipo **deliberativo basado en planes**, se explica en detalle posteriormente.

4.4 Estándar de Comunicación

La comunicación entre los diversos agentes se realiza de modo directo mediante el uso del estándar FIPA-ACL [FIPA, 2002] [Bauer y Huget, 2003] [Huget *et al*., 2003]. El estándar FIPA-ACL define una forma común para el intercambio de mensajes entre sistemas multiagentes lo que permitiría la interconexión entre diferentes sistemas. El estándar FIPA-ACL define una serie de parámetros configurables en los mensajes, en la siguiente tabla.

Parámetro	*Categoría*
performative	Tipo de comunicación
sender	Emisor
receiver	Receptor
reply-to	Agente a responder

content	Contenido
language	Descripción del contenido
encoding	Descripción del contenido
ontology	Descripción del contenido
protocol	Control de la comunicación
conversation-id	Control de la comunicación
reply-with	Control de la comunicación
in-reply-to	Control de la comunicación
reply-by	Control de la comunicación

Tabla 1. Parámetros mensajes FIPA-ACL

Entre los campos más importantes, se tienen *performative* que indica el tipo de comunicación y suele ser usado para el filtrado y la selección de mensajes, *language* la definición del lenguaje usado, *encoding* la codificación o el formato del mensaje y finalmente la ontología que sigue. La ontología se puede usar como un campo de filtrado. Una completa descripción de cada uno de los parámetros se puede ver en la especificación de FIPA-ACL [FIPA, 2002].

El formato de los mensajes usado para el envío se define de modo independiente al contenido de los mensajes. El formato puede ser XML aunque existen otros diferentes según la plataforma de agentes.

La comunicación entre los agentes y los servicios se realiza siguiendo el protocolo estándar SOAP, para ello se dispone de un módulo de comunicación que convierte el formato de los mensajes de FIPA-ACL a SOAP y viceversa. El módulo permite interconectar servicios y agentes de modo sencillo, este módulo viene implementado en algunas plataformas de agentes como JADE por lo que no siempre sería necesaria su implementación.

4.5 Modelo de procesamiento: flujo de información

En este apartado se muestra un ejemplo del comportamiento de la plataforma que recibe una cola de tareas por realizar y que tiene que realizar su distribución entre los agentes disponibles que representan a los trabajadores. El primer paso es determinar cómo las tareas han llegado al sistema. Cada vez que los agentes con rol **gestor** reciben una tarea del exterior la añaden a la cola de tareas global que hay que replanificar. Esta

asignación se realiza de modo automático mediante la serie de reglas predefinidas para su comportamiento.

A medida que la cola de tareas varía, el agente con rol **distribuidor**, en caso de determinar problemas en la planificación, puede activar alguna de las reglas definidas y ordenar replanificaciones al agente con el rol de planificador. El agente con el rol de **planificador** acto seguido tomaría la lista de tareas y haría una estimación del flujo de trabajo para cada uno de los tipos de tareas existentes en la cola, determinando en base a esta información el número de agentes necesarios para realizar cada una de las tareas de los diferentes tipos. A partir de dicha información, el agente planificador haría una petición de asignación de roles al agente **coordinador** que realizaría una selección de los agentes con mejores capacidades para las tareas previstas. Una vez seleccionados los agentes, se realiza la distribución de las tareas de modo que se minimicen los retrasos o bien se minimicen las pérdidas en función de la posibilidad o imposibilidad de que los agentes completen las tareas previstas.

Una vez realizada la planificación, el agente con rol **distribuidor** tomaría los datos de la cola de tareas y continuaría con la asignación de los agentes con rol **tramitador**. Mientas la planificación no esté finalizada se continúa con la asignación según la planificación previa.

Los agentes **tramitadores** realizan las tareas asignadas y avisan de su finalización al agente **distribuidor** y **gestor**. El agente **distribuidor** interpreta el comportamiento reactivo definido y asigna nuevas tareas al agente tramitador en función de cola existente, de igual modo, el agente **gestor** interpreta las reglas y ejecuta las acciones oportunas.

4.6 Modelo de planificación

El modelo de planificación expuesto en este apartado está asociado al rol planificador. El agente con el rol planificador tiene la capacidad de aprender de las planificaciones llevadas a cabo en etapas anteriores. Para ello, los agentes adoptan el modelo de planificación basado en planes que es una especialización del modelo de razonamiento basado en casos (CBR) [Kolodner, 1993] que se integra como motor de razonamiento en un modelo deliberativo BDI. Un modelo deliberativo integrado con un motor de razonamiento de un CBP se define del siguiente modo:

El entorno M y los cambios que se producen dentro de él se representan por medio de un conjunto de variables.

$$M = \{\tau_1, \tau_2, \cdots, \tau_s\} \text{ con } s < \infty \qquad (3)$$

Las creencias son vectores de valores correspondientes a un subconjunto o la totalidad de las variables del entorno.

$$B = \{b_i / b_i = \{\tau_1^i, \tau_2^i, \cdots, \tau_n^i\}, n \le s \quad \forall i \in N\}_{i \in N} \subseteq M \tag{4}$$

Un estado del mundo $e_j \in E$ se representa en el agente como un conjunto de creencias que son ciertas en un determinando instante de tiempo t.

Sea $E = \{e_j\}_{j \in N}$ el conjunto de estados del mundo, fijando el valor de t el estado j viene definido por

$$e_j^t = \{b_1^{jt}, b_2^{jt}, \cdots b_r^{jt}\}_{r \in N} \subseteq B \quad \forall j, t \tag{5}$$

Los deseos son establecidos al inicio de la aplicación. Son aplicaciones entre un estado del mundo y el estado que se quiere alcanzar.

$$d : \underset{e_0}{E} \rightarrow \underset{e^*}{E} \tag{6}$$

Las intenciones es el modo en que el conocimiento de los agentes es manejado para alcanzar los objetivos. Un deseo es alcanzable si existe una aplicación i definida a través de n creencias de modo que

$$i : \underset{(b_1, b_2, \cdots, b_n, e_0)}{BxBx^{n)}\cdots xBxE} \rightarrow \underset{e^*}{E} \tag{7}$$

En nuestro modelo, las intenciones garantizan que existe suficiente conocimiento en la base de creencias de modo que un deseo se puede alcanzar mediante un plan de acciones.

Las acciones se definen como los mecanismos que provocan cambios en el mundo generando cambios de estado.

$$a_j : \underset{e_i}{E} \rightarrow \underset{a_j(e_i)=e_j}{E} \tag{8}$$

Un plan de un agente es una secuencia de acciones que definen la ruta de estados a través de los cuales el agente pasa desde el estado actual, e_0, hasta otro estado del mundo.

$$p_n : \underset{e_0}{E} \rightarrow \underset{p_n(e_0)=e_n}{E} \tag{9}$$

$$p_n(e_0) = e_n = a_n(e_{n-1}) = \cdots = (a_n \circ \cdots \circ a_1)(e_0) \quad p_n \equiv a_n \circ \cdots \circ a_1$$

Partiendo de esta representación, los agentes tienen capacidades de planificación basados en el modelo BDI. El estado inicial se corresponde con plan en el que no se han asignado agentes a las tareas, el estado final se corresponde con la cola de tareas en la que se ha asignado un agente a cada uno de las tareas. Las acciones se

corresponden con las diferentes funciones que permiten asignar a los agentes con rol tramitador tareas.

Para realizar el proceso de planificación el agente sigue el modelo de planificación CBR-BDI. En la Figura 7 se muestra un esquema del funcionamiento del agente y las tareas realizadas en cada una de las etapas del ciclo de razonamiento.

Planificación basado en el modelo CBR-BDI

Input: Nueva lista de tareas

1. retrieve: lista tareas
2. reuse: cálculo de servicios necesarios, selección de los agentes y distribución de tareas.
3. revise: automática en función de la planificación realizada.
4. retain: actualiza la memoria de casos con la lista de tareas.

Figura 7 Sistema CBR

El primer punto en la definición de un modelo CBR-BDI consiste en la definición del concepto de caso para así tener definida la memoria almacenada en el sistema. Cada caso almacena la información asociada al conjunto de tareas realizadas durante una jornada de trabajo en la organización, por tanto, se tendrá un caso por jornada laboral. La información de cada caso sigue la siguiente tupla:

$$C = \{t_i / t_i = (ids, idt, b, B, f, p, d, a), i = 1...n\} \quad (10)$$

Donde t_i presenta la tarea i, ids es el identificador del tipo de tarea, idt el trámite, b beneficio, B el beneficio acumulado, f fecha de entrada, p plazo, d duración, a identificador del agente que realizó la tarea.

Cuando el agente planificador recibe el nuevo caso c_{n+1} compuesto de los elementos indicados en (10) procede a realizar la planificación mediante la aplicación de las diferentes etapas del ciclo de razonamiento según se describe en los apartados siguientes:

Durante la **recuperación** se procede a recuperar los casos más similares al caso actual c_{n+1}. Los casos más similares son aquellos que contienen más tareas del mismo tipo que las tareas actualmente encoladas en el sistema. Para mejorar el rendimiento de la búsqueda se añade a la memoria de casos la información del número de tareas existentes de cada tipo de actividad y sólo se realiza la búsqueda sobre tareas almacenadas en un período de tiempo prefijado. El tiempo prefijado varía según el tipo de servicios que se esté dando en una organización. El número de casos recuperados es predefinido para posteriormente realizar la fase de adaptación. Al conjunto de casos recuperados se les denomina $C_r \subseteq C$.

En la etapa de **reutilización** se adapta la información recuperada C_r de la memoria de casos para generar el plan completo correspondiente al caso c_{n+1}. La información recuperada se adapta mediante la aplicación teoría de colas y algoritmos genéticos. Por un lado, la información recuperada se utiliza para determinar la tasa de llegada y de servicio de cada uno de los trámites y así determinar mediante teoría de colas el número de servicios necesarios para ejecutar los trámites indicados. A la hora de determinar el tiempo medio de servicio se usan los datos obtenidos de los diferentes casos, para estimar el error sobre el valor obtenido bajo un determinado nivel de confianza se aplica muestreo estadístico para estimar dicho valor, en caso de que se supere un determinado umbral, se procede seleccionar el tamaño de la muestra mínimo para reducir el error al valor indicado. En la Figura 8 se puede ver el algoritmo seguido para la definición del número de servicios totales.

```
/* C_r:  casos recuperados                                          */
/* c_{n+1}:  nuevo caso                                             */
/* N: tamaño de la población                                        */
/* eMax:  error máximo                                              */
/* coste:  coste de los agentes (constante k en teoría de
   colas)                                                           */
/* Numero:  conjunto con el número de agentes por tarea             */
/* ListaTareas:  lista con la relación tarea-agente                 */
/* c_{n+1}:  caso nuevo                                             */
/* N: tamaño población                                              */
/* Medio:  conjunto con el tiempo medio de servicio para cada
   tipo de tarea                                                    */
Input: C_r, c_{n+1}, N, eMax, coste
Output: Numero, ListaTareas, Medio

Error ← ∅;
Medio ← ∅;
ListaTareas ← recuperarTareas(c_{n+1});
foreach x ∈ ListaTareas do
    e ← calcularError(x, C_r, N_x);
    if e > eMax then
        TareasSimilares ← recupearTareasSimilares(x, eMax);
        Medio ← calcularMedio(c_r, TareasSimilares);
    end
    else
        Medio ← calcularMedio(C_r);
    end
    Error ← Error ∪ {e};
    Medio ← Medio ∪ {Medio};
end
[Numero, ListaTareas] ← calcularAgentes(C_r, Medio, coste);
```

Figura 8 Definición del número de servicios totales

La función calcular servicios se realiza en función del algoritmo representado en la Figura 9.

```
/* C_r:  casos recuperados                                          */
/* Medio:  conjunto con el tiempo medio de servicio para cada
   tipo de tarea                                                    */
/* coste:  coste de los agentes (constante k en teoría de
   colas)                                                           */
/* Numero:  conjunto con el número de agentes por tarea             */
/* Tareas:  Lista con las tareas a realizar                         */
/* Agentes:  conjunto de agentes seleccionados por el
   coordinador para realizar las tareas                             */
Input: C_r, Medio, coste
Output: Numero, Tareas, Agentes

TareasOrdenadas ← ordenarTareasBeneficio();
f ← 0;
Tareas ← ∅;
for i=1 to #TareasOrdenadas do
    L_q ← elementosCola(C_r, Medio[i], s=1);
    W_q ← calcularEsperaMedia(L_q, λ);
    L ← elementosSistema(L_q, λ, Medio[i]);
    duracion ← calcularDuracionMedia(C_r, TareasOrdenadas[i], eMax);
    beneficio ← calcularBeneficioMedio(C_r, TareasOrdenadas[i], eMax);
    [f_b, servicios] ← calcularBeneficio(L, Medio[i], duracion, beneficio);
    p ← actualizarFactorUtilizacion(λ, Medio[i]);
    if ρ > 1 then endFor;
    f ← f+f_b;
    Tareas ← Tareas ∪ TareasOrdenadas[i];
    Numero ← Numero ∪ servicios;
end
Agentes ← seleccionarAgentesCoordinador(Numero, Tareas);
```

Figura 9 Cálculo del número de servicios

Por otra parte, los casos recuperados, sirven de base para construir los cromosomas iniciales en los algoritmos genéticos. El algoritmo seguido durante la fase de adaptación correspondiente a la aplicación de los algoritmos genéticos se puede ver en la Figura 10.

```
Input: C_r, c_{n+1}, Tareas, numero, probabilidadCruce,
       probabilidadMutacion
Output: c_{n+1}

TiempoMedioTareaAgente ← calcularTiempoMedioTareaAgente(C_r,
c_{n+1});
Poblacion ← crearPoblacion(C_r, c_{n+1}, Tareas, numero,
TiempoMedioTareaAgente);
k ← cte;
for i=1 to tope do
    Aptitud ← calcularAptitudCromosomas(Poblacion);
    if max(Aptitud)>aptitudParada then return;
    NuevaPoblacion ← ∅;
    NuevaPoblacion ← obtenerCromosomasElitistas(Poblacion, k);
    for i=n to #Poblacion do
        Padres ← seleccionarPadres(Poblacion, Aptitud);
        Hijos ← curzarCromosomas(Padres, probabilidadCruce);
        Hijos ← mutarCromosomas(Hijos, Tareas, numero,
        probabilidadMutacion);
        NuevaPoblacion ← NuevaPoblacion ∪ Hijos;
    end
end
```

Figura 10 Asignación de las tareas

La etapa de **revisión** se realiza de modo automático a medida que se van finalizando tareas por parte de los agentes. El agente actualiza la duración de las tareas según se van completando y a su vez, realiza replanificaciones en caso de recibir un aviso por parte de los agentes tramitadores sobre la imposibilidad de completar un plan bajo las restricciones temporales previstas.

La fase de **aprendizaje** se limita a almacenar el caso una vez se ha finalizado la jornada. La nueva memoria de casos C' se define del siguiente modo: $C' = C \cup c_{n+1}$ Para limitar el tamaño de la memoria de casos se eliminan de la memoria a medida que superan una antigüedad predefinida.

4.6.1 Cálculo de error máximo

Para realizar la estimación del número de agentes con rol tramitador y realizar la asignación de las tareas, es necesario realizar una estimación del tiempo medio de duración de las tareas de modo global y de modo particular para cada uno de los agentes. La estimación de la duración de las tareas se realiza determinando el nivel de error máximo permitido bajo un determinado nivel de confianza. Para ello, se fija un

nivel de confianza y se determina el nivel de error obtenido para un determinado tamaño de muestra. La ecuación (11) define el intervalo para la media dado un tamaño muestral y población.

$$\hat{\mu} = \bar{x} \pm k \frac{s}{\sqrt{n-1}} \sqrt{\frac{N-n}{N-1}} \tag{11}$$

Donde $\bar{x}$ representa la media, k es el valor Z de un N(0,1) que deja bajo ella el nivel de confianza preestablecido, N es el tamaño poblacional, n el tamaño muestral y s es la desviación.

La ecuación (11), se simplifica si se toma que N>>n del orden de 100 veces. En caso de que se dé dicha condición la ecuación se reduce a la expresión (12).

$$\hat{\mu} = \bar{x} \pm k \frac{s}{\sqrt{n_\infty}} = \bar{x} \pm e \tag{12}$$

Donde n_∞ representa que N>>n del orden de al menos 100 veces.

Por tanto, despejando se tiene la expresión (13) que permite determinar el tamaño muestral para fijar un error máximo.

$$n_\infty = \frac{k^2 S_c^2}{e^2} \tag{13}$$

S_c representa la cuasivarianza, el resto de las variables se ha comentado anteriormente.

Finalmente, en etapas iniciales del sistema para los que no se dé la condición de que N>>n se selecciona el tamaño de la muestra siguiendo la ecuación (14).

$$n = \frac{1}{\frac{1}{n_\infty} + \frac{1}{N}} \tag{14}$$

Las ecuaciones mostradas en este apartado, permiten determinar el número de elementos a recuperar de la memoria para acotar los errores obtenidos. Esto permite limitar el acceso a la memoria y permite eliminar planificaciones que ya no sean accedidas.

4.6.2 Planificación dinámica de roles

El número de agentes que deben estar disponibles en el sistema se estima de manera dinámica. El objetivo es que el número de agentes se adecúe a la demanda tratando de garantizar que el factor de utilización del sistema sea menor que 1. Para ello se utilizará una planificación basada en teoría de colas ya que ésta es especialmente válida para el estudio de los puntos de servicio necesarios para atender una demanda incierta.

Como ya se describió en el apartado 2.3.2, en un sistema de colas se pueden distinguir diferentes parámetros que determinan su funcionamiento:

- *Tasa de llegada*: La distribución de llegada que sigue la entrada de tareas en el sistema. Normalmente la función de distribución seguida es una de Poisson.
- *Tiempo de servicio*: Tiempo empleado en llevar a cabo la tarea una vez el agente correspondiente se hace cargo de ella. La distribución seguida normalmente es una exponencial.
- *Servidores*: Uno o varios agentes con la posibilidad de ausencia.
- *Disciplina de la cola*: FIFO (first in first out), LIFO (last in first out), RR (Round Robin), RSS (random selection of service), with priority.
- *Capacidad de la cola*: Longitud de la cola finita o infinita.
- *Tamaño de la fuente de entrada al sistema*

Para la especificación de los parámetros se suele seguir la denominada notación de Kendall que consiste en una séxtupla que almacena los datos indicados anteriormente. Se representa mediante la siguiente sintaxis 1/2/3/4/5/6. Los valores de cada uno de los términos de la expresión varían de la forma siguiente para cada uno de los términos:

1. M: Poisson, D tiempos determinísticos, E_k distribución de Erlang, GI distribución general
2. M: Exponencial, D tiempos determinísticos, E_k distribución de Erlang, G distribución general
3. 1 para un sólo servidor, S para varios.
4. En la disciplina se ponen las siglas del método seleccionado
5. Capacidad de la cola un número ó ∞.
6. Número máximo de consumidores en el sistema, un número ó ∞.

Existen muchas variantes en la teoría de colas según sean las opciones elegidas. El modelo M/M/1, es el más sencillo de usar, es aquel en el que la tasa de llegadas sigue

una distribución de Poisson (15), la de servicio una exponencial (16) y el servidor es único.

$$P(x=k)=\frac{\lambda^k e^{-\lambda}}{k!} \tag{15}$$

$$F(x)=1-e^{-\lambda x} \tag{16}$$

En el caso que se presenta, se dispondrá de varios agentes por tanto, nos encontraremos con modelos M/M/s o bien M/G/s. Supondremos siempre que la llegada de las peticiones sigue una distribución de Poisson (15) pues dicha distribución mide el número de llegadas en un determinado tiempo y es la más usada en todos los trabajos similares [Menasce, 2002]. Además, se supondrá que el tiempo de servicio sigue una distribución exponencial, lo más adecuado para el caso genérico.

La teoría de colas se planeta según el modelo de nacimiento y muerte. El problema de nacimiento muerte define la relación entre las llegadas de tareas y su servicio de modo que se tiene un problema balanceado en el que todo lo que llega se atiende. Se plantea según lo mostrado en la Figura 11.

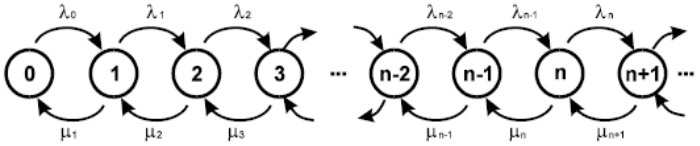

Figura 11 Nacimiento muerte

En función de la figura se puede establecer la relación entre la probabilidad de que el sistema se encuentre en el estado k, el estado k representa el conjunto de tareas que se tienen en la cola de espera.

En el caso de que no haya ninguna tarea en la cola de espera se produce la siguiente relación entre lo que llega y lo que sale del estado 0.

$$\mu_1 P_1 = \lambda_0 P_0 \tag{17}$$

$$P_1 = \frac{\lambda_0}{\mu_1} P_0 \tag{18}$$

En este caso se tiene que la probabilidad de que haya 0 tareas en la cola (P_0) por la tasa de llegada tiene que ser igual a la probabilidad de que haya una tarea en la cola que se va a atender de inmediato.

Para el estado n-1 se tiene lo siguiente

$$\lambda_{n-2}P_{n-2} + \mu_n P_n = \lambda_{n-1}P_{n-1} + \mu_{n-1}P_{n-1} \tag{19}$$

$$\mu_n P_n = \lambda_{n-1}P_{n-1} + \mu_{n-1}P_{n-1} - \lambda_{n-2}P_{n-2} \tag{20}$$

$$P_n = (\lambda_{n-1}P_{n-1} + \mu_{n-1}P_{n-1} - \lambda_{n-2}P_{n-2}) / \mu_n \tag{21}$$

Despejando en (21) P_n y sustituyendo los P_{n-1} se tiene que

$$P_n = \frac{\lambda_0 \cdots \lambda_{n-1}}{\mu_1 \cdots \mu_n} P_0 = c_n P_0 \tag{22}$$

El problema de planificación de múltiples tareas se puede reducir al caso de planificación de una única tarea por cada uno de los tipos. Así para cada una de las tareas se realiza una planificación independiente de modo que se calcula el tiempo medio de espera y longitud media de cola de modo independiente. El tiempo medio de espera y la longitud media global se reduce a calcular la media de los valores calculados para cada una de las tareas. En el caso del modelo M/M/s donde s=1,2,3,… es el número de agentes y dada una tasa de llegada $\lambda_n = \lambda$ =cte, la tasa de servicio cuando se encuentran n procesos viene definida por la siguiente ecuación (23) [Martín, 2003].

$$\mu_n = \begin{cases} n\mu & n = 1,2, \ldots, s-1 \\ s\mu & n \geq s \end{cases} \tag{23}$$

μ representa el tasa media de servicio para los s agentes disponibles. Este valor depende tanto de los agentes como de la máquina en la que se encuentran. En los cálculos no se hace distinción de los tiempos de servicio independientes ya que no se puede estimar los agentes ocupados y por tanto, la tasa de servicio general cuando el número de tareas en el sistema es inferior al número de agentes.

Partiendo de que

$$\sum_{n=0}^{\infty} P_n = 1 \tag{24}$$

Suponiendo que en el sistema se encuentra en condiciones de estabilidad, es decir se cumple que el factor de utilización es menor que $1\ \rho = (\lambda / \mu s) < 1$, la probabilidad de que hayan n tareas (P_n) en el sistema viene dado por la siguiente ecuación:

$$P_n = c_n P_0 = \begin{cases} \dfrac{\lambda^n}{n!\,\mu^n} P_0 & n = 0,1, \ldots, s-1 \\ \dfrac{\lambda^s}{s!\,\mu^s}\left(\dfrac{\lambda}{s\mu}\right)^{n-s} P_0 & n \geq s \end{cases} \tag{25}$$

Sustituyendo en (24) el valor de P_n por (25) se tiene

$$P_0 = \frac{1}{\sum_{n=0}^{s-1} \frac{\lambda^n}{n!\,\mu^n} + \sum_{n=s}^{\infty} \frac{\lambda^s}{s!\,\mu}\left(\frac{\lambda}{s\mu}\right)^{n-s}} = \frac{1}{\sum_{n=0}^{s-1} \frac{\lambda^n}{n!\,\mu^n} + \frac{\lambda^s}{s!\,\mu}\frac{1}{1-\frac{\lambda}{s\mu}}} \tag{26}$$

$$c_n = \begin{cases} \dfrac{\lambda^n}{n!\,\mu^n} & n = 1,2, \ldots, s-1 \\ \dfrac{\lambda^s}{n!\,\mu^s}\left(\dfrac{\lambda}{s\mu}\right)^{n-s} & n \geq s \end{cases} \tag{27}$$

En caso de que no se cumpla la condición del factor de utilización del sistema será necesario aumentar el número de agentes del sistema.

Una vez definida la probabilidad de que haya n tareas en el sistema, se pueden definir los parámetros que realmente importan, en este caso se trata del número de tareas que se van a encontrar en la cola del sistema. Viene definido por la siguiente ecuación.

$$L_q = \sum_{n=s}^{\infty} (n-s) P_n = P_0 \frac{\lambda^s}{s!\,\mu^s} \frac{\lambda}{s\mu} \frac{1}{\left(1-\frac{\lambda}{s\mu}\right)^2} \tag{28}$$

A partir de la ecuación (28) se puede definir de modo sencillo el tiempo medio de espera de las tareas en la cola del sistema.

$$W_q = \frac{L_q}{\lambda} \tag{29}$$

En base al número medio de unidades en la cola (28) y el tiempo medio de espera en el sistema (29), se establece el número de tareas en el sistema según la ecuación (30).

$$L = L_q + \lambda / \mu \tag{30}$$

En base a ese tiempo medio W_q y el número de agentes, se determina el número final de agentes a utilizar. Para determinar el número de agentes óptimo se realiza una estimación en la que se minimiza la función de coste que depende del número de servicios arrancados y del tiempo de espera en la cola. La función se define de forma particular para cada servicio dependiendo de los costes reales de cada uno de los servicios del sistema, aunque se proporciona la siguiente función de coste (31).

$$f(L, P_0, .., P_{s-1}, \mu', \overline{p}, \overline{b}) = f_b(L, \mu', \overline{p}, \overline{b}) - k \cdot s \tag{31}$$

$$f_b(L, \mu', \overline{p}, \overline{b}) = \begin{cases} (\overline{p}/u')\overline{b} \cdot s \cdot (1-\rho) & \text{si } L \cdot u' > \overline{p} \cdot s \cdot (1-\rho) \\ L\overline{b} & \text{si } L \cdot u' \leq \overline{p} \cdot s \cdot (1-\rho) \end{cases} \tag{32}$$

Siendo k una constante asociada al coste de tener trabajando un agente, $\overline{b}$ el beneficio de ejecución de la tarea, μ' es el tiempo medio de realización de la tarea obtenido a partir de la tasa de servicio, $\overline{p}$ el plazo medio de ejecución de una tarea, en caso de superar las condiciones de estabilidad, en f_b sólo se contabiliza hasta alcanzar el factor de utilización a 1. El factor de utilización ρ varía según se añaden nuevos servicios a la cola hasta alcanzar el factor de utilización de 1.

Siguiendo la función de coste indicada en (31), se introduce la función de coste global que tiene en cuenta la ejecución de los diferentes servicios, el modo siguiente:

$$f(f_1, ..., f_2) = \sum_{j=1}^{k} f_i \tag{33}$$

donde f_i se calcula siguiente la ecuación mostrada en (31). Debido a que en ocasiones puede que no se den las condiciones de estabilidad, es necesario calcular los términos por orden de beneficio según el tipo de tarea de modo que cuando se alcance el factor de utilización del 100% se termine de añadir términos al sumatorio.

Una vez establecida la función a optimizar definida en (33) se calcula el valor mínimo de la función. El valor se calcula iterativamente iniciando el número de agentes a 1, el valor fijado es el primer mínimo local que se corresponde con el mínimo global.

4.6.3 Asignación de tareas

Una vez establecido el número de agentes a iniciar para minimizar los costes, se procede a realizar una asignación de las tareas entre los agentes disponibles. Los problemas de asignación se realizan habitualmente mediante programación lineal [Martín, 2003], programación cuadrática o bien mediante el uso de otras técnicas de programación no lineal como los multiplicadores de Lagrange [Wan *et al*., 2009], Kuhn-Tucker [Shi *et al.,* 2005].

En caso de que el factor de utilización del sistema no supere el valor de 1, la distribución de las tareas entre los agentes se realiza de modo que se garantice en la medida de lo posible que se pueden realizar las tareas asignadas en caso de que se produzcan retrasos o el tiempo de realizar una tarea se incremente. Se realiza de modo que se maximice la función (34):

$$\max \sum_{i=1}^{k} f_i \; donde \; fi = \begin{cases} \log(1 - |\, x_i - \bar{x}\,|) & (x_i - \bar{x}) \geq 0 \\ -\log(1 - |\, x_i - \bar{x}\,|) & (x_i - \bar{x}) < 0 \end{cases} \tag{34}$$

Donde $x_i = t_i - a_{i-1} - c_i$ con t_i el tiempo máximo de finalización de la tarea i, a_{i-1} el tiempo acumulado para ejecutar las tareas i-1 anteriores y finalmente c_i el tiempo para ejecutar la tarea i, que se personaliza según el agente elegido y se calcula a partir del valor medio de las tareas ejecutadas anteriormente.

Teniendo en cuenta la definición de la función (34), maximizando las diferencias se consigue que todas las tareas tengan una distribución uniforme de los tiempos sobrantes por lo que se facilita que la consecución de las mismas sea más probable. Analizando la definición de la función log(x_i) , para maximizar la función dada, se deben de tomar valores de x_i de modo que sean homogéneos ya que la función suaviza los valores altos y minimiza los valores bajos.

El valor de cada x_i de la función objetivo varía según el orden de ejecución de las tareas por lo que no se puede establecer el valor de antemano. Si se planteara el problema desde el punto de vista de la programación lineal o programación lineal entera, sería necesario establecer todas las permutaciones posibles para calcular los x_i porque el valor del sumatorio varía en función de los valores calculados anteriormente. Las técnicas de programación no lineal como los multiplicadores de Lagrange [Wan *et al*., 2009] y o Kuhn-Tucker [Shi *et al.,* 2005] tampoco se pueden aplicar debido al mismo problema en la definición de la función objetivo. De modo alternativo, se podría plantear el uso de otros algoritmos para la resolución problema como el método húngaro [Jonker y Volgenant, 1986]. El algoritmo también presenta los mismos problemas que el Simplex [Kabadi y Punnen, 2008] ya que es necesario realizar todas las

combinaciones en la función objetivo. El algoritmo es iterativo pero no se establece el orden de ejecución de tareas hasta el último paso por lo que no se puede modificar de modo iterativo la tabla de costes en función del orden de las tareas. Por ello, se hace necesario plantear métodos alternativos para alcanzar una solución.

En caso de que el factor de utilización del sistema sea mayor que 1, la función de aptitud se redefine para minimizar las pérdidas posibles del trabajo ya realizado con anterioridad.

$$\min \sum_{i=1}^{k} - B_i \tag{35}$$

Al igual que en el caso anterior para conocer el valor de cada B_i es necesario establecer el orden de ejecución de los trámites. En caso de que de tiempo para finalizar dicha tarea el valor de B_i no se tendrá en cuenta.

Para resolver el problema se puede recurrir a métodos heurísticos. Entre los métodos aplicables, se tienen los algoritmos genéticos. Los algoritmos genéticos permiten resolver el problema de optimización mediante la computación evolutiva y la aplicación de operadores sobre los elementos de la población. La definición del algoritmo evolutivo se fundamente en la definición de los siguientes módulos:

- Módulo evolutivo: Técnica de la codificación y función de la función de aptitud
- Módulo poblacional: Técnica de representación, de arranque, criterio de selección y de reemplazo.
- Módulo reproductivo: Operadores genéticos

La codificación de los cromosomas se realiza de modo que cada gen se compone de los elementos indicados por t_i identificados en (10). La secuencia de elementos en el cromosoma representa también el orden de ejecución de las tareas, así cada uno de los cromosomas se representa según la Figura 12:

Tarea 1	Tarea 2	Tarea 3	...	Tarea n

Figura 12 Cromosoma

De modo abreviado y para simplificar la explicación de los operadores, cada cromosoma se representará según lo indicado en la Figura 13. El primer campo representa al identificador de las tareas que han llegado al sistema.

Tarea - Agente	...	Tarea - Agente	Tarea - Agente	Tarea - Agente

Figura 13 Definición simplificada de los cromosomas

Una vez definida la codificación de los cromosomas, es necesario definir los operadores de cruce y mutación. El operador de cruce se define de modo similar al cruce multipunto usado en otros problemas. El operador queda definido de la siguiente manera:

Se elige para cada descendiente un tramo de uno de los progenitores y a la vez se preserva el orden relativo del otro.

Los pasos a seguir para realizar este cruce se resumen a continuación:

- Se selecciona un recorrido parcial
- Intercambio de los segmentos directo de tareas en caso de coincidir identificador
- Los intercambios definen una serie de emparejamientos que relaciona cada uno de los genes de un cromosoma con el que ocupa la misma posición en el otro progenitor

Definición de los cromosomas según el esquema mostrado en la figura 13:

```
Cromosoma 1 (C1)=(1-1, 2-1, 3-3, | 4-1, 5-2, 6-1, 7-1, | 8-2, 9-2)

Cromosoma 2 (C1)=(1-2, 5-3, 2-3, | 4-1, 3-2, 6-2, 7-2, | 8-3, 9-2)
```

```
Emparejamientos: (Identificador de tarea C1 - Identificador de tarea C2)

4↔4, 5↔3, 6↔6, 7↔7
```

Intercambio de segmentos:

```
(x-x, x-x, x-x, | 4-1, 3-2, 6-2, 7-2, | x-x, x-x)

(x-x, x-x, x-x, | 4-1, 5-2, 6-1, 7-1, | x-x, x-x)
```

Eliminación de las redundancias:

```
(1-1, 2-1, x-x, | 4-1, 3-2, 6-2, 7-2, | 8-2, 9-2)

(1-2, x-x, 2-3, | 4-1, 5-2, 6-1, 7-1, | 8-3, 9-2)
```

Intercambio de los emparejamientos:

```
(1-1, 2-1, 5-3, | 4-1, 3-2, 6-2, 7-2, | 8-2, 9-2)

(1-2, 3-3, 2-3, | 4-1, 5-2, 6-1, 7-1, | 8-3, 9-2)
```

Hay que tener en cuenta que se pueden producir encadenamientos de emparejamientos a la hora de realizar los reemplazamientos de los cromosomas, esto es, se puede dar el caso que tengamos 1↔4, 4↔2 y por tanto tendríamos 1↔2.

Como operadores de mutación, se definen varios de modo que se ejecutan aleatoriamente y sólo se seleccionan las mutaciones que mejoran la aptitud de los cromosomas. Los operadores de mutación definidos son los siguientes:

- Intercambio de orden de tareas. Ejemplo

(1-1, 2-1, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)

(2-1, 1-1, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)

- Intercambio de asignación de las tareas contiguas. Ejemplo

(1-1, 2-1, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)

(1-3, 2-1, 5-1, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)

- Cambio de asignación de una tarea. Ejemplo

(1-1, 2-1, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)

(1-1, 2-3, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)

Además, se define el operador de elitismo para guardar el porcentaje de las soluciones más eficientes en cada generación de la población.

Adicionalmente, se define el tamaño de la población como constante. La definición de la población de tamaño constante implica la sustitución de los cromosomas padres por lo hijos en las generaciones a excepción de los cromosomas elitistas que se mantienen. En la sucesiva evolución de la población se realizan inmigraciones para la caída en mínimos locales a causa de la convergencia de los cromosomas de partida.

El criterio de selección elegido es el de la ruleta. El método de la ruleta define una probabilidad de selección de cada uno de los cromosomas proporcional a la función de aptitud. El criterio de selección se define de modo que calcule correctamente las probabilidades teniendo en cuenta que la función de aptitud toma valores negativos para aquellos casos en los que el factor de utilización es menor que 1.

Como último paso, queda definir la iniciación de los cromosomas. La iniciación de los cromosomas se hace en función de las tareas recibidas, cada cromosoma se inicia con las tareas del nuevo caso. Por cada una de las tareas del cromosoma se busca secuencialmente en el caso y se asigna la tarea a un agente de los existentes de modo que esta asociación se mantiene para el resto de las tareas. En caso de que el caso previo tuviera más o menos agentes estas tareas se asignan de modo aleatorio a los agentes restantes.

4.7 Resumen y conclusiones

En este capítulo se ha expuesto el modelo de arquitectura multiagente y el sistema de planificación automática para los diferentes agentes de la arquitectura. El modelo propuesto se trata de un modelo genérico sin ser aplicado en particular a ningún caso de estudio y tiene la posibilidad de modificar el comportamiento mediante la incorporación de servicios con nuevos mecanismos de planificación que pueden integrar nuevos mecanismos deliberativos o reactivos con la definición de reglas de modo dinámico.

La arquitectura modela una organización virtual permitiendo variar los agentes y los roles que empeñan de modo dinámico. El modelo está formado por un conjunto de agentes que integran diferentes roles y forman el núcleo de la organización. El comportamiento de los roles no se especifica en los propios agentes sino que se asocia a servicios lo que permite variar la funcionalidad de los agentes con la simple modificación del servicio o la selección de un servicio alternativo.

Los servicios integran modelos razonamiento variados debido a la funcionalidad que deben desempeñar. Los comportamientos reactivos basados en reglas facilitan la toma de decisiones inmediata mediante el uso de BRMS (Business Rule Management System) e introducen comportamientos deliberativos para aquellas partes que requieren una mayor carga computacional debido a la complejidad del comportamiento.

La arquitectura permite seleccionar agentes y asigna roles en función de las actividades que hay que realizar de modo que se maximice la eficiencia económica y la replanificación, no obstante, se podría readaptar a cualquier otro tipo de problema con la modificación de las funciones objetivos del problema. El uso de teoría de colas permite simular el comportamiento del sistema y prever posibles sobrecargas del mismo permitiendo seleccionar el número de agentes necesarios para cumplir un conjunto de tareas de antemano, facilitando por ello la estabilidad ya que evita la continua modificación de agentes en función de sobrecargas puntuales. Por otra parte, los algoritmos genéticos permiten realizar tareas de asignación de modo eficiente en tiempo razonable frente a otras heurísticas, no obstante, sería uno de los puntos susceptibles de explorar posibles mejoras del sistema.

En el siguiente capítulo se expondrán los comportamientos de los agentes en el modelo propuesto para llevar a cabo la planificación para un caso de estudio en concreto.

Capítulo V

Caso de Estudio

VNiVERSiDAD
D SALAMANCA

Introducción

El sistema propuesto puede ser aplicado a diferentes ámbitos de empresas y organizaciones donde es necesario realizar una planificación de tareas. Sin embargo, inicialmente, este trabajo surgió de la necesidad de disponer de una planificación que permitiera una distribución efectiva del trabajo para el nuevo enfoque generado por la implantación de la e-Administración. Por ello, en este capítulo se muestra la aplicación del modelo propuesto a un caso de estudio que presenta una sociedad orientada a tramitar todos los expedientes que llegan a la administración pública por medios telemáticos.

El capítulo se ha dividido de la siguiente forma: en primer lugar presentará una revisión de sistemas multiagente aplicados a este campo, seguido de la exposición del caso y la justificación de la motivación. A continuación, se muestran los resultados obtenidos a partir de los que se establecerán las conclusiones del trabajo que se exponen en el siguiente capítulo.

5.1 E-Administración y Sistemas Multiagente

La expresión *Administración electrónica*, *e-Administración* o *e-Government*, se aplica a la utilización de las tecnologías de la información y las comunicaciones en las actividades que competen a la Administración y, en particular, en su relación con los ciudadanos [Marchionini *et al.,* 2003] [De Meo *et al.,* 2008]. La extensión de la e-Administración a los dispositivos móviles da lugar a la m-Administración o m-Government que se puede definir como el uso extensivo de tecnologías, canales y dispositivos móviles, en conexión con la e-Administración, incluyendo las transacciones de procesos sobre redes inalámbricas [Kumar y Sinha, 2007] [Kharma y Hassan, 2010].

La administración electrónica tiene como objetivo transformar las oficinas físicas tradicionales, basadas en procesos en papel, en oficinas virtuales, basadas en documentos electrónicos, cubriendo tanto las comunicaciones internas y como las comunicaciones entre oficinas de diferentes organizaciones y la comunicación con los ciudadanos. La e-Administración se ha visto impulsada por la aparición de las tecnologías de la información y comunicación facilitando a los clientes (ciudadanos) la interacción con las organizaciones y mejorando las condiciones de los trabajadores: flexibilidad horaria, teletrabajo, movilidad, etc. Las ventajas directas que obtiene un cliente de e-Administración son:

- Total disponibilidad temporal de los servicios ofrecidos por la organización, eliminando horarios y días festivos.
- Facilidad de acceso: las oficinas están accesibles desde cualquier punto del mundo gracias al teléfono o internet.
- Ahorro de tiempo: cualquier gestión puede realizarse sin requerir un desplazamiento físico a la oficina y sin la necesidad de esperar largas colas para recibir una atención personal.
- Facilidad en el seguimiento de expedientes. Tanto los ciudadanos como los trabajadores tienen mayor facilidad para acceder a la información relacionada con el estado y procesos asociados a cada expediente.

La propia Administración también obtiene ventajas al poder facilitar un mayor volumen de servicios a gran escala sin requerir el incremento proporcional de recursos físicos (personal, oficinas) que serían necesarios en otro caso.

La aplicación de las TIC a los servicios de la Administración se puede dividir en tres categorías: acceso a la información, acceso a los servicios y participación ciudadana [Marchionini *et al.,* 2003].

El acceso a la información es la aplicación más frecuente, tanto en lo que se refiere a su oferta como a su demanda. La Administración produce unas enormes cantidades de información, de la cual, una parte cada vez más importante está disponible a través de la Web u otros medios electrónicos. El manejo de estas grandes cantidades de información requiere de complejos sistemas de gestión que garanticen su óptimo mantenimiento y faciliten la localización de los elementos de interés.

La incorporación de los medios telemáticos a las operaciones administrativas permite a los ciudadanos acceder a diversos servicios on-line para realizar gran parte de los trámites requeridos en su relación con la Administración. El establecimiento de sistemas técnicamente fiables y jurídicamente válidos de identificación personal ha abierto el abanico de las gestiones susceptibles de realización telemática. Así, por ejemplo, se ofrece a empresas y ciudadanos el tratamiento electrónico del IVA, de las operaciones de aduanas o de las declaraciones fiscales. Los propios servicios que las diversas administraciones públicas (local, regional, estatal) se prestan entre sí, también están afectados por la e-Administración.

La participación ciudadana en las decisiones de gobierno por medios telemáticos es, probablemente, la aplicación menos desarrollada y también la más controvertida. Quizás el máximo exponente de aplicación de este tipo sea el voto electrónico, todavía poco implantado y lejos de ser una verdadera aplicación de e-Administración, ya que requiere la utilización de dispositivos específicos, lo que no permite eliminar la necesidad del desplazamiento de los votantes a las ubicaciones de estos dispositivos. En este apartado, se pueden encuadrar también las facilidades de comunicación directa de

los ciudadanos con sus gobernantes y representantes, normalmente limitadas a la posibilidad de realizar comunicaciones vía e-mail aunque, generalmente, con escasas consecuencias.

El diseño de aplicaciones de e-Administración implica la consideración de factores específicos de este tipo de sistemas, como son las características de los potenciales usuarios, el volumen y distribución de las transacciones a realizar o, en algunos casos, la concentración temporal de la utilización de determinados servicios.

Las administraciones públicas trasladan su propia complejidad intrínseca a la estructuración de la información y servicios que ofrecen a los ciudadanos, empresas o a otras administraciones. Ello hace que sea, en ocasiones, una labor muy complicada el simple hecho de localizar el punto de prestación de un servicio. Por otra parte, los potenciales usuarios de estos servicios se corresponden con la totalidad de la población lo que eleva enormemente el establecimiento de perfiles de usuario y transforma en un reto la tarea de diseñar interfaces adaptadas a los diferentes perfiles.

En este contexto, las características de adaptabilidad, escalabilidad y aprendizaje propias de los sistemas multiagente hace que estos se revelen como una alternativa interesante en el diseño de sistemas para la e-Administración.

Es posible encontrar numerosos trabajos en los que se utilizan SMA para abordar diversos aspectos de sistemas de e-Administración. Una línea de trabajo es la referida al modelado de sistemas que faciliten la implementación de lo que podríamos denominar “ventanilla única”, mediante los cuales, el ciudadano podría, desde un único punto, acceder a cualquier servicio o información disponible, y la localización del servicio sería responsabilidad del propio sistema. Algunos ejemplos de esto podemos encontrar en [Medjahed y Bouguettaya, 2005] [De Meo *et al.,* 2006] [De Meo *et al.,* 2008]. La interoperabilidad entre administraciones es otro tema para el que también existen propuestas basadas en sistemas multiagente [Gao *et al.,* 2006], en algunos casos construidos sobre la base de Servicios Web [Usman *et al.,* 2006]. Sin embargo, no se han podido localizar referencias a las estrategias de planificación utilizadas en los sistemas multiagente propuestos.

5.1.1 La e-Administración en España

La e-Administración en España está teniendo su mayor impulso en los últimos años, motivado en parte por un marco legal que ha permitido llevar las garantías jurídicas que existen en el mundo real al mundo virtual. En este sentido, la Ley de firma electrónica (Ley 59/2003), establece el concepto de firma electrónica reconocida y la equipara jurídicamente a la firma en papel, dotándola así de plena validez legal para las transacciones electrónicas públicas y privadas. Por otra parte, con la Ley Orgánica de

Protección de Datos de Carácter Personal (Ley Orgánica 15/1999) y su reglamento de desarrollo, se establecen las garantías de confidencialidad de los datos proporcionados por las personas físicas y jurídicas en estas transacciones. Por último, hay que destacar la Ley de Acceso Electrónico de los Ciudadanos a los Servicios Públicos (Ley 11/2007), a la que en muchas ocasiones se refiere como "Ley de Administración Electrónica", que consagra el concepto de e-Administración en el marco jurídico español y la eleva a la categoría de derecho de los ciudadanos. Establece que los ciudadanos tienen el derecho de acceder electrónicamente a los servicios de la Administración Pública y ésta tiene la obligación de proporcionar esas vías de acceso desde del 31 de diciembre del 2009.

5.2 Descripción del caso de estudio

El problema que se pretende solucionar se basa en encontrar una planificación de tareas dentro de una entidad formada por trabajadores que deben coordinarse para finalizar un conjunto de tareas con el máximo beneficio y efectividad posible. Como se comentó anteriormente, las organizaciones virtuales (OV) se presentan como un mecanismo idóneo para el simulado social y como una forma de potenciar la coordinación y la adaptación de los sistemas multiagente abiertos.

El caso de estudio presenta una sociedad orientada a tramitar todos los expedientes que llegan a la administración pública por medios telemáticos. Esta sociedad de agentes se ha diseñado utilizando una metodología de diseño denominada GORMAS [Argente, 2008a], que mediante sus etapas de análisis, diseño de la estructura y diseño de la dinámica permite especificar los servicios que ofrece la organización, su estructura y normas que rigen su comportamiento. El sistema se implementará basándose en la arquitectura THOMAS [Carrascosa *et al.,* 2009] [Giret *et al.,* 2010] comentada en el apartado 3.2.3 y con el modelo propuesto en el capítulo 4.

Los objetivos a alcanzar en el caso de estudio son:

- Desarrollo de la sociedad de agentes, lo que incluye:
  - Definición de roles de agentes: se definirán las funcionalidades generales necesarias para el correcto funcionamiento de la sociedad y las propias del caso de estudio definido. De esta forma quedarán determinados los comportamientos de los agentes que formarán la organización. Hay que tener en cuenta que el sistema será abierto por lo que la definición de estos roles no debe ser condición de imposibilidad para que puedan utilizarse sistemas externos sobre el mismo.

- Definición de los servicios necesarios para el adecuado funcionamiento de una organización.
- Definición de las normas que regirán la sociedad y que influirán en las interacciones y comportamiento de los agentes.
- Definición de los mensajes e interacciones propios de los agentes.

- Simulación del comportamiento de la organización en el caso concreto de coordinación y adaptación de sus agentes.
- Validación del modelo propuesto mediante la simulación de la organización del caso de estudio.

Como se explicó en el capítulo 4, donde se propone el modelo de organización, se dispone de una cola de entrada que contiene todas las tareas pendientes. En la e-Administración cada vez que un usuario introduce una nueva solicitud se crea un nuevo expediente. A partir de ese momento, todas las acciones posteriores referentes a esa solicitud, tanto de personal propio de la Administración Pública, del sistema informático o del propio usuario, estarán ligadas a ese expediente. Por esta razón, cada tarea estará ligada a un expediente. A su vez, los expedientes se tramitan siguiendo unos pasos legales predeterminados que deben ser resueltos en un plazo estipulado previamente. Por ello, cada tarea de la cola de entrada tendrá las siguientes características:

- *Expediente asociado*: se corresponde con un identificador que permite conocer todos los datos del expediente incluido el tipo de expediente. Para el caso de estudio se tendrán en cuenta tres tipos de expedientes: becas, impuestos y multas.
- *Estado o paso en el que se encuentra el expediente*: determina concretamente qué actividad debe realizar el agente y permite conocer cuál es el plazo de tiempo disponible para ello. En el ciclo de vida del expediente cada paso o tarea realizada permite que éste cambie de estado. Cada tipo de expediente tendrá unos pasos y estados determinados.

La adaptación del modelo propuesto para este caso de estudio, se consigue mediante especializaciones de los agentes Tramitadores. Como se explicó en el apartado anterior, los agentes con rol Tramitador se encargan de llevar a cabo las actividades o pasos que requiere cada tarea, es decir, atendiendo a su rol y a la tarea encomendada implementan los servicios necesarios para llevarla a cabo. Las especializaciones identificadas están relacionadas con los tres tipos de expedientes que se tendrán en cuenta:

- *Tramitador de Becas*: agente Tramitador que se encarga de resolver y tratar con los expedientes de tipo Beca.
- *Tramitador de Impuestos*: se encarga de los expedientes de tipo Impuestos.
- *Tramitador de Multas*: se encarga de tramitar los expedientes clasificados como Multa.

En la Figura 14 se muestra un ejemplo los pasos que seguiría un expediente de becas y por lo tanto, los servicios que un Agente Tramitador de Becas debería realizar. En este caso de estudio, sólo se contempla un tipo de beca por lo que sólo existe un diagrama de flujo para determinar los pasos, sin embargo, al modelarse en forma de servicios la adaptación y ampliación del servicio simplemente se consigue añadiendo más servicios, con sus normas, proveedores y clientes.

En el diagrama los recuadros azules representan los pasos o tareas que deben desarrollar los agentes o trabajadores para completar el expediente. Los nombres que acompañan las flechas representan los estados por los que pasa el expediente para identificar el paso que corresponde ejecutar en cada momento. Los recuadros amarillos representan las interacciones que el interesado o ciudadano debe realizar para seguir adelante con la tramitación.

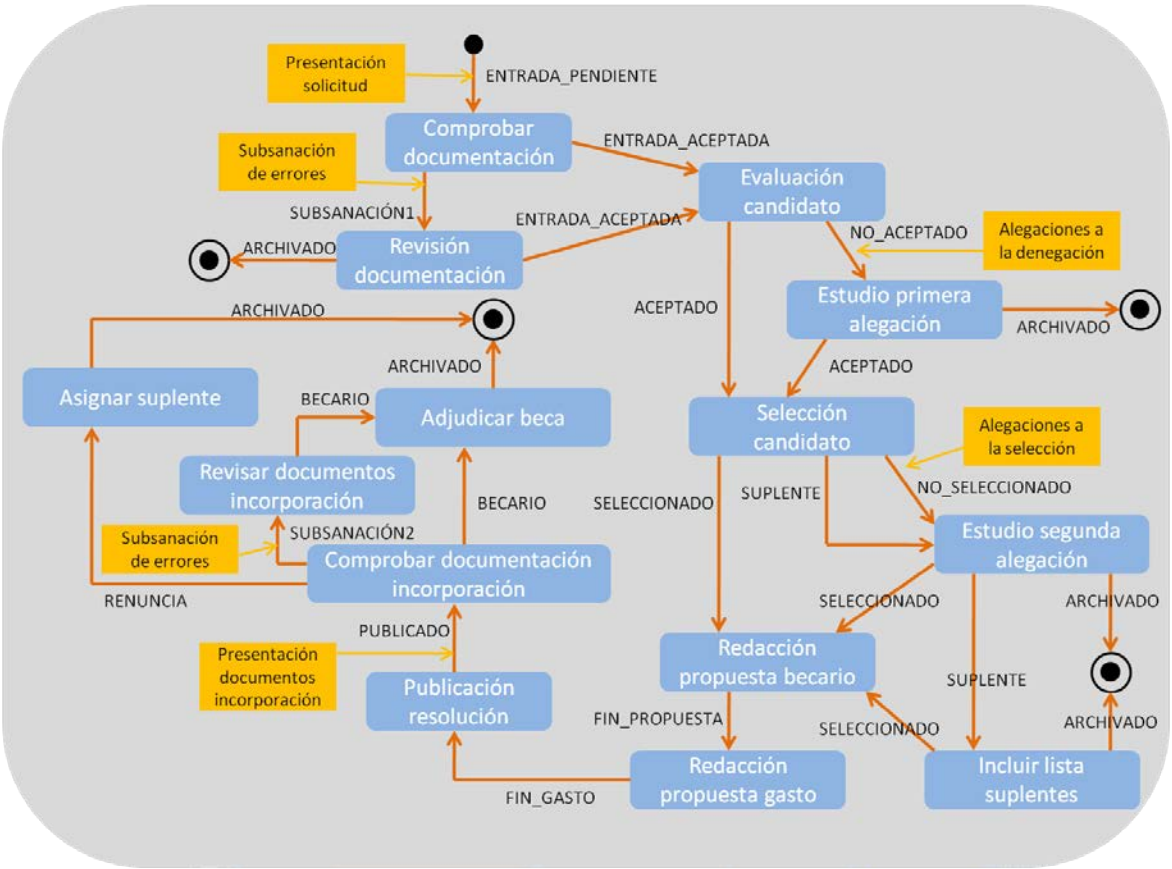

Figura 14 Pasos del expediente tipo becas

Dentro de la organización virtual existirán diferentes servicios que pueden ser invocados por diferentes roles. De esta manera, cuando un Tramitador recibe una tarea, que en el caso de estudio se corresponde con un expediente en un determinado estado, existirá un servicio registrado en el módulo *Service Facilitator* de la arquitectura THOMAS que podrá ser ejecutado por el rol que tramite ese expediente y contendrá todas las operaciones necesarias para que el agente lleve a cabo su tarea.

Siguiendo las pautas de la guía metodológica [Argente, 2008a], en la Figura 15 se observa el resultado de una de las primeras tareas que se debe realizar, instanciar la vista funcional (misión) del modelo de organización donde se representan los productos y servicios que ofrece el sistema, el tipo de entorno, los objetivos globales que persigue, grupos de interés y la información que consume [Rodríguez, 2010].

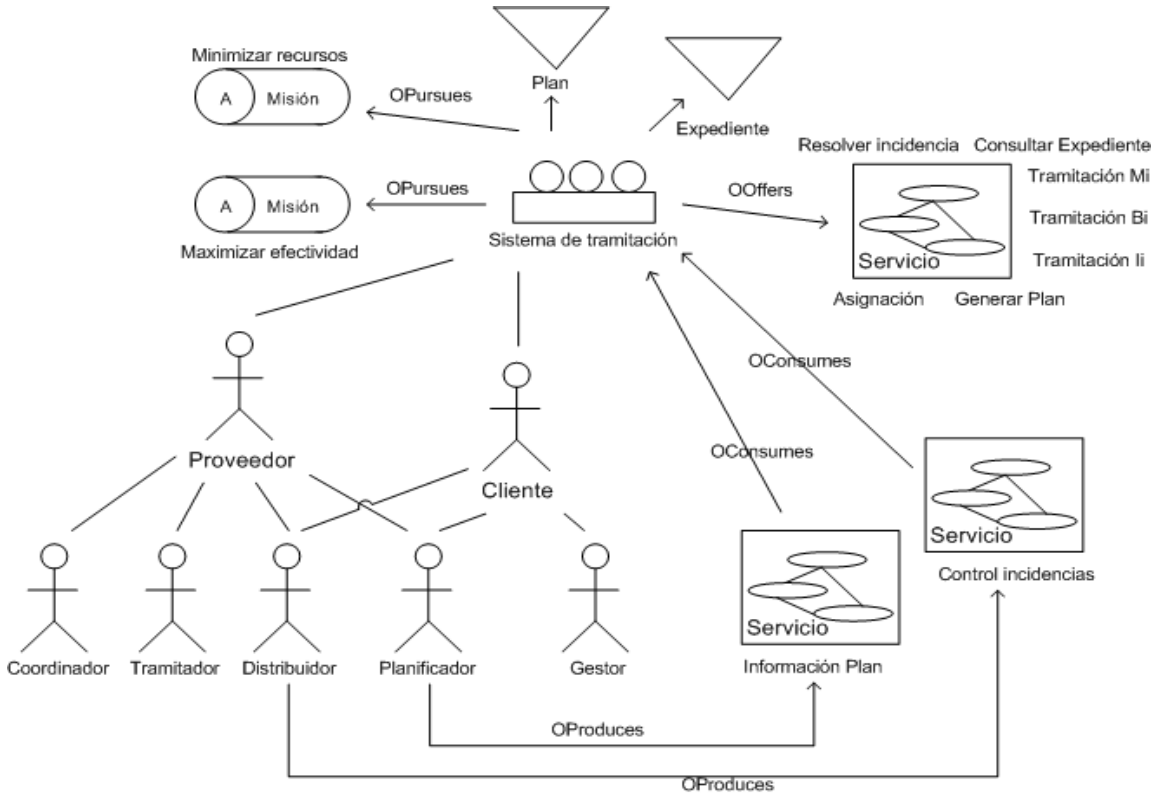

Figura 15 Diagrama del modelo de organización. Misión del sistema.

El sistema ofrece como producto principal un plan para tramitar todos los expedientes y a su vez, ofrece información y servicios que permitan facilitar el trabajo de tramitación. El agente encargado de mostrar la información del expediente será el Gestor y el encargado de los servicios de tramitación será el Tramitador.

La organización ofrecerá todos los servicios necesarios para los distintos trámites, expedientes y planes. También debe resolver incidencias generadas como posibles retrasos o bajas de agentes.

El principal cliente será el Gestor que consume servicios ofrecidos tanto por el Tramitador al finalizar el expediente y por el Planificador en caso de necesitar consultar el plan recomendado. Asimismo el Planificador y el Distribuidor se consideran de forma simultánea clientes y proveedores puesto que, el Planificador ofrece servicios de planificación al Distribuidor y el Distribuidor ofrece servicios de control y asignación al Planificador.

La misión de la organización será, por un lado minimizar los recursos necesarios para la tramitación, intentando optimizar la distribución de tareas y por otro lado, maximizar la efectividad, de forma que se tramiten con más urgencia aquellos expedientes que pueden considerarse más beneficiosos. Los dos objetivos están intrínsecamente relacionados.

Siguiendo las pautas para el modelado de la organización, otro de los pasos que se incluyen es la definición de la funcionalidad, en la que los servicios se asocian con los roles correspondientes mediantes las relaciones WFUtiliza y WFProporciona. Este diagrama se muestra en la Figura 16 .

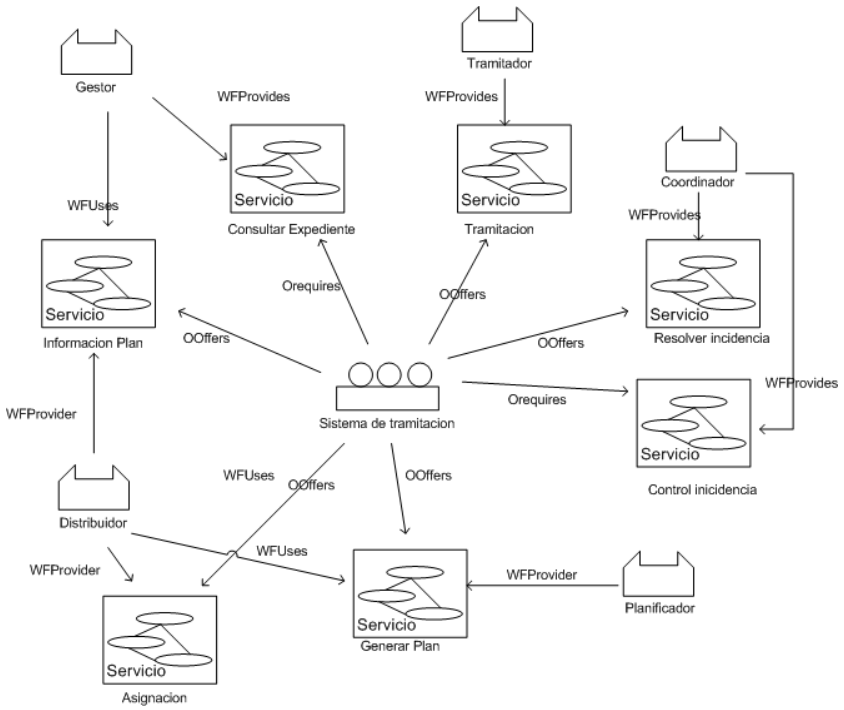

Figura 16 Diagrama del modelado de la organización. Vista funcional.

5.3 Resultados y conclusiones

En este apartado se presenta el sistema construido con el fin de probar la planificación de tareas y los resultados obtenidos con ésta. El programa se encarga de realizar una simulación del comportamiento que tendría la organización virtual y en concreto, el agente planificador. Asimismo, mientras realiza la simulación se recogen las acciones realizadas para mostrar las estadísticas sobre el funcionamiento y se muestran recomendaciones sobre la distribución.

En la Figura 17 se muestra la interfaz principal del programa de simulación.

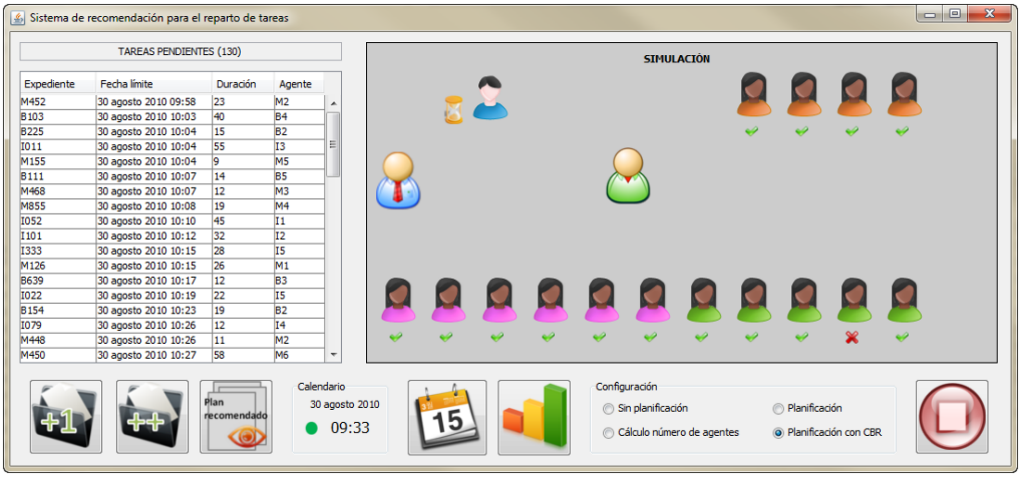

Figura 17 Interfaz del sistema de simulación

La tabla que se encuentra en la parte superior izquierda se corresponde con el estado de la cola de expedientes que esperan tramitación. Esta tabla contiene la siguiente información:

- *Expediente*: identificador único del expediente. En el caso de estudio se tienen en cuenta tres tipos de expediente que definen la letra de inicio del identificador: M para expedientes de multas, B para expediente de becas, I para impuestos.
- *Fecha límite*: cada uno de los pasos en los que se divide un expediente tiene asociado un límite de tiempo en el que debe ser resuelto. En esta columna se muestra la fecha límite en que el paso actual del expediente debe ser

tramitado, calculada a partir de la fecha de terminación del paso anterior y la fecha legal establecida para la resolución del paso correspondiente.

- *Tiempo establecido*: es el tiempo que se prevé que tarde el agente asignado a tramitar el expediente en finalizar el trabajo. Este campo depende del modo de simulación escogido, si iniciamos la simulación "Planificación con CBR", será el agente planificador el encargado de recuperar el tiempo estimado que tardará el agente, dato que se encuentra incluido en los casos del CBR. En caso contrario, el tiempo establecido es un tiempo medio igual para todos los agentes pero dependiente del tipo de tarea o paso del expediente.
- *Agente asignado*: identificador correspondiente al agente que el plan generado ha asignado a realizar esa tarea. De forma similar a los expedientes, este identificador está relacionado con el rol que desempeña el agente: M para el rol correspondiente a multas, I para los impuestos y B para las becas.

Hacia la derecha de la interfaz se encuentra el panel donde se desarrollará la simulación y se mostrarán todos los agentes que existen en la organización en cada momento. En este panel pueden aparecer las siguientes figuras:


Tramitador multas


Tramitador impuestos


Tramitador becas


Coordinador


Distribuidor

Planificador

Figura 18 Iconos representantes de cada rol

Las figuras de los agentes Tramitadores van acompañadas de un "tick" verde en caso de seguir la planificación o de una cruz roja en caso de retraso. La figura del "tick" verde es un botón, que si se pulsa se puede introducir un retraso con el fin de hacer pruebas en la simulación.

Toda la parte inferior de la interfaz está dedicada a mostrar las opciones para la configuración de la simulación. De izquierda a derecha se muestran los siguientes botones o paneles:

- Botón para introducir una tarea: pulsando este botón se muestra una ventana que simula la llegada al sistema de un nuevo expediente y donde se debe seleccionar el tipo de expediente. La fecha de llegada es automáticamente detectada por el sistema y el paso en el que se encuentra es siempre el inicial para cada tipo de expediente.
- Botón para introducir una lista de tareas: para introducir automáticamente un grupo de tareas se puede utilizar este botón que permite cargar un fichero con todos los datos asociados a las tareas. Cada línea del fichero corresponde con una tarea en la que el campo inicial es una fecha y hora que determina el momento en que la simulación debe introducir el expediente en la cola.
- Botón de consulta del plan recomendado: en cualquier momento se puede consultar el plan recomendado correspondiente al instante de tiempo en el que se pulsó el botón. El planificador cada vez que diseña un plan lo almacena en un fichero para su consulta, por ello, en caso de pulsar el botón mientras el planificador está generando un plan se obtendría el plan inmediatamente anterior pero que todavía ejecuta la aplicación puesto que el planificador está generando el nuevo.
- Panel calendario: muestra el paso del tiempo de la simulación, donde cada 10 segundos se incrementa un minuto en la simulación. Un recuadro rojo indica que la simulación esta parada mientras que un círculo verde indica que está en funcionamiento.
- Botón calendario: permite establecer la fecha de inicio de la simulación y establecer las horas entre las que los agentes trabajan.
- Botón de consulta de estadísticas: con el fin de evaluar el funcionamiento, pulsando este botón se muestran los siguientes datos:

- Número de agentes utilizados y el tiempo durante el que se utilizaron esos agentes.
- Número de tareas resueltas en total.
- Número de tareas resueltas por cada agente Tramitador.
- Número de tareas retrasadas.
- Tiempo de la simulación.
- Beneficios obtenidos.

- Panel de configuración: permite escoger el modo de simulación, existen las siguientes opciones:
  - Sin planificación: el agente Distribuidor reordena la cola de entrada de acuerdo a la fecha límite de resolución de expediente y según los agentes tramitadores van acabando, el Distribuidor les asigna la tarea que se encuentra al inicio de la cola. El número de agentes se decide en base al número de expedientes que en hay en la cola de entrada. En este caso, es muy probable que algún expediente se retrase debido a que no hay agentes suficientes o que los agentes estén ociosos y disminuya la eficiencia.
  - Cálculo número de agentes: con este modo, se calculan el número de agentes que se prevé serán necesarios para satisfacer la cola de expedientes. Se tiene en cuenta también un beneficio asociado al expediente, de manera que se ordena en base a la fecha límite y posteriormente por el beneficio asociado. La asignación se realiza del mismo modo que el caso anterior.
  - Planificación: entra en funcionamiento el agente Planificador pero el tiempo estimado para la resolución viene determinado por el estado o paso en el que se encuentre el expediente y es el mismo para todos los agentes. No existe capacidad de aprendizaje o adaptación.
  - Planificación con CBR: se incluye el CBR dentro del agente planificador, de manera que con el tiempo el sistema mejora por la capacidad de aprendizaje con respecto al tiempo que emplea cada agente en la resolución de la tarea y es capaz de conocer con más exactitud la tasa de llegada de los expedientes.
- Botón de inicio o fin de la simulación: pulsando este botón se inicia o se para la simulación.

En la imagen del programa en funcionamiento se puede observar hay 130 expedientes en cola para ser tramitados y en el momento de la captura un agente

Tramitador de Impuestos se ha retrasado con lo que el agente Planificador se encuentra replanificando (lo indica el reloj de arena que aparece junto al Planificador) puesto que ya es imposible continuar el plan diseñado.

De cara a evaluar el mecanismo de planificación propuesto y teniendo en cuenta que como objetivo final se pretende tramitar todos los expedientes dentro de los límites de tiempo establecidos o en caso de imposibilidad, maximizar el beneficio obtenido, se llevaron a cabo diferentes pruebas.

El sistema se ha probado con dos listas de expedientes diferentes para los cuatro modos de simulación, que se indican de la siguiente manera:

- Modo 1 equivale al modo sin planificación pero con un número limitado de agentes.
- Modo 2 corresponde al modo en el que utilizando la teoría de colas se calcula el número de agentes.
- Modo 3 equivale al modo con planificación utilizando la teoría de colas y el algoritmo genético.
- Modo 4 corresponde a la planificación introduciendo un sistema CBR en el agente planificador.

En el primer caso (Test 1) la lista de prueba contaba con 500 expedientes introducidos en un plazo de 120 minutos y el segundo caso (Test 2) contaba con 1500 expedientes introducidos al sistema en un plazo de 210 minutos. Previamente se contaba con una memoria de 700 casos, que se basaba en casos en los cuales se habían alterado valores aleatoriamente siguiendo una distribución normal.

En la siguiente figura se muestra la comparativa del número de expedientes que no pudieron ser tramitados a tiempo para cada modo en las dos pruebas realizadas.

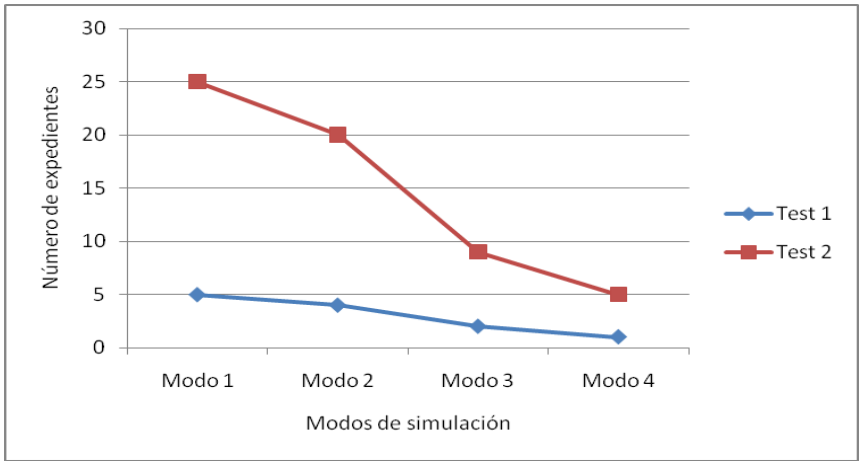

Figura 19 Número de expedientes fuera de plazo

Se puede observar que el salto cualitativo se produce, en ambos casos, por la introducción en el tercer modo de simulación de los algoritmos genéticos para llevar a cabo la distribución. La introducción del CBR también produce una mejora en el último modo ya que permite trabajar con parámetros que se ajustan más a la realidad debido al aprendizaje realizado por el sistema. En el Test 2 para el Modo 4 se observa que el número de expedientes no tramitados a tiempo es superior al Test 1, esto se debe principalmente a que se superó la tasa de llegadas recuperada de los expedientes de la memoria del CBR.

El gráfico que se presenta a continuación (Figura 20), es todavía más significativo puesto que revela los beneficios obtenidos en los distintos modos. Para asignar un valor al campo beneficio de cada expediente se utiliza un sistema de puntos que tiene en cuenta diferentes aspectos, como el tipo de expediente (dando importancia a los impuestos), las ganancias económicas que supone, trabajo realizado anteriormente, etc. De manera que el beneficio se calcula aplicando un sistema basado en reglas.

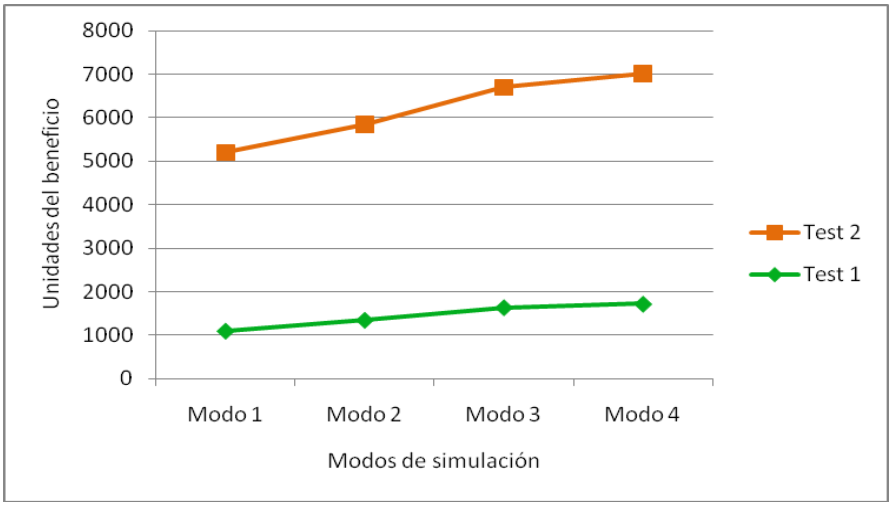

Figura 20 Beneficio obtenido

El peor beneficio como era de esperar se obtiene en el modo sin planificación. En este caso, el hecho de estimar de manera genérica el número de agentes resulta muy impreciso por lo que en muchos casos, se producen pérdidas por no tener agentes suficientes o por tener demasiados. Además puesto que no se tiene en cuenta el beneficio de cada expediente, se le asigna prioridad a expedientes que suponen pérdidas.

En el segundo caso, la utilización de la teoría de colas mejora los beneficios al poder equilibrar el número de agentes necesarios y la ordenación por beneficio permite asignar una prioridad que mejore las cifras.

En el tercer modo, como sucedía con el gráfico anterior, revela que la introducción de una distribución de tareas basada en un algoritmo genético es el punto fuerte de la planificación. Finalmente, todavía se obtienen unos beneficios mejores teniendo en cuenta la adaptación a partir de casos pasados que proporciona el CBR.

Capítulo VI

Conclusiones

VNiVERSiDAD
D SALAMANCA

En este trabajo se ha presentado un modelo de asignación de roles y asignación de tareas en organizaciones virtuales que permite trabajar en tiempo de ejecución. Las organizaciones virtuales requieren del uso de mecanismos que controlen la conexión y desconexión de agentes a las organizaciones de modo que se garantice el cumplimiento de determinadas reglas dentro de la organización. En este caso se ha buscado optimizar el rendimiento en función de los agentes.

El modelo de planificación definido, se integra en una arquitectura multiagente genérica orientada a la distribución de tareas e integrada con una arquitectura SOA. La integración del procesamiento mediante servicios web facilita la extensibilidad y adaptabilidad del modelo permitiendo incorporar nuevos mecanismos de planificación según las necesidades de las organizaciones. El corazón de la arquitectura son los agentes CBR-BDI y agentes con capacidad de planificación basados en casos bajo el modelo BDI, agentes deliberativos que incorporan mecanismos de razonamiento basado en casos y de planificación basada en casos. Estos agentes dotan a la arquitectura de una gran capacidad de adaptación al contexto y facilitan el proceso de toma de decisiones. Se trata de elementos de computación que trabajan de forma distribuida.

En este trabajo, se han estudiado diferentes alternativas metaheurísticas y su integración con diferentes técnicas en agentes CBR-BDI para realizar tareas de asignación y distribución. Los métodos metaheurísticos permiten resolver problemas genéricos, por lo que son aplicables a diferentes tipos de problemas, y se han integrado con técnicas estadísticas como la teoría de colas para realizar la búsqueda de la solución final en tiempos de ejecución eficientes. Para llevar a cabo dicha tarea, el modelo de planificación integra un sistema de planificación basado en el modelo de planificación basado en casos bajo el modelo BDI. El modelo de planificación basado en casos BDI integra a partir de su representación simbólica del entorno definida en la arquitectura BDI un motor de razonamiento basado en los CBR. EL motor de razonamiento se caracteriza por la definición de un ciclo de razonamiento formado por una serie de etapas que permiten recuperar planes pasados y adaptarlos para resolver problemas actuales. Este modelo híbrido ha permitido crear un modelo de planificación automático que permite, a partir de los planes recuperados, definir una nueva planificación estimada como más eficiente en función de métricas de eficiencia definidas.

Para poder llevar a cabo la planificación automática, es fundamental el modelo simbólico BDI que permite definir de modo simbólico los planes llevados a cabo mediante una base de creencias. La base de información del modelo simbólico es aplicable a agentes de diferentes tipos como es el caso de los agentes reactivos, ya que la información existente en la memoria sirve para la adaptación dinámica de las reglas de condición acción para este tipo de agentes. Los planes almacenados en la memoria no son más que un conjunto de asociaciones tareas-agentes que determina la secuencia en la que las tareas se han ido llevando a cabo al final de una jornada laboral.

Los planes no son más que un conjunto ordenado de acciones que se aplican sobre un estado inicial que viene establecido por un conjunto de variables que define el

entorno. En este caso, las acciones vienen determinadas por la selección de agentes, asignación de roles y la asignación de las actividades a cada uno de los roles. Mediante esta definición formal de los planes y haciendo uso del motor de razonamiento basado en planes se ha conseguido generar planes de mayor eficiencia mediante la incorporación de técnicas estadísticas, algoritmos evolutivos y teoría de colas en las diferentes etapas del CBR. Por tanto, la arquitectura propuesta, permite por un lado descomponer el problema en agentes que colaboran entre sí para la obtención de un resultado final, permite separar la funcionalidad de la lógica de negocio mediante la incorporación de servicios web y finalmente permite aprender de las experiencias pasadas mediante la planificación llevada a cabo con el modelo de planificación basado en planes con el modelo BDI. Por tanto, la arquitectura propuesta facilita la incorporación de nuevos roles que utilicen diferentes técnicas de modelado y estrategias de aprendizaje de forma que se puedan realizar experimentos adicionales con nuevas técnicas y puedan ser comparados con los resultados iniciales se han presentado en este documento. De la misma forma los agentes CBR-BDI y los agentes con capacidad de planificación basados en el modelo BDI facilitan la incorporación de nuevos algoritmos a cualquiera de las etapas del ciclo CBR mediante la inclusión de nuevos servicios web ya que la definición del comportamiento de los roles se encuentra fuera de los agentes.

Finalmente, a parte de la investigación realizada en los modelos de planificación, se ha desarrollado un mecanismo de planificación basado en diferentes paradigmas de la Inteligencia Artificial Distribuida mediante el uso de sistemas multiagentes y servicios web que incorporan de Inteligencia Artificial y técnicas estadísticas como mecanismos de razonamiento.

6.1 Contribuciones de la Investigación

Este trabajo hace nuevas contribuciones en el ámbito de las organizaciones virtuales adaptativas. Se ha propuesto un modelo de planificación integrado en una arquitectura multiagente que incorpora diversas técnicas para llevar a cabo la distribución de tareas de modo automático. A continuación se resumen las aportaciones realizadas:

1. Marco para la gestión de organizaciones virtuales
   Se ha realizado un estudio de las técnicas metahuerísticas existentes para realizar tareas de asignación de roles automáticos.
   a. Se han aplicado técnicas estadísticas para el cálculo de error máximo en base a las planificaciones recuperadas.

   b. Se ha aplicado teoría de colas para la predicción del número óptimo de agentes por cada tipo de tarea en base a previsiones de flujo de llegada.
   c. Se han aplicado algoritmos genéticos para realizar tareas de asignación y distribución de tareas en base a la eficiencia de los agentes y al beneficio acumulado de modo que las tareas sean lo más replanificables posibles y se minimicen las pérdidas.

2. Marco de la Inteligencia artificial distribuida
   Se ha realizado un estudio de las técnicas existentes para la elaboración de sistemas con capacidad de aprendizaje y colaboración. Posteriormente, estas técnicas se han aplicado para elaborar mecanismos de planificación automática.
3. Marco para la planificación automática
   Se ha desarrollado un sistema de planificación automático que permite la distribución de roles y asignación de tareas en base a la eficiencia de los agentes disponibles. El modelo propuesto incorpora conceptos como el modelo de planificación basado en casos, teoría de colas y algoritmos genéticos.
4. Diseño de agentes inteligentes
   Mediante la incorporación de los mecanismos de planificación automática comentados anteriormente.
5. Propuesta de una arquitectura multiagente aplicable a entornos inteligentes
   Se ha propuesto una arquitectura multiagente que facilita la planificación automática que integra las arquitecturas SOA lo que facilita la separación de los agentes y su comportamiento
6. Aplicación de la arquitectura
   Se ha aplicado la arquitectura a casos de estudio reales con los que se han validado tantos las técnicas empleadas como el proceso de planificación automática.

Finalmente, es posible concluir que se ha realizado un trabajo de investigación, en el que se ha estudiado el estado del arte de la teoría de agentes, sistemas multiagente, y técnicas aplicadas a la planificación. Se han estudiado los mecanismos deliberativos aplicables a los agentes, las herramientas para la construcción de sistemas multiagente y de entornos inteligentes, y un problema referente a la importancia de la creación de entornos de planificación automática aplicado al egoverment para el control y distribución de trabajo. A partir de estos estudios, se ha procedido a construir una arquitectura multiagente con capacidad de planificación mediante la incorporación del modelo de planificación basado en casos y el modelo BDI.

6.2 Trabajo futuro

Este trabajo deja abiertas una gran cantidad de puertas para un trabajo futuro. A continuación se muestran algunas de ellas:

1. Comprobar y validar los sistemas multiagente de este trabajo dentro de los entornos estudiados, de tal forma que se puedan evaluar los modelos propuestos en términos de tiempo, facilidad y calidad de análisis y diseño, tiempo de cálculo de respuestas, calidad de las respuestas, etc.
2. Negociación
   Como ya se ha indicado a lo largo de este trabajo, se ha decidido enfocar las interacciones hacia el ámbito dentro del entorno. Sin embargo los distintos sistemas multiagente deben cooperar entre sí para obtener comunicación y computación ubicua. Sería deseable que se produjese una interacción entre sistemas multiagente de diferentes características y, a ser posible, una negociación de parámetros entre ellos.
3. Seguridad
   Se podrían dotar de seguridad las comunicaciones realizadas dentro de nuestro sistema multiagente, ya que los datos intercambiados pueden considerarse importantes y debe garantizarse privacidad.
4. Aumentar la cantidad de algoritmos
   Se considera adecuado aumentar la cantidad de algoritmos para realizar planificaciones en función del tiempo disponible. Se está trabajando en la sustitución de los algoritmos genéticos por redes neuronales para realizar tareas de asignación de modo más eficiente.
5. Resolución de nuevos problemas prácticos
   Para comprobar la validez de la arquitectura propuesta, sería deseable aplicarla a nuevos problemas prácticos. De esta forma sería posible comprobar si se adapta de forma adecuada a la resolución de problemas de características distintas, o bien si se ha construido una arquitectura muy restringida al problema concreto que se ha estudiado durante la realización de este trabajo.
6. Extender las funciones objetivo
   Incorporar nuevos tipos de funciones objetivos que permitan maximizar o minimizar diferentes aspectos de los problemas y que además permitan incorporar restricciones de asignación de tareas a agentes de modo implícito y no explícito como se realiza actualmente. En la actualidad la asignación de las tareas se hace en base a datos de asignaciones previas pero no se incorporan restricciones particulares de cada planificación.

7. Comunicación
   Incorporar nuevos protocolos de comunicación que mejoren la interconectividad del sistema con otras tecnologías para facilitar la computación distribuida mediante Grid Computing y permitir el uso de servicios web de tipo REST.

Capítulo VII

Research Overview

VNiVERSiDAD
D SALAMANCA

When working in conjunction with multiagent systems, virtual organizations attempt to simulate the functions and interactions of entities in different environments. Recent studies have addressed the problem of assigning roles to agents that form part of the organization, and incorporating new agents to carry out certain tasks. However, these studies are limited to defining the norms and rules that determine the behaviour of the organization. The present study proposes a virtual organization model for e-government environments to assign resources and minimize the required personnel by forecasting workloads. To this end, queuing theory, a genetic algorithm and CBR are used to obtain an efficient distribution. Queuing theory can establish the number of agents with a specific role that are necessary to maximize profits, while the genetic algorithm distributes roles among agents according to their respective efficiency. The final part of the paper is focused on validating the plan developed inside a case study centered on e-Government in order to obtain empirical results.

Introduction

Various studies have explored the concept of planning in virtual enterprises or organizations [Camarinha-Matos & Afsarmanesh, 2002] [Wu & Sun,2002] [Verter & Dincer, 1992] [Fenga & Yamashiro, 2006]. Nevertheless, these studies have focused on defining the objective parameters and functions of problems, which inhibits their broader application. Furthermore, these systems neglect to address potential demands and thus provide plans based on the current workload at a given time. Other studies seek to optimize the allocation of resources to maximize efficiency and minimize costs by means of different heuristic or precise optimization techniques [Fenga & Yamashiro, 2006]. The main problem in these cases stems from the dynamic aspect of work scenarios and the difficulty in finding a balance between time spent on planning and time spent on implementing plans, which is the key to adapting to the needs of organizations such as those that require frequent replanning. We therefore propose the need to create a system that is capable of carrying out an efficient planning system not only for current and existing work, but for the prediction of future work as well.

Planning strategies are commonly based on minimizing objective and multi-objective functions, and matching offers and demands [Camarinha-Matos & Afsarmanesh, 2002]. The problems that arise from objective functions are resolved by exact or heuristic means. The exact methods, such as linear programming [Medjahed & Bouguettaya 2005], nonlinear programming [Wan *et al.,* 2009] [Shi *et al.,* 2005] and graph theory, ensure that an optimal solution is provided in execution time according to the number of existing variables. Early research in virtual organizations was based on optimization through whole linear programming [Fenga & Yamashiro, 2006] [Verter & Dincer, 1992]. However, techniques such as linear or nonlinear programming are not applicable to NP-Hard problems. Moreover, it is inherently difficult for these techniques to define constraints and objective functions because the objective function must present all of the existing combinations in order to define the variables, and this number of combinations can be quite high. As a result, it is necessary to either resort to heuristics to solve optimization problems [Wu & Sun, 2002] in which the parameters of the objective function are calculated, or to combine heuristic techniques with linear programming to solve these types of problems [Verter & Dincer, 1992]. It is advisable to use metaheuristic techniques [Kim & Park, 2004] [Porto *et al.,* 2000] [Taillard*.,* 1994] [Kolonko, 2009] to solve these types of problems, as they can obtain efficient solutions in reasonable execution times.

This study proposes a multiagent based Virtual Organization (VO) model with planning and resource distribution capabilities, as well as the ability to estimate work demands in order to both determine the number of resources needed and, according to the number of demands, to distribute work resources in such a way that maximizes profits and minimizes delays. The planning system will be developed according to a case-based reasoning (CBR) system [Kolodner, 1993] which can be integrated into the various stages of reasoning techniques in order to estimate resources. This estimation is based on queuing theory and planning, and uses a variant of a multilayer perceptron that incorporates unsupervised learning. The planning mechanism is applied to virtual organizations [Zambonelli *et al.,* 2003] [Boella *et al.,*. 2005] of agents [Durfee *et al.,* 1989] to simulate the behaviour of the organization in its planning and allocation of work for the given e-government case study. Due to the high number of procedures and users involved in this type of entity, it is essential to have a good planning system that can ensure the work is carried out successfully and in a timely manner, and that the staff allocated to the project is the minimal number required to successfully meet future challenges.

Section 2 of this work presents the state of the art for the different concepts used: multiagent systems, virtual organizations, and task planning methods. Section 3 presents the proposed model and includes solutions for a dynamic distribution of roles and subsequent task assignments. The solution is explained in detail in section 4, which also includes the case study and corresponding results. Finally, section 5 presents our conclusions and future work.

7.1 Virtual Organizations and Multi-Agent Architectures

The technology of dynamic agent organizations that auto-adjust to obtain advantages from their environment is more than suitable for coping with the development of this type of system. These organizations could appear in emergent or dynamic agent societies, such as grid domains, peer-to-peer networks or other contexts where agents dynamically group together to offer compound services. However, although technology, on a theoretical level, seems to be able to allow the development of this new kind of system, it is still necessary to investigate theories, models, mechanisms, methods and tools in order to develop systems with reorganization capabilities that provide them with the ability to adapt to environmental changes.

Multi-agent systems are characterized by being autonomous, reactive, proactive, socially skilled, etc., [Wooldridge & Jennings, 1995] and therefore are perfectly suited for creating organizational models in which each agent can send and receive messages with an individual, organization or actor in the real society that is being modelled [David *et al.,* 2004]. Additionally, the interaction between agents can correspond to interactions existing in the real world [David *et al.,* 2004]. There are many agent-based social simulation models that try to analyze different social phenomena [Epstein, Axtell, 1996] [David *et al.,* 2004]. Schelling [Schelling, 1978] carried out the first agent-based social simulation in which each person was represented by an agent and the interaction between these agents represented relevant social processes. Nevertheless, there is still much work to be done, especially in the field of automated re-organization of virtual organizations [David *et al.,* 2004].

The open MAS [Zambonelli *et al* 2003] [Bajo *et al.,* 2009a] should allow the participation of heterogeneous agents, which change over time, with architectures and even with different languages. For this reason, we cannot rely on agent behaviour when it is necessary to establish controls on the basis of norms or social rules that can affect the organizational architecture and the agents residing within it. This is one of the reasons that encourage the use of virtual organizations (VO). A VO [Esteva *et al.,* 2001] is an open system designed for grouping, for the collaboration of heterogeneous entities, and for when there is a separation between form and function that defines their behaviour. Other trends such as [Boella *et al.,*. 2005] define standards of cooperation between agents according to their interaction, and use BDI models to establish behaviour. Camarinha-Matos *et al* [Camarinha-Matos & Afsarmanesh, 2002] raised the issue of coordination within virtual organizations and, since it is no longer a question of problems that are limited to a simple matching between needs and resources but one that involves multiple variables, suggested that these systems be used to assist in the decision-making process.

Coordination mechanisms with agents have been proposed in different ways [Jennings & Wooldridge, 1998], the most common of which are mediated methods. The intermediary methods play an important role in artificial societies in a way similar to

what occurs in human societies. There are many coordinator models, of which the two primary and opposing methods are global and individual coordination. In global coordination, the MAS determines and plans the actions of the agents, while in the individual coordination, the MAS completes the autonomy of the agents.

The present study proposes a global planning model developed by THOMAS that can carry out a global coordination in an organization. THOMAS [Carrascosa *et al.,* 2009] arose from the need to support open multi-agent systems in virtual organizations. The architecture presents the infrastructure needed to develop the concepts of agents, and applies splitting, abstraction and organization techniques while taking all requirements into account.

The use of THOMAS implies that the organizations are composed of Hierarchical Organizational Units in which a supervisor agent has control over all other members, coordinates the tasks, and centralizes the planning and decision process. The THOMAS [Carrascosa *et al.,* 2009] platform additionally provides new adaptation methods that allow the structure to be changed so that it may adapt to the external changes.

Approaches towards the integration of multi-agent systems with SOA and Web Service have recently been explored [Ardissono *et al.,* 2004]. Some developments are centred on communication between these models, while others are centred on the integration of distributed services, especially Web Services, into the structure of the agents. Oliva *et al* [Oliva *et al.,* 2008] have developed a Java-based framework to create SOA and Web Services compliant applications, which are modelled as agents. Communication between agents and services is performed by using what they call "artifacts" and WSDL (Web Service Definition Language). Shafiq *et al* [Shafiq *et al.,* 2006] propose a gateway that allows interoperability between Web Services and multi-agent systems. Xiang [Xiang, 2007] proposes a multi-agent architecture to develop inter-enterprise cooperation systems using SOA and Web Service components and communication protocols. Walton [Walton, 2006] presents a technique to build multi-agent systems using Web Services, defining a language to represent the dialogs among agents. There are also frameworks, such as Sun's Jini and IBM's WebSphere, which provide several tools to develop SOA-based systems. Jini uses Java technology to develop distributed and adaptive systems over dynamic environments. Rigole *et al* [Rigole *et al.,* 2002] have used Jini to create agents on demand in a home automation system, where each agent is defined as a service in the network. WebSphere provides tools for several operating systems and programming languages. However, the systems developed using these frameworks are not open at all because the framework is closed and services and applications must be programmed using a specific programming language that supports their respective proprietary APIs (Application Programming Interface).

7.2 Planification Models

In its broadest sense, the problem of planning consists of establishing an appropriate order of execution to a set of activities according to specific criteria such as efficiency, quality, time, cost, etc. Recent interest in the field of business and economics, and the corresponding need to provide answers to the problems of planning, has led many researchers to propose new planning mechanisms using Artificial Intelligence [Tupia, 2004]. One such example is the task scheduler, which can be defined as a set of tasks that must be executed by a group of heterogeneous entities, and whose purpose it is to find an order and distribution that minimizes the number of available resources and can meet the restrictions of each individual task.

When looking to solve planning problems, there are generally two available options to choose from: exact methods and approximate methods. The first is an optimal method that consists of calculating the exact solution; this may involve lengthy computations, which make this approach impractical. The second alternative involves the use of heuristic methods to calculate a near-optimal solution; this option is less time-consuming, which justifies its use.

The nature of the problems is combinatorial and therefore the time required to find the optimal solution grows exponentially with the number of tasks considered. These problems fall within the scope of combinatorial optimization and can be considered, at best, NP (non-deterministic polynomial) problems. These problems do not have a polynomial algorithm that resolves them, but they do have an algorithm that provides a non-optimal solution, although it is very time-consuming for the few instances it provides [Rios & Bard, 2000]. However, the majority of the combinatorial optimization problems belong to NP-hard problems (a subset of NP problems). This situation justifies the application of heuristic algorithms for the search of solution in reasonable time.

Heuristics is one of several procedures of approximation. Although the solutions provided by a heuristic approach are generally quite good, heuristics cannot guarantee that the optimal solution for a problem will actually be found. Furthermore, in most cases it is difficult to adapt these algorithms to a problem that is only slightly different from those for which the algorithm was created in the first place. One example of a heuristic algorithm is the voracious algorithm (AVM), which selects the optimal option during each step of a problem solving process with the goal of finding an optimal general solution. This algorithm is also referred to as greedy myopic: greedy because it always selects the best candidate to form part of the solution; and myopic because the selection is unique and unmodifiable, as it does not analyze beyond the effects of having selected an element as part of the solution. Some examples of these applications can be found in these studies [Tupia & Mauricio, 2004].

The heuristic algorithms that are easily adapted to other problems are called meta-heuristics. Meta-heuristics are intelligent strategies for designing, improving and optimizing general heuristic procedures and providing them with a higher performance

compared to traditional heuristics. The most common meta-heuristics algorithms are commented below.

GRASP algorithm. The term GRASP Algorithm is an acronym for Greedy, because of the criteria used by the algorithm in choosing the optimal value after each step of selecting a candidate; Randomized, because the algorithm randomly chooses a candidate from the list it has obtained; Adaptive, because it is capable of adapting to different application contexts or relevant modifications of the model; and Search Procedure, because it searches within a space or neighborhood, randomly evaluating a set of possible solutions.

While the AVM algorithm focuses on selecting the candidate with the best value at a given time, the GRASP algorithm broadens the restriction to select not necessarily the best solution, but one within a given set of values. Each possible solution is evaluated by an objective function, thus ensuring the selection of a better solution than that provided by the voracious algorithm. These values are not necessarily local optimal values, but they are the primary means of finding one by using a local search. Examples of the application of this algorithm in a planning process can be found in these studies [Kim & Park, 2004] [Andres *et al.,* 2008] [Binato *et al.,* 2000].

Tabu Search. This algorithm performs a local search based on the notion that a specific solution can be improved by applying minor changes. For each solution a set of neighbor solutions are calculated; the algorithm then selects the best solution to continue with the next step. The algorithm stores a search history to prevent the iteration from entering into a loop and becoming confined to local values. There are usually two types of memory: the short term memory that stores the most recent searches; and the long term memory that contains older searches. The short term memory consists of a tabu list that stores information regarding the most recently accessed solutions, so that, any solutions matching a case in the tabu list are summarily rejected. The following publications reviewed some task planning cases in which the application of these algorithms had an effective result [Taillard*,* 1994] [Kolonko, 2009] [Chambers & Barnes, 1999].

Simulated Annealing. Simulated Annealing is based on a close analogy between the process of thermodynamic physics and that of a combinatorial optimization problem. The traditional local search algorithms start with an initial solution that is gradually transformed into other solutions that are in turn improved by the addition of minor changes or mutations. If the new solution is better than the previous result, the solution is updated and the system repeats the process until it is no longer possible to obtain a better solution. In this way, the search ends with a local optimum that is not necessarily a global optimum. One way of avoiding this type of problem is to direct the result to a worse solution. However, if the search is in fact leading to a good solution, these kinds of escape movements must be very carefully applied because the system could be leading to a global minimum. Simulated Annealing must be carried out with a probability function that reduces the probability of the escape movements leading to worse solutions during the search, which would in turn assume that the solution is closer to a global optimum. Some studies related to the use of this method for task

planning can be found in [Kolonko, 2009] [van Laarhoven *et al.,* 2002] [Steinhofel *et al.,* 1999].

Genetic Algorithms. The general purpose of evolutionary algorithms is to guide the stochastical search in producing a set of structures from which the most appropriate solutions are selected iteratively and a quasi-optimal solution is ultimately found. Unlike other methods, there is not just one solution, but an entire set of solutions for each iteration of the algorithm. These methods are based on generating, selecting, combining, and replacing solutions. Because they manage several solutions at a time during the search process, the execution time tends to be greater than other meta-heuristic algorithms. In general terms, a genetic algorithm consists of a population of solutions coded in the same way as chromosomes. The chromosomes have a fitness value that quantifies the 'goodness' of a solution to the problem. The number of opportunities for reproduction will be determined according to this value. There is, furthermore, some probability that these chromosomes will mutate. The main advantages of genetic algorithms include their ease in adapting to both general and specific problems, their broad theoretical base, which can easily hybrid with other paradigms, their easy implementation, and the number of empirical tests that provide specific operators. These algorithms have been used in several planning problems, such as those found in [Lee *et al.,* 2008] [Yu & Buyya, 2006] [Shi *et al.,* 2005].

Many task planning theories have been proposed with varying rates of success. In [Seda, 2007] the author present an interesting mathematical model applied to the problem of task planning. A review of different comparative studies [Pacheco & Casado, 2003] [Rodríguez, 2003] on the efficiency of various meta-heuristics indicates that no single technique stands out clearly above the rest.

7.3 Proposed Architecture

The proposed architecture is an organization model for e-goverment environments that incorporates agents with capabilities for automatically generating work plans and distributing tasks. The core of the system is a novel plan-based planning mechanism that interacts with a BDI model through the implementation of web services, which allows for self-adaptive capabilities in different environments. The system additionally provides communication mechanisms to facilitate integration with the SOA architecture.

The proposed model was designed to develop a planning mechanism to coordinate the agents located inside the virtual organizations. To this end, a set of roles and activities are defined according to these needs. Agents with certain roles act as service coordinators and controllers; the services are responsible for implementing the behaviour of the agents and processing the information, which facilitates tasks involving replication and modulation. In the proposed system, the roles in the platform are hierarchical: the higher organizational layer contains roles from the platforms

responsible for the task distribution functionalities; and the lower layers contain the processes roles, which are responsible for developing the tasks.

The model proposed in this paper focuses on developing a planning mechanism to coordinate the agents found in the VO. We will first identify the roles that these agents can assume:

- **Processor role.** Responsible for carrying out the activities required for each specific task. For this reason, the agent will specialize in a specific activity according to the type of tasks the system must resolve.
- **Planner role.** Designs the overall plan to be implemented by the organization. Sets the number of processor agents and distributes tasks according to the role they play. Replans according to the size of the input queue or the inability to accomplish an activity with a plan.
- **Distributor role:** Distributes tasks according to their completion by the agents and checks that each task is being processed within the time limits established for the plan.
- **Coordinator:** Performs the general control of the system. Communicates with the THOMAS platform elements to carry out control actions (connect, disconnect, exceptions, etc.).
- **Manager role:** This agent manages all the information of the task and communicates to the user.

Figure 1 illustrates the different agents of the system and the interactions among them. The task list, which stores the activities to carry out in the multi-agent system, is shown in the upper corner; the agents and the interconnections can be seen in the center of the image.

The service layer is responsible for carrying out the functionality of the different agents. This ensures a separation between the functionality and the business logic. The behavior of the agents is independent of the agent itself since the functionality depends on the roles that the agent has at a particular moment. In order to achieve this separation, the role distributor assigns web services that integrate the behavior of CBP-BDI and reactive models associated with each role. The reactive models are based on the definition of dynamic rules that are analyzed and interpreted by the BRMS (Business Rule Management System). There two different services associated with the roles: planner and reactive. The planner service incorporates tasks from the CBP-BDI model assigned to the planner role, while the reactive model incorporates a business rules reasoning engine that interprets the business logic of the system.

The service layer includes a service called Facilitator Directory that provides information on the various services available, and manages the XML file for the UDDI (Universal Description Discovery and Integration). To facilitate communication between agents and services, the architecture integrates a communication layer that provides support for the FIPA-ACL (Foundation for Intelligent Physical Agents-Agent Communication Language) and SOAP (Simple Object Access Protocol) protocols.

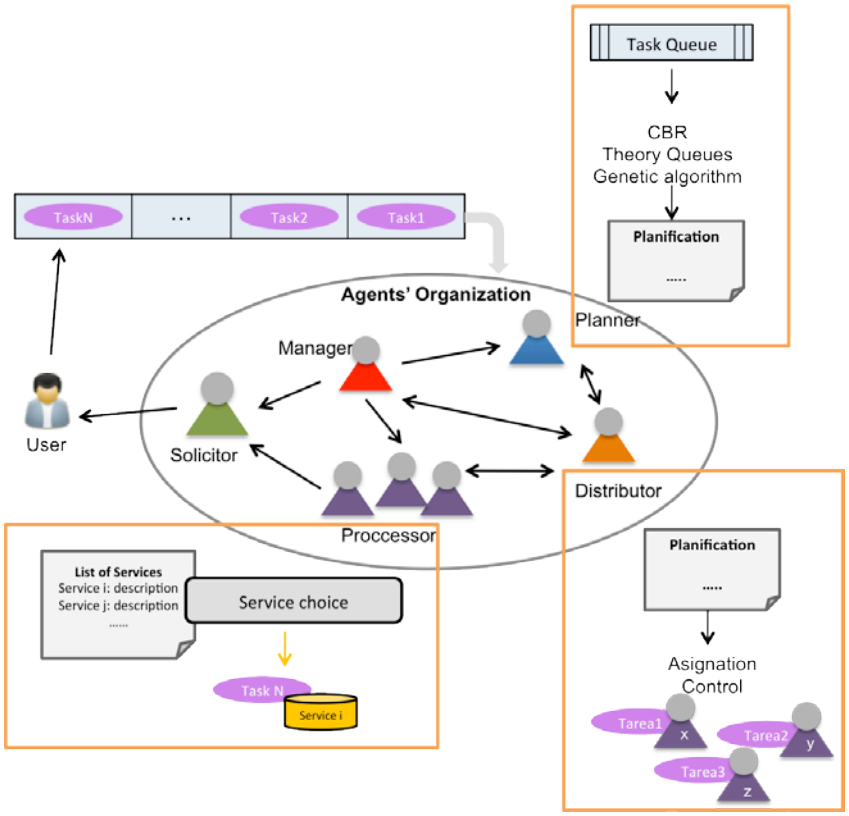

Figura 21 Outline of proposed model

One of the major problems in the development of an architecture based on a multi-agent system is that there are currently no clear standards or well developed methodologies for defining the steps of analysis and design that need to be taken. There are at present a number of methodologies: Gaia [Wooldridge, 2009], AUML (Agent UML) [Corchado y Laza, 2003], INGENIAS [Pavón *et al.,* 2007], TROPOS [Giorginia *et al.,* 2005], MESSAGE [EURESCOM, 2001]. The main approach of agent-oriented methodologies is centred on the individual actions of the agents, many of which have been developed from extensions of object-oriented methodologies or from knowledge engineering, while others have started from totally independent developments. The majority of these methodologies define the phases of analysis and design, whereas only some (like Tropos, Prometheus, MaSE, MASSIVE) also detail the implementation phase. They assume that the agents have common objectives, are benevolent, and cooperate with each other to reach these common objectives. The social rules are not specifically defined, and the social structure, which is presented by the system, emerges from the interactions of the agents, yet remains undefined as well, both in the analysis and in the design of the actual methodology. For this reason, these types of methodologies are not useful by themselves for the development of open multiagent systems, and they only allow the development of closed systems in which the participation of external agents is not admitted, is normally unreliable and uncooperative, and includes self-interested

behaviours.

The main approach of an organization-oriented methodology is centred on its own organization of the system, and takes its objectives, structures and social norms into account. These methodologies primarily detail the analysis phase and in some cases the design phase. However, it is necessary to have an agent-oriented methodology to complete the internal aspects of the agents and the implementation of the system, even though in the majority of the cases there is no explanation as to how this support is actually done. In addition, some of the proposals are merely extensions of agent-oriented methodologies, although many are independent developments that apply organizational theories and surroundings as the bases.

When the concept of organization is identified as the central point of the methodology, the social structure will be specifically defined for this purpose, indicating the objectives, roles, hierarchies, groups, interactions and topology of the system. Assuming the agents are heterogeneous, the participation of external and internal agents is allowed. Furthermore, the social norms define not only the mechanisms for including external agents within the society, but also the control mechanisms for the behaviour of the agents according to the restrictions imposed by the system. For these reasons, these methodologies are the most suitable for modelling both open and closed multiagent systems. Nevertheless, they are still in an incipient phase of development and lack some formalisms and appropriate methodological guidelines for carrying out the analysis, design and complete implementation of a multiagent system and, especially, of an open system [Argente *et al.,* 2008a].

The present study uses GORMAS methodology because it covers all the specific needs that arise with virtual organizations, having been created precisely for this purpose. In particular, the methodological guide offers an iterative procedure that makes it possible to: (i) specify the mission of the system; (ii) analyze the required tasks and processes, on the basis of services and products; (iii) determine the dimensions of the organization (departmentalization, specialization, centralization, coordination and normalization); (iv) select the suitable structure according to the specified dimensions; (v) identify the processes of information and decision; (vi) specify the open characteristics of the system (functionality to publish; control of external agents); (vii) determine the mechanisms (rules) for controlling agent behaviours; and (viii) specify the reward system to promote the behaviours that most interesting to the organization.

The following sections detail the two most innovative elements in the system: the CBP-BDI task planning role that specializes in coordinating tasks; and a dynamic distribution mechanism for tasks and roles based on the creation of rules.

7.3.1 Role Planner

The role planner is responsible for carrying out the planning as the tasks are received. This agent is in charge of receiving the task list, establishing the number of agents needed to accomplish the activities, and putting the tasks in order so that any delays have the least possible impact on the overall plans. In Figure 2, we can see the information stored in the task list. The information for each task is:

- IdTask: Indentifies the global task in the system with a unique ID.
- TypeTask: Each task requires a set of activities that can be performed sequentially or in parallel. These activities can be considered as subtasks of the global task. This field establishes the current state of the global task and, consequently, the next activity or step that must be performed. From this point forward, the word task will be employed to refer to any activity performed to resolve the final task, bearing in mind that each task is defined by IdTask and TypeTask.
- Profit: Total profit gained from executing a task for a specific case. This field is important when determining the number of agents needed to minimize losses from not performing certain tasks.
- Accumulated Profit: Total accumulated profit from previously completed subtasks that will be lost if the current task is not completed on time.
- Arrival Time: This date establishes the arrival time of the task to the system. This field is important for determining the arrival rate of specific activities for a given date.
- Deadline: Maximum amount of time allowed for resolving the case. This value is preset for each type of task.
- Duration: Total time used in finalizing the task.
- Agent: The agent responsible for carrying out a task.

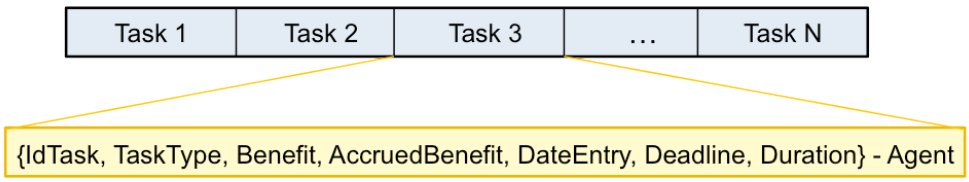

Figura 22 Task list information

Based on the list of cases that need to be developed, the role planning agent selects the number of agents needed to support demand while minimizing costs, and distributes tasks among the agents. The tasks are distributed so as to minimize any

replanning required to complete the tasks in the previously determined time. This process is summarized in the Figure 3, which displays a task list received by the planner agent, who then determines the most suitable number of processor agents and distributes the tasks.

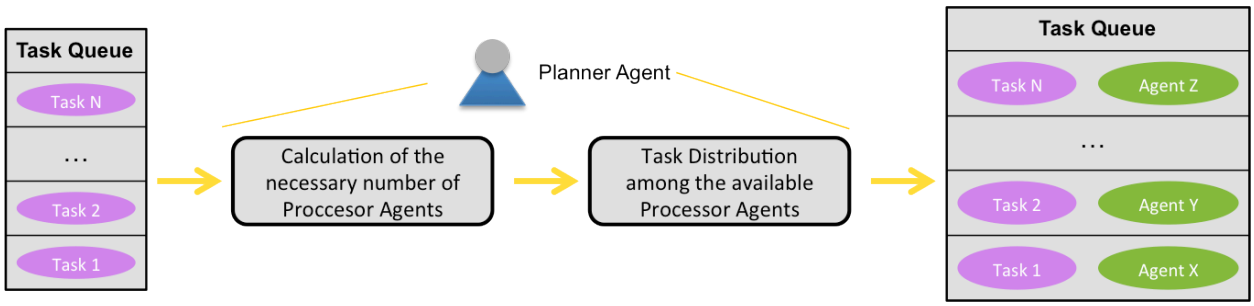

Figura 23 Planning process

If there are not enough agents to finalize the tasks, the system puts the cases in the best order to minimize losses. The system should always work under stable conditions, which is to say, those in which the utilization rate is less than 1.

The replanning process can be carried out according to different situations:

- New task: The system replans when it receives a new task. This mechanism would not be very useful if the arrival rate were high because the system would spend too much time replanning.
- Periodically: The system replans every specific period of time. If the system were to follow this procedure, it would be necessary to establish the period of time based on the arrival rate. However, as the arrival rate is not constant, this procedure could be ineffective. For example, it may be necessary to replan with the addition of only one task, while twenty new tasks are waiting to be reviewed.
- Accumulated number of unassigned tasks: Instead of replanning each time a new task arrives, the system replans only after there are a specific number of tasks waiting to be assigned to an agent.
- Inability to schedule: the system initiates the replanning process when delays by the processing agents have made it impossible to complete the most recent plans.

The first two options can be considered as simple systems with few tasks. In order to build a system that can adapt to different scenarios, the proposed system replans according to the number of tasks waiting to be assigned, and whether the previous plan can be successfully completed. The system establishes a threshold of 5%, based on the number tasks in the previous queue, before initiating a replanning process. It also creates a new distribution upon receiving an urgent task whose duration is similar to the maximum resolution term.

The planning model explained in this section is associated with the planner role. The role planner agent has the capability to learn from previous cases. The agents adopt the CBP (Case-Based Planning) model, which is a special type of CBR [Kolodner, 1993].

While CBP is derived from CBR, it is specially designed to generate plans (sequences of actions) [Corchado *et al.,* 2008a] [Kolodner, 1993]. In CBP, the proposed solution for solving a given problem is a plan. This solution is generated by taking into account the plans applied for solving similar problems in the past. The problems and their corresponding plans are stored in a memory of plans. The reasoning mechanism generates plans using past experiences and planning strategies, which is how the concept of Case-Based Planning is obtained [Glez-Bedia & Corchado, 2002]. The problem description (initial state) and solution (situation when final state is achieved) are represented as beliefs, the final state as a goal (or set of goals), and the sequences of actions as plans. The CBP cycle is implemented through goals and plans. When the goal corresponding to one of the stages is activated, different plans (algorithms) can be executed concurrently to achieve the goal or objective. Each plan can activate new sub-goals and, consequently, cause the execution of new plans. In practice, what is stored is not only a specific problem with a specific solution, but also additional information about how the plans have been derived. As with case-based reasoning, the case representation, the organization of the memory of plans, and the algorithms used in every stage of the case-based planning cycle are essential in defining an efficient planner.

To carry out the planning process the agent follows the CBR-BDI planning model. The first point in the definition of a CBR-BDI model is the definition of case (1).

$$C = \{t_i / t_i = (ids, idt, b, B, f, p, d, a), i = 1...n\} \quad (1)$$

where t_i presents the task i, ids is the identifier of the task type, idt the process, b benefit, B the accrued benefit , f the entry, p the term, d the duration, and a the ID of the agent that performed the task. The next step is to apply the different stages of reasoning.Upon receiving a new case c_{n+1} composed of the elements indicated in (1), the planning agent creates a new plan by applying the different stages of the reasoning cycle as described in the following sections:

In the **retrieve** phase the system retrieves the most similar cases to the current case c_{n+1}. The most similar cases are those that contain the greatest number of tasks most similar to those in the current queue. In order to improve the search process, information about the number of tasks for each type of activity is added to the information in the memory of plans. The search of the stored tasks is limited to a predefined period of time, which can vary according to the type of services that have been given to the organization. The number of recovered cases is defined in advance in order to subsequently carry out the reuse phase. The set of retrieved cases is defined as $C_r \subseteq C$.

During the **reuse** phase, the system uses the information retrieved from the memory of cases C_r in order to generate the plan associated with the new case c_{n+1}. The

retrieved information is adapted by means of the queuing theory and genetic algorithms. The retrieved information is used, first of all, to determine the arrival rate and the service rate for each case. The queuing theory then determines the number of services needed for a case. The information obtained from the retrieved cases is used to calculate the average service time. Statistic sampling is used to calculate the maximum error for the value obtained with a specified confidence level. If after calculating the error the specified threshold is exceeded, the system will select the size of the minimum sample to reduce the error down to the indicated value so that the threshold is no longer exceeded. Figure 24 shows the algorithm used to calculate the total number of required services.

```
Input: C_r, c_{n+1}, N, eMax, cost
Output: number, TaskList, Average

Error ← ∅;
Average ← ∅;
TaskList ← retrieveTask(c_{n+1});
foreach x ∈ TaskList do
    e ← calculateError(x, C_r, N_x);
    if e > eMax then
        SimilarTasks ← retrieveSimilarTasks(x, eMax);
        Average ← calculateAverage(c_r, SimilarTasks);
    end
    else
        Average ← calculateAverage(C_r);
    end
    Error ← Error ∪ {e};
    Average ← Average ∪ {Average};
end
[number, TaskList] ← calculateAgents(C_r, Average, cost);
```

Figura 24 Definition of the total number of services

The calculateServices function is carried out by the algorithm shown in Figure 25;

```
Input: C_r, Average, cost
Output: number, Tasks

SortedTasks ← sortProfitTask();
f ← 0;
Tasks ← ∅;
for i=1 to #SortedTasks do
    L_q ← QueuingElements(C_r, Medio[i], s=1);
    W_q ← calculateAverageWait(L_q, λ);
    L ← SystemElements(L_q, λ, Average[i]);
    duration ← calculateAverageDuration(C_r, SortedTasks[i], eMax);
    profit ← calculateAverageProfit(C_r, SortedTasks[i], eMax);
    [f_b, services] ← calculateProfit(L, Averaga[i], duration, profit);
    p ← updateUtilizationRate(λ, Average[i]);
    if ρ > 1 then endFor;
    f ← f+f_b;
    Tasks ← Tasks ∪ SortedTasks[i];
    number ← number + services;
end
```

Figura 25 Calculation of the number of services

The retrieved cases serve as a basis for building the initial chromosomes in the genetic algorithms. The algorithm used during the revise phase and corresponding to the application of the genetic algorithms can be seen in Figure 26.

```
Input: C_r, c_{n+1}, Taks, number, likelihoodCross, likelihoodMutation
Output: c_{n+1}

AverageTimeTaskAgent ← calculateTimeTaskAgent(C_r, c_{n+1});
Population ← createPopulation(C_r, c_{n+1}, Tasks, number,
AverageTimeTaskAgent);
k ← cte;
for i=1 to tope do
    Aptitude ← calculateAptitudeChromosomes(Population);
    if max(Aptitude)>aptitudeMin then return;
    NewPopulation ← ∅;
    NewPopulation ← obtainElitismChromosom(Population, k);
    for i=n to #Poblacion do
        Parent ← selectParent(Population, Aptitude);
        Sons ← crossChromosome(Parent, likelihoodCross);
        Sons ← mutarCromosomas(Sons, Taks, number,
        likelihoodMutation);
        NewPopulation ← NewPopulation ∪ Hijos;
    end
end
```

Figura 26 Task assignment

The **revise** phase is automatically initiated as the agents finish their tasks. The agent updates the duration of the tasks as they are completed and then replans if it receives a message from the processor agent regarding the impossibility of completing a plan under the existing temporal restrictions.

The **retain** phase stores the plan at the end of the business day. The new memory of cases C' is defined as follows: $C' = C \cup c_{n+1}$ Any plans that originated prior to a specific date are removed in order to limit the size of the memory.

7.3.1.1 Calculation of maximum error

In order to estimate the number of processor agents needed and to assign tasks, it is first necessary to estimate the average duration for each kind of task in general and for each of the agents in particular. The process of calculating the duration of tasks is done by first determining the level of maximum error allowed for a given confidence level. To this end, the system establishes the level of confidence and calculates the level of error obtained from a sample size. Equation (2) defines the average interval of a given sample and population.

$$\hat{\mu} = \bar{x} \pm k \frac{s}{\sqrt{n-1}} \sqrt{\frac{N-n}{N-1}} \tag{2}$$

where $\bar{X}$ represents the average, k is the value Z of a N(0,1) that belongs to a predetermined level of confidence, N is the size of the population, n is the sample size, and s is the deviation.

Equation (2) is simplified if N>>n by approximately 100 times. Under this condition, the equation can be reduced to the following expression (3).

$$\hat{\mu} = \bar{x} \pm k \frac{s}{\sqrt{n_\infty}} = \bar{x} \pm e \tag{3}$$

where n_∞ represents that N>>n by approximately 100 times.

Therefore, we can obtain the expression (4) that determines the sample size used to establish a maximum error.

$$n_\infty = \frac{k^2 S_c^2}{e^2} \tag{4}$$

S_c represents the quasivariance; the others variables have been previously indicated.

Finally, the sample size is defined by equation (5) in the first stages of the system when the condition N>>n is not satisfied.

$$n = \frac{1}{\frac{1}{n_\infty} + \frac{1}{N}} \tag{5}$$

The equations presented in this section establish the number of elements that must be retrieved in order to limit any obtained errors. This process can limit access to the memory and remove old cases that have not already been accessed.

7.3.1.2 Dynamic Planning Roles

The number of agents that should be available in the system is estimated dynamically. The intention is for the number of agents to meet the demand and ensure that the system utilization factor ρ is less than 1. The queuing theory is used to calculate the prediction. The queuing theory originated from a study of telephony in Denmark in 1909 [Erlang, 1909]. The study involved the provision of services to clients with an unknown set of demands. Queuing theory was expressed as a birth-death process, which defines the relationship between the arrival of tasks and their service so that the problem balances out in such a way that all tasks can be addressed. Figure 27 illustrates this problem.

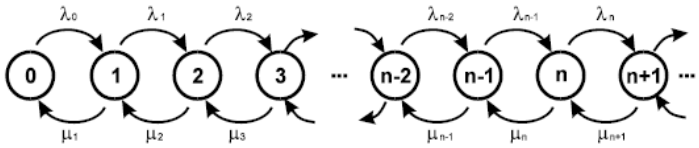

Figura 27 Birth-death process

The likelihood of the system belonging to state k can be defined according to Figure 7, where state k means that there are k tasks in the system queue.

If there are no tasks in the system queue, the relationship between the input to state 0 and the output to this state can be defined as:

$$\mu_1 P_1 = \lambda_0 P_0 \tag{6}$$

$$P_1 = \frac{\lambda_0}{\mu_1} P_0 \tag{7}$$

Using the previous expression (6) we can assume that the probability of finding zero tasks in the system queue (P_0) multiplied by the arrival rate must be equal to the probability of finding one task in the system queue that will be addressed immediately.

The following expression can be defined in the state n-1

$$\lambda_{n-2}P_{n-2} + \mu_n P_n = \lambda_{n-1}P_{n-1} + \mu_{n-1}P_{n-1} \tag{8}$$

$$\mu_n P_n = \lambda_{n-1}P_{n-1} + \mu_{n-1}P_{n-1} - \lambda_{n-2}P_{n-2} \tag{9}$$

$$P_n = (\lambda_{n-1}P_{n-1} + \mu_{n-1}P_{n-1} - \lambda_{n-2}P_{n-2}) / \mu_n \tag{10}$$

Using (10) and replacing P_{n-1} we have

$$P_n = \frac{\lambda_0 \cdots \lambda_{n-1}}{\mu_1 \cdots \mu_n} P_0 = c_n P_0 \tag{11}$$

The problem of planning multiple tasks can be simplified to a case of planning a single task for each type of task. Thus, a plan is performed independently for each task so that the average waiting time and average queue length can be calculated independently. The average waiting time and the overall average length is simplified to calculating the average values for each of the tasks. In the case of the M/G/s model where s = 1,2,3, ... is the number of agents, the arrival rate $\lambda_n = \lambda$ = etc, the service rate for when there are n processes is defined by the following equation (12).

$$\mu_n = \begin{cases} n\mu & n = 1,2, \ldots, s-1 \\ s\mu & n \geq s \end{cases} \tag{12}$$

where μ represents the average service rate for s available agents. This value depends on both the agents and the machine found.

Assuming that the system is in a stable condition (i.e., it meets the utilization factor $\rho = (\lambda / \mu s) < 1$), the probability that n tasks exists (P_n) in the system is given by equation (13).

$$P_n = c_n P_0 = \begin{cases} \frac{\lambda^n}{n!\,\mu^n} P_0 & n = 0,1, \ldots, s-1 \\ \frac{\lambda^s}{s!\,\mu^s}\left(\frac{\lambda}{s\mu}\right)^{n-s} P_0 & n \geq s \end{cases} \tag{13}$$

where:

$$P_0 = \frac{1}{\sum_{n=0}^{s-1} \frac{\lambda^n}{n!} + \frac{\lambda^s}{s!\mu} \frac{1}{1-\frac{\lambda}{s\mu}}}$$

$$c_n = \begin{cases} \frac{\lambda^n}{n!\,\mu^n} & n = 1,2,\ldots,s-1 \\ \frac{\lambda^s}{n!\,\mu^s}\left(\frac{\lambda}{s\mu}\right)^{n-s} & n \geq s \end{cases} \quad (14)$$

Having defined the probability that n tasks exist in the system, it is possible to define the number of tasks L_q in the system queue and the average waiting time W_q of tasks at the end of the system (15) [Martín, 2003].

$$L_q = \sum_{n=s}^{\infty} (n-s) P_n = P_0 \frac{\lambda^s}{s!\mu^s} \frac{\lambda}{s\mu} \frac{1}{\left(1-\frac{\lambda}{s\mu}\right)^2} \quad W_q = \frac{L_q}{\lambda} \quad (15)$$

To determine the optimal number of agents we make an estimate that minimizes the cost function, which depends on both the number of agents used and the waiting time in the queue. The function is defined in a particular way for each service depending on the actual costs of each agent in the system. The following profit function is provided (16).

$$f(L, P_0, .., P_{s-1}, \mu', \overline{p}, \overline{b}) = f_b(L, \mu', \overline{p}, \overline{b}) - k \cdot s \quad (16)$$

$$f_b(L, \mu', \overline{p}, \overline{b}) = \begin{cases} (\overline{p}/u')\overline{b} \cdot s \cdot (1-\rho) & \text{si } L \cdot u' > \overline{p} \cdot s \cdot (1-\rho) \\ L\overline{b} & \text{si } L \cdot u' \leq \overline{p} \cdot s \cdot (1-\rho) \end{cases} \quad (17)$$

where k is a constant associated with the cost of having an agent working, $\overline{b}$ the average benefit of performing the task, μ' is the average time to complete the task, obtained from the service rate, $\overline{p}$ the average time to execute a task. If we exceed the conditions of stability, f_b is counted only up to the utilization factor 1. The utilization factor ρ varies according to the new services added to the queue until it reaches the utilization factor of 1.

Following the cost function given in (7), we introduce the global cost function (18) that takes into account the implementation of the various services.

$$L_q = \sum_{n=s}^{\infty}(n-s)P_n = P_0 \frac{\lambda^s}{s!\mu^s}\frac{\lambda}{s\mu}\frac{1}{\left(1-\frac{\lambda}{s\mu}\right)^2} \quad W_q = \frac{L_q}{\lambda} \tag{18}$$

Where f_i is calculated from equation (7). Because the stability conditions may not always be given, it is necessary to calculate the terms in order of benefit depending on the type of the task so that when we reach the utilization factor of 100% we stop calculating the summation terms. Once the optimization function defined in (9) the maximum value is calculated iteratively starting with the number of agents equal to 1; the fixed value is the first local maximum that corresponds to the global maximum.

7.3.1.3 Task Assignment

Once it has been determined that the number of starting agents will minimize costs, we proceed to allocate tasks among the available agents. If the system utilization factor does not exceed the value of 1, the distribution of tasks among agents is performed so as to ensure, to the greatest extent possible, that it can perform assigned tasks in case of delays or an increase in the time needed to perform a task. The distribution is performed so as to maximize the following function (19):

$$\max \sum_{i=1}^{k} f_i \; donde \; fi = \begin{cases} \log(1-|x_i - \bar{x}|) & (x_i - \bar{x}) \geq 0 \\ -\log(1-|x_i - \bar{x}|) & (x_i - \bar{x}) < 0 \end{cases} \tag{19}$$

where $x_i = t_i - a_{i-1} - c_i$ with t_i the maximum time for completion of the task i, a_{i-1} the cumulative time to perform the tasks i-1 above, and finally c_i the time to run task i, which is customized according to the agent selected and calculated from the average value of the previously executed tasks. Minimizing the differences gets all the tasks to have a uniform distribution of the remaining, which makes the tasks easier to achieve.

If the system utilization factor is greater than 1, the aptitude function is redefined to minimize possible losses of the work already done

$$\min \sum_{i=1}^{k} - B_i \tag{20}$$

In order to know the value of each B_i it is necessary, as with the previous case, to establish the order of execution of the procedures. If the system has had time to complete task B_i then the value B_i will not be taken into account in the equation.

The chromosome encoding is performed so that each gene is composed of the elements listed by t_i and identified in (1). The sequence represents the order of the execution of tasks. Each chromosome is represented as in Figure 28. The first field for each gene contains the id of the task; the second field contains the id of the assigned agent.

Task-Agent	...	Task-Agent	Task-Agent	Task-Agent

Figura 28 Simplified definition of the chromosones

After the code has been defined, it is necessary to establish the multi-point and the mutation operators. The crossover operator is defined similarly to the multi-junction operator used in other problems. The operator is defined as follows:

A frame is selected for each chromosome and the relative order of the other parent chromosome is kept. The phases are summarized below:

- Select a partial route
- Exchange direct task segments where the IDs match
- The exchanges define a series of matches that relates each gene of the chromosome that shares the same position with the other parent

For example:

```
chromosome 1 (C1)=(1-1, 2-1, 3-3, | 4-1, 5-2, 6-1, 7-1, | 8-2, 9-
2)
chromosome 2 (C1)=(1-2, 5-3, 2-3, | 4-1, 3-2, 6-2, 7-2, | 8-3, 9-
2)


Pairings (Task Id C1-Id  from Task C2)
4↔4, 5↔3, 6↔6, 7↔7
Segments exchange
  (x-x, x-x, x-x, | 4-1, 3-2, 6-2, 7-2, | x-x, x-x)
  (x-x, x-x, x-x, | 4-1, 5-2, 6-1, 7-1, | x-x, x-x)
Eliminating redundancy
  (1-1, 2-1, x-x, | 4-1, 3-2, 6-2, 7-2, | 8-2, 9-2)
  (1-2, x-x, 2-3, | 4-1, 5-2, 6-1, 7-1, | 8-3, 9-2)

Exchange pairings
  (1-1,  2-1,  5-3,  |  4-1,  3-2,  6-2,  7-2,  |  8-2,  9-2)
  (1-2, 3-3, 2-3, | 4-1, 5-2, 6-1, 7-1, | 8-3, 9-2)
```

We must bear in mind the possibility that linked matches may be generated during the process of replacing chromosomes. We could, for example, have the links 1↔4, 4↔2 and therefore we would have 1↔2.

Various mutation operators are defined and randomly executed, but only the mutations that improve the aptitude of the chromosomes will be selected. The following mutation operators are defined:

- Exchange order of tasks. Example:

```
(1-1, 2-1, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)
(2-1, 1-1, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)
```

- Exchange of assigning contiguous tasks. Example:

```
(1-1, 2-1, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)
(1-3, 2-1, 5-1, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)
```

- Changing the allocation of a task. Example:

```
(1-1, 2-1, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)
(1-1, 2-3, 5-3, 4-1, 3-2, 6-2, 7-2, 8-2, 9-2)
```

The elitism operator is defined to keep the percentage of the most efficient solutions for every generation of the population. Moreover, the population size is also defined as a constant, which involves replacing the parent chromosomes with the children chromosomes during the generation, with the exception of the elitist chromosomes which remain unchanged.

The roulette strategy was chosen as the selection criteria because it defines a proportional selection probability for each of the chromosomes according to the aptitude function. The selection criteria was defined to correctly calculate the probabilities while taking into account that the aptitude function can have a negative value for cases in which the utilization factor is less than 1. As a last step we need to define the initiation of the chromosomes. The initiation of chromosomes is based on the received tasks; each chromosome is initiated with the tasks of the new case. For each task of the chromosome, a sequential search is made in the case, and the task is assigned to an existing agent. This association is maintained for the remaining tasks.

7.4 Case Study

The proposed system can be applied to different business environments and organizations that require task planning. However, this study was primarily motivated by the need for a platform that could create an effective distribution of work and fall in line with the new business goals generated by a recently installed e-Administration system. The result of this particular innovation within the field of Public Administration is to establish a new work mechanism that requires changes in the organization.

The case study presents a society that focuses on processing cases of Public Administration that have been received by telematics means. This society of agents was designed using the GORMAS design methodology [Argente *et al.,* 2008a], which uses various stages of analysis, structural design and dynamic design to specify the services offered by the organization, its structure, and the norms that dictate its behaviour. The system was implemented and based on the previously mentioned THOMAS architecture [Carrascosa *et al.,* 2009] [Giret *et al.,* 2010] and on the model proposed in section 3. The system is meant to simulate the behaviour of the Planner Agent, which ensures that the entire organization is provided with a global plan to successfully solve all cases.

The model proposed for this case study was adapted according to the particular specializations of the Processing Agents. These specializations are related to the three types of cases that will be taken into account: Scholarship Processor, Tax Processor, and Fine Processor. Each task in the entry queue is characterized by the case with which it is associated, and its current state or phase.

There are different services within the virtual organization, each of which can assume different roles. This way, when a Processer Agent receives a case, a service will be registered within the SF module of the THOMAS architecture. The service can be executed by the role that processes the case and will contain all the operations that the agent needs to carry out its task.

Following the methodological guidelines [Argente *et al.,* 2008a] we can observe the results in Figure 29 for one of the first tasks that must be performed. The figure instantiates the functional view (mission) of the organizational model, which includes the products and services offered by the system, the type of environment, the global objectives, interest groups and the information they consume.

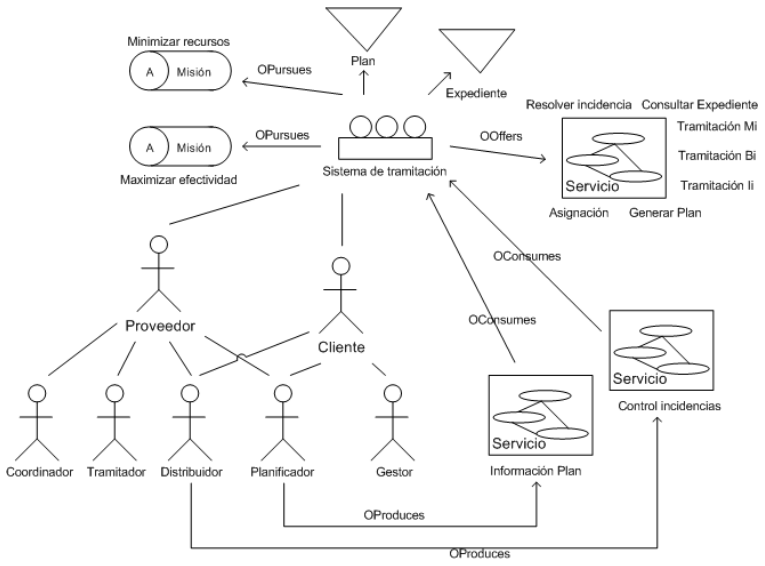

Figura 29 Diagram of the organizational model. System mission.

7.5 Results

The program used in this study simulates the behaviour of the Planner agent within a virtual organization. While the simulation is taking place, the actions are statistically analyzed for their performance and the system makes recommendations regarding the distribution of tasks.

Figure 30 displays an image of the program in operation. The task queue shows the tasks waiting to be processed. Once the task was assigned, the Tax Processor was immediately delayed (as indicated by the red cross), requiring the Planner agent to replan (as indicated by the hourglass next to the Planner) since it has now become impossible to continue with the original plan.

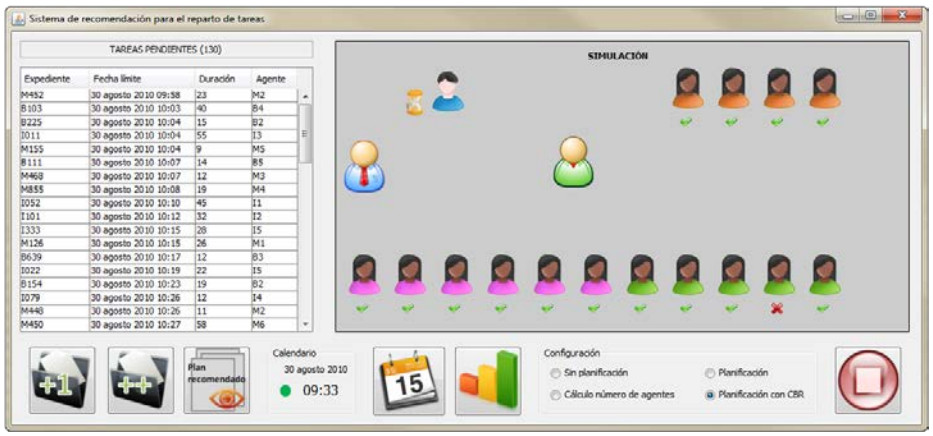

Figura 30 System interface

The system was tested with two different records for four simulation Modes: Mode 1 – No planning; Mode 2 - Calculating the number of agents needed within queuing theory; Mode 3 - Planning (including queuing theory and genetic algorithms); Mode 4 – The whole planning with CBR. In the first case (Test 1) a list of 500 records were entered within a period of 120 minutes; the second case (Test 2) had 1500 records into the system within 210 minutes. There was a previous memory of 700 cases, based on cases where values were altered randomly following a normal distribution.

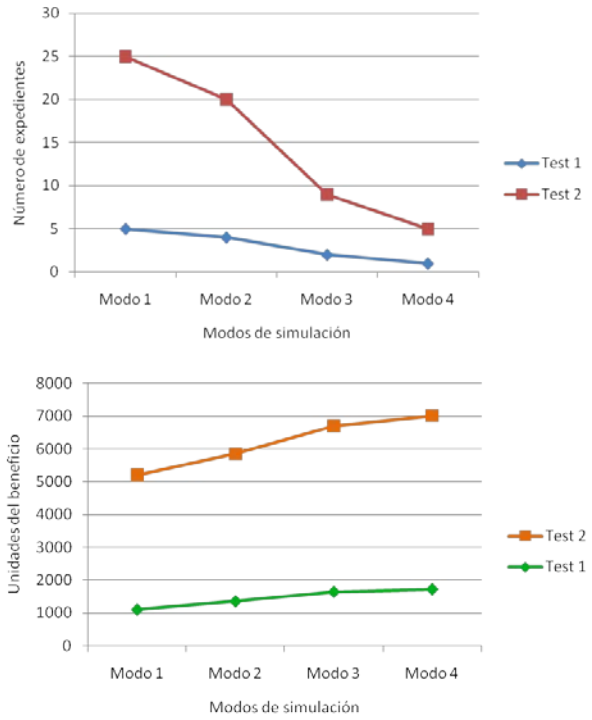

Figura 31 Number of unprocessed cases

The top chart in figure 31 shows the comparative number of cases that could not be processed in time for each mode in the two tests.

There is a qualitative leap in both cases, which occurs with the introduction of the genetic algorithms to carry out the distribution in the third mode of simulation. The introduction of the CBR also produces an improvement in the last mode because it allows a more real estimate of the value of the parameter to be made. . Test 2 for Mode 4 shows that the number of cases not processed in time is higher than Test 1, this is mainly because the recovered arrival rate exceeded the records retrieved from the memory of the CBR.

The bottom chart in Figure 31 is even more significant because it reveals the benefits obtained in different modes. In order to assign a value to the benefit field for each record, a point system that takes various aspects into account, such as the type of record, the economic gain expected, previous work, etc., is used.

As expected, the worst benefit is obtained without a planning mode. In this case, the generic estimate of the number of players is very vague and produced many cases in which there are losses from either not having enough agents or having too many. Also since the estimate does not take into account the benefit of each record, it gives priority to some records which are considered losses. In the second case, the use of queuing theory improves the benefits and is able to balance the number of agents required with the ordination benefit, thus assigning priority to improve the figures. The top chart

shows how the introduction of a genetic algorithm-based task distribution in the third mode is the strong point of planning. Finally, even better benefits are obtained taking into account the adaptation from past cases provided by the CBR.

7.6 Conclusions and Future Work

This study has presented a virtual organization model with a task planning mechanism that provides recommendations for distributing tasks within a dynamic environment. Virtual organizations are proposed as a mechanism for modeling a business or institution, including all of the characteristics that are defined in a real life situation. The advantages of virtual organizations make them the ideal mechanism for simulating the behavior of a business or institution where planning and coordination are key elements.

The use of a genetic algorithm as a mechanism for exploring the search area and optimization proved very effective within the proposed problem, which was characterized by a complexity due to the numerous combinations of options available. The use of the queuing theory made it possible to establish the number of agents needed to optimize resources. Both methods were satisfactorily incorporated within the CBR reasoning cycle, specifically during the reuse phase. This demonstrates once again the ease in extending the CBR-BDI architecture to incorporate the different operational techniques.

The proposed model was developed by applying an e-Administration model to a real life scenario, providing a detailed explanation of the steps needed to adapt the model, and identifying the roles and details characterizing each task.

Finally, one of the principal achievements of this study consists of validating the proposed planning model by implementing a simulation. The results obtained confirm an improvement in the planning process, and demonstrate that by refining the cases included within the CBR cycle, it is possible to achieve a high level of learning and adaption within a dynamic environment.

In our future research, we intend to:

- Explore other heuristic and metahuerstic methods to test different formulas for assigning tasks. Establish a value for the effectiveness of each method, and provide the user with recommendations for different planning processes according to a success percentage.
- Include a developed model within a complete system for e-Administration. This system will provide an exhaustive control of all personnel, cases and resources, thus enabling the organization to reach its full efficiency and potential.
- Completely design and implement the organization, including a model of all necessary standards and services.
-

Capítulo VIII

Referencias

VNiVERSiDAD
D SALAMANCA

[Aamodt y Plaza, 1994] Aamodt, A. y Plaza, E. (1994) Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. Artificial Intelligence Communications. Vol. 7 (1) 39-52

[Adams *et al.,* 1998] Adams, J., Balas E. y Zawack D. (1998) The Shifting Bottleneck Procedure for Job Shop Scheduling. Management Science Vol. 34 (3) 391-401

[Andres *et al.,* 2008] Andres, C., Miralles, C. y Pastor, R. (2008) Balancing and scheduling tasks in assembly lines with sequence-dependent setup times. European Journal of Operational Research. Vol. 187 (3) 1212-1223

[Antoniol, 2004] Antoniol, G., Cimitile, A., Di Lucca, G.A. y Di Pent, A. (2004) Assessing Staffing Needs for a Software Maintenance Project through Queuing Simulation. IEEE Transactions on Software Engineering. Vol. 30 (1) 43-58

[Ardissono *et al.,* 2004] Ardissono, L., Petrone, G. y Segnan, M. (2004) A conversational approach to the interaction with Web Services. Computational Intelligence Vol. 20 (4) 693-709

[Argente *et al.,* 2008a] Argente E., Julian V. y Botti V. (2008) GORMAS: Guías para el desarrollo de Sistemas Multiagente abiertos basados en organizaciones. Tesis Doctoral. Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia

[Argente *et al.,* 2008b] Argente, E., Criado, N., Julian, V. y Botti, V. (2008) Designing Norms in Virtual Organizations. CCIA Vol. 184 16-23.

[Bajo, 2007] Bajo, J. (2007) Arquitectura Multiagente Ligera: Convergiendo hacia la Inteligencia Ambiental. Tesis Doctoral. Departamento de Informática y automática. Universidad de Salamanca.

[Bajo *et al.,* 2009a] Bajo, J., Corchado, J.M., Botti, V. y Ossowski, S. (2009) Practical Applications of Agents and MAS: Methods, Techniques and Tools for Open MAS. Journal of Physical Agents, Vol. 3 (2) 1-2

[Bajo *et al.,* 2009b] Bajo J., De Paz J.F., De Paz Y., Corchado J.M. (2009) Integrating Case Planning and RPTW Neuronal Networks to Construct an Intelligent Environment for Health Care. Expert Systems with Applications. Vol. 36 (3) 5844-5858.

[Bajo *et al.,* 2007a] Bajo J., Corchado J., Tapia D., Rodríguez S., De Paz J., Sánchez J., Saavedra A., (2007) Arquitectura Multiagente para Entornos Dinámicos: Tecnología e Inteligencia Aplicadas. En: Ubiquitous Computing and Ambient Intelligence, Congreso Español de Informática (CEDI-UCAMI 2007), Thompson-Paraninfo, Zaragoza, España.

[Bajo *et al.,* 2007b] Bajo J., Tapia D., de Luis A., De Paz F., Corchado J. (2007) Hybrid Architecture for a Reasoning Planner Agent. En: Knowledge-Based Intelligent Information and Engineering Systems (pp. 461-468). Springer Berlin Heidelberg.

[Bajo *et al.,* 2007c] Bajo J., De Paz J., Tapia D., Corchado J. (2007) Deliberative Agents for Distributed Monitoring and Evaluation of the Air-Sea Interaction. International Journal of Computer Science (INFOCOMP) 16-25

[Bajo *et al.,* 2006a] Bajo, J., De Paz, Y., De Paz, J. F., Martín, Q. y Corchado, J. M. (2006) SMas: A Shopping Mall Multiagent Systems. En: Intelligent Data Engineering and Automated Learning–IDEAL 2006 (pp. 1166-1173). Springer Berlin Heidelberg.

[Bajo *et al.,* 2006b] Bajo, J., De Luis, A., Tapia, D. I., y Corchado, J. M. (2006) Wireless Multi-Agent Systems based on CBR-BDI Agents: from Theory to Practice. Proceedings of IWPAAMS'06, 85-96.

[Bauer y Huget, 2003] Bauer B., Huget M.P. (2003) FIPA Modeling: Agent Class Diagrams. Working Draft, foundation for Intelligent Physical Agents. www.auml.org.

[Binato *et al.,* 2000] Binato, S., Hery, W., Loewenstern, D. y Resende, M.G. (2000) A GRASP for Job Scheduling. Technical Report N° 00.6.1 AT&T Labs Research.

[Blum y Furst, 1997] Blum, A. L. y Furst, M. L. (1997) Fast planning through planning graph analysis. Artificial Intelligence. Vol. 90 281-300

[Blum, 1995] Blum A. y Furst, M. (1995) Fast Planning through Plan-graph Analysis. Article of memories from 14th International Joint Conference on Artificial Intelligence, 1636-1642. Morgan-Kaufmann.

[Boella *et al.,* 2005] Boella, G., Hulstijn, J. y Van Der Torre, L. (2005) Virtual organizations as normative multiagent systems. En System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on (pp. 192c-192c). IEEE.

[Bonrostro, 1998] Bonrostro, J.A. (1998) Concentración heurística: Descripción y comparación con otros metaheurísticos. Revista Rect@, Vol.1 35-41.

[Bratman, 1999] Bratman, M. E. (1999) Intention, plans and practical reason. CSLI Publications, Stanford (CA), USA.

[Bratmann, 1988] Bratman, M. E., Israel, D. y Pollack, M. (1988) Plans and resource-bounded practical reasoning. Computational Intelligence. Vol. 4 (4) 349-355.

[Braun y Rossak, 2005] Braun P. y Rossak W. (2005) Mobile Agents: Basic Concepts, Mobility Models, and the Tracy Toolkit. Morgan Kaufmann Publishers Inc.

[Brenner *et al.,* 1998] Brenner W., Zarnekow R., Wittig H. (1998) Intelligent Software Agents : Foundations and Applications. Springer 98.

[Bresciani, *et al.,* 2002] Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F. y Mylopoulos, J. (2002) Modeling Early Requirements in Tropos: A Transformation Based Approach. En: Agent-Oriented Software Engineering II (pp. 151-168). Springer Berlin Heidelberg.

[Brooks, 1991] Brooks R.A. (1991) Intelligence without reason. Proceedings of the Twelve International Joint Conference on Artificial Intelligence. 569-595

[Bueno 2009] Bueno, E. (2009) Curso básico de economía de la empresa. Un enfoque de la organización. Editorial Pirámide. Madrid.

[Bussman y Müller, 1993]. Bussman, S. y Müller, H. J. (1993) A communication achitecture for cooperating agentes. Computational Artificial Intelligence Vol. 12 (1) 37-54.

[Cabri *et al.,* 2011] Cabri, G., Puviani, M. y Zambonelli, F. (2011) A Taxonomy of Agent-based Adaptive Collaboration Patterns. International Conference on Collaborative Technologies and Systems, Philadelphia (USA), IEEE CS Press, May 2011

[Camarinha-Matos y Afsarmanesh, 2002] Camarinha-Matos, L., Afsarmanesh, H. (2002) Design of a virtual community infrastructure for elderly care. Collaborative Business Ecosystems and Virtual Enterprises, 439-450.

[Campbell y Diaby, 2002] Campbell, G.M. y Diaby, M. (2002) Development and evaluation of an assignment heuristic for allocation cross-trained workers. European Journal of Operational Research, Vol. 138 (1) 9-20.

[Campello, y Maculan, 2002] Campello, R. y Maculan, N. (2002) Algoritmos e Heurísticas. Desenvolvimento e avaliaçao de performance. Apolo Nacional Editores, Brasil.

[Carbonell, 1983] Carbonell, J. G. (1983) Learning by Analogy: Formulating and Generalizing Plans from Past Experience. (pp. 137-161). Springer Berlin Heidelberg.

[Carrascosa *et al.,* 2008] Carrascosa C., Bajo J., Julián V., Corchado J.M. y Botti V. (2008) Hybrid multi-agent architecture as a real-time problem-solving model. Expert Systems with Applications. Vol. 34 (1) 2-17.

[Carrascosa *et al.,* 2009] Carrascosa, C., Giret, A., Julian, V., Rebollo, M, Argente, E. y Botti, V. (2009) Service Oriented MAS: An open architecture. En: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2 (pp. 1291-1292). International Foundation for Autonomous Agents and Multiagent Systems.

[Carrascosa, 2004] Carrascosa, C. (2004) Meta-razonamiento en agentes con restricciones temporales críticas. Tesis Doctoral. Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia

[Castro *et al.,* 2009] Castro, J.L., Navarro, M., Sánchez, J.M. y Zurita, J.M. (2009) Loss and gain functions for CBR retrieval. Information Science Vol. 179 (11) 1738-1750

[Cavedon y Rao, 1996] Cavedon, L. y Rao, A. S. (1996) Bringing about rationality: Incorporating plans into a BDI agent architecture. En: PRICAI'96: Topics in Artificial Intelligence (pp. 601-612). Springer Berlin Heidelberg.

[Cetkovic y Parmee , 2002] Cvetkovic D. y Parmee I. (2002) Agent-based support within an interactive evolutionary design system. Artificial Intelligence for Engineering Design, Analysis and Manufacturing. Vol. 16 (5) 331-342

[Chambers y Barnes, 1999] Chambers, J. y Barnes, W. (1999) Taboo Search for the Flexible-Routing Job Shop Problem. Informe técnico TAY 2.124. Department of Computer Sciences, University of Texas- USA.

[Conte y Paolucci, 2001] Conte, R. y Paolucci, M. (2001) Intelligent social learning. Artificial Society and Social Simulation. Vol. 4 (1) 1-23.

[Coral *et al.,* 2007] Coral, M., Jaúregui, Y. y Mauricio, D. (2007) Una solución al problema de selección de cursos basado en heurísticas para el caso de la FISCT. II Congreso Internacional de Iniciación Científica en Computación.

[Corchado *et al.,* 2008a] Corchado, J.M., Bajo, J., De Paz, J.F. y Tapia, D.I. (2008) Intelligent Environment for Monitoring Alzheimer Patients, Agent Technology for Health Care. Decision Support Systems. Vol. 44 (2) 382-396.

[Corchado *et al.,* 2008b] Corchado J. M., Gonzalez-Bedia M., De Paz Y., Bajo J. y De Paz J.F. (2008) Replanning mechanism for deliberative agents in dynamic changing environments. Computational Intelligence. Vol. 24 (2) 77-107

[Corchado *et al.,* 2005] Corchado, J. M., Pavón, J., Corchado, E., y Castillo, L.F. (2005) Development of CBR-BDI agents: A tourist guide application. En: Advances in Case-Based Reasoning (pp. 547-559). Springer Berlin Heidelberg.

[Corchado y Laza, 2003] Corchado, J. M., Laza, R. (2003) Constructing Deliberative Agents with Case-based Reasoning Technology. International Journal of Intelligent Systems. Vol. 18 (12) 1227-1241.

[Cormen *et al.,* 2001] Cormen T., Leiserson, Ch., Rivest, R., y Stein, C. (2001) Introduction to Algorithms. MIT Press, Editorial McGraw Hill.

[Corominas *et al.,* 2003] Corominas, A., Lusa, A. y Pastor, R. (2003) Using MILP to plan annualised working hours. Journal of the Operational Research Society. Vol. 53 (10) 1101-1108.

[Cox *et al.,* 2006] Cox, M. T., Muñoz-Avila, H. y Bergmann, R. (2006) Case-Based Planning. Knowledge Engineering Review. Vol. 20 (3) 283-287.

[Daft, 2010] Daft, R.L. (2010) Teoría y diseño organizacional. Ediciones Paraninfo SA.

[Dastani *et al.,* 2003] Dastani, M., Dignum, V., y Dignum, F. (2003) Role Assignment in Open Agent Societies. Proceedings of the second international joint conference on Autonomous agents and multiagent systems (pp. 489-496). ACM.

[David *et al.,* 2004] David, N., Marietto, M.B., Sichman, J.S. y Coelho, H. (2004) The Structure and Logic of Interdisciplinary Research in Agent-Based Social Simulation. Journal of Artificial Societies and Social Simulation Vol. 7 (3).

[Davis, 1995] Davis, L. (1995) Job shop scheduling with genetic algorithms. First International Conference on Genetic Algorithms and their Applications, 136-140. Morgan - Kaufmann USA.

[de la Rosa *et al.,* 2013] de la Rosa, T., García-Olaya, A. y Borrajo D. (2013) A case-based approach to heuristic planning. Applied Intelligence. Vol 39 (1) 184-201

[De Meo *et al* 2006] De Meo, P., Quattrone, G., Fadil, H. and Ursino, D. (2006) A multi-agent system for efficiently managing query answering in an e-government scenario. Proceedings of the 2006 ACM symposium on Applied computing (pp. 308-312). ACM.

[De Meo *et al* 2008] De Meo, P., Quattrone, G. and Ursino, D. (2008) A Multiagent System for Assisting Citizens in Their Search of E-Government Services. IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews, Vol. 38 (5) 686-698

[De Zuani, 2005] De Zuani, R. (2005) Introducción a la administración de organizaciones. Valetta Ediciones.

[Decker *et al.,* 1997] Decker K., Pannu A., Sycara K. y Williamson M. (1997) Designing Behaviors for Information Agents. Proceedings of the first international conference on Autonomous agents (pp. 404-412). ACM Press.

[Delgado *et al.,* 2007] Delgado, E., Cortés, C.J. y Duarte, O. (2007) Aplicación de Algoritmos Genéticos para la Programación de Tareas en una celda de manufactura. Ingeniería e Investigación. Vol. 25 (2) 24-31

[Dignum y Dignum, 2006] Dignum, V., y Dignum, F. (2006) A landscape of agent systems for the real world. Technical Report UU-CS-2006-061, Institute of Information and Computing Sciences, Utrecht University.

[Dignum, 2004] V. Dignum. (2004) A model for organizational interaction: based on agents, founded in logic. Tesis Doctoral. Universiteit Utrecht.

[Doran *et al.,* 1997] Doran, J.E., Franklin, N., Jennings, N.R. y Norman, T. (1997) On cooperation in multi-agent systems. The Knowledge Engineering Review. Vol. 12 (3) 309-314

[Dowsland y Adenso, 2003] Dowsland, K.A. y Adenso, B. (2003) Diseño de heurística y fundamentos del Recocido Simulado. Revista Iberoamericana de Inteligencia Artificial. Vol. 7 (19) 93-102

[Drumond y Girardi, 2008] Drumond, L. y Girardi, R. (2008) A multi-agent legal recommender system. Artificial Intelligence and Law. Vol. 16 (2) 175-207

[Duong y Grefenstette, 2005] Duong, D.V. y Grefenstette, J. (2005) The emulation of social institutions as a method of coevolution. Proceedings of the 2005 conference on Genetic and evolutionary computation (pp. 555-556). ACM Press.

[Durfee *et al.,* 1989] Durfee, E.H., Lesser, V.R. y Corkill, D.D. (1989) Trends in Cooperative Distributed Problem Solving. IEEE Transactions on Knowledge and Data Engineering. Vol. 1 (1) 63-83

[Epstein y Axtell, 1996] Epstein, J.M. y Axtell, R.L. (1996) Growing Artificial Societies: Social science from the bottom up. The Brookings Institution, Washington DC, USA.

[Erlang, 1909] Erlang, A.K. (1909) The theory of probabilities and telephone conversations. Nyt Tidsskrift for Matematik B. Vol. 20 131-137

[Esteva *et al.,* 2001] Esteva, M., Rodríguez, J., Sierra, C., Garcia, P. y Arcos, J. (2001) On the formal specifications of electronic institutions. Agent mediated electronic commerce (pp. 126-147). Springer Berlin Heidelberg.

[Esteva *et al.,* 2002] Esteva, M., de la Cruz, D. y Sierra, C. (2002) ISLANDER: an Electronic Institutions Editor. Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3 (pp. 1045-1052). ACM Press.

[EURESCOM, 2001] EURESCOM (2001) MESSAGE: Methodology for engineering systems of software agents. Technical report P907-TI1, EURESCOM

[Fenga y Yamashiro, 2006] Fenga, D.Z. y Yamashiro, M. (2006) A pragmatic approach for optimal selection of plant-specific process plans in a virtual enterprise. Journal of Materials Processing Technology. Vol. 173 (2) 194-200

[Feo y Resende, 1995] Feo T. y Resende M. (1995) Greedy Randomized Adaptive Search Procedure. Journal of Global Optimization. Vol. 6 109-133

[Ferber *et al.,* 2003] Ferber, J., Gutknecht, O. y Michel, F. (2003) From agents to organizations: an organizational view of multi-agent systems. Agent-Oriented Software Engineering IV (pp. 214-230). Springer Berlin Heidelberg.

[Ferguson, 1992] Ferguson, I.A. (1992) TouringMachines: An Architecture for Dynamic, Rational, Mobile Agents. Technical report No. 273. University of Cambridge Computer Laboratory, Cambridge, Reino Unido.

[FIPA, 2002] FIPA, (2002) Foundation for Intelligent Physical Agents. FIPA ACL Message Structure Specification. Document No. SC00061G. http://www.fipa.org/specs/fipa00061/SC00061G.pdf

[Foster *et al.,* 2001] Foster, I., Kesselman, C. y Tuecke, S. (2001) The anatomy of the grid: Enabling scalable virtual organizations. International journal of high performance computing applications 15 (3) 200-222.

[Fox, 1981] Fox, M. (1981) An organizational view of distributed systems. IEEE Transactions on System, Man and Cybernetics. Vol. 11 (1) 70-80.

[Franklin *et al.,* 1996] Franklin, S. y Graesser, A. (1996) Is it an agent, or just a program?: A taxonomy for autonomous agents. Intelligent agents III agent theories, architectures, and languages (pp. 21-35). Springer Berlin Heidelberg.

[Gao *et al* 2006] Gao, G., Wang, Y., Wang, N., Li, H. and Wang, D. (2006) An Intelligent and Cooperative Information System for Dynamic Management of E-government Services in Inter-administration. Procedures IEEE International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2006).

[Garey y Johnsonn, 1979] Garey, M. R. y Johnson, D.S. (1979) Computers and Intractability: A guide to the theory in NP-Completeness. W.H. Freeman and Company, New York.

[Gasser y Ishida, 1991] Gasser, L. y Ishida, T. (1991) A dynamic organizational architecture for adaptive problem solving. AAAI (Vol. 91, pp. 185-190).

[Georgeff y Rao, 1998] Georgeff, M. y Rao, A. (1998) Rational software agents: from theory to practice. N. R. Jennings, y M. J. Wooldridge (Eds.), Agent Technology: Foundations, Applications, and Markets, 139-160, Springer-Verlag New York

[Giorginia *et al.,* 2005] Giorginia, P., Mylopoulos, J. and Sebastiani, R. (2005) Goal-oriented requirements analysis and reasoning in the Tropos term methodology. Engineering Applications of Artificial Intelligence. Vol. 18 (2) 159-171

[Giret *et al.,* 2010] Giret, A, Julian, V. Rebollo, M., Argente, E., Carrascosa, C. y Botti, V. (2010) An Open Architecture for Service-Oriented Virtual Organizations. Programming Multi-Agent Systems (pp. 118-132). Springer Berlin Heidelberg.

[Glez-Bedia y Corchado, 2002] González-Bedia, M. y Corchado, J.M. (2002) A planning strategy based on variational calculus for deliberative agents. Computing and Information Systems Journal. Vol. 10 (1) 2-14

[Glover, 1989] Glover, F. (1989) Tabu Search — part I. ORSA Journal on computing. Vol. 1 (3) 190-206.

[Glover, 1990] Glover, F. (1990) Tabu Search — part II. ORSA Journal on computing. Vol 2 (1) 4-32.

[Gonçalves *et al.,* 2002] Gonçalves J., Magalhäes J. y Resende M. (2002) Hibrid Genetic algorithm for the Job Shop Scheduling. AT&T Labs Research Technical Report TD-5EAL6J

[Grant *et al.,* 2010] Grant, J., Kraus, S., Perlis, D. y Wooldridge, M. (2010) Postulates for revising BDI structures. Synthese. Vol. 175 (1) 39-62.

[Grosz y Kraus, 1996] Grosz, B. y Kraus, S. (1996) Collaborative Plans for Complex Group Actions. Artificial Intelligence. Vol. 86 (2) 269-358.

[GTI-IA, 2009] GTI-IA. (2009) An Abstract Architecture for Virtual Organizations: The THOMAS project. Disponible en: http://www.fipa.org/docs/THOMASarchitecture.pdf

[Gutiérrez, 2005] Gutiérrez, M. (2005) Optimización con recocido simulado para el problema de conjunto independiente. Revista en Línea no. 3, articulo 2. http://www.azc.uam.mx/publicaciones/enlinea2/3-2rec.htm

[Guzmán, 1983] Guzmán, I. (1983) Reflexiones en torno al Orden Social. Editorial Jus, México.

[Hammond, 1989] Hammond, K. J. (1989) Case-Based Planning: Viewing Planning as a Memory Task. New York, USA: Academic Press Professional, Inc.

[Hendler, 2006] Hendler J. A. (2006) Introduction to the Special Issue: AI, Agents, and the Web. IEEE Intelligent Systems. Vol. 21 (1) 11-17

[Hernández *et al* 2006] Hernandez, L., Botti, V. y Garcia-Fornes, A. (2006) A deliberative scheduling technique for a real-time agent architecture. Engineering Applications of Artificial Intelligence. Vol. 19 (5) 521-534

[Hodge *et al.,* 2003] Hodge, B. J., Anthony, W. y Gales, L. (2003) Teoría de la Organización: un enfoque estratégico. Pearson Educación.

[Holland, 1992] Holland, J. H. (1992) Adaptation in Natural and Artificial Systems. An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. MIT Editions.

[Horling y Lesser, 2004] Horling, B y Lesser, V. (2004) A survey of multiagent organizational paradigms. The Knowledge Engineering Review. Vol. 19 (4) 281-316

[Hübner *et al.,* 2006] Hübner, J., Sichman, J. y Boissier, O. (2006) S-moise+: A middleware for developing organised multi-agent systems. Coordination, organizations, institutions, and norms in multi-agent systems (pp. 64-77). Springer Berlin Heidelberg.

[Huget *et al.,* 2003] Huget, M.P., Bauer, B., Odell, J., Levy, R., Turci, P., Cervenka, R. y Zhu, H. (2003). FIPA modeling: Interaction diagrams. Foundation for Intelligent Physical Agent, Geneva, Switzerland.

[Huhns y Stephens, 1999] Huhns, M., Stephens, L. (1999) Multiagent Systems and Societies of Agents. Weiss, G. (Ed.), Multi-agent Systems: a Modern Approach to Distributed Artificial Intelligence, MIT Press

[Iglesias, 2010] Iglesias, J.A. (2010) Modelado automático del comportamiento de agentes inteligentes. Tesis Doctoral. Departamento de Informática. Universidad Carlos III. Madrid.

[Jennings *et al.,* 1998] Jennings, N. R., Sycara, K., y Wooldridge, M. (1998) A Roadmap of Agent Research and Development. Autonomous Agents and Multi-Agent Systems Journal. Vol. 1 (1) 7-38

[Jennings y Bussmann, 2003] Jennings, N. R. y Bussmann, S. (2003) Agent-based control systems. Agent-based control systems. IEEE Control Systems. Vol. 23 (3) 61-74

[Jennings y Wooldridge, 1998] Jennings, N. y Wooldridge, M. (1998) Applications of Intelligent Agents. Applications of intelligent agents (pp. 3-28). Springer Berlin Heidelberg.

[Jennings, 1993] Jennings, N. R. (1993) Specification and implementation of a belief-desire-jointintention architecture for collaborative problem solving. International Journal of Intelligent and Cooperative Information Systems. Vol. 3 (2) 289-318

[Jiao y Mei, 2004] Jiao, W.P. y Mei, H. Automated adaptations to dynamic software architectures by using autonomous agents. Engineering Applications of Artificial Intelligence. Vol. 17 (7) 749-770

[Joh, 1997] Joh, D. Y. (1997) CBR in a Changing Environment. Case-Based Reasoning Research and Development (pp. 53-62). Springer Berlin Heidelberg.

[Jones y Rabelo, 1999] Jones, A. y Rabelo, L.C. (1999) Survey of Job Shop Scheduling Techniques. Wiley Encyclopedia of Electrical and Electronics Engineering.

[Jonker y Volgenant, 1986] Jonker, R. y Volgenant, T (1986) Improving the Hungarian assignment algorithm. Operations Research Letters. Vol.5 (4) 171-175

[Kabadi y Punnen, 2008] Kabadi, S. y Punnen, A. (2008) A strongly polynomial simplex method for the linear fractional assignment problem. Operations Research Letters. Vol. 36 (4) 402-407

[Kephart y Walsh, 2004] Kephart J.O. y Walsh W.E. (2004) An Artificial Intelligence Perspective on Autonomic Computing Policies. Fifth IEEE International Workshop on Policies for Distributed Systems and Networks, 2004. POLICY 2004. Proceedings. (pp. 3-12). IEEE.

[Kharma y Hassan 2010] Kharma, Q., Hassan, M. (2010) Adaptive Mediation Architecture for Mobile- Government Framework. The International Journal of ACM Jordan (ISSN 2078-7952). Vol. 1(4) 154-164

[Kim y Park, 2004] Kim, K.H. y Park, Y. (2004) A crane scheduling method for port container terminals. European Journal of Operational Research. Vol. 156 (3) 752-768

[Kinny y Georgeff , 1991] Kinny, D. y Georgeff, M. (1991) Commitment and effectiveness of situated agents. Proceedings of Thirteenth International Joint Conference on Artificial Intelligence (IJCAI'91), 82-88

[Klein y Calderwood, 1988] Klein, G. A. y Calderwood, R. (1988) How do people use analogues to make decisions. Proceedings of the DARPA Case-Based Reasoning Workshop. Vol. 209 223

[Klein, 1988] Klein, G. A., Whitaker, L. A. y King, J. A. (1988) Using Analogues to Predict and Plan . Proceedings of the DARPA Case-Based Reasoning Workshop, 224-232

[Klusch y Sycara, 2001] Klusch, M. y Sycara, K. (2001) Brokering and matchmaking for coordination of agent societies: a survey. Coordination of Internet Agents: Models, Technologies, and Applications 197-224. Springer Verlag.

[Kolodner, 1983a] Kolodner J. (1983a) Maintaining organization in a dynamic long-term memory. Cognitive Science. Vol. 7 (4) 243-280

[Kolodner, 1983b] Kolodner J. (1983b) Reconstructive memory, a computer model. Cognitive Science. Vol. 7 (4) 281-328

[Kolodner, 1993] Kolodner, J. (1993) Case-Based Reasoning. Morgan Kaufmann, San Francisco.

[Kolonko, 2009] Kolonko, M. (2009) Some new results on simulated annealing applied to the job shop scheduling problem. European Journal of Operational Research. Vol. 113 (1) 123-136

[Kumar y Sinha 2007] Kumar, M., Sinha, O. (2007) M-government - Mobile Technology for E-government. Proceedings of the 5th international Conference on e-Governance (ICEG-2007), Hyderabad, India, 294-301

[Kumara *et al.,* 1998] Kumara, S.T., Soyster, A.L. y Kashyap, R.L. (1998) Artificial Intelligence: Manufacturing theory and practice. Editorial NorthCross Institute of industrial Engineers, USA

[Labidi y Lejouad, 1993]. Labidi, S. y Lejouad, W. (1993) De l'Intelligence Artificielle Distribuée aux Systèmes Multi-Agents. Institut National de Recherche en Informatique et en Automatique, Rapport de Recherche nº 2004.

[Labrou *et al.,* 1999] Labrou, Y., Finin, T. y Yun Peng. (1999) Agent communication languages: the current landscape. Intelligent Systems and their Applications, IEEE Vol. 14 (2) 45-52

[Laza y Corchado, 2001] Laza R., Corchado J.M. (2001) Creation of deliberative agents using a CBR model. Computing and Information Systems Journal. Vol. 8 (2) 33-39

[Lee *et al.,* 2008] Lee, D.H., Wang H.Q. y Miao L.X. (2008) Quay crane scheduling with non-interference constraints in port container terminals. Transportation Research Part E: Logistics and Transportation Review, Vol. 44 (1) 124-135

[López de Mántaras *et al.,* 2006] López de Mántaras R., Perner P. y Cunningham P. (2006) Emergent Case-Based Reasoning Applications. Knowledge Engineering Review. Vol. 20 (3) 325-328

[López *et al* , 2004] López, S., Sánchez, P. y Conde, J. (2004) Secuenciación de tareas mediante metaheurísticos. VIII Congreso de Ingeniería de Organización, 1021-1031

[López *et al.,* 2006] López, F., Luck, M. y d'Inverno, M. (2006) A normative framework for agent-based systems. Computational and Mathematical Organization Theory. Vol. 12 (2-3) 227-250

[Maes, 1991] Maes, P. (1991) Designing Autonomous Agents. Theory and Practice from Biology to Engineering and Back. MIT Press.

[Maes, 1994] Maes, P. (1994) Agents that reduce work and information overload. Communications of the ACM. Vol. 37 (7) 30-40

[Malone y Crowston, 1994] Malone, T. y Crowston, K. (1994) The Interdisciplinary Study of Coordination. ACM Computing Surveys. Vol. 26 (1) 87-119

[Malone, 1988]. Malone, T. W (1988) What is coordination theory? . National Science Foundation Coordination theory Workshop. MIT Press.

[Marchionini *et al* 2003] Marchionini, G., Samet, H. and Brandt, L. (2003) Introduction to the Special issue on Digital Government. Communications of the ACM. Vol. 46 (1) 24-27

[Mariano *et al.,* 2003] Mariano, P., Pereira, C. Correira, A.L., Ribeiro, R., Abramov, V., Szirbik, N., Goossenaerts, J., Marwala, T. y De Wilde, P. (2001) Simulation of a trading multi-agent system. IEEE International Conference on Systems, Man, and Cybernetics. Vol. 5 3378-3384

[Marti y Moreno, 1996] Marti, R. y Moreno J. (1996) Metaheurísticas en Optimización Combinatoria. Conferencia: 25 años de Matemáticas en la Universidad de La Laguna, 443-451

[Martí, 2003] Martí, R. (2003) Procedimientos Metaheurísticos en Optimización Combinatoria. Matemàtiques. Vol.1 (1) 3-62

[Martín, 2003] Martín, Q. (2003) Investigación Operativa. Prentice-Hall

[Mas, 2005] Mas, A. (2005) Agentes software y sistemas multiagente : conceptos, arquitecturas y aplicaciones. Pearson Educacion

[Massie, 1973] Massie, J. (1973) Bases esenciales de la Administración. Ediciones Diana, México.

[McAllester y Rosenblatt, 1991] McAllester, D. y Rosenblatt, D. (1991) Systematic nonlinear planning. Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91). Vol. 2 634-639. AAAI Press/MIT Press

[McCarthy, 1978] McCarthy J. (1978) Ascribing mental qualities to machines. Technical report, Stanford University AI Lab., Stanford, CA 94305

[Medjahed y Bouguettaya 2005] Medjahed, B. y Bouguettaya, A. (2005) Customized Delivery of E-Government Web Services. IEEE Web Services. Nov-Dec 2005, 77-84

[Melián *et al.,*2003] Melián, B., Moreno, J. y Moreno, J. (2003) Metaheuristicas: Una visión global. Revista Iberoamericana de Inteligencia Artificial. Vol. 7 (19) 7-28

[Menasce, 2002] Menasce, D.A. (2002) Trade-offs in designing Web clusters. Internet Computing, IEEE. Vol. 6 (5) 76-80

[Monostori *et al.,* 2006] Monostori L., Váncza, J. y Kumara, S.R.T. (2006) Agent-Based Systems for Manufacturing. CIRP Annals Manufacturing Technology. Vol. 55 (2) 697-720

[Müller, 1996] Müller, J. P. (1996) The Design of Intelligent Agents: A Layered Approach. Vol. 1177. Springer-Verlag, New York

[Muller, 1996] Muller, H.J. (1996) Negotiation principles. En: Foundations of Distributed Artificial Intelligence. (G.M.P. O'Hare, and N.R. Jennings, eds.), John Wiley & Sons, San Francisco, CA, USA

[Nwana, 1995] Nwana H. S. (1995) Software Agents: An Overview. Knowledge Engineering Review. Vol. 11 (2) 205-244

[O'Brien y Nicol, 1998] O'Brien, P.D. y Nicol, R.C. (1998) FIPA - Towards a Standard for Software Agents. BT Technology Journal. Vol. 16 (13) 51-59

[Odell *et al.,* 2004] Odell J., Levy R. y Nodine M. (2004) FIPA Modeling: Agent Class Superstructure Metamodel. FIPA Modelling TC

[Oliva *et al.,* 2008] Oliva, E., Natali, A., Ricci, A. y Viroli, M. (2008) An Adaptation Logic Framework for Java-based Component Systems. Journal of Universal Computer Science. Vol. 14 (13) 2158-2181

[Ossowski, 1999] Ossowski, S. (1999) Coordination in Artificial Agent Societies: social structures and its implications for autonomous problem-solving agents. Springer-Verlag.

[Ossowski y García-Serrano, 1998] Ossowski, S. y García-Serrano, A. (1998) Social Coordination among Autonomous Problem-Solving Agents. Proceedings of the Workshops on Commonsense Reasoning, intelligent Agents, and Distributed Artificial intelligence: Agents and Multi-Agent Systems Formalisms, Methodologies, and Applications. Lecture Notes in Computer Science. 1441, 134-148. Springer-Verlag, London

[Pacheco y Casado, 2003] Pacheco, J.A. y Casado, S. (2003) Estudio Comparativo de Diferentes Metaheurísticas para la Resolución del Labor Scheduling Problem. Estudios de Economia Aplicada. Vol. 21 (3) 537-557

[Parunak y Odell, 2002] Parunak, H. V. D. y Odell, J. (2002) Representing Social Structures in UML. Agent-Oriented Software Engineering II, 1-16, Springer Berlín Heidelberg

[Pattison *et al.,* 1987] Pattison, H.E., Corkill, D.D. y Lesser, V.R. (1987) Instantiating Descriptions of Organizational Structures. En: Distributed Artificial Intelligence (M.N. Huhns ed.) 59-96. M Kauffman Publishers Inc., San Mateo (CA), USA

[Pavón *et al.,* 2007] Pavón J., Gómez J., Fernández A. y Valencia J. (2007) Development of intelligent multi-sensor surveillance systems with agents. Journal of Robotics and Autonomous Systems. Vol. 55 (12) 892-903

[Peiró, 1992] Peiró, J. M. (1992) Organizaciones. Nuevas perspectivas psicosociológicas. PPU, Barcelona.

[Peiró, 1991] Peiró, J. (1991) Psicología de la Organización. Universidad Nacional de Educación a Distancia. Ed. Toran, S.A., Madrid.

[Penberthy y Weld, 1992] Penberthy, J. S. y Weld, D. (1992) UCPOP: A Sound, Complete, Partial-Order Planner for ADL. Third International Conference on Knowledge Representation and Reasoning (KR-92), 92, 103-114.

[Pérez-Carballo, 2013] Pérez-Carballo, J.F. (2013) Control de la Gestión Empresarial. Editorial ESIC, Madrid.

[Pinedo, 2012] Pinedo M. (2012) Scheduling, Theory, Algorithms and Systems. Springer, New York.

[Pinzón *et al.,* 2011] Pinzón C., Bajo J, De Paz J.F. y Corchado J.M. (2011) S-MAS: An Adaptive Hierarchical Distributed Multiagent Architecture for Blocking Malicious SOAP Messages within Web Services Environments. Expert Systems with Applications. Vol. 38 (5) 5486-5499

[Pitsoulis y Resende, 2002] Pitsoulis, L.S. y Resende, M.G.C. (2002) Greedy randomized adaptive search procedures. En: Handbook of Applied Optimization (Pardalos, P.M. y Resende, M.G.C., eds.) Oxford University Press, New York, 168-182

[Pokahr *et al.,* 2003] Pokahr, A., Braubach, L. y Lamersdorf W. (2003) Jadex Implementing a BDI-Infrastructure for JADE Agents EXP. Search of Innovation (Special Issue on JADE). Vol. 3 (3) 76-85.

[Popa *et al.,* 2008] Popa, H.E., Negru, V., Pop, D. y Muscalagiu, I. (2008) DL-AgentRecom - A Multi-Agent Based Recommendation System for Scientific Documents. 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, 2008. SYNASC'08. IEEE. 320-324.

[Porto *et al.,* 2000] Porto S, Kitajima J.P. y Ribeiro C. (2000) Performance evaluation of a parallel tabu search task scheduling algorithm. Parallel Computing. Vol. 26 (1) 73-90

[Pradenas *et al.,* 2008] Pradenas, L., Hidalgo, S. y Jensen M. (2008) Asignación de supervisores forestales mediante un algoritmo Tabú Search. Ingeniare. Revista chilena de ingeniería. Vol. 16 (3) 404-414

[Puviani *et al.,* 2013] Puviani, M., Cabri, G. y Zambonelli, F. (2013) A Taxonomy of Architectural Patterns for Self-adaptive Systems. Proceedings of the International C* Conference on Computer Science and Software Engineering. ACM Press. 77-85

[Ramírez, 2006] Ramírez, C.O. (2006) Un algoritmo Grasp con doble relajación para resolver el problema del Flow Shop Scheduling. Tesis Doctoral. Universidad Pontificia de Perú

[Rao y Georgeff, 1991] Rao A.S. y Georgeff M.P. (1991) Modeling Rational Agents within a BDI-Architecture. Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR91) 473-484

[Rao y Georgeff, 1995] Rao A.S. y Georgeff M.P. (1995) BDI Agents from Theory to Practice. Proceedings of the First International Conference on Multi-Agents Systems (ICMAS) Vol. 95, 312-319

[Razavi *et al.,* 2005] Razavi, R., Perrot, J.F. y Guelfi, N. (2005) Adaptive modeling: an approach and a method for implementing adaptive agents. En: Massively Multi-Agent Systems I (Ishida, T., Gasser, L. y Nakashima, H. eds.) Springer, New York, 136-148

[Resende y Ribeiro, 2003] Resende, M.G.C. y Ribeiro, C.C. (2003) Greedy randomized adaptive search procedures. En: Handbook of Metaheuristics (Glover, F. and Klochenberber, G.A. eds.). Kluwer academic Publishers, New York, 219-249

[Resnick y Varian, 1997] Resnick, P. y Varian, H. R. (1997) Recommender systems. Communications of the ACM. ACM. Vol. 40 (3) 56-58

[Restrepo *et al.,* 2004] Restrepo, C. Jorge, H. y Sánchez, C. (2004) Solución al problema de entrega de pedidos utilizando recocido simulado. Scientia et Technica. Vol. 10 (24) 225-230

[Riesbeck y Schank, 1989] Riesbeck, C. K. y Schank, R. C. (1989) Inside Case-Based Reasoning. Lawrence Erlbaum Associates Inc Publishers, Hillsdale, NJ, USA.

[Rigole *et al.,* 2002] Rigole, P., Holvoet, T. y Berbers, Y. (2002) Using Jini to Integrate Home Automation in a Distributed Software-System. En: Distributed Communities on the Web: 4th International Workshop, DCW 2002 Sydney, Australia, April 3-5, 2002, Revised Papers (Lecture Notes in Computer Science) (Plaice, J., Kropf, P.G., Schulthess, P. y Slonim J. eds). Springer Berlin Heidelberg 291-303

[Rios y Bard, 2000] Rios, R. y Bard, J. (2000) Heurística para secuenciamiento de tareas en Líneas de flujo. Ciencia UNAL. Facultad de Ingeniería Mecánica y Eléctrica. Universidad Autónoma de nuevo León. Vol. 3 (4) 420-427

[Robbins, 2009] Robbins, S.P. (2009) Comportamiento Organizacional. Pearson Educación

[Rodríguez *et al.,* 2012] Rodríguez, S., Tapia, D.I., de Paz, J.F., Abraham, A., Bajo, J. y Corchado, J.M. (2012). Self-organizing Multi-agent System for Management and Planning of Surveillance Routes. Computing and Informatics. Vol. 31 1081-1100

[Rodríguez, 2003] Rodríguez, P. (2003) Discusión y Análisis de la metaheurística SN. Departamento de Investigación Operativa, InCo, FI, UdelaR. Reporte Tecnico 03-02

[Rodríguez, 2010] Rodríguez, S. (2010) Modelo Adaptativo para Organizaciones Virtuales de Agentes. Tesis Doctoral. Departamento de Informática y Automática. Universidad de Salamanca

[Ross, 1989] Ross, B.H. (1989) Some psychological results on case-based reasoning. Proceedings of the DARPA Case-Based Reasoning Workshop, 144

[Rueda *et al.,* 2002] Rueda, S., García, A.J. y Simari, G.R. (2002) Argument-based negotiation among bdi agents. Journal of Computer Science and Technology. Vol. 2 (7) 1-8

[Russell y Norvig, 2014] Russell, S. y Norvig, P. (2014) Artificial Intelligence: A modern approach. Pearson Education Ltd., Harlow, Essex, UK

[Sánchez y López, 2005] Sánchez, P. y López, S. (2005) Programación de tareas, un reto diario en la empresa. Anales de Mecánica y Electricidad, Vol. LXXXII, nº. III, 24-30

[Sansores y Pavón, 2005] Sansores, C. y Pavón, J. (2005) Simulación social basada en agentes. Inteligencia artificial: Revista Iberoamericana de Inteligencia Artificial. Vol. 9 (25) 71-78

[Schein, 2010] Schein. E.H. (2010) Organizational culture and leadership. John Wiley & Sons, San Francisco, CA, USA.

[Schelling, 1978] Schelling, T.C. (1978) Micromotives and macrobehavior, WW Norton Inc. Publishers, New York

[Seda, 2007] Seda, M. (2007) Mathematical Models of Flow Shop and Job Scheduling Problems. International Journal of AM&CS, Vol. 4 (4) 241-246

[Seguido, 2009] Seguido, M. (2009) Sistemas de recomendación para webs de información sobre la salud. Tesis de Máster. Departamento de Lenguajes y Sistemas Informáticos. Universidad Politécnica de Cataluña

[Shafiq *et al.,* 2006] Shafiq, M.O., Ding, Y. y Fensel, D. (2006) Bridging Multi Agent Systems and Web Services: towards interoperability between Software Agents and Semantic Web Services. Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06). IEEE Computer Society, Washington, DC. 85-96

[Shanthikumar, *et al.,* 2007] Shanthikumar, J.G., Shengwei, D. y Zhang, M.T. (2007) Queueing Theory for Semiconductor Manufacturing Systems: A Survey and Open Problems. IEEE Transactions on Automation Science and Engineering. Vol 4 (4) 513-522.

[Shi *et al.,* 2005] Shi, C., Lu, J. y Zhang, G. (2005) An extended Kuhn-Tucker approach for linear bilevel programming. Applied Mathematics and Computation. Vol. 162 (1) 51-63

[Shi, 2007] Shi, G. (2007) A Genetic Algorithm Applied to the Classic Job-Shop Scheduling Problem. International Journal of Systems Science. Vol. 28 (1) 25-32

[Steinhofel *et al.,* 1999] Steinhofel, K., Albrecht, A. y Wong, C.K. (1999) Two simulated annealing-based heuristics for the job shop scheduling problem. European Journal of Operational Research. Vol. 118 (3) 524-548

[Tadj, 1995] Tadj, L. (1995) Waiting in line (queuing theory). Potentials, IEEE. Vol 14 (5) 11-13

[Taillard, 1994] Taillard, E. (1994) Parallel Taboo Search Technique for the Job shop Scheduling Problem. ORSA Journal on Computing Vol. 6 (2) 108-117

[Tambe, 1997] Tambe, M. (1997) Towards Flexible Teamwork. Journal of Artificial Intelligence Research. Vol. 7 83-124

[Tapia *et al.,* 2012] Tapia, D.I., Fraile, J.A., Rodríguez, S., Alonso, R.S. y Corchado, J.M. (2012) Integrating Hardware Agents into an Enhanced Multi-Agent Architecture for Ambient Intelligence Systems. Information Sciences Vol. 222 47-65

[Thomas-Tin, 2013] Disponible en: http://thomas-tin.usal.es/. Última vez visitado: 10 de diciembre de 2013

[Tupia y Mauricio, 2004] Tupia, M. y Mauricio, D. (2004) Un algoritmo voraz para resolver el problema de la programación de tareas dependientes en máquinas diferentes. RISI Vol. 1 (1) 9-18

[Tupia, 2004] Tupia M. (2004) Un algoritmo GRASP para resolver el problema de la programación de tareas dependientes en máquinas diferentes. Proceedings of Conferencia Latino Americana de Informática CLEI (2004, Perú), 129-139

[van der Hoek y Wooldridge 2012] van der Hoek, W. y Wooldridge, M. (2012) Logics for Multiagent Systems. AI Magazine. Vol. 33 (3) 92

[van Laarhoven *et al.,* 2002] van Laarhoven, P., Aarts, E. y Lenstra, J.K. (2002) Job Shop Scheduling by Simulated Annealing. Operations Research. Vol. 40 (1) 113-125

[Verter y Dincer, 1992] Verter, V. y Dincer, M.C. (1992) An integrated evaluation of facility location, capacity acquisition and technology selection for designing global manufacturing strategies. European Journal of Operational Research. Vol. 60 (1) 1-18

[Wagner y Hollenbeck, 2004] Wagner, J., y Hollenbeck, J. (2004) Comportamiento Organizativo. Paraninfo SA, Madrid.

[Walton, 2006] Walton, C. (2006) Agency and the Semantic Web. Oxford University Press, Inc., New York, USA.

[Wan *et al.,* 2009] Wan, S., Yang, F. e Izquierdo, E. (2009) Lagrange multiplier selection in wavelet-based scalable video coding for quality scalability. Signal Processing: Image Communication. Vol. 24 (9) 730-739

[Weiser, 1993] Weiser, M. (1993) Ubiquitous Computing. IEEE Computer Hot Topics

[Weyns y Georgeff, 2010] Weyns, D. y Georgeff, M. (2010) Self-Adaptation Using Multiagent Systems. IEEE Software. Vol. 27 (1) 86-91

[Wooldridge y Jennings, 1995] Wooldridge, M. y Jennings, N.R. (1995) Intelligent Agents: Theory and Practice. The Knowledge Engineering Review. Vol. 10 (2) 115-152

[Wooldridge, 2009] Wooldridge M. (2009) An Introduction to MultiAgent Systems. John Wiley & Sons, San Francisco, CA, USA

[Wu y Jennings 2013] Wu, F. y Jennings, N.R. (2013) Regret-Based Multi-Agent Coordination with Uncertain Task Rewards. arXiv preprint arXiv:1309

[Wu y Sun, 2002] Wu, N.Q. y Sun, J. (2002) Grouping the activities in virtual enterprise paradigm. Production Planning & Control. Vol. 13 (4) 407-415

[Xiang, 2007] Xiang, L.I.U. (2007) Multi-Agent-Based Service-Oriented Architecture for Inter-Enterprise Cooperation System . Proceedings of the Second international Conference on Digital Telecommunications (ICDT'07). IEEE Computer Society, Washington, DC, 22.

[Xin *et al.,* 2006] Xin, M., Wu, C. and Li, W. (2006) An approach to self-adaptive active control mechanism to support e-Government based on Multi-Agent System Procedures of the 5th IEEE International Conference on Cognitive Informatics (ICCI'06), 711-715

[Yu y Buyya, 2006] Yu, J. y Buyya, R. (2006) Scheduling scientific workflow applications with deadline and budget constraints using genetic algorithms. Scientific Programming. Vol. 14 (3) 217-230

[Zambonelli *et al.,* 2003] Zambonelli, F., Jennings, N. R. y M. Wooldridge (2003) Developing multiagent systems: The Gaia methodology. ACM Transactions on Software Engineering and Methodology. Vol. 12 (3) 317-370

[Zambonelli, 2002] Zambonelli, F. (2002) Abstractions and Infrastructures for the Design and Development of Mobile Agent Organizations. En: Agent-Oriented Software Engineering II (Wooldridge, M.J., Weiss, G. and Ciancarini, P., eds.) Springer-Verlag, Berlin-Heidelberg, 245-262

[Zambonelli *et al.,* 2000] Zambonelli, F., Jennings, N.R. y Wooldridge, M. (2000) Organisational abstractions for the analysis and design of multiagent systems. En: Agent-Oriented Software Engineering (Ciancarini, P., and Wooldridge, M.J. eds.), Springer-Verlag, Berlin-Heidelberg, 127-141

[Zamudio *et al.,* 2006] Zamudio, J., Rivera, L. y Mauricio, D.S. (2006) Algoritmo Grasp para la distribución eficiente de objetos en una interfaz gráfica de usuarios. CLEI 2006 Santiago de Chile