Universidad de las Ciencias Informáticas

Facultad 15

"Propuesta de arquitectura para un Sistema Tutor Inteligente de apoyo al proceso de aprendizaje"

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autores:

- **Yuriesky Duarte Correa**
- **Ricardo Segovia Vega**

Tutor: Lic. Rolan Rober Bullain Diéguez

Junio 2010. La Habana, Cuba.
"Año 52 de la Revolución"

"El Mundo está en las manos de aquellos que tienen el coraje de soñar y correr el riesgo de vivir sus sueños"

Paulo Coelho

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Yuriesky Duarte Correa
Firma del Autor

Ricardo Segovia Vega
Firma del Autor

Lic. Rolan Rober Bullain Diéguez
Firma del Tutor

A nuestro tutor Rolan, por su apoyo en la investigación, e inculcarnos el deseo de hacer cosas con calidad. Por su comprensión y dedicación durante el desarrollo del trabajo y por los consejos útiles que nos proporcionó.

A la Doctora en Ciencias Técnicas Natalia Martínez Sánchez, y los doctores Maikel León Espinosa y Zenaida García Valdivia y el MSc. Ariel Castellanos por su dedicación, cooperación y apoyo en la investigación.

De Ricardo:

A todas aquellas personas que influyeron en mi proceso educativo.

A mi familia que me ha apoyado en todo momento para llegar hasta aquí, en especial a mi madre por darme la fuerza para superar todos los obstáculos.

A mis amigos Yuriesky, Ángel, Minardo, Lianni y Michel por estar a mi lado cada vez que necesité de ellos.

A todas mis amistades de la facultad, aprendí mucho de todos.

A todos mis profesores que pusieron su empeño en darme una buena educación.

De Yuriesky:

A mis padres Milagros Correa Fuentes y Alberto Duarte Gaínza por su apoyo incondicional y siempre haber creído en mí, por ser lo más grande que tengo en la vida; por el tiempo dedicado a mí educación, por estar cada día más orgullosos de su hijo y por haber confiado eternamente en mí.

A mi hermano del alma Yoelito, por ser una de las cosas más lindas que tengo en la vida. Por su constante e inconfundible amor y por ese carisma puro que solo sale de su corazón.

A mi abuelita Nena, quien supo ser como una madre para mí y de quien aprendí tanto en esta vida.

A mí chuchi linda Lianni: por dedicarme cada uno de sus días, por su inconfundible amor y cariño, por siempre estar a mí lado cuando más la necesité, por disfrutar de mis alegrías y sufrir con mis tristezas, por tantos momentos felices. Por la pureza de sus sentimientos y su eterna dedicación.

A mi segunda gran familia, mis tíos Ana, Rogelio y mi primo Rogelito. Por su apoyo constante y los momentos felices que me regalaron, por su dedicación en todo momento, por propiciar el calor familiar en momentos difíciles, por amarme como si fuera su propio hijo.

A toda mi familia, mi abuela, mis tíos y primos, quienes han dado muestra en todo momento de preocupación y entrega a mi porvenir.

A la familia de mi novia Lianni por su confianza y su cariño.

A todas mis amistades que durante la carrera muchos jugaron el papel de amigo y familia, Ricardo, Michel, Alexis, Ángel, Minardo, Rey, Carmen, Yuri, Armando, Danirys, Dunia, Anisley, Bianka, Maidolis, por compartir juntos momentos de preocupación y alegría.

A mi compañero de tesis, Ricardo por su constante dedicación al trabajo, su apoyo y amistad durante la carrera.

A Raúl de la Cruz quien supo guiarme y aconsejarme. Por su apoyo y confianza.

De Yuriesky:

A mi hermano Yoelito: con la convicción profunda de que puede soñar y vivir sus sueños, y de que no existe cosa en el mundo que se proponga y no pueda hacer. A ti, que vas creciendo ahora como una semilla que germina de mi corazón y que puedes estar seguro, que ese corazón te ha reservado la materia necesaria para que crezcas toda la vida.

Para ti, todo mi amor.

De Ricardo:

... A mi madre por su ejemplo y dedicación. Y a todas las personas que creyeron en mí.

La Inteligencia Artificial y los Sistemas Tutores Inteligentes (STI) constituyen actualmente un área de creciente interés dentro de los procesos de Enseñanza-Aprendizaje. La creación de numerosos sistemas, como medios de apoyo a este campo aumenta considerablemente cada día en busca de facilitar el proceso de instrucción e innovar nuevas técnicas para su mejoramiento. El Paradigma Orientado a Agentes, como parte de esta ciencia de la computación, ha hecho numerosos aportes en el desarrollo de estos sistemas.

Los Agentes Inteligentes constituyen un avance significativo en el mundo de la informática, los cuales están enfocados a la realización de complejas tareas que requieren de cierto grado de inteligencia, además de ser considerados sistemas autónomos que se centran en su propio aprendizaje y resultan de gran interés por contribuir a la resolución de problemas complejos de difícil solución mediante técnicas clásicas. Su integración en Sistemas Multi-Agentes, ha sido un elemento esencial en el desarrollo de estas aplicaciones, caracterizándose por un alto nivel de autonomía y factibilidad en complejos procesos, lo que favorece la creación de estos sistemas.

Los Sistemas Tutores Inteligentes han impactado desde sus inicios en las aplicaciones de la Informática Educativa y con el afán de mejorar como sistema surgieron nuevas arquitecturas y técnicas de desarrollo. Tras el estudio de varios elementos relacionados con el tema, la investigación abarca elementos conceptuales de los STI y se propone una arquitectura para desarrollar sistemas de esta índole, garantizando la flexibilidad del mismo. Además se desarrolla la arquitectura propuesta bajo la metodología PASSI (Proceso para la Especificación e Implementación de Sociedades de Agentes), mediante la cual se generan los artefactos correspondientes a la misma y se posibilita una mayor comprensión y entendimiento de dicha arquitectura, así como su puesta en práctica por parte de los desarrolladores del sistema para su futura implementación.

PALABRAS CLAVES:

Sistemas Tutores Inteligentes, Sistemas Multi-Agentes, Inteligencia Artificial.

Introducción

El desarrollo de las nuevas tecnologías conlleva una Revolución no solo en áreas como la industria, el comercio, el sector social y el económico; también la educación se ve beneficiada de estos avances, permitiendo generar nuevas técnicas para transmitir el conocimiento.

La idea de automatizar el proceso de instrucción surge incluso antes de 1947, año en que se crea la primera computadora electrónica. Ya en 1912, E. L. Thorndike formulaba la idea de un material auto-guiado o de una enseñanza programada de forma automática. Pero no es hasta la década de 1950 que B. F. Skinner en su artículo "The Science of Learning and the Art of Teaching" propone que las técnicas tradicionales de enseñanza podían mejorarse con el uso de lo que entonces se denominaba "teaching machines".

Esta idea se desarrolló hasta impulsar lo que actualmente se conoce como *Instrucción Asistida por Computadoras* (IAC). Este tipo de enseñanza se basa en aplicaciones que contienen programación de cursos, realización de exámenes comprobatorios, simulación de modelos y procesos educativos, así como desarrollo de tutoriales. Las principales críticas que atacaron a la IAC según Fernández (Fernández, y colegas, 2006) fueron bajo la base de que el estudiante carecía de iniciativa propia o era muy limitada, no se podía utilizar el lenguaje natural en las respuestas, los sistemas de IAC eran demasiado rígidos ya que su comportamiento estaba preprogramado, y no poseían conocimiento real.

La Inteligencia Artificial (IA) es una rama de la computación que aparece a finales de la década del 50 y sus técnicas progresaron rápidamente con aportes en muchos sectores sociales. La Instrucción Asistida por Computadoras recibe un gran impacto con estas técnicas, se facilitan los procesos de instrucción y se garantiza mayor eficiencia en los mismos. Es en 1970 que Carbonell da la iniciativa de aplicar las técnicas de la Inteligencia Artificial a los sistemas de IAC en su artículo "AI in CAI: An Artificial Intelligence Approach to Computer Aided Instruction" (Carbonell, 1970); a él se le atribuye la creación del primer Tutor Inteligente: el SCHOLAR (un tutor inteligente para la enseñanza de la geografía de América del Sur), aunque el término de "Sistema Tutor Inteligente" (STI) es definido posteriormente por Brown.

Los STI usan la Inteligencia Artificial para incorporar conocimiento y lograr adquirir técnicas educativas para impartir una materia. Logran así controlar el nivel de aprendizaje de cada uno de los estudiantes y de esta forma se garantiza una instrucción personalizada. Estos sistemas se caracterizan por poseer un elevado nivel de especialización en el contenido que imparten, convirtiéndose en sistemas expertos.

La Universidad de las Ciencias Informáticas (UCI) tiene una misión social de llevar a cabo la informatización y desarrollo tecnológico a todos los sectores económico-sociales en Cuba. La Educación es una de las esferas priorizadas por la sociedad cubana y la UCI se ha sumado desde sus inicios con el objetivo de contribuir al desarrollo de este campo con el uso de las nuevas tecnologías.

Las multimedia y otros sistemas como los Entornos Virtuales de Aprendizaje constituyen medios de apoyo a los procesos educativos, pero no son capaces de lograr la independencia que se puede alcanzar con un tutor, en su gran mayoría no ofrecen herramientas para una atención personalizada al estudiante y carecen de "inteligencia" propia para poder tomar decisiones en el proceso de instrucción. Limitarse solo al uso de estas aplicaciones constituye un freno al creciente desarrollo en las técnicas instructivas que requiere una sociedad moderna con ansias de conocimientos.

Por otra parte, en el país se lleva a cabo un programa educativo masivo que requiere el gasto de muchos recursos y esfuerzo. La eficaz aplicación de los STI en el apoyo a la instrucción a distancia y a los centros educacionales físicos ha sido probada a nivel internacional. Sin embargo, este tipo de aplicaciones no se ha comenzado a explotar, de forma masiva, en el sistema de enseñanza de Cuba según la bibliografía consultada.

Los STI serían de gran apoyo al proceso educacional cubano. Pueden ser usados como complemento de la instrucción brindada por el maestro, ya sea simplemente para reforzar conocimiento, dar asistencia a los estudiantes más talentosos o para dar asistencia a los estudiantes más lentos en el aprendizaje; e incluso pueden llegar a sustituir en muchos casos la presencia de un profesor. Un sistema de esta índole, también, podría darle al maestro información sobre el desempeño del estudiante para que pueda aplicar las medidas que considere apropiadas.

También debe analizarse que a nivel mundial la construcción de Sistemas Tutores Inteligentes es un proceso muy costoso; requiere el uso de muchos recursos como tiempo, personal altamente calificado y tecnologías informáticas de alto nivel. Estos sistemas generalmente son construidos de forma rígida, especializados en un único contenido. La creación de un STI para cada una de las materias implica un elevado costo y esfuerzo.

Una vez analizados todos estos elementos se define como **problema científico** que los Sistemas Tutores Inteligentes que son usados como apoyo en los procesos de instrucción están definidos para impartir una determinada materia y carecen de la posibilidad de adaptarse a diferentes dominios.

A partir del problema definido se enmarca como **objeto de estudio** las técnicas de la Inteligencia Artificial y como **campo de acción** los Sistemas Tutores Inteligentes.

El **objetivo** de la investigación está dirigido a proponer una arquitectura de un STI genérico para apoyar al proceso de aprendizaje.

Para conducir la investigación se plantea la siguiente **hipótesis**: Si se define una arquitectura flexible de un Sistema Tutor Inteligente entonces se facilitará el desarrollo de nuevos STI con la capacidad de adaptarse y expandirse en diversos contextos.

Se plantean los siguientes **objetivos específicos** para lograr el cumplimiento del objetivo general:

- Realizar el estudio del estado del arte de las técnicas de Inteligencia Artificial y los Sistemas Tutores Inteligentes.
- Seleccionar la metodología, herramientas y tecnologías que satisfagan las necesidades del sistema.
- Utilizar la metodología seleccionada para el modelado del Sistema Tutor Inteligente.
- Desarrollar la arquitectura del sistema.
- Validar los resultados obtenidos durante el desarrollo de la arquitectura del STI.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

La Inteligencia Artificial (IA) y los Sistemas Tutores Inteligentes (STI) constituyen actualmente un área de creciente interés dentro del proceso de Enseñanza-Aprendizaje. Para una mejor comprensión de esta tecnología, se abordan conceptos y aspectos relevantes de la IA y los STI y su aplicación en la educación, se llega finalmente a la elección de una metodología que permita modelar un sistema de apoyo a la enseñanza y las herramientas necesarias para el desarrollo de la arquitectura.

1.2 Inteligencia Artificial

A Culloch y Pitts de acuerdo a Stuart (Stuart, y colegas, 1995), se les atribuyen los primeros trabajos relacionados con la Inteligencia Artificial. Sus principales investigaciones se enmarcan en el conocimiento de la fisiología y las funciones básicas de las neuronas, el análisis formal de la lógica proposicional y la teoría de computación de Turing. Posteriormente en 1949 se creó una regla simple para modificar el peso (o intensidad) de las conexiones entre las neuronas del cerebro (Hebb, 1949). Alan Turing, en 1953 (Turing, y colegas, 1953), realizó programas que su principal aporte estaba en que seguían el paradigma de Von Newman. Un año más tarde, en 1954, se creó una red neuronal que simulaba 40 neuronas, para su confección se utilizaron 3000 tubos de vacío y partes de un bombardero B-24, a este proyecto se le llamó Stochastic Neural Analog Reinforcement Calculator (SNARC) (Minsky, 1954).

De acuerdo a Stuart (Stuart, y colegas, 1995) es en 1954 cuando Newell y Simon desarrollaron programas que eran capaces de utilizar razonamiento lógico. La declaración hecha por Simon de haber inventado un programa computacional capaz de pensar de manera no numérica provocó un gran impacto. Stuart plantea que es en los últimos años de 1950 cuando se tomó el nombre que McCarthy le había dado a este campo de la computación por más de 20 años: Inteligencia Artificial.

En los años 1960, las aplicaciones y utilidades de la IA se esparcieron por todo el mundo. Rápidamente se crearon laboratorios de Inteligencia Artificial, robótica, y se fortalecieron los temas de investigación en esta rama. Sus manifestaciones más importantes se evidenciaron en las universidades más grandes del mundo como el Instituto Tecnológico de Massachusetts (MIT) y Universidad de Standford, además del programa de Slagle llamado Simbolic Automatic Integrator (SAINT) creado en 1963 (Slagle, 1963), este

último era capaz de solucionar problemas de cálculo que se le entregaban a los estudiantes del primer año.

Las próximas décadas fueron muy vertiginosas para la IA y las aplicaciones de esta rama se esparcieron por las diferentes esferas sociales. Algunos de los sistemas desarrollados en esta etapa fueron: Dendral, programa que determina la estructura de una sustancia con la información proveniente de un espectrómetro de masa, es considerado como precursor de los Sistemas Expertos Basados en el Conocimiento (Buchanan, y colegas, 1969); MyCin (Buchanan, y colegas, 1984), sistema para diagnosticar las infecciones de la sangre y Prospector, creado por Duda en 1979 (Duda, y colegas, 1980), entre otros.

La Inteligencia Artificial ha recibido diferentes interpretaciones por quienes se han dedicado a su estudio, dando lugar a una gran diversidad de conceptos. Este término está compuesto por dos vocablos (inteligencia y artificial) que para muchas personas se torna un poco difícil emitir una definición al respecto. De acuerdo al diccionario de la Real Academia de la Lengua Española en sus tres primeras accesiones: la inteligencia es la capacidad de entender o comprender, resolver problemas, es conocimiento; mientras que asume por artificial a lo hecho por mano o arte del hombre, no natural, falso. Es entonces un poco complejo conceptualizar ambas palabras en el contexto científico-tecnológico, con el fin de aportar las ideas claves que abarca esta rama de la computación.

En 1978 Bellman (Bellman, 1978) define la Inteligencia Artificial como: “La automatización de las actividades que asociamos con el pensamiento humano, actividades como la toma de decisiones, la solución de problemas y el aprendizaje” y siete años más tarde Haugeland se refiere a la IA como: “El nuevo y excitante esfuerzo de hacer pensar a las computadoras”…”computadoras con mente, en el sentido completo y literal de la frase”.

Un concepto más completo de esta ciencia es dado por Barr y Feigenbaun (Barr y Feigenbaun, 1981), quienes consideran que “La Inteligencia Artificial es la parte de la Ciencia que se ocupa del diseño de sistemas de computación inteligentes, es decir sistemas que exhiben las características que asociamos a la inteligencia y el comportamiento humano que se refiere a la comprensión del lenguaje, el aprendizaje, el razonamiento, la resolución de problemas, etc.”

1.2.1 Técnicas de la IA aplicadas a los Tutores Inteligentes

1.2.1.1 Redes Neuronales Artificiales

En el contexto de los sistemas inteligentes se encuentran las redes neuronales (RN), que son interconexiones masivas en paralelo de elementos simples y que responden a una cierta jerarquía que intentan interactuar con los objetos reales tal como lo haría un sistema neuronal psicológico (Kohonen, 2001).

El perceptrón, la RN más antigua, fue creado en 1957 por Frank Rosenblatt. Esta red posee la característica de asimilar conocimiento en base a las experiencias mediante la generalización de casos, posee algunas limitaciones como su incapacidad para resolver el problema de la función OR-exclusiva y, en general, es incapaz de clasificar clases no separables linealmente. A pesar de esto se sigue usando en la actualidad.

Las redes neuronales más usadas en el desarrollo de los STI según la bibliografía consultada son las redes Backpropagation (propagación del error hacia atrás) y las SOM (Mapas Autoorganizativos).

1.2.1.1.1 Red Backpropagation

En 1986, Rumelhart, Hinton y Williams, formalizaron un método para que una red neuronal aprendiera la asociación que existe entre los patrones de entrada y las clases correspondientes, para ello utilizaron varios niveles de neuronas. El método backpropagation ha ampliado de forma considerable el rango de aplicaciones de las redes neuronales.

El funcionamiento de esta red consiste en el aprendizaje de un conjunto predefinido de pares de entrada-salida dados como ejemplo. Primero se aplica un patrón de entrada como estímulo para la primera capa de las neuronas de la red, se va propagando a través de todas las capas superiores hasta generar una salida, se compara el resultado en las neuronas de salida con la salida que se desea obtener y se calcula un valor de error para cada neurona de salida. A continuación, estos errores se transmiten hacia atrás, parten de la capa de salida hacia todas las neuronas de la capa intermedia que contribuyan directamente a la salida. Este proceso se repite, capa por capa, hasta que todas las neuronas de la red hayan recibido un error que describa su aportación relativa al error total. Basándose en el valor del error recibido, se reajustan los pesos de conexión de cada neurona, de manera que en la siguiente vez que se presente el mismo patrón, la salida esté más cercana a la deseada.

La importancia de la red backpropagation consiste en su capacidad de autoadaptar los pesos de las neuronas de las capas intermedias para aprender la relación que existe entre un conjunto de patrones de entrada y sus salidas correspondientes. Es importante la capacidad de generalización, facilidad de dar salidas satisfactorias a entradas que el sistema no ha visto nunca en su fase de entrenamiento. La red debe encontrar una representación interna que le permita generar las salidas deseadas cuando se le dan entradas de entrenamiento, y que pueda aplicar, además, a entradas no presentadas durante la etapa de aprendizaje para clasificarlas.

1.2.1.1.2 Red SOM

Los modelos de Mapas Autoorganizativos (SOM, por sus siglas en inglés) fueron introducidos por T. Kohonen (Kohonen, 2001) y son un tipo especial de redes neuronales artificiales de aprendizaje no supervisado que ha sido exitosamente aplicado como una herramienta de Minería de Datos. Las ventajas de los mapas autoorganizativos radican en que son capaces de preservar la topología del espacio de los datos, proyectan datos altamente dimensionales a un esquema de representación de baja dimensión y tienen la habilidad de encontrar similitudes en los datos.

El algoritmo de las SOM consiste básicamente en un procedimiento iterativo capaz de representar la estructura topológica del espacio de entrada por medio de un conjunto discreto de prototipos de vectores de peso, la estructura es asociada a neuronas de la red. Las SOM mapean los patrones de entradas vecinos a neuronas vecinas. El mapa es generado estableciendo una correspondencia entre las señales de entrada, las neuronas se localizan en una cuadrícula discreta. La correspondencia es obtenida a través de un algoritmo de aprendizaje competitivo consistente en una secuencia de pasos de entrenamiento que modifica iterativamente el vector de pesos de neuronas. Para un mejor entendimiento se muestra el algoritmo planteado por Kohonen:

Requiere: iniciar pesos $W_{i,j}$ para cada neurona en la etapa de competición
  Mientras los pesos $W_{i,j}$ no varíen significativamente **Hacer**
    Para cada patrón de entrada **Hacer**
      presentar el patrón a la red
      obtener la neurona ganadora
      actualizar peso de las neuronas ganadoras y sus vecinos
    Fin Para
  Fin Mientras

1.2.1.2 Árboles de decisión

La familia de los Top Down Induction Trees (TDIDT), en español conocido como Inducción de Árboles de Decisión, pertenece a los métodos inductivos del aprendizaje automático que aprenden a partir de ejemplos preclasificados. Son usados en minería de datos para modelar las clasificaciones en los datos mediante árboles de decisión.

Los árboles de decisión representan una estructura de datos que organiza eficazmente los parámetros (descriptores); estos árboles son construidos de forma tal que en cada nodo se realiza una prueba sobre el valor de los descriptores y de acuerdo con la respuesta se desciende por las ramas hasta llegar al final del camino donde se encuentra el valor del clasificador.

Los árboles TDIDT, a los cuales pertenecen los generados por el ID3 y su descendiente C4.5, se construyen a partir del método de Hunt (Hunt, y colegas, 1966). Este método se encuentra explicado detalladamente también por FELGAER (FELGAER, 2005). La idea general para construir un árbol de decisión consiste en tener un conjunto T de casos preseleccionados y las clases {C1, C2,…, Ck}, donde existen tres posibilidades:

- T contiene uno o más casos, todos pertenecientes a una única clase Cn.
- T no contiene ningún caso.
- T contiene casos pertenecientes a varias clases.

En los casos en los que el conjunto T contiene ejemplos pertenecientes a distintas clases se realiza una prueba sobre las distintas variables y se realiza una partición según la “mejor” variable. Para encontrar la “mejor” variable se utiliza la teoría de la información que sostiene que la información se maximiza cuando la entropía se minimiza (la entropía determina desestructuración de un conjunto).

1.2.1.3 Razonamiento Basado en Casos

El Razonamiento Basado en Casos (RBC) es una técnica de IA muy usada en la construcción de STIs. Posee un paradigma para resolver problemas que tiene una capacidad inherente de aprender. El principio básico del RBC es solucionar problemas desconocidos mediante la aplicación de soluciones que fueron usadas para resolver problemas en el pasado (Kolodner, 1993). El proceso principal de este método

consiste en la búsqueda de casos análogos al problema presente y adaptar las soluciones halladas para resolver la situación actual.

A la hora de construir un Sistema Tutor Inteligente que use esta técnica debe tenerse en cuenta el análisis hecho por Casali (Casali, 2010) referente a dos características principales que deben cumplir dichos sistemas:

- El aprendizaje no es un módulo que se agrega a un sistema basado en conocimiento, sino que constituye una parte básica del desarrollo, integrándose a la metodología con que se aborda la resolución de problemas.
- El papel dinámico y central que adquiere la memoria, que no es ya un simple receptáculo de hechos establecidos en el curso de una consulta, sino que debe organizarse para permitir la memorización de episodios, y su correcto indexado, para poder evocarlos cuando resulten útiles en el análisis de casos futuros.

El RBC según Peña (Peña, y colegas, 2002) es en efecto un proceso cíclico e integrado para resolver un problema, aprender de su experiencia, resolver un nuevo problema y así sucesivamente. Si dos problemas se parecen, sus soluciones son similares y entonces es posible aplicar una adaptación de la anterior solución a la solución del problema actual.

Esta técnica funciona con un conjunto de casos. A un caso lo van a distinguir tres características principales: la descripción del problema, la solución que se aplicó y el resultado de la solución. Todos los casos son organizados en una estructura denominada Base de Casos (BC).

La BC según Martínez (Martínez, y colegas, 2009) contiene las experiencias, ejemplos o casos a partir de los cuales el sistema hace sus inferencias. Esta base puede ser generada a partir de casos o ejemplos resultantes del trabajo de expertos humanos o por un procedimiento automático o semiautomático que construye los casos desde datos existentes registrados, por ejemplo, en una base de datos. Su composición es un proceso cíclico como se muestra en la Figura 1 y cuenta con cuatro módulos (Ovalle y Jiménez, 2007):

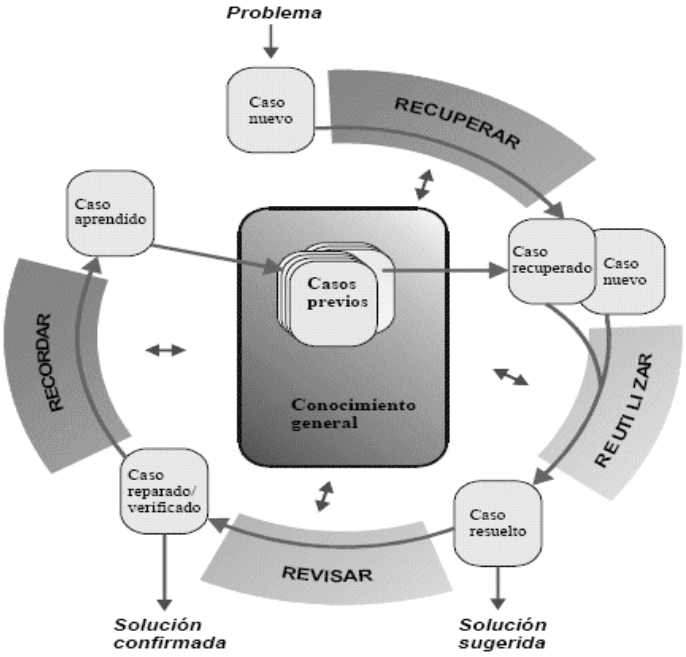
Recuperación: En la fase de recuperación se analiza el problema actual y se utilizan sus características principales para buscar en la BC aquellos casos que más se parezcan a la situación actual, entre los casos recuperados, se selecciona el más apropiado. Si se recuperan casos similares, el sistema estima el grado de similitud de los casos buscados usando valores fijos en sus umbrales o extremos.

Adaptación: Una vez que se ha determinado el caso más parecido al problema actual, el sistema debe adaptarlo para que se ajuste a sus peculiaridades.

Revisión: Después de aplicar la solución, el siguiente paso consiste en la revisión de los resultados obtenidos en la aplicación para comprobar si la solución propuesta ha tenido éxito o no. Si la solución ha fracasado, se intentan explicar las causas del fallo y se repara la solución para evitarlo en el futuro.

Almacenamiento: En esta última fase, el sistema almacena en la BC la nueva experiencia a través de un caso que incorpora el problema actual, la solución y sus resultados. Si la solución fracasó, se almacena la información necesaria para prevenir fracasos similares. El nuevo caso se integra en la BC y se actualizan los índices.

Figura 1: Ciclo del RBC.



Fuente: Tomado de (Ovalle y Jiménez, 2007).

1.2.1.4 Agentes Inteligentes

Russell y Norvig (Russell y Norvig, 2003) señalan que un agente es un sistema capaz de percibir a través de sensores las informaciones que provienen del ambiente donde está insertado y reaccionar a través de efectores, por lo que se lo puede definir como una entidad de software que exhibe un comportamiento

autónomo, situado en un ambiente en el cual es capaz de realizar acciones para alcanzar sus propios objetivos y a partir del cual percibe los cambios.

Los sensores de cada agente recolectan información, que se convierte en conocimiento para alcanzar su objetivo, razonan y actúan para modificar el propio entorno. Se comunican por medio del protocolo de paso de mensajes y realizan sus acciones concurrentemente. Poseen propiedades como: autonomía, habilidad social, reactividad, proactividad, movilidad, continuidad temporal, adaptabilidad y aprendizaje.

1.2.1.5 Inteligencia Artificial Distribuida

La Inteligencia Artificial Distribuida (IAD) es definida por Iglesias (Iglesias, 1998) como "aquella parte de la IA que se centra en comportamientos inteligentes colectivos que son producto de la cooperación de diversos agentes". Centrándose fundamentalmente en las relaciones existentes entre estos agentes, así como su colaboración.

En el análisis realizado por García y Ossowsky (García y Ossowsky, 2010) se considera que los sistemas IAD se caracterizan por una arquitectura formada por componentes inteligentes modulares que interactúan de forma coordinada. No existe una terminología compartida en IAD, ni esquemas de clasificación reconocidos por la mayoría, aunque no hay discrepancias en cuanto a las ventajas que ofrece el enfoque IAD sobre paradigmas convencionales:

- Como los sistemas distribuidos convencionales, los sistemas de la IAD aprovechan la distribución natural del dominio (espacial, temporal, funcional) con los fines siguientes: mejorar el rendimiento, la robustez, facilitar reusabilidad y mantenimiento. Además una arquitectura distribuida, facilita el aprovechamiento del paralelismo inherente en la estructura de un problema.
- Los sistemas IAD generan un valor añadido, que se manifiesta en: una mejor aceptabilidad en la sociedad, favorecer la adaptación de estructuras preexistentes en las organizaciones humanas y facilita la interacción hombre máquina.

De acuerdo a Iglesias la IAD está enmarcada en tres etapas: IAD clásica, IAD autónoma e IAD comercial. La IAD clásica tiene dos áreas principales de investigación: la resolución (cooperativa) de problemas distribuidos y los Sistemas Multi-Agentes (Iglesias, 1998).

Resumiendo, la IAD es una rama de la Inteligencia Artificial que abarca lo referido a la resolución de problemas con un enfoque cooperativo en un cierto ambiente utilizando los agentes inteligentes. Las técnicas desarrolladas por la IAD permiten resolver aplicaciones en distintos niveles a través de: 1) la

expansión de las funcionalidades existentes en el sistema a través del encapsulamiento de estas mismas aplicaciones en IAD y 2) la generación de sistemas que incorporen las técnicas de la IAD desde su concepción hasta su implementación. Esto permite obtener una mayor agilidad, flexibilidad, inteligencia y rendimiento del sistema global que se pueden mejorar a medida que estos factores permitan alcanzar los objetivos a través de: a) la construcción de sistemas descentralizados en lugar de los sistemas más usuales con todos sus componentes centralizados, b) la obtención de soluciones emergentes resultantes de las interacciones entre los agentes entre sí o de los agentes con los humanos y c) la ejecución concurrente en lugar de ejecución secuencial que es un punto muy importante a destacar en los desarrollos actuales (Bolan, y colegas, 2004).

1.3 Sistemas Tutores Inteligentes

Muchos sistemas fueron desarrollados desde que se comenzó a investigar sobre los STIs y sentaron las bases conceptuales de los sistemas desarrollados actualmente. Permitieron también un mejor entendimiento sobre las características que deben cumplir y los objetivos para los cuales son creados.

Entre los primeros conceptos que se definieron de un STI se destacan los de Wolf (Wolf, 1984), que los define como “sistemas que modelan la enseñanza, el aprendizaje, la comunicación y el dominio del conocimiento del especialista y el entendimiento del estudiante sobre ese dominio”; VanLehn plantea que un STI es “un sistema de software que utiliza técnicas de inteligencia artificial (IA) para representar el conocimiento e interactúa con los estudiantes para enseñárselo” (VanLehn, 1988); mientras que Burns (Burns, 1988) plantea que un sistema tutor se puede considerar “inteligente” según la medida en que cumpla tres criterios: 1) el dominio a enseñar debe ser conocido por el sistema, es decir, éste debe poder hacer inferencias o resolver problemas en este dominio, 2) el sistema debe poder deducir la cercanía del conocimiento del estudiante con el propio y 3) el sistema debe poder adaptar su estrategia de enseñanza para reducir la distancia entre su conocimiento y el del estudiante.

Por otra parte ya en la década del 90 Guardia Robles (Guardia Robles, 1993) resume un conjunto de características que deben cumplir todos los Sistemas Tutores Inteligentes (STI):

- Deben ser “inteligentes” en comparación con los sistemas tradicionales de instrucción por computadora (IAC), siendo el diferencial de inteligencia los métodos de la rama de la Inteligencia Artificial (IA).

- Deben poseer la capacidad tanto para resolver el problema que se le presenta a un estudiante como también la capacidad de explicar cómo lo resolvió.
- Como en los IAC tradicionales, permiten una mayor individualización en la instrucción a través del entendimiento de las metas y creencias del estudiante.
- Se usan técnicas de Inteligencia Artificial para planeación, optimización y búsquedas, deja así que el sistema decida el orden de presentación del contenido al alumno.
- La interacción puede ser muy variada en un STI: desde sistemas pasivos (que esperan para que el alumno realice una acción), hasta los que constantemente presentan nueva información (tutor oportunista), con casos intermedios en los que se enseña un concepto en un momento determinado o solo cuando el alumno lo pide.
- Utilizan nuevas tecnologías, con los ejemplos de interfaces orientadas a la utilización de multimedia y del WWW.
- No basta con indicarle un error al estudiante, el sistema debe hacer hipótesis basadas en el historial de errores del alumno y detectar la fuente del problema.

Con estas consideraciones en mente, Guardia Robles (Guardia Robles, 1993) presenta una definición mucho más completa para los tutores inteligentes: “Un Sistema Tutor Inteligente es un sistema de enseñanza asistida por computadora, que utiliza técnicas de Inteligencia Artificial, principalmente para representar el conocimiento y dirigir una estrategia de enseñanza; y es capaz de comportarse como un experto, tanto en el dominio del conocimiento que enseña (mostrando al alumno cómo aplicar dicho conocimiento), como en el dominio pedagógico, donde es capaz de diagnosticar la situación en la que se encuentra el estudiante y de acuerdo a ello ofrecer una acción o solución que le permita progresar en el aprendizaje.”

El objetivo de un STI es darle la capacidad a una computadora de involucrarse en el proceso de enseñanza usando técnicas de Inteligencia Artificial. Convirtiéndola no solo en un vehículo estático de información, sino en un instructor dinámico capaz de simular a un profesor humano. Un buen STI debe ser capaz, en primer lugar, de convertirse en un experto en la materia que va a impartir; pero además tiene que ser experto en las técnicas educativas; tiene que tomar varias decisiones y aprender de los casos que se le presenten, debe saber elaborar un programa para impartir un determinado conocimiento, y evaluar constantemente al estudiante para poder decidir qué contenido debe enseñarse en cada momento y cuál es la mejor forma de mostrárselo a cada estudiante en particular.

1.3.1 Modelos Arquitectónicos de STI

La idea de modelar la mente humana todavía es una tarea muy difícil de cumplir. Según Bolan la mayoría de los STI no presentan el nivel esperado de inteligencia (Bolan, y colegas, 2004). Es muy importante realizar una buena elección para modelar un sistema de expectativas tan altas.

De forma general existen dos formas de modelar la arquitectura de un STI: la primera se refiere a seguir fielmente la arquitectura básica propuesta por Carbonell en 1970 usando un modelo orientado a objetos; y la segunda propone el uso de los agentes inteligentes para crear elementos en la arquitectura que sean más independientes entre sí.

Independientemente del paradigma de programación que se utilice para la implementación de los Sistemas Tutores Inteligentes, siempre se deberán respetar las estructuras de los módulos, las interfaces y los distintos submódulos que componen la estructura propuesta por Carbonell.

1.3.1.1 Modelo de 3 capas propuesto por Carbonell

Este modelo fue propuesto por Carbonell (Carbonell, 1970), constituyendo el punto de partida para la creación de todos los Sistemas Tutores Inteligentes creados posteriormente; posibilita por primera vez un sistema informático para la enseñanza capaz de poseer inteligencia propia y así lograr adaptarse a las características de cada estudiante. Una de sus principales características es el alto grado de especialización en el contenido que imparte. Se basa en tres capas: Módulo del Tutor, Módulo del Estudiante y Módulo del Dominio como se muestra en la Figura 2.a.

El Módulo del Tutor define y aplica una estrategia pedagógica de enseñanza, contiene los objetivos a ser alcanzados y los planes utilizados para alcanzarlos. Selecciona los problemas, monitorea el desempeño, provee asistencia y selecciona el material de aprendizaje para el estudiante. Integra el conocimiento acerca del método de enseñanza, las técnicas didácticas y del dominio a ser enseñado. Consta de Protocolos Pedagógicos, un Planificador de Lección y un Analizador de Perfil.

El Módulo Estudiante tiene por objetivo realizar el diagnóstico cognitivo del alumno, y el modelado del mismo para una adecuada retroalimentación del sistema. Se plantean para el Módulo Estudiante los siguientes submódulos:

- Estilos de aprendizaje: compuesto por una base de datos con los estilos de aprendizajes disponibles en el sistema, los métodos de selección de estilos y las características de cada uno de ellos.

- Estado de conocimientos: contiene el mapa de conocimientos obtenido inicialmente a partir del módulo del dominio y que progresivamente el actualizador de conocimientos irá modificando a través de los resultados obtenidos en las evaluaciones efectuadas por el módulo del tutor, quien le enviará dichos resultados procesados.

Mientras que el Módulo del Dominio tiene el objetivo global de almacenar todos los conocimientos dependientes e independientes del campo de aplicación del STI. Básicamente deberá tener los submódulos: Parámetros Básicos del Sistema, Conocimientos y Elementos Didácticos.

Entre los STI desarrollados bajo este modelo se pueden destacar: Scholar, Why, Sophie, Guidon, West, Buggy, Debuggy, Steamer, Meno, Proust y Sierra.

SCHOLAR es considerado el primer STI. Fue diseñado por Jaime Carbonell en el Instituto Tecnológico de Massachusetts para enseñar la geografía de América del Sur, abarca conceptos como país, estado y población; se basó en una aplicación conocida como ELIZA, diseñada para el estudio de la comunicación entre persona-máquina. Según Wenger la representación del conocimiento se hace a través de una red semántica, donde los nodos representan conceptos y objetos geográficos, y los arcos representan una jerarquía parcial de relaciones tales como superparte, superconcepto y superatributo (Wenger, 1987).

Inicialmente, SCHOLAR crea un modelo con todas las preguntas (con sus respuestas) del tópico a enseñar. El estudiante contesta una serie de preguntas y el tutor compara las respuestas con su modelo, genera así la retroalimentación apropiada (aprobar o corregir la respuesta).

El sistema cuenta con una interfaz de lenguaje natural con vocabulario limitado. El texto se genera a partir de plantillas de oraciones y preguntas. El control de la sesión está repartido, pues tanto el estudiante como el sistema pueden tomar la iniciativa de hacer preguntas.

Por otra parte el STI Sophisticated Instructional Environment (SOPHIE) fue diseñado por John Seely. Casares plantea que fue el primer sistema de Enseñanza Asistida por Computadoras (EAC) inteligente en pretender entender su dominio (Casares, 1999). Se diseñó para enseñar a diagnosticar fallas en sistemas eléctricos, pero su arquitectura podría aplicarse en muchos otros campos. Sophie permite al estudiante hacer pruebas con un circuito descompuesto, al tiempo que evalúa sus hipótesis a partir de la información disponible y lo aconseja.

En lugar de tratar de imitar a un maestro, Sophie crea un ambiente en el cual el estudiante aprende probando sus ideas. Para lograr esto, tiene un poderoso modelo del dominio del conocimiento, junto con

numerosas heurísticas para contestar a las preguntas del estudiante, generar contraejemplos y para criticar constructivamente sus decisiones.

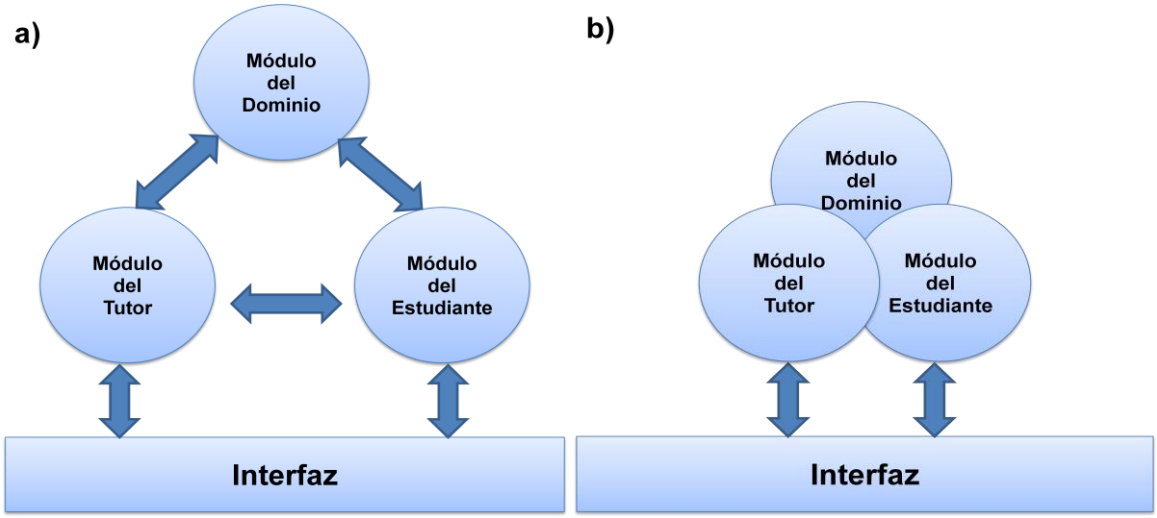
Tiene un esquema de inferencia que utiliza múltiples representaciones del conocimiento: Modelo de simulación del microcosmos (circuitos eléctricos), Especialistas procedurales, Redes y la Gramática semántica.

Stevens y Sollin desarrollaron el sistema tutorial para la enseñanza del proceso pluvial WHY (Beutelspacher, y colegas, 1995). El paradigma educativo en que Why se basa es el método socrático, en el que la instrucción consiste en cuestionar las respuestas del estudiante para hacerlo examinar su validez, encontrar contradicciones y hacer inferencias correctas. Para lograrlo utiliza una limitada interfaz de lenguaje natural.

La representación del dominio del conocimiento de Why se basa en scripts jerárquicos, series de ranuras que describen las secuencias de eventos que pueden suceder en una situación conocida y que establecen las relaciones temporales y causales entre los distintos eventos. Los eventos representan a su vez condiciones de entrada, resultados, propiedades y actividades. Además, cuenta con una base de reglas de producción, donde la condición es la última respuesta del alumno y en consecuencia, la acción que Why propondrá a continuación.

El modelo propuesto por Carbonell presenta algunas limitaciones. Una de ellas es que no se ha podido llevar a la práctica este modelo sin solapamiento de sus módulos como se muestra en la Figura 2.b, pues no se deja claro las funcionalidades de cada uno de ellos. Lo que dificulta la reutilización del código y la expansión del sistema.

Figura 2: Modelo de STI de 3 capas propuesto por Carbonell. Figura 2.a) Arquitectura teórica del STI. Figura 2.b) Arquitectura real implementada con solapamiento del STI.



Fuente: Elaboración propia.

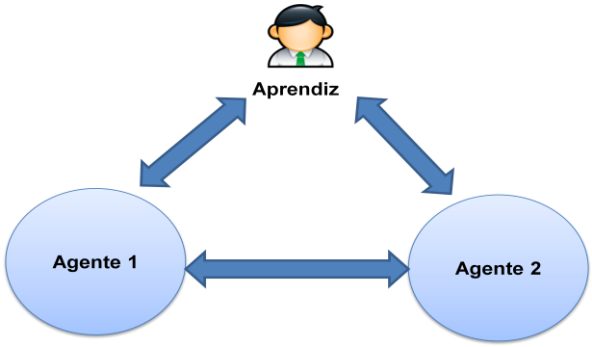
1.3.1.2 Modelo basado en Agentes

Villareal plantea que los agentes inteligentes en el marco educativo y de los STI se ven como: "*fragmentos de software con características humanas que facilitan el aprendizaje. Estas características pueden expresarse desplegando texto, gráfico, iconos, voz, animación, multimedia o realidad virtual*" (Villareal, 2003).

Las metodologías orientadas a agentes no han surgido como metodologías totalmente nuevas, sino que se han planteado como extensiones de metodologías existentes. Se trata de rediseñar la arquitectura básica propuesta por Carbonell a fin de obtener módulos con funcionalidades e interfaces bien definidas y sin solapamiento de funciones con la tecnología de agentes. Aunque en el artículo "Diseño de Sistemas Tutores Inteligentes con Tecnología de Agentes: Los Agentes Docentes en el Módulo Tutor" (Cataldi, y colegas, 2007) se plantea que resulta muy dificultoso en dominios complejos, donde los volúmenes de información requeridos son muy grandes, efectuar las revisiones de todas las situaciones posibles para que los agentes puedan evolucionar y adaptarse al entorno.

Este modelo no posee una estructura rígida en su arquitectura, es simplemente la interrelación de múltiples agentes entre sí y con los aprendices, en la Figura 3 se visualizan estos elementos.

Figura 3: Arquitectura general de un STI basado en Agentes.



Fuente: Elaboración propia.

Múltiples han sido los Sistemas Tutores Inteligentes desarrollados basados en Agentes. Los más relevantes en la bibliografía consultada son: MACES, AMPLIA, BAGHERA, JADE, MAS-PLANG y ALLEGRO.

En el sistema MAS-PLANG (Aguilar, y colegas, 2005) diferentes estudiantes interactúan con el sistema por medio de varios agentes (ver Anexo 1), que poseen doble función: 1) interactuar entre ellos y con el sistema en nombre del estudiante y 2) filtrar la información de aprendizaje que los estudiantes reciben de otros agentes y del sistema. Los agentes son individuales (cada estudiante tiene los suyos) y tienen conocimiento acerca de los objetivos y de los estilos de aprendizaje de los estudiantes que ellos representan y también son capaces de aprender de sus interacciones con el entorno.

Contiene dos niveles de agentes: los de nivel superior o asistentes personales y los de nivel inferior o agentes de información. Los agentes del nivel superior actúan como los típicos agentes personales digitales que aprenden del entorno e interactúan con el estudiante para ayudarlo en el desarrollo de sus actividades de aprendizaje. Mientras que los agentes del nivel inferior actúan de intermediarios entre los agentes del nivel superior y las bases de datos del modelo de dominio y del modelo de usuario para recomendar las unidades docentes adaptadas a las preferencias del estudiante de acuerdo a su estilo de aprendizaje.

La arquitectura del Sistema Multi-Agente Pedagógico ALLEGRO (ver Anexo 2) está conformada por dos tipos de agentes: Humanos (aprendices, docentes, auxiliares de docencia y expertos invitador) y de Software (tutor, experto, colaborativo, modelo de aprendiz, diagnóstico e interfaz). Su paradigma instruccional de acuerdo a Ovalle y Jiménez (Ovalle y Jiménez, 2005) se fundamenta en tres teorías de

aprendizaje: conductismo, cognitivismo (cognición distribuida y aprendizaje basado en problemas-PBL) e histórico-social. La planificación instruccional de ALLEGRO utiliza la técnica de Razonamiento Basado en Casos, con el fin de utilizar la experiencia almacenada de la solución exitosa de problemas similares pasados. En este sentido se puede afirmar que el sistema aprende de forma autónoma a partir de la experiencia con los aprendices, convirtiendo a la planificación instruccional en una herramienta flexible con capacidad de adaptar los conocimientos con determinado grado de abstracción dependiendo del alumno.

Los STI basados en agentes, de acuerdo a Cataldi (Cataldi, y colegas, 2007) cuentan con tolerancia a fallas dado que un sistema creado por agentes autónomos no se colapsará cuando uno o más de sus componentes falle. Presentan facilidades en la corrección de errores y en la modificación del sistema para proveer mejoras, dada la estructura modular del mismo basada en una estrategia "Divide y Vencerás"[1]. Se caracterizan por una fuerte especialización de cada módulo del sistema y posibilitan que el mismo aprenda de las sucesivas evaluaciones que se realizan. El sistema es extensible, es decir que se le pueden agregar nuevas funcionalidades sin comprometer su funcionamiento previo. Además, permite integrar múltiples plataformas, dada la estandarización de los mensajes entre agentes.

Una vez analizadas las características de los modelos de Sistemas Tutores Inteligentes, se escoge como propuesta de solución el Modelo basado en Agentes, bajo la base de que este modelo redefine al anterior propuesto por Carbonell en 1970 dividido en tres capas, con el objetivo de brindar una mayor especialización a cada uno de los módulos e independencia entre ellos. Estas propiedades van a permitir el desarrollo de un STI genérico al que se le puedan adicionar nuevos módulos o funcionalidades para aumentar su dominio sin que esto afecte el funcionamiento del STI.

1.4 Metodologías, herramientas y técnicas utilizadas

1.4.1 Análisis de las principales metodologías de desarrollo de Sistemas Multi-Agentes

La Ingeniería de Software ha permitido un mayor entendimiento acerca de la complejidad de las aplicaciones. En la producción de software moderna se hace imprescindible un proceso de diseño controlable, documentado y reproducible a su vez por los diversos desarrolladores, que permita definir

[1] La estrategia "Divide y Vencerás" consiste en resolver un problema dividiéndolo en subproblemas más sencillos y combinando sus soluciones. En el caso de los sistemas multi-agentes se identifican diferentes roles de cada agente repartiendo las responsabilidades entre todos.

etapas evolutivas sobre la base de abstracciones y de herramientas. Para lograr este objetivo, diversos expertos han creado las metodologías de desarrollo de software.

Para el desarrollo de sistemas con tecnología Orientada a Objetos existen varias metodologías de desarrollo. La más conocida es el Proceso Unificado de Desarrollo (RUP, por sus siglas en inglés); también se encuentran las metodologías ágiles como Programación Extrema (XP) y SCRUM. El paradigma Orientado a Agentes plantea nuevos conceptos para el desarrollo de software que no se satisfacen con la aplicación de metodologías orientadas a objetos. Se hace necesario adoptar una metodología que incorpore a los agentes como centro del proceso de desarrollo.

En los últimos tiempos ha surgido un amplio número de metodologías orientadas a agentes; entre las más referenciadas se encuentra MAS-CommonKADS, que extiende los modelos de CommonKADS para adaptarse a los sistemas Multi-Agentes; en el 2002 surge Prometheus, que sustenta el diseño, la documentación y la implementación de SMA; PASSI, que en español se corresponde con Proceso para la Especificación e Implementación de Sociedades de Agentes, fue desarrollada también en este año y propone un proceso paso a paso para el diseño e implementación de SMA, integra conceptos del Paradigma Orientado a Objetos y de Inteligencia Artificial. También han surgido otras metodologías como MaSE, GAIA, Tropos, Ingenias, MESSAGE, ZEUS, Vowel Engineering, RoMAS y RAP. En la Tabla 1 se hace un análisis acerca de las principales metodologías para desarrollar Sistemas Multi-Agentes, las fases del proceso de desarrollo que soportan y los distintos aspectos que modelan.

Tabla 1: Comparación entre metodologías para desarrollar SMA.

Aspectos		Tropos	GAIA	MaSE	INGENIAS	PASSI	MAS-CommonKADS
Ciclo de vida		Secuencial	Secuencial	Secuencial	Iterativo	Iterativo e Incremental	Secuencial
FASES	Requisitos	X	-	X	X	X	-
	Análisis	X	X	X	X	X	X
	Diseño	X	X	X	X	X	X
	Implementación	-	-	X	X	X	-
	Prueba	-	-	-	-	X	-
MODELOS	Ambiente	X	X	-	X	X	X
	Inteligencia	-	-	-	X	-	X
	Interacción	X	X	X	X	X	X
	Ontología	-	-	-	-	X	-
Total		5	4	5	7	8	5

Fuente: Elaboración propia.

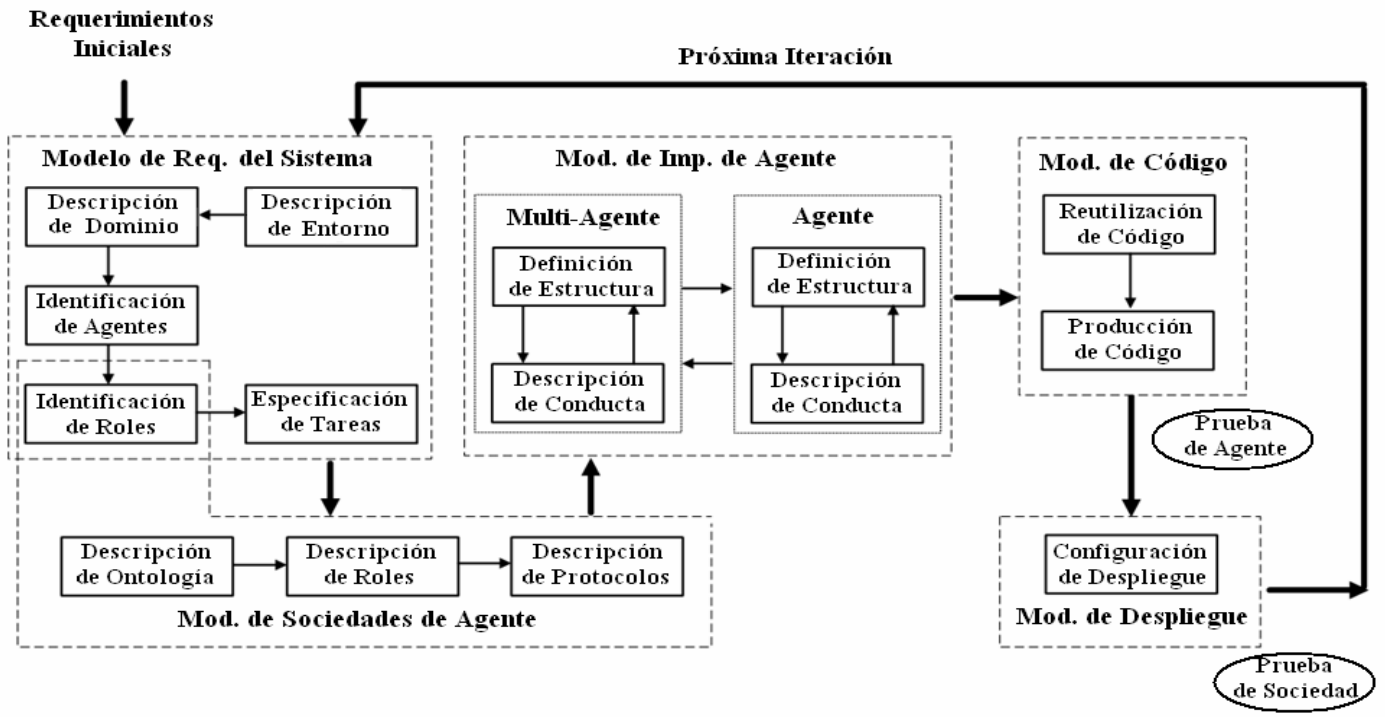
Teniendo en cuenta que 1) PASSI e Ingenias abarcan un mayor conjunto de características que son necesarias en el desarrollo de software, entre ellas un ciclo de vida más avanzado y fases tan importantes como análisis, diseño e implementación y 2) MAS-CommonKADS de acuerdo a Iglesias (Iglesias, y colegas, 2005) es la metodología más fuerte en el modelado del conocimiento (inteligencia) del agente con su Modelo de Experiencia, se realiza un breve análisis de dichas metodologías.

MAS-CommonKADS: es una metodología orientada a Sistemas Multi-Agentes, una extensión a CommonKADS (Iglesias, 1998). Con ella es posible describir todo el sistema basándose en representaciones gráficas y plantillas que el usuario debe rellenar. Utiliza los siguientes modelos para el desarrollo de SMA: Modelo de Agente, Modelo de Organización, Modelo de Tareas, Modelo de Experiencia, Modelo de Comunicación, Modelo de Coordinación, Modelo de Diseño (Iglesias,1998).

Ingenias: fue desarrollada por el grupo GRASIA de la Universidad Complutense de Madrid. Sus principales etapas son el Análisis y Diseño; presenta tres fases internas: Inicio, Elaboración y Construcción, recayendo el mayor peso sobre la fase de Elaboración (Gómez, 2003). Ingenias define un conjunto de meta-modelos (una descripción de alto nivel de qué elementos tiene un modelo) que son construidos con GOPRR (Graph, Object, Property, Relationship, and Role) en los que hay que describir el sistema. Los meta-modelos indican qué hace falta para describir: agentes aislados, organizaciones de agentes, el entorno, interacciones entre agentes o roles, tareas y objetivos.

PASSI: (Process for Agents Societies Specification and Implementation) opera sobre la base de la integración de Modelos de diseño y conceptos de Ingeniería de Software Orientada a Agentes (ISOA) lo que unido a las tendencias actuales de la Inteligencia Artificial en apoyo con el nivel de especificación que brinda el lenguaje de modelado UML (Unified Modeling Language), le atribuye un nivel de descripción acertado del objeto de aplicación (Gutierrez y Llarch, 2008). PASSI incorpora cinco modelos que responden a diferentes niveles de diseño, los que a su vez se subdividen en quince fases que resultan en el desarrollo del ciclo de vida de un SMA como se muestra en la Figura 4. Los mecanismos que implementa permiten la personalización de la representación del diseño orientado a agentes y evitan con ello la adopción de un lenguaje de modelado totalmente nuevo. Estructura el sistema a partir de casos de uso e incorpora además los principales conceptos del paradigma de agentes, como son: Recurso, Usuario, Rol, Escenario, Tarea e Interacción.

Figura 4: Fases del ciclo de vida de la metodología PASSI.



Fuente: Tomado de (Pérez, 2006).

1.4.2 Selección de la metodología de desarrollo de software

Con el estudio realizado sobre las metodologías Ingenias, MAS-CommonKADS, PASSI y con el análisis de la Tabla 1 se concluye que la metodología a utilizar es PASSI, al ser la más completa en el ciclo de vida y aspectos que modela. Es la única metodología que modela la fase de Pruebas, permitiendo la salida de productos con mayor calidad. Cuenta con una herramienta para el modelado: PASSI ToolKit y abundante documentación.

1.4.3 Herramientas

PASSI ToolKit

Por PASSI estar soportada por una herramienta CASE (Computer Aided Software Engineering): PASSI ToolKit (PKT), la cual constituye un sólido sostén a la metodología; encontrarse gratis en Internet y estar accesible a través de la página principal de PASSI, se escoge para el modelado del STI. PKT divide los artefactos en los paquetes: Vista de Casos de Uso, Vista Lógica, Vista de Componentes y Vista de

Despliegue. En estos paquetes se encuentran los modelos obtenidos en cada una de las fases de la metodología. Es capaz de compilar modelos correspondientes a diversos niveles de aplicación del proceso de desarrollo y brinda la reusabilidad de patrones de agentes de diversas aplicaciones a través del Agent-Factory.

Visual Paradigm

Se escoge la herramienta de modelado Visual Paradigm para el modelado de los datos ya que es una herramienta multiplataforma que ofrece una interfaz amigable, es compatible con diferentes gestores de Base de Datos (BD), dentro de estos se encuentran Oracle, PostgreSQL, MySQL, DB2, MS SQL, SQLite, Derby, entre otros. Permite la generación de la BD y su posterior sincronización. Además genera todas las clases persistentes en distintos lenguajes de programación y los archivos de mapeo de Hibernate.

Framework JADE

Dentro de los entornos de Sistemas Multi-Agentes, una opción es la plataforma OpenSource JADE (Java Agent Development Framework) que ofrece algunas ventajas que justifican la elección de la misma para el desarrollo del Sistema Multi-Agentes. La plataforma JADE, de acuerdo a Cataldi (Cataldi, y colegas, 2006), posee API´s (Application Programming Interface) para la creación de agentes y elementos relacionados con los mismos, cuenta además con una interfaz gráfica y con herramientas para controlar y depurar el sistema. Los sistemas responden al estándar FIPA (Foundation for Intelligent Physical Agents) al igual que los mensajes intercambiados por los agentes; maneja para cada agente una cola de mensajes ACL (Agent Communication Language) la cual se puede acceder de manera síncrona, asíncrona o temporizada.

Framework Hibernate

Hibernate es una herramienta ORM (del inglés Object-Relational Mapping) el cual, en tiempo récord, se ha establecido como el producto OpenSource líder en el mapeo de atributos entre una base de datos relacional y el modelo de objetos de una aplicación. Proporciona grandes beneficios como es la independencia de la base de datos, bajo acoplamiento entre negocio y persistencia, y un desarrollo rápido, ya que con Hibernate se puede cubrir de manera sencilla y rápida el 80 - 90% de la persistencia de cualquier aplicación (Rosés, 2004). Esto permite centrar los esfuerzos en optimizar las consultas que realmente lo merecen. Posee su propio lenguaje HQL (Hibernate Query Language) que lo hace multimotor. Soporta un amplio número de bases de datos relacionales: Oracle, Microsoft SQL Server,

Sybase, MySQL, TimesTen, HypersonicSQL, InterSystems Cache y SAP DB. Hibernate ofrece facilidades para la recuperación y actualización de datos, control de transacciones, repositorios de conexiones a bases de datos, consultas programáticas y declarativas, y un control de relaciones de entidades declarativas. Permite desarrollar objetos persistentes siguiendo el lenguaje común de Java: incluyendo asociación, herencia, polimorfismo, y composición. Por todas estas ventajas se selecciona el framework Hibernate como capa de abstracción de datos.

1.4.4 Lenguaje de programación

Java

Java es uno de los lenguajes más usados para el desarrollo de software a nivel mundial, presenta importantes características que lo convierten en un lenguaje potente, cumple con el paradigma Orientado a Objetos y posee una arquitectura neutral que posibilita que la compilación de las aplicaciones desarrolladas en java sea completamente independiente de la arquitectura del ordenador donde se ejecute. La neutralidad de la arquitectura mide en gran medida el grado de portabilidad de este lenguaje. Se caracteriza por ser un lenguaje seguro, elimina el uso de los punteros con el objetivo de eliminar el uso indebido de recursos en memoria. Además la máquina virtual se encarga de verificar que el software no posea fragmentos de código ilegal (código que falsea punteros, viola derechos de acceso sobre objetos o intenta cambiar el tipo o clase de un objeto) antes de ejecutarlo. Java es un lenguaje interpretado y compilado a la vez; es compilado, en la medida en que su código fuente se transforma en una especie de código máquina (bytecodes), semejantes a las instrucciones de ensamblador; por otra parte, es interpretado, ya que los bytecodes se pueden ejecutar directamente sobre cualquier máquina a la cual se haya instalado la máquina virtual de java. Por todas estas características y principalmente por soportar el framework JADE para el desarrollo de SMA se propone como lenguaje de programación del sistema.

1.5 Conclusiones

Con el estudio de los diferentes aspectos relacionados con los Sistemas Tutores Inteligentes se arrojó a la conclusión de que el modelo basado en agentes es el más apropiado para garantizar una arquitectura flexible para desarrollar STI adaptables a diversos entornos.

La metodología PASSI, como proceso de especificación e implementación de Sistemas Multi-Agentes, modela el ciclo de vida de los agentes inteligentes de forma completa e independiente del dominio de aplicación.

Capítulo 2: Arquitectura propuesta para el Sistema Multi-Agentes

2.1 Introducción

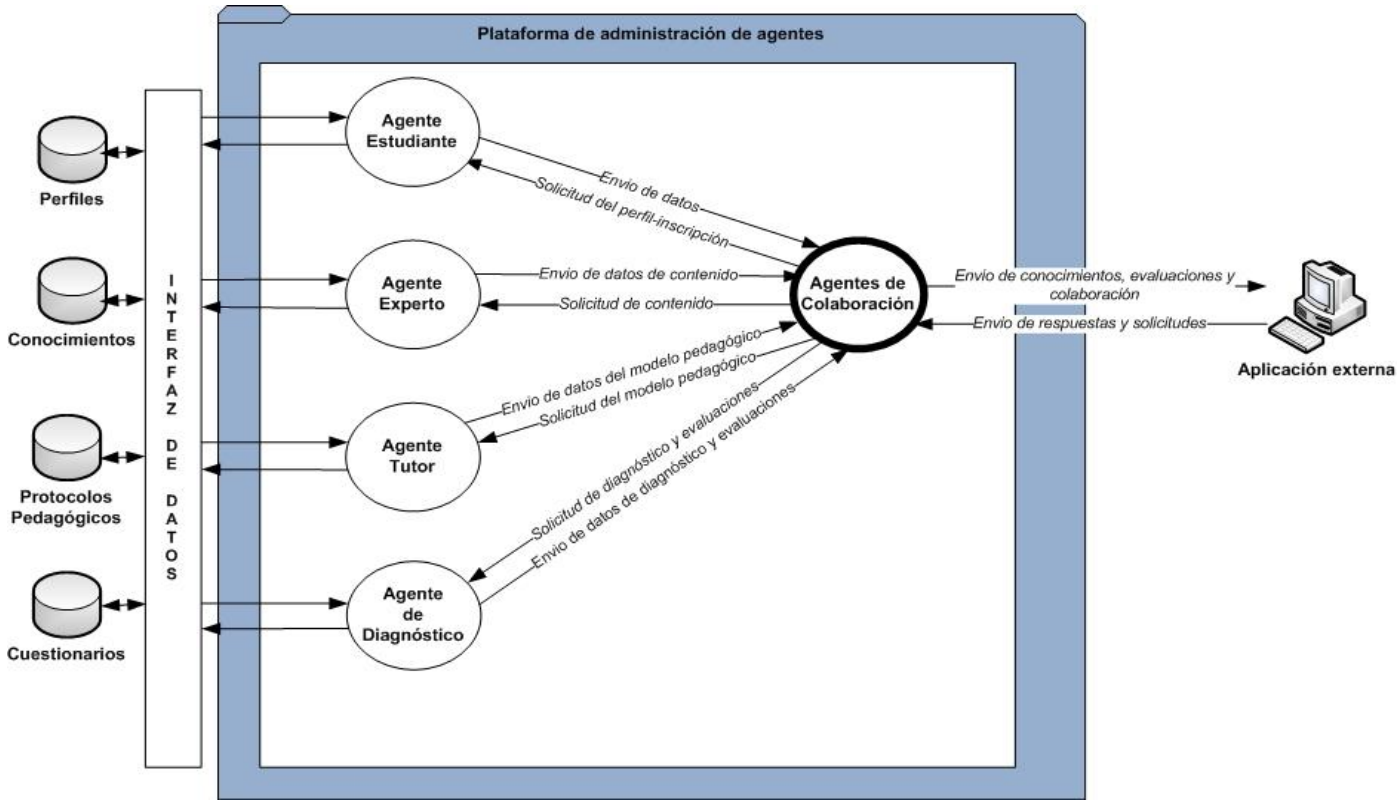
Tras el estudio de las arquitecturas generales que se han usado en el desarrollo de Sistemas Tutores Inteligentes en este capítulo se describe la arquitectura propuesta para un STI genérico. Además, para una mayor especificación del sistema, se detallan los elementos más significativos de la misma a través de la metodología PASSI, generándose una serie de artefactos que facilitan su comprensión.

2.2 Descripción general de la arquitectura de STI propuesta

En la arquitectura se usan algunos estilos para facilitar y estandarizar su desarrollo. Está separada en capas, esto garantiza su extensión y corrección de errores. Además es sustentada en un modelo cliente-servidor que permite que la capacidad de proceso esté repartida entre los clientes y los servidores. También usa estándares abiertos como FIPA (siglas de Foundation for Intelligent Physical Agents) y XML para garantizar interoperabilidad con sistemas externos.

Con el objetivo de garantizar flexibilidad en la arquitectura se propone que el Sistema Tutor Inteligente se divida en cuatro elementos fundamentales: el sistema de bases de datos, aplicaciones externas y la plataforma de administración de agentes, dentro de la cual se encuentra el núcleo del STI, como se muestra en la Figura 5.

Figura 5: Arquitectura general propuesta del STI.



Fuente: Elaboración propia.

La **plataforma de administración de agentes** tiene la responsabilidad de incorporar nuevos agentes al Sistema Multi-Agentes (el núcleo del STI), activar o desactivar a dichos agentes, así como facilitar y controlar la comunicación entre ellos.

Se propone utilizar la Plataforma de Agentes brindada por JADE. Esta plataforma brinda el Agent Management System (AMS), que garantiza que cada agente tenga un único nombre, es el encargado de proporcionar los servicios de páginas blancas y ciclo de vida, y de mantener el directorio de los identificadores de agentes (AID: Agent Identifier) y su estado; cada agente debe registrarse con el AMS para obtener un AID válido. El Directory Facilitator (DF) también es proporcionado por la plataforma de JADE; este agente proporciona los servicios de páginas amarillas, de esta forma permite que un agente encuentre a otros agentes que proporcionan los servicios que requiere para cumplir sus objetivos. También cuenta con el Agent Communication Channel (ACC) para controlar el intercambio de mensajes.

Para soportar todos estos servicios la Plataforma de Agentes brindada por JADE posee varias herramientas de interfaz gráfica: Remote Monitoring Agent (RMA), Directory Facilitator (GUI DF), Dummy Agent, Sniffer Agent e Introspector Agent.

El **núcleo del STI** está compuesto por un sistema de agentes inteligentes usando la Inteligencia Artificial Distribuida (IAD), se convierte así en el componente inteligente del sistema. Contiene varios agentes de colaboración encargados de tramitar las solicitudes y respuestas de las aplicaciones externas hacia el agente más apropiado, mantener la comunicación entre ellos, así como organizar la información dada por los agentes internos para enviarla de forma coherente hacia el exterior.

El agente estudiante modela toda la información referida a los aprendices para garantizar que el tutorado se realice de forma personalizada, consta de dos partes principales: la gestión de los estilos de aprendizaje y el modelo del conocimiento que posee el estudiante.

Al registrar un nuevo estudiante la aplicación externa debe informar el estilo de aprendizaje que posee el mismo. Para lograr esto se propone la aplicación al estudiante del test adaptado en la Universidad de las Ciencias Informáticas (UCI) del modelo desarrollado por los Drs. Juan Silvio Cabrera y Gloria Fariñas.

En el transcurso del proceso de tutoría el STI actualiza los estilos de aprendizaje del estudiante en dependencia de las acciones realizadas por el mismo. El Razonamiento Basado en Casos (RBC) le permite al sistema asociar el comportamiento que sigue un estudiante con los estilos de aprendizaje a partir de la base de casos que tuvieron un aprendizaje satisfactorio con un comportamiento similar.

Para representar el conocimiento del estudiante se acoge el modelo de superposición (overlay), donde se considera que el objetivo es lograr que el estudiante llegue a adquirir el mismo conocimiento del experto. De esta forma se considera al conocimiento del estudiante como un subconjunto de lo que conoce el experto.

El agente tutor es el encargado de planificar la lección y las evaluaciones. A la hora de planificar una clase o evaluación se tiene en cuenta el estado cognitivo del estudiante y sus preferencias de aprendizaje, cuenta con un conjunto de reglas pedagógicas específicas para cada contenido o examen que le permite al tutor personalizar la instrucción a partir del contenido que se debe impartir, las reglas correspondientes al mismo, las evaluaciones pendientes y los estilos de aprendizaje del estudiante.

Las evaluaciones están concebidas en el STI mediante un agente de diagnóstico. Estas se almacenan y se asocian a un contenido del experto. Este agente es responsable además de gestionar las reglas de producción que permiten realizar un diagnóstico del estudiante en dependencia de su perfil y las respuestas dadas en las evaluaciones efectuadas.

El sistema cuenta además con un agente experto, el cual posee el contenido que se desea impartir. Para modelar el conocimiento se propone la técnica caja de cristal. Esta variante es muy común en los sistemas expertos, donde el conocimiento se representa con reglas de producción, muy comprensibles para un humano. Cuenta con un árbol de dependencia entre los contenidos para poder determinar cuál debe impartirse a un estudiante en dependencia de su estado de conocimientos adquiridos. Se considera que cada dominio está organizado por temas, y estos a su vez poseen objetivos o metas a cumplir a través de diferentes contenidos.

La interacción con el Tutor Inteligente está concebida a través de distintas **aplicaciones externas** que se comunican usando el protocolo estándar FIPA con los agentes de colaboración del sistema. De esta forma se brinda la posibilidad de acceder al STI desde cualquier plataforma y realizar las extensiones requeridas en un futuro, sin importar elementos como el lenguaje de programación utilizado en la aplicación. Esta fase de la investigación no abarca el estudio ni desarrollo de las aplicaciones externas.

El **sistema de bases de datos** necesario para el funcionamiento del STI debe establecer una comunicación con el núcleo a través de una interfaz de datos para permitir el uso de diferentes servidores de bases de datos. Para lograr este objetivo se utiliza el framework Hibernate por soportar un gran número de sistemas de bases de datos y permitir fácilmente la abstracción de los mismos a través del mapeo usando el lenguaje XML hacia el modelo de objetos. El modelo de datos se refleja en el epígrafe 2.4.

2.3 La metodología PASSI aplicada al STI

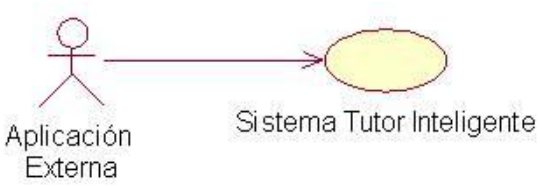
Para la especificación de la arquitectura propuesta se hace uso de la metodología PASSI con cada uno de los artefactos que esta genera a través de los modelos de requerimientos del sistema, modelo de sociedad de agente, modelo de implementación de agente, modelo de código y el modelo de despliegue.

2.3.1 Modelo de requerimiento del sistema

2.3.1.1 Descripción de entorno

Este modelo proporciona una visión del sistema al máximo nivel de abstracción, donde se representan los actores internos y externos, así como sus relaciones con el sistema, considerado este como un todo único como se muestra en la Figura 6.

Figura 6: Diagrama de entorno.



Fuente: Elaboración propia.

La aplicación externa es cualquier tipo de aplicación que se comunique con el sistema a través de protocolos preestablecidos. Esta está a cargo de establecer la línea fundamental de comunicación con el STI. Interactúa con el Caso de Uso Sistema Tutor Inteligente, el cual representa el sistema multi-agentes como un todo.

2.3.1.2 Descripción de dominio

La fase de Descripción de Dominio es el punto de partida de la metodología PASSI. En ella se modelan los requerimientos del sistema a través de diagramas de casos de uso mediante estereotipos importados del Lenguaje Unificado de Modelado (UML). El diagrama resultante (Figura 7) proporciona una primera aproximación de la identificación de los agentes involucrados en el sistema.

De la identificación de las funcionalidades esperadas del sistema a automatizar, se obtiene una serie de Casos de Uso (CU) que se prevé satisfagan las mismas. Los CU identificados son los siguientes:

- CU Enviar información
- CU Procesar información de agentes
- CU Tramitar solicitud
- CU Autenticar
- CU Gestionar usuario

- CU Gestionar sesión
- CU Gestionar log
- CU Gestionar estudiante
- CU Gestionar estilos de aprendizaje de estudiante
- CU Gestionar estado cognitivo de estudiante
- CU Gestionar protocolos pedagógicos
- CU Gestionar evaluaciones
- CU Gestionar dominio
- CU Tramitar aspectos de la clase
- CU Seleccionar material de consulta
- CU Seleccionar evaluaciones
- CU Definir estrategias pedagógicas
- CU Realizar diagnóstico

De acuerdo al impacto que provocan en la arquitectura del sistema los CU Tramitar solicitud y Tramitar aspectos de la clase son descritos detalladamente a continuación:

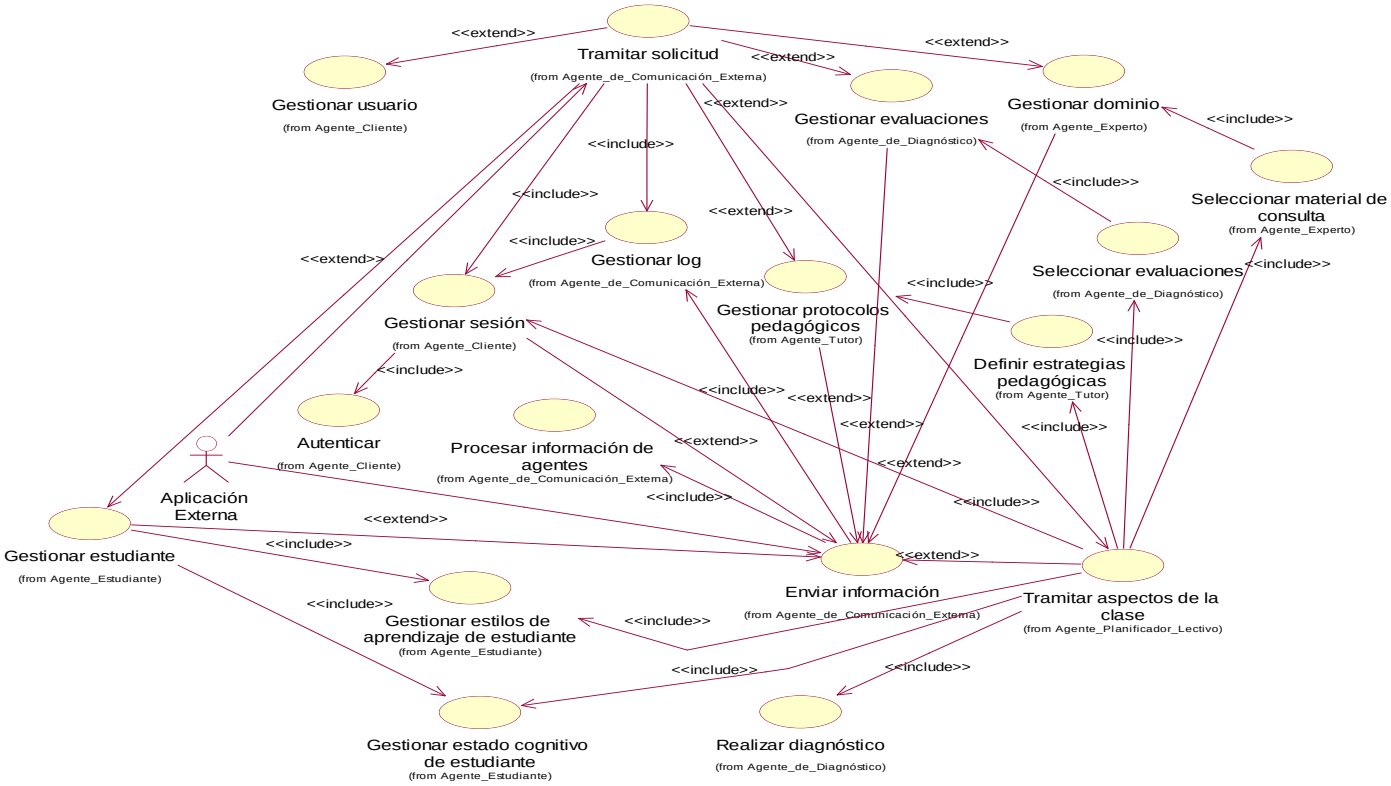
Tabla 2: Descripción del CU Tramitar solicitud.

CU: Tramitar solicitud		
Descripción:	Este caso de uso es el responsable de recibir todas las solicitudes realizadas desde las aplicaciones externas y decidir hacia qué agente se debe dirigir la solicitud. De esta forma se garantiza un control sobre el flujo de información que entra al núcleo del STI desde el exterior.	
Comunicaciones:	**Iniciador**	**Participantes**
	1. Aplicación Externa	1. CU Tramitar solicitud
	2. CU Tramitar solicitud	2. CU Gestionar usuario
	3. CU Tramitar solicitud	3. CU Gestionar sesión
	4. CU Tramitar solicitud	4. CU Gestionar estudiante
	5. CU Tramitar solicitud	5. CU Tramitar aspectos de la clase
	6. CU Tramitar solicitud	6. CU Gestionar log
	7. CU Tramitar solicitud	7. CU Gestionar protocolos pedagógicos
	8. CU Tramitar solicitud	8. CU Gestionar evaluaciones
	9. CU Tramitar solicitud	9. CU Gestionar dominio

Tabla 3: Descripción del CU Tramitar aspectos de la clase.

CU: Tramitar aspectos de la clase		
Descripción:	Mediante el CU Tramitar aspectos de la clase se gestiona todo el proceso de instrucción de una lección. Es el encargado de tramitar el flujo de información entre los agentes encargados de la instrucción, garantiza el orden adecuado del mismo: primeramente solicita el perfil del estudiante, con esta información pide que se seleccionen los materiales que corresponde enseñar al estudiante según su perfil al igual que las evaluaciones; por último manda a definir la estrategia pedagógica adecuada según el estudiante y organiza la lección con todos estos elementos.	
Comunicaciones:	**Iniciador**	**Participantes**
	1. CU Tramitar solicitud 2. CU Tramitar aspectos de la clase 3. CU Tramitar aspectos de la clase 4. CU Tramitar aspectos de la clase 5. CU Tramitar aspectos de la clase 6. CU Tramitar aspectos de la clase 7. CU Tramitar aspectos de la clase 8. CU Tramitar aspectos de la clase 9. CU Tramitar aspectos de la clase	1. CU Tramitar aspectos de la clase 2. CU Gestionar estilos de aprendizaje de estudiante 3. CU Gestionar estado cognitivo de estudiante 4. CU Enviar información 5. CU Seleccionar material de consulta 6. CU Seleccionar evaluaciones 7. CU Definir estrategias pedagógicas 8. CU Realizar diagnóstico 9. CU Gestionar sesión

Figura 7: Diagrama de descripción de dominio.



Fuente: Elaboración propia.

2.3.1.3 Identificación de agentes

Esta etapa permite la agrupación de las funcionalidades del sistema por agentes, los cuales son representados con el estereotipo de paquetes UML. Siguiendo esta idea, un agente puede representar a un caso de uso o un paquete de casos de uso de los que fueron identificados en la descripción del dominio, se genera así un nuevo diagrama.

A continuación se describen los agentes identificados.

Agente Cliente

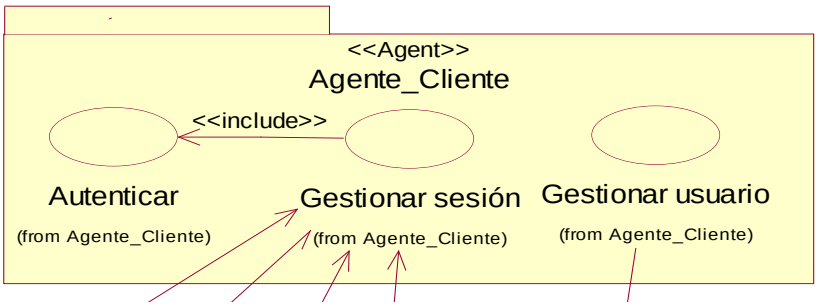
Este agente agrupa los siguientes CU:

1. CU Autenticar
2. CU Gestionar sesión

3. CU Gestionar usuario

Estos casos de uso se agrupan en el agente cliente porque son las funcionalidades que gestionan toda la información referente al cliente y los usuarios del sistema. Dicho agente controla y mantiene informados a los demás agentes sobre la información necesaria de los clientes que se encuentren conectados al sistema.

Figura 8: Agente Cliente (fragmento del Diagrama de identificación de agentes).



Fuente: Elaboración propia.

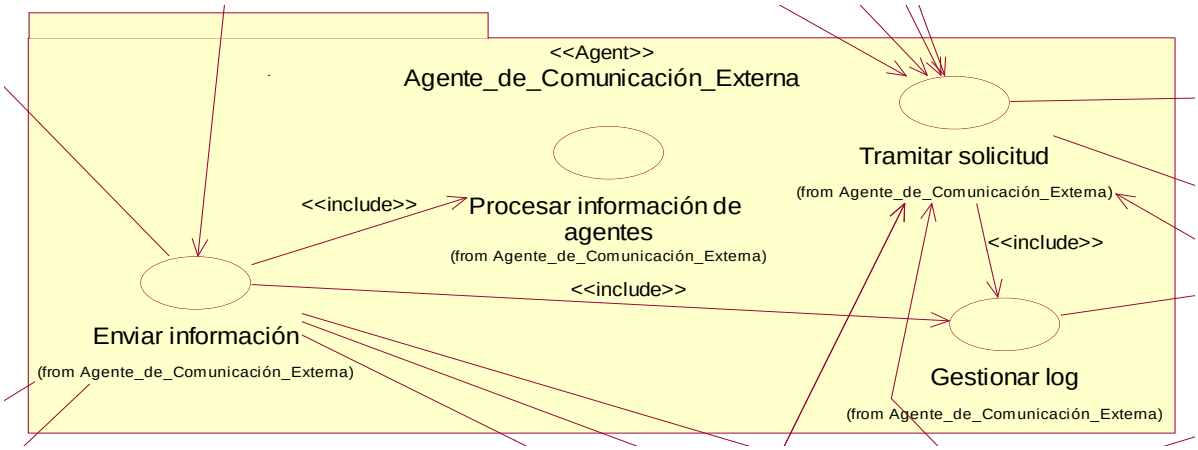
Agente de Comunicación Externa

Este agente agrupa los siguientes CU:

1. CU Enviar información
2. CU Procesar información de agentes
3. CU Tramitar solicitud
4. CU Gestionar log

Los CU Procesar información de agentes, Enviar información, Gestionar log y Tramitar solicitud son agrupados en el agente de comunicación externa por referirse a la comunicación del sistema con los recursos externos. Este agente es el encargado de manejar la información brindada por los agentes y responder a las solicitudes realizadas desde el exterior. También es quien decide cuál es el agente indicado para enviarle las solicitudes que llegan al sistema y se encarga de registrar la actividad realizada por los clientes en el sistema.

Figura 9: Agente de Comunicación Externa (fragmento del Diagrama de identificación de agentes).



Fuente: Elaboración propia.

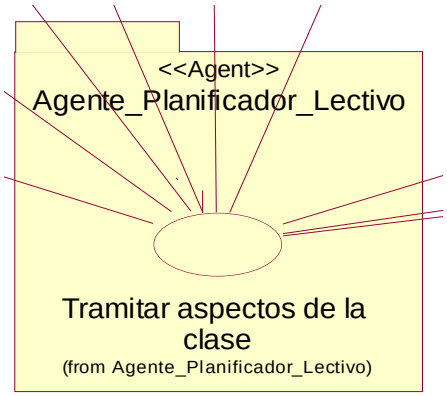
Agente Planificador Lectivo

Este agente contiene al CU:

1. CU Tramitar aspectos de la clase

El agente planificador lectivo es considerado el más importante para el Sistema Tutor Inteligente; está a cargo de todo el proceso de planificación de la instrucción. Establece la coordinación entre los agentes estudiante, experto, de diagnóstico y tutor con el objetivo de confeccionar una clase y mantener un flujo de retroalimentación en dependencia de las acciones del estudiante.

Figura 10: Agente Planificador Lectivo (fragmento del Diagrama de identificación de agentes).



Fuente: Elaboración propia.

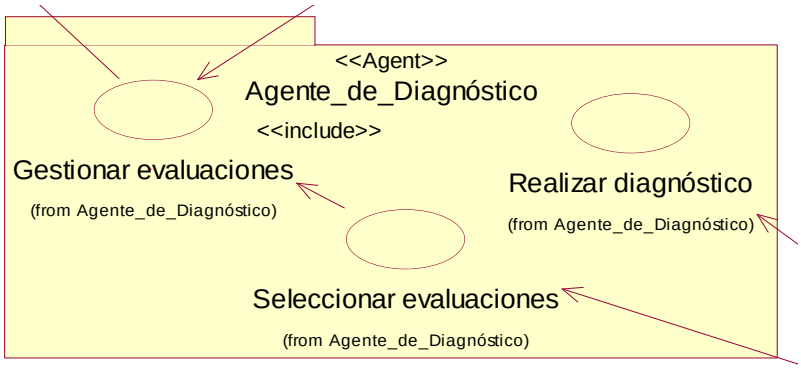
Agente de Diagnóstico

Este agente agrupa los siguientes CU:

1. CU Gestionar evaluaciones
2. CU Seleccionar evaluaciones
3. CU Realizar diagnóstico

Estos casos de uso son agrupados en el agente de diagnóstico por estar estrechamente relacionados con el proceso de evaluación y retroalimentación del estado cognitivo del estudiante. Este agente administra las evaluaciones que debe realizar el estudiante en dependencia de su perfil y es el encargado de proveer el diagnóstico del mismo en todo momento. Almacena y gestiona además todas las evaluaciones del STI clasificadas y organizadas por tópicos, objetivos y preferencias de aprendizaje.

Figura 11: Agente de Diagnóstico (fragmento del Diagrama de identificación de agentes).



Fuente: Elaboración propia.

Agente Estudiante

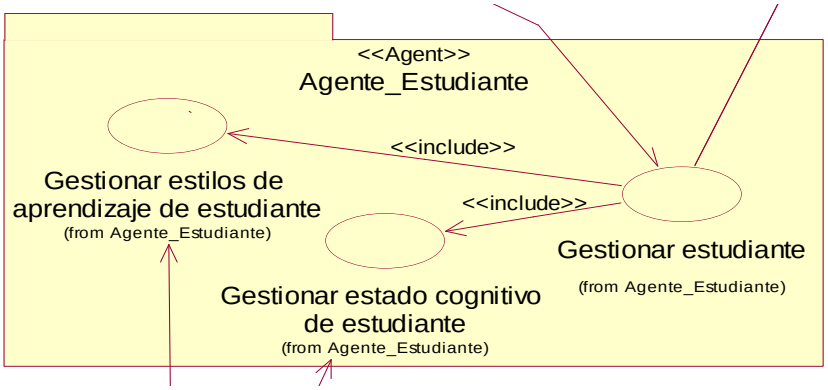
Este agente agrupa los siguientes CU:

1. CU Gestionar estudiante
2. CU Gestionar estilos de aprendizaje de estudiante
3. CU Gestionar estado cognitivo de estudiante

Estos casos de uso gestionan toda la información correspondiente a los estudiantes. El agente estudiante agrupa a dichos CU como se muestra en la Figura 12, se modela de esta forma el comportamiento de un

aprendiz. Almacena y gestiona los estilos de aprendizaje y el estado cognitivo que posee para poder generar un perfil del mismo que permita al STI brindarle una atención y seguimiento personalizado.

Figura 12: Agente Estudiante (fragmento del Diagrama de identificación de agentes).



Fuente: Elaboración propia.

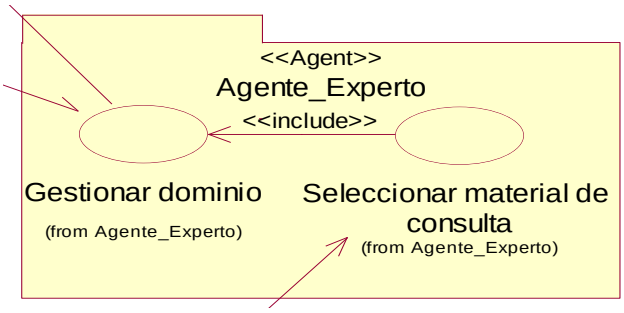
Agente Experto

Este agente agrupa los siguientes CU:

1. CU Gestionar dominio
2. CU Seleccionar material de consulta

El agente experto agrupa a todos los casos de uso que permiten modelar el conocimiento del sistema. Este agente simula a un experto en varios dominios, presenta la capacidad de aprender, estructura y organiza todo el conocimiento de tal forma que pueda brindarlo de una manera comprensible por el resto de los agentes a la hora de planificar la instrucción.

Figura 13: Agente Experto (fragmento del Diagrama de identificación de agentes).



Fuente: Elaboración propia.

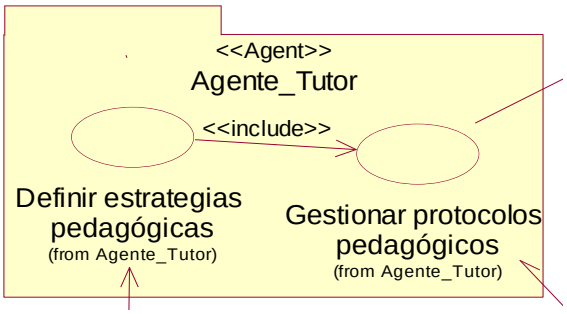
Agente Tutor

Este agente agrupa los siguientes CU:

1. CU Gestionar protocolos pedagógicos
2. CU Definir estrategias pedagógicas

Los casos de uso Gestionar protocolos pedagógicos y Definir estrategias pedagógicas se contemplan en el agente tutor por manejar todo lo referente a los protocolos pedagógicos que son utilizados en el proceso de instrucción. Este agente ayuda a planificar una lección según las preferencias del estudiante, y el conocimiento que le corresponda recibir o evaluar. Almacena y gestiona un conjunto de reglas que le permite tomar decisiones al respecto que son conocidas como protocolos pedagógicos.

Figura 14: Agente Tutor (fragmento del Diagrama de identificación de agentes).



Fuente: Elaboración propia.

2.3.1.4 Identificación de roles

La Identificación de Roles consiste en conocer cuáles son las responsabilidades que tienen los agentes involucrados en el sistema. Para ello se describen escenarios específicos para cada rol mediante diagramas de secuencia. Los roles son obtenidos tras la composición de varias tareas en una conducta resultante. Esto muestra las capacidades de un rol en particular, así como su posible utilidad en la identificación de modelos reutilizables. A cada escenario identificado se le confecciona un diagrama, en el que cada objeto representa un rol específico de un agente. Estos diagramas se corresponden al flujo normal de eventos del escenario mostrado, suponiendo con ello la realización satisfactoria de las tareas involucradas.

El objetivo primordial de esta etapa es modelar el ciclo de vida de cada agente e identificar los roles que puede jugar, las colaboraciones que necesita, y las comunicaciones en las que participa.

De los agentes propuestos en la sección 2.3.1.3 se identifican los siguientes roles:

Agente Cliente

- Rol Autenticar
- Rol Gestionar sesiones
- Rol Gestionar usuario

Agente de Comunicación Externa

- Rol Tramitar solicitud
- Rol Controlar flujo de información hacia el exterior
- Rol Gestionar log

Agente Planificador Lectivo

- Rol Planificar lección

Agente de Diagnóstico

- Rol Definir evaluación
- Rol Diagnosticar
- Rol Gestionar evaluación

Agente Estudiante

- Rol Gestionar estudiante
- Rol Gestionar estilos de aprendizaje de estudiante
- Rol Gestionar estado cognitivo de estudiante

Agente Experto

- Rol Gestionar dominio
- Rol Definir conocimiento a impartir

Agente Tutor

- Rol Gestionar protocolos pedagógicos
- Rol Definir estrategias pedagógicas

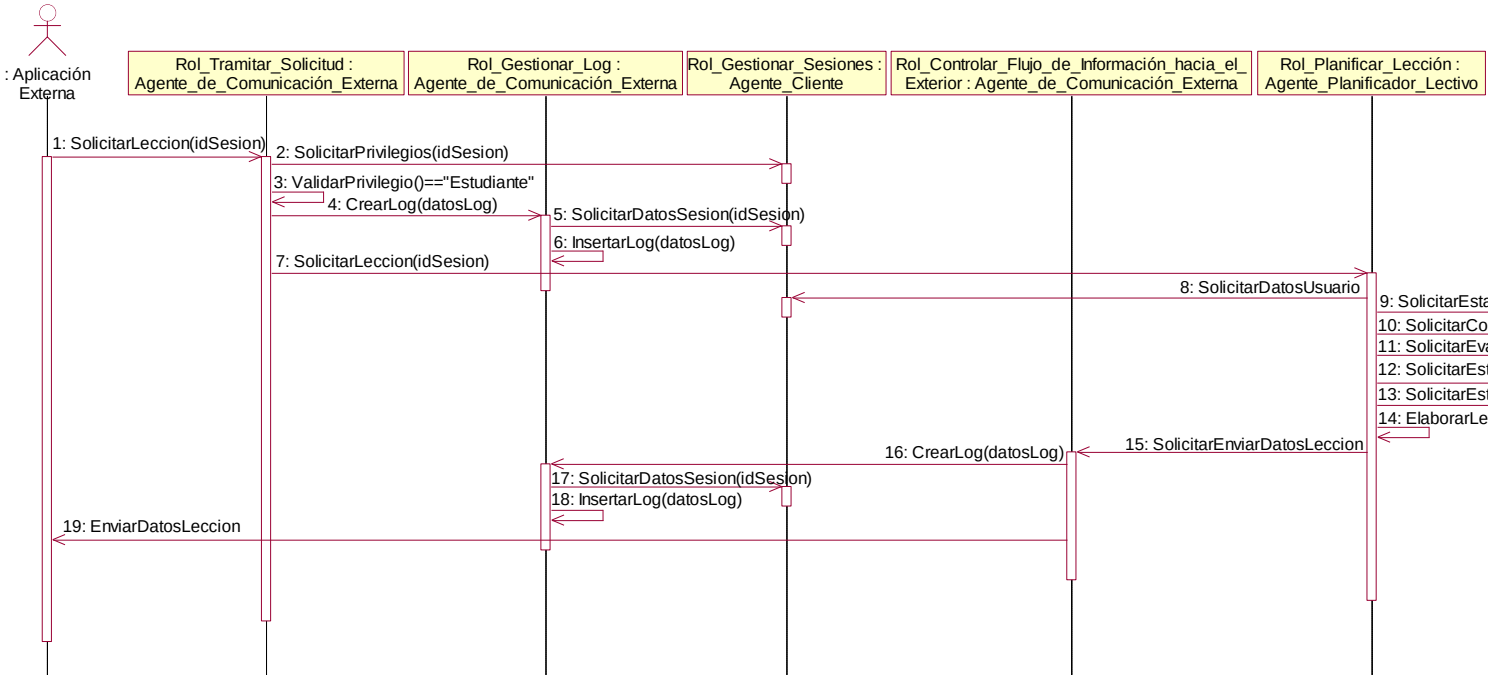
Estos roles intervienen en los escenarios: Crear sesión, Gestionar dominio, Gestionar usuario, Gestionar evaluación, Gestionar protocolo pedagógico e Impartir lección. Por su importancia e impacto en la arquitectura del sistema, se especifica la descripción de los escenarios Crear sesión e Impartir lección con algunos de sus respectivos diagramas de secuencia.

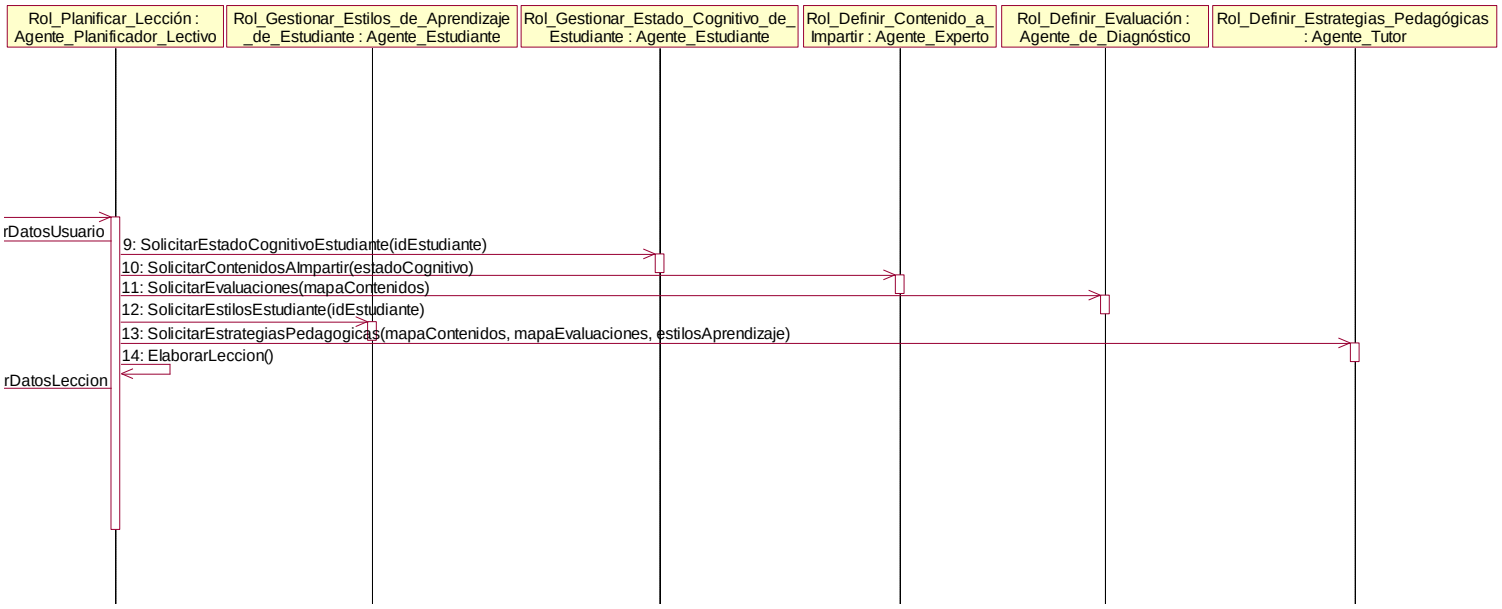
Tabla 4: Especificación del escenario Impartir lección.

Escenario: Impartir lección	
Descripción	Este escenario representa el flujo básico de eventos que se llevan a cabo en el sistema cuando una aplicación externa autenticada representando a un estudiante solicita una lección de una materia.
CU Asociados	CU Autenticar, CU Gestionar sesión, CU Gestionar log, CU Enviar información, CU Procesar información de agentes, CU Tramitar solicitud, CU Gestionar estilos de aprendizaje de estudiante, CU Gestionar estado cognitivo de estudiante, CU Gestionar protocolos pedagógicos, CU Gestionar evaluaciones, CU Gestionar dominio, CU Tramitar aspectos de la clase, CU Seleccionar material de consulta, CU Seleccionar evaluaciones, CU Definir estrategias pedagógicas, CU Realizar diagnóstico.
Roles Asociados	**Descripción**
Rol Gestionar sesiones (Agente Cliente)	Este rol se responsabiliza en este caso en brindar toda la información referente a la sesión mediante la cual está accediendo el cliente.
Rol Tramitar solicitud (Agente de Comunicación Externa)	Este rol es responsable de atender las solicitudes realizadas desde las aplicaciones externas y tramitarla al agente más apropiado (en este caso, al Agente Planificador Lectivo).
Rol Controlar flujo de información hacia el exterior (Agente de Comunicación Externa)	Este rol se encarga de organizar la información proporcionada por los agentes del sistema para enviarla de forma coherente hacia la aplicación externa.
Rol Gestionar log (Agente de Comunicación Externa)	Este rol tiene la función de insertar un registro para cada actividad que realizan las aplicaciones externas sobre el sistema.
Rol Planificar lección (Agente Planificador Lectivo)	El Rol Planificar lección tiene la responsabilidad de solicitar todos los datos necesarios a los agentes Estudiante, Experto, de Diagnóstico y Tutor que le permiten estructurar la lección personalizada y para lograr una retroalimentación solicita al Agente de Diagnóstico un diagnóstico oportuno del estudiante para actualizar su perfil.
Rol Gestionar estilos de aprendizaje de estudiante	Este rol es el encargado en este caso de proporcionar al

(Agente Estudiante)	Agente Planificador Lectivo las preferencias del estudiante en cuanto a los estilos de aprendizaje.
Rol Gestionar estado cognitivo de estudiante (Agente Estudiante)	Este rol es el encargado de proporcionar al Agente Planificador Lectivo el estado actual de conocimientos que posee el estudiante, es capaz además modificar el mismo al recibir un nuevo diagnóstico.
Rol Definir conocimiento a impartir (Agente Experto)	Este es el rol responsable de, a partir del estado cognitivo de un estudiante, definir qué conocimientos debe recibir y los materiales didácticos necesarios.
Rol Definir evaluación (Agente de Diagnóstico)	El Rol Definir evaluación tiene en cuenta el estado cognitivo del estudiante conjuntamente con el conocimiento que se le debe impartir para decidir cuáles evaluaciones deben aplicarse al estudiante.
Rol Definir estrategias pedagógicas (Agente Tutor)	Este rol tiene la responsabilidad de determinar las estrategias pedagógicas que se deben seguir, teniendo en cuenta las preferencias del aprendiz, para impartir cada contenido o evaluación.
Rol Diagnosticar (Agente de Diagnóstico)	Este rol se encarga de realizar un diagnóstico al estudiante en dependencia de su estado cognitivo y las respuestas dadas en las evaluaciones realizadas.

Figura 15: Diagrama identificación de roles del escenario Impartir lección. Sección Planificar.





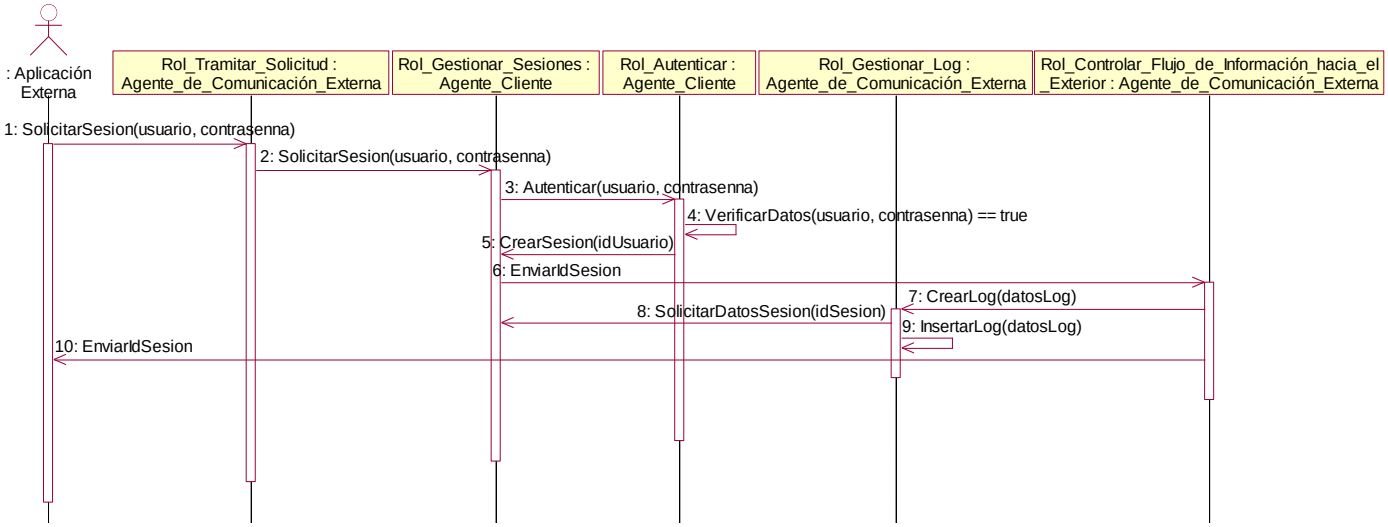
Fuente: Elaboración propia.

Tabla 5: Especificación del escenario Crear sesión.

Escenario: Crear sesión	
Descripción	Este escenario muestra el flujo normal de eventos que sigue el sistema cuando una aplicación externa se autentica y solicita la creación de una sesión para facilitar su comunicación con el sistema.
CU Asociados	CU Autenticar, CU Gestionar sesión, CU Gestionar log, CU Enviar información, CU Procesar información de agentes, CU Tramitar solicitud.
Roles Asociados	**Descripción**
Rol Autenticar (Agente Cliente)	Este rol es el encargado de verificar los datos de autenticación del usuario.
Rol Gestionar sesiones (Agente Cliente)	Este rol se responsabiliza de crear una sesión que mantenga los datos del cliente cada vez que se conecte al sistema.
Rol Tramitar solicitud (Agente de Comunicación Externa)	Este rol es responsable de atender las solicitudes realizadas desde las aplicaciones externas y tramitarla al agente más apropiado (en este caso, al Agente Cliente).
Rol Controlar flujo de información hacia el exterior (Agente de Comunicación Externa)	Este rol se encarga de organizar la información proporcionada por los agentes del sistema para enviarla de forma coherente hacia la aplicación externa.

Rol Gestionar log (Agente de Comunicación Externa)	Este rol tiene la función de insertar un registro para cada actividad que realizan las aplicaciones externas sobre el sistema.

Figura 16: Diagrama identificación de roles del escenario Crear sesión.



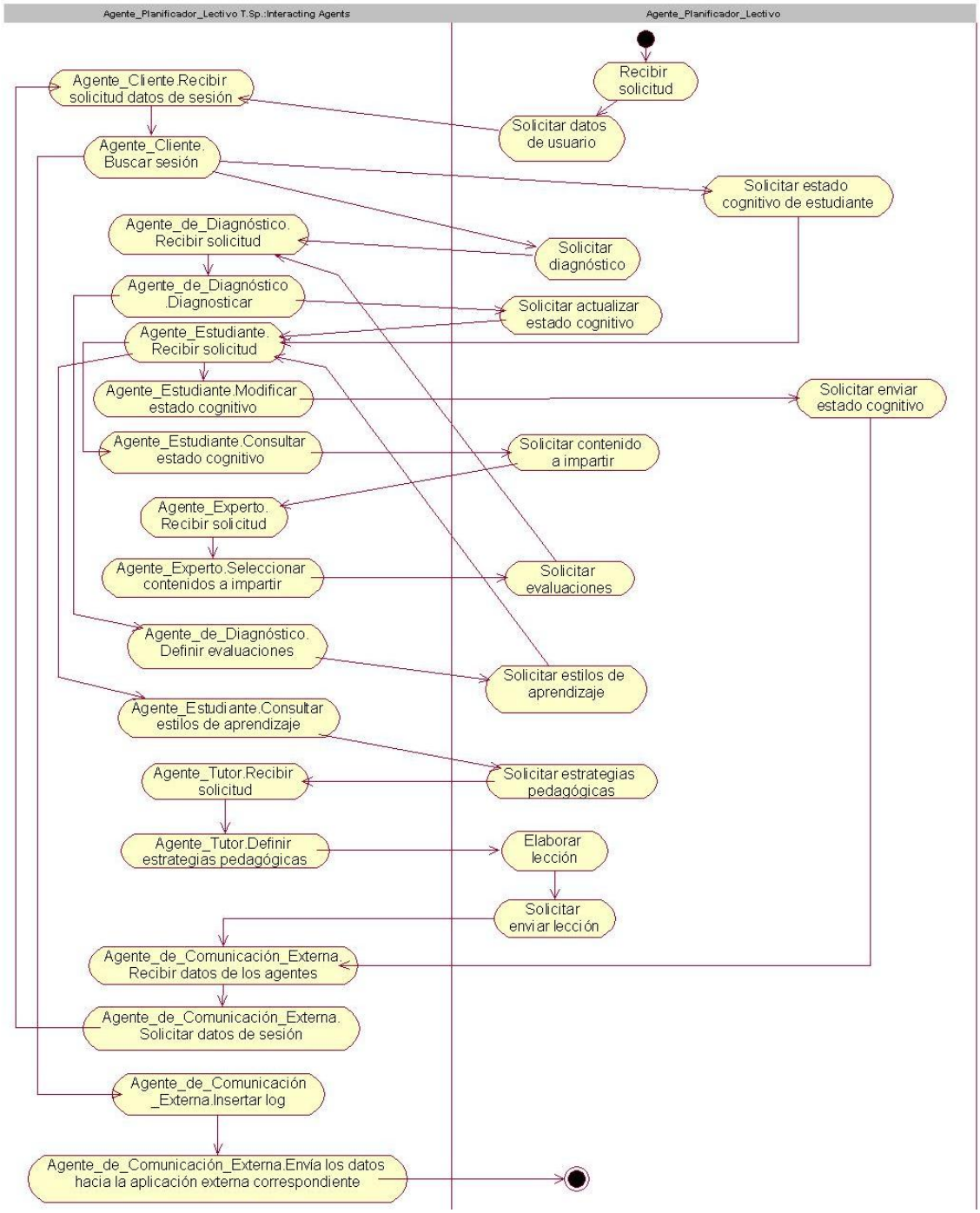
Fuente: Elaboración propia.

2.3.1.5 Especificación de tareas

Esta etapa especifica a través de Diagramas de actividad las capacidades de cada agente. Además tiene el alcance de modelar el ciclo de vida de los agentes para reflejar las colaboraciones que necesitan, y las comunicaciones en que participan. A partir del Diagrama de identificación de roles y la exploración de todos los escenarios en que participan los agentes, el análisis de todas las interacciones y acciones internas que realiza, se confecciona el Diagrama de especificación de tareas.

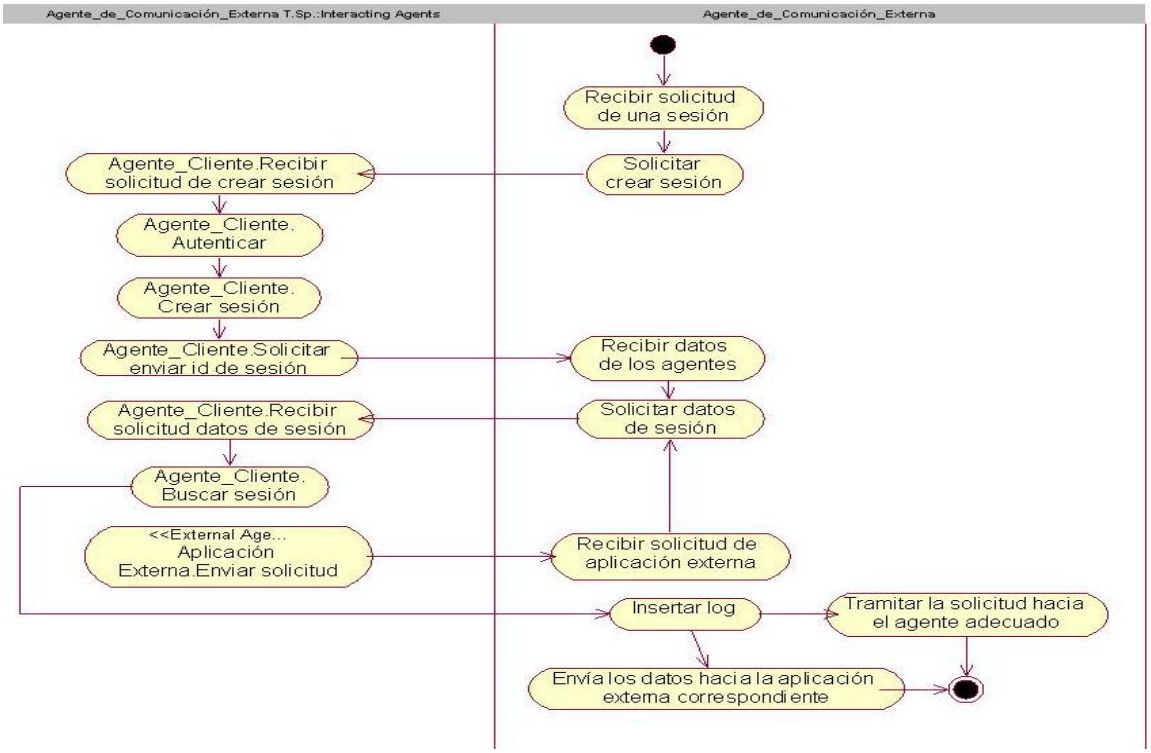
Por su importancia se muestran los Diagramas de especificación de tareas de los agentes Planificador Lectivo y el de Comunicación Externa en la Figura 17 y Figura 18 respectivamente.

Figura 17: Diagrama de especificación de tareas del Agente Planificador Lectivo.



Fuente: Elaboración propia.

Figura 18: Diagrama de especificación de tareas del Agente de Comunicación Externa.



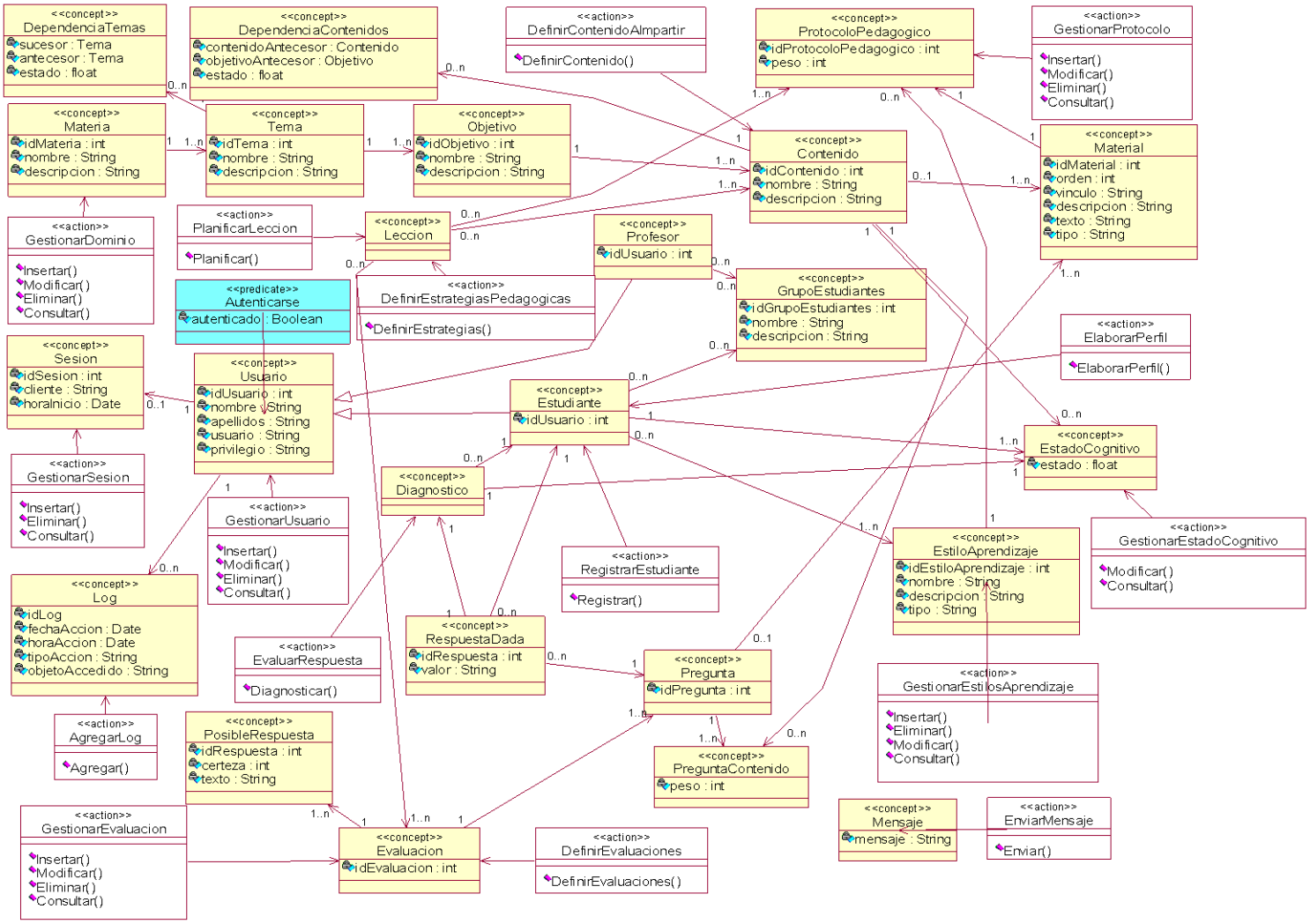
Fuente: Elaboración propia.

2.3.2 Modelo de sociedad de agentes.

2.3.2.1 Descripción de ontología

La descripción de la ontología se centra en modelar las interacciones sociales y dependencias entre los agentes involucrados en la solución del sistema. Para lograrlo se confeccionan dos diagramas: Diagrama de descripción de ontología de dominio (Figura 19) y Diagrama de descripción de ontología de comunicación. El primero representa la ontología del dominio del sistema, mientras que el segundo enfoca el conocimiento de los agentes y las relaciones comunicativas entre los mismos.

Figura 19: Diagrama de descripción de ontología de dominio.



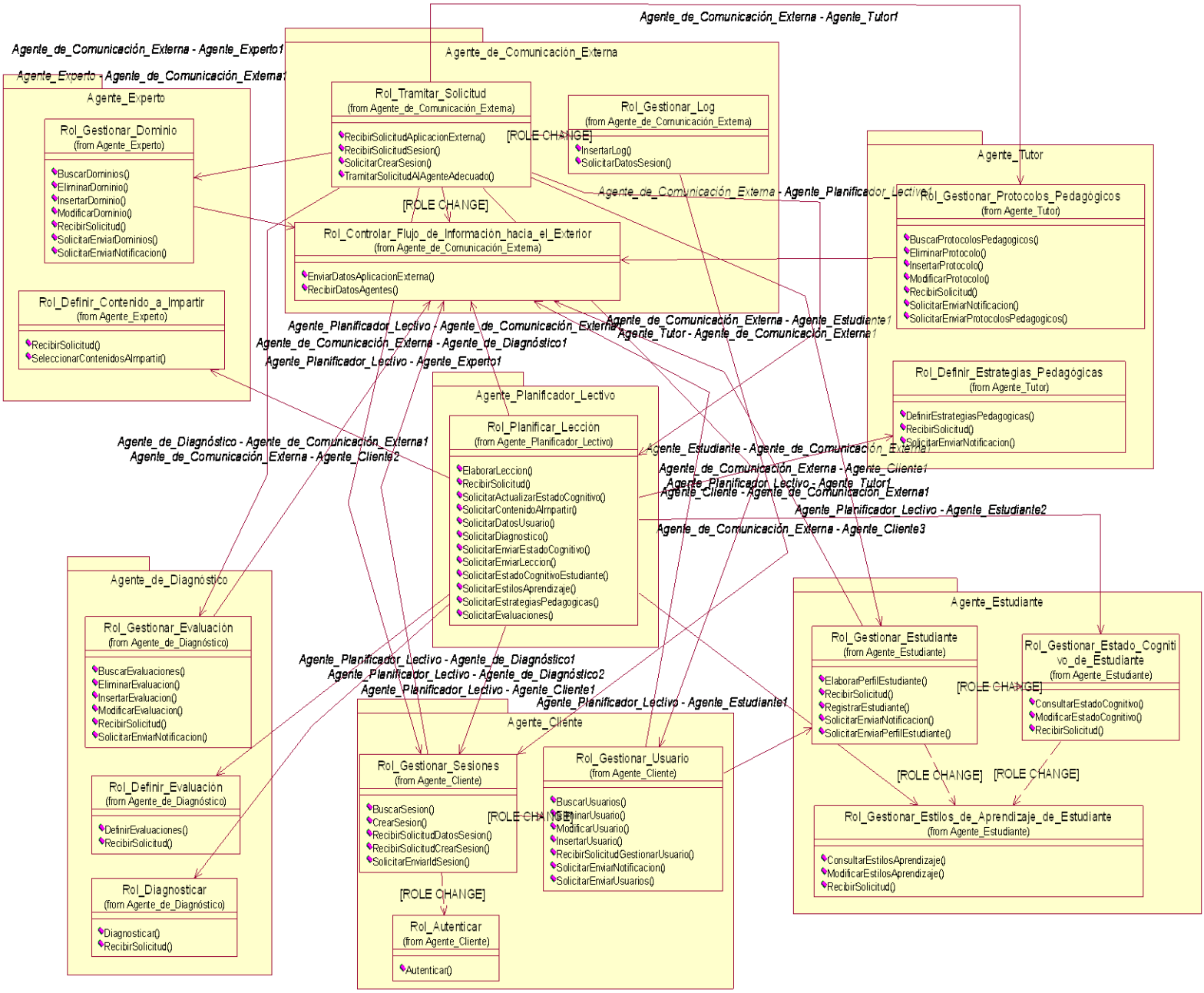
Fuente: Elaboración propia.

2.3.2.2 Descripción de roles

La descripción de roles tiene como resultado un Diagrama de paquetes. Este está compuesto por paquetes, los cuales corresponden a los agentes que fueron identificados en la fase Identificación de Agentes, y que incluyen un artefacto, clase de UML, para cada rol identificado.

En la construcción del diagrama se ha tomado como convenio representar las interacciones entre roles como asociaciones.

Figura 20: Diagrama de descripción de roles.

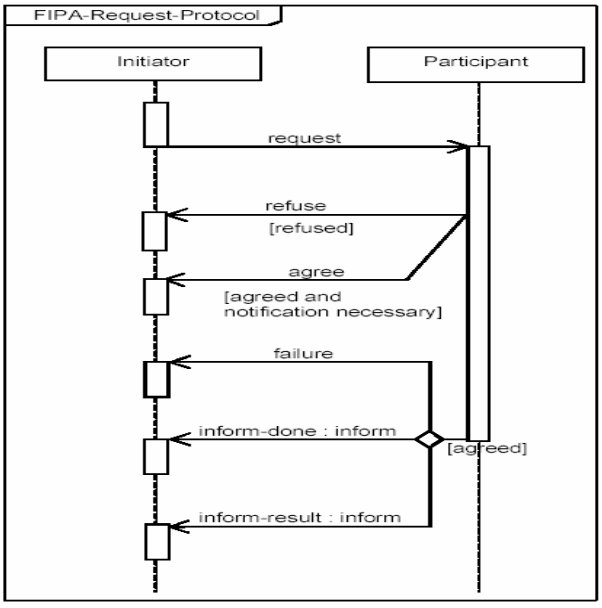


Fuente: Elaboración propia.

2.3.2.3 Descripción de protocolos

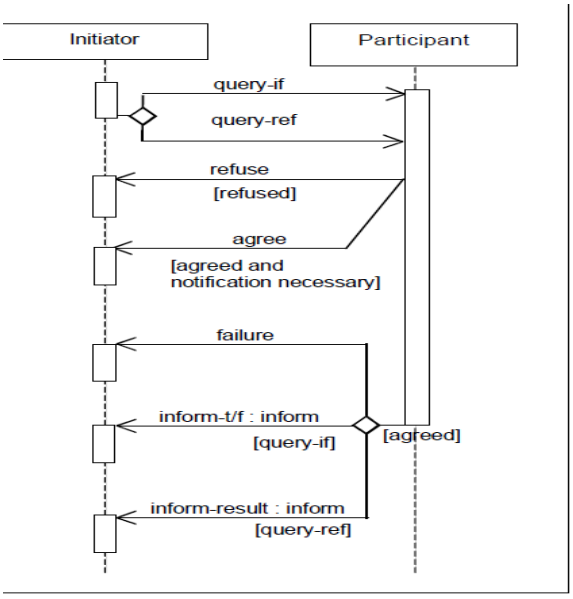
A través de Diagramas de secuencia en la fase de descripción de protocolos se detallan la gramática de cada protocolo de comunicación, de esta forma queda con mayor claridad la vía de comunicación que se establece entre los agentes. Esos diagramas se elaboraron con la utilización de AUML, la cual es una extensión de UML para agentes, por este motivo el diseñador no necesita especificarlos, pues bastaría con incorporar en el diseño los estándares propuestos por Foundation for Intelligent Physical Agents (FIPA). Este protocolo proporciona un conjunto de estándares para la construcción de Sistemas Multi-Agentes dentro de ellos se encuentran estándares para comunicación entre agentes, gestión de agentes en plataformas, coordinación entre agentes y expresión de conocimiento en cuerpo de mensajes. Específicamente se usaron los protocolos FIPARequest (Figura 21) y FIPAQuery (Figura 22). El primero de ellos permite a un agente solicitar un servicio y/o recurso a otro para satisfacer una necesidad determinada. El participante procesa la demanda, y determina si está en condiciones de satisfacerla o no. El segundo (FIPA Query) es similar al anterior solo que la acción a ejecutar es de un tipo concreto: *informe* para informar sobre algo. A continuación se presentan los diagramas de secuencia correspondientes a estos protocolos:

Figura 21: Protocolo FIPARequest.



Fuente: Tomado de (FIPA Request, 2002).

Figura 22: Protocolo FIPAQuery.



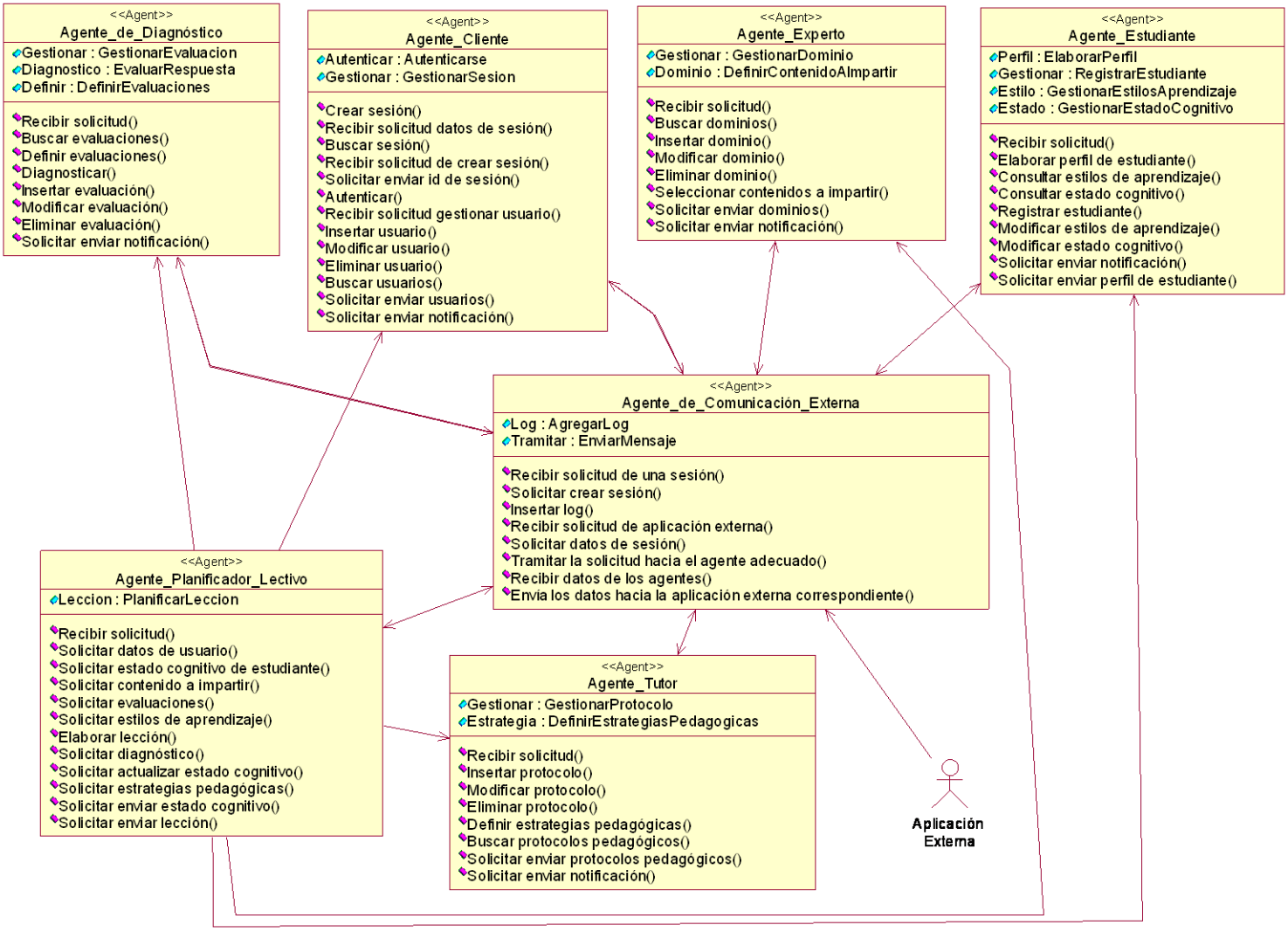
Fuente: Tomado de (FIPA Query, 2002).

2.3.3 Modelo de implementación de agentes

2.3.3.1 Definición de estructura del sistema multi-agentes

En esta etapa se representa la estructura del SMA en conjunto mediante un Diagrama de clases, donde cada clase simboliza uno de los agentes identificados en la fase Identificación de Agentes. El conocimiento de cada agente puede representarse mediante atributos considerando el uso de operaciones para la identificación de las tareas del agente.

Figura 23: Diagrama de definición de estructura del sistema multi-agentes.



Fuente: Elaboración propia.

2.3.3.2 Definición de estructura del agente

En esta fase se ilustra la estructura interior del agente a través de las clases que lo constituyen, o sea, la clase principal del agente y las clases internas que identifican sus tareas. Se utiliza un Diagrama de clase para cada agente, con sus métodos y atributos, y las clases responsables de las tareas.

Por lo extenso de estos diagramas no se muestran en el informe, pero se pueden encontrar en los artefactos generados por la investigación y adjuntos a este trabajo.

2.3.4 Modelo de código

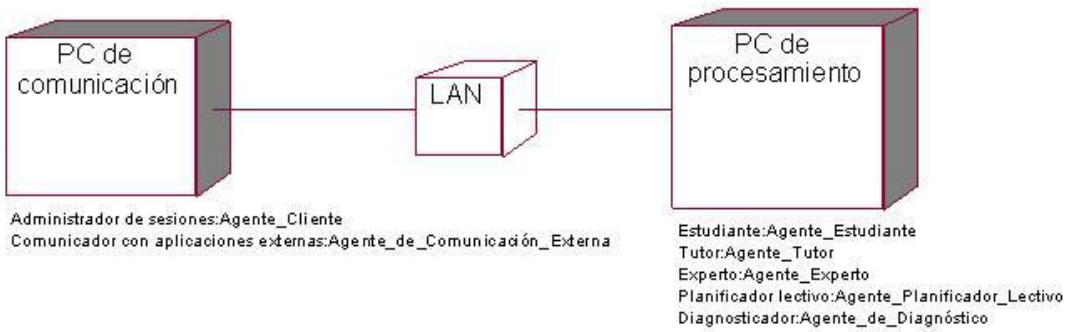
Esta etapa de la metodología no se corresponde con los objetivos trazados para este capítulo, pues la implementación del sistema en cuestión no está concebida como parte de esta fase de la investigación, por lo que ha sido recomendado desarrollarla en posteriores trabajos.

2.3.5 Modelo de despliegue

2.3.5.1 Configuración de despliegue

Esta etapa describe la asignación de agentes a las unidades físicas de procesamiento disponibles y los modos de comunicación entre agentes en unidades de procesamiento diferentes como se muestra en la Figura 24.

Figura 24: Diagrama de configuración de despliegue.

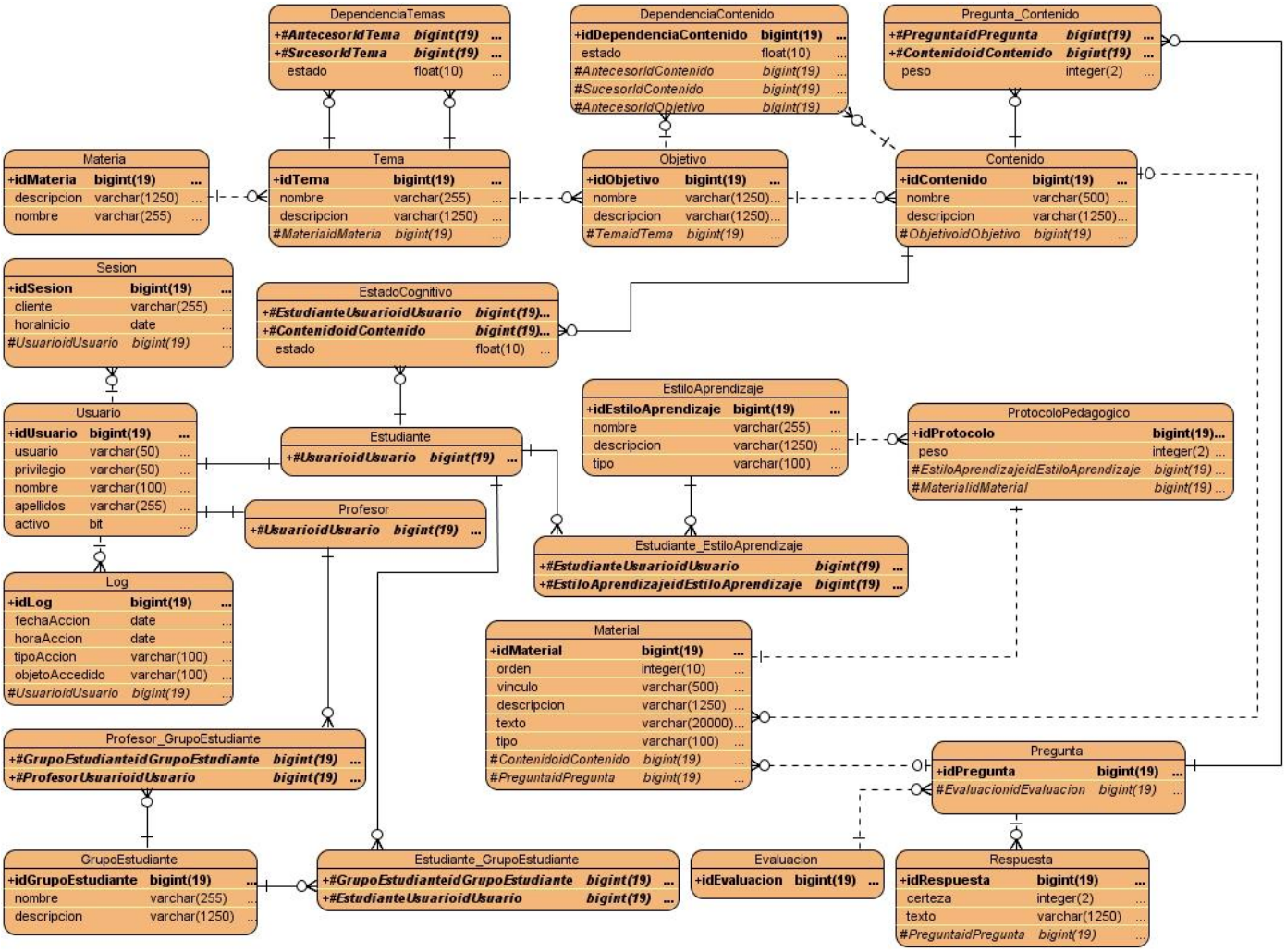


Fuente: Elaboración propia.

2.4 Modelo de datos

Un elemento importante dentro del sistema es la base de datos, de donde los agentes adquieren toda la información para poder tomar las mejores decisiones rápidamente. En la Figura 25 se muestra el modelo de datos del STI. Gracias a la utilización de la herramienta de modelado Visual Paradigm la generación y sincronización de la base de datos será automática, al igual que la generación de las clases persistentes en java y los archivos XML de mapeo que usa el framework Hibernate.

Figura 25: Modelo de datos.



Fuente: Elaboración propia.

2.5 Análisis de los resultados obtenidos

Una vez terminada la investigación es de vital importancia realizar un estudio que permita evaluar el nivel de relevancia que representa según el problema existente y el aporte que brinda a la solución del mismo.

2.5.1 Validación por listas de chequeo

Al aplicar la metodología PASSI para modelar el sistema multi-agentes se genera un gran número de artefactos de vital importancia. Para la validación de los mismos se aplicó una lista de chequeo (ver Anexo

3) para cada una de sus fases con las exigencias de esta metodología. Las preguntas se agruparon bajo una categoría relativa a una característica de calidad. En cada una de las revisiones fueron corregidas las no conformidades hasta que los artefactos generados al aplicar la metodología PASSI satisficieron cada una de las características de calidad contenidas en la lista.

2.5.2 Método de validación de expertos

Para conocer cuan eficaz ha sido el trabajo realizado se procede a la validación por el método de expertos, con el objetivo primordial de conocer el criterio de personalidades y especialistas en este campo. En este caso para realizar la validación mediante expertos se decidió realizar un análisis a partir de la puesta en práctica del Método Delphi, el cual fue desarrollado por Olaf Helmer a mediados de la década de 1960, es considerado un método de investigación sociológica, que independientemente de que pertenece al tipo de entrevista de profundidad en grupo, se aparta de ellas agregando características particulares. Es una técnica grupal de análisis de opinión, parte de un supuesto fundamental y de que el criterio de un individuo particular es menos fiable que el de un grupo de personas en igualdad de condiciones, en general utiliza e investiga la opinión de expertos (Ruiz, 1989).

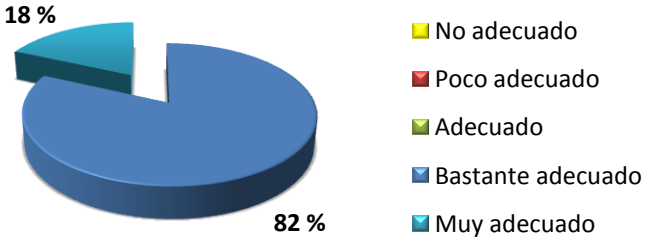
Dándole cumplimiento a los aspectos básicos desde el punto de vista metodológico del Método Delphi, se procede a la selección de los expertos y a la creación de las encuestas.

Para la selección de los expertos a encuestar se localizaron personas conocedoras en Informática Educativa, con reconocida competencia y experiencia en el tema de forma que se garantice la confiabilidad de los resultados. Se aplicó la metodología elaborada por el Comité Estatal para la Ciencia y la Técnica de Rusia para seleccionar expertos en un tema, efectuándoles preguntas de autoevaluación para conocer su Coeficiente de Conocimiento y de Argumentación y realizar el cálculo del Coeficiente de Competencia. Sólo se escogieron aquellos expertos cuyo Coeficiente de Competencia es alto o medio. De los 7 expertos que realizaron la encuesta 5 de ellos presentan el grado científico de Doctores y 2 la categoría académica de Máster; con importantes publicaciones nacionales e internacionales sobre el tema. Son además profesionales prestigiosos de universidades cubanas.

La elaboración del cuestionario se realizó usando preguntas claras, precisas e independientes. Cuenta con un total de 11 preguntas dirigidas a evaluar los criterios de mérito científico, implantación, generalización e impacto de la investigación desarrollada y permite dar una evaluación final de bueno, excelente, aceptable, cuestionable o malo.

Una vez analizadas las evaluaciones de cada criterio de las encuestas (ver Anexo 4) y tras la aplicación de la función de distribución normal estándar invertida se obtuvo como resultado que 9 de los 11 criterios se evaluaron de “Bastante adecuado” y 2 de “Muy adecuado” como se muestra en la Figura 26. Vistos estos elementos se considera que el criterio de los expertos converge a que el trabajo realizado es “Bastante adecuado”.

Figura 26: Evaluación final de los criterios de expertos



Fuente: Elaboración propia.

2.6 Conclusiones

La arquitectura propuesta, conformada por cuatro módulos, está diseñada para garantizar flexibilidad en el desarrollo de Sistemas Tutores Inteligentes y posibilita su adaptación a diferentes dominios. Los artefactos generados usando la metodología PASSI en el modelado de la arquitectura del STI propuesto permitirán obtener mayor claridad y entendimiento para los desarrolladores en la fase de implementación del sistema.

Con la aplicación de la lista de chequeos a los artefactos generados y la corrección oportuna de los errores encontrados se garantizó que todos los artefactos cumplen las normativas que plantea la metodología PASSI.

Conclusiones

Tras el estudio realizado en la investigación y la puesta en práctica para el desarrollo de la misma se arribó a las siguientes conclusiones:

- Tras el estudio de las técnicas de IA y los STI existentes se determinó que el modelo basado en agentes con el uso de la Inteligencia Artificial Distribuida proporciona flexibilidad en la construcción de Sistemas Tutores Inteligentes.
- El uso de la metodología PASSI, modelada en la herramienta PASSI ToolKit, en conjunto con la plataforma de agentes y el framework JADE facilita el desarrollo de sistemas multi-agentes complejos.
- La arquitectura propuesta garantiza la construcción de STI con la posibilidad de adaptarse y expandirse a nuevos dominios.
- El procesamiento de los artefactos de validación y el análisis de los resultados obtenidos en las encuestas realizadas a expertos convergen al carácter ingenioso y la calidad de la investigación.

Recomendaciones

Se recomienda que a partir de lo desarrollado en el presente trabajo, se proceda con la implementación del Sistema Tutor Inteligente Multi-Agente Genérico y una vez implementado se adicionen diferentes dominios para su utilización en el proceso de enseñanza-aprendizaje. Se debe profundizar en las técnicas de la IA que pueden usarse en la implementación de las tareas de los agentes inteligentes.

Referencias bibliográficas

AGUILAR, María; PEÑA, Clara I.; FABREGAT, Ramón (2005) "SMIT: un agente sintético antropomórfico para un entorno virtual de aprendizaje" [en línea]. Universidad de Girona, Departamento de electrónica, Informática y Automática. Abril 2005. [Fecha de consulta: 13/11/2009].
Disponible en: http://eia.udg.es/~clarenes/pdfs/publicaciones/maagpa-cameraready.pdf

BARR, A.; FEIGENBAUN, E. A. (1981). The Handbook of Artificial Intelligence. Volumen I. William Kaufman, Los Altos, CA.

BELLMAN, Richard E. (1978). An Introduction to Artificial Intelligence: Can Computers Think? Boyd & Fraser Publishing Company. 1978.

BEUTELSPACHER, M.; FRANZONI, A. L. y MORALES, A. (1995). *Sistema de Apoyo Generalizado para la Enseñanza Individualizada (SAGE)*. Tesis de grado (Ingeniería en computación). Ciudad México, México: Instituto Tecnológico Autónomo de México, 1995.

BOLAN FRIGO, L.; POZZEBON, E.; BITTENCOURT, G. (2004). O Papel dos Agentes Inteligentes nos Sistemas Tutores Inteligentes. En: World Congress on Engineering and Technology Education (WCETE' 2004). Guarujá, Santos, SP. Proceedings of the World Congress on Engineering and Technology Education.

BUCHANAN, B. G.; SHORTLIFFE, E. H. (1984). Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project. Addison-Weslwy, Reading, MA.

BUCHANAN, B. G.; SUTHERLAND, G. L.; FEIGENBAUM, E. A. (1969). Heuristic DENDRAL: a program for generating explanatory hypotheses in organic chemistry. In Meltzer, B., Michie, D., and Swann, M., editors, Machine Intelligence 4, p. 209-254. Edinburgh University Press, Edinburgh, Scotland.

BURNS, H. L.; CAPPS, C.G. (1988). "Foundations of STI: An Introduction", en Polson, M. C, y Richardson, J. J. (editores), Foundations of Intelligent Tutoring Systems, Lawrence Erlbaum Associates, Estados Unidos, 1988.

CARBONELL, J. R. (1970). AI in CAI: An artificial intelligence approach to computer assisted instruction. IEEE transaction on Man Machine System. V11 n.4, p 190-202.
ISSN: 0536-1540

CASALI, Ana. (2010). Integración de redes bayesianas y razonamiento basado en casos, en sistemas de agentes. 10 de febrero de 2010.
Disponible en: http://www.iiia.csic.es/~puyol/SEIAD2001/Treballs/AnaCasali/IntegracionCBR-BN.doc.pdf

CASARES, JUAN P. (1999). *AMIVA: Ambiente para la instrucción visual de algoritmos.* Tesis de grado (Ingeniería en computación) [en línea]. Ciudad México, México: Instituto Tecnológico Autónomo de México , Julio 1999. 162 p. [Fecha de consulta: 15/11/2009].
Disponible en: http://usablehack.com/amiva/AMIVA.pdf

CATALDI, Zulma; CALVO, Patricia; SALGUEIRO, Fernando A.; LAGE, Fernando J. (2007) "Diseño de Sistemas Tutores Inteligentes con Tecnología de Agentes: Los Agentes Docentes en el Módulo Tutor" [artículo en línea]. RESI, Revista Electrónica de Sistemas de Información. Núm. 10/ 2007. [Fecha de consulta: 13/11/2009].
Disponible en: http://revistas.facecla.com.br/index.php/reinfo/article/viewFile/191/99
ISSN 1677-3071.

CATALDI, Zulma; LAGE, Fernando J. (2009) "Sistemas tutores inteligentes orientados a la enseñanza para la comprensión" [artículo en línea]. EDUTEC, Revista Electrónica de Tecnología Educativa. Núm. 28/ Marzo 2009. [Fecha de consulta: 30/10/2009].
Disponible en: http://edutec.rediris.es/revelec2/revelec28/
ISSN 1135-9250.

CATALDI, Zulma; SALGUEIRO, Fernando; LAGE, Fernando J. (2006) "Sistemas tutoriales multiagentes con modelado del estudiante y del autor" [artículo en línea]. EDUTEC, Revista Electrónica de Tecnología Educativa. Núm. 20/ Marzo 2006. [Fecha de consulta: 30/10/2009].
Disponible en: http://edutec.rediris.es/Revelec2/revelec20/
ISSN 1135-9250.

DUDA, R. O.; GASCHINIG, J. G.; HART, P. E. (1980). Model Design in the Prospector Consultant System for Mineral Exploration. Michie, D. Expert Systems in the Microelectronic Age. Edinburgh University Press. Edinburgh, p. 153-167.

FELGAER, Pablo Ezequiel. (2005). *Optimización de redes bayesianas basado en técnicas de aprendizaje por inducción.* Tesis de grado (Ingeniería informática) [en línea]. Buenos Aires, Argentina: Universidad de Buenos Aires, Febrero 2005. 157 p. [Fecha de consulta: 15/11/2009].
Disponible en: http://tesis.uci.cu/document/tesisDoct/2.pdf

FERNANDEZ, Raúl R.; SERVER, Pedro Marino; CARBALLO, Elme (2006) "Aprendizaje con nuevas tecnologías paradigma emergente. ¿Nuevas modalidades de aprendizaje?" [artículo en línea]. EDUTEC, Revista Electrónica de Tecnología Educativa. Núm. 20/ Enero 2006. [Fecha de consulta: 30/10/2009].
Disponible en: http://edutec.rediris.es/Revelec2/revelec20/
ISSN 1135-9250.

FIPA Query Interaction Protocol Specification (2002). Foundation for Intelligent Physical Agents, 2002.
Disponible en: http://www.fipa.org/specs/fipa00027/SC00027H.pdf

FIPA Request Interaction Protocol Specification (2002). Foundation for Intelligent Physical Agents, 2002.
Disponible en: http://www.fipa.org/specs/fipa00026/SC00026H.pdf

FRIENDBERG, R. M.; DUNHAM, B.; NORTH, T. (1959). A learning machine: Parte 2. IBM Journal of Research and Development, 3 (3): 282-287

GARCÍA, Ana; OSSOWSKY, Sascha. 2010. Inteligencia Artificial Distribuida y Sistemas Multiagentes. 10 de febrero de 2010.
Disponible en: http://www.dia.fi.upm.es/~agarcia/publications/archivos/REV3.pdf

GÓMEZ, Jorge J. (2003). "Metodologías para el desarrollo de sistemas multi-agentes" [artículo en línea]. Inteligencia Artificial: Revista Iberoamericana de Inteligencia Artificial. Núm.18/2003, pp. 51-63. [Fecha de consulta: 11/02/2010].
Disponible en: http://www.cdainfo.com/down/1-Desarrollo/multiagentes.pdf
ISSN: 1137-3601

GUARDIA ROBLES, B. (1993). *Asesores Inteligentes para apoyar al proceso de enseñanza de lenguajes de programación.* Tesis de grado. México: Instituto Tecnológico de Monterrey (ITESM).

GUTIERREZ, Ana; LLARCH, Silvia M. (2008). *Especificación de un Sistema Multi-Agente para la Gestión de Ventas Personalizadas.* Tesis de grado (Ingeniería en Ciencias Informáticas). Ciudad de La Habana, Cuba: Universidad de las Ciencias Informáticas, 2008.

HEBB, D. O. (1949). The Organitation of Behavior. NY: Jhon Wiley & Sons, 1949. 335 p.
ISBN 0-8058-4300-0

HUNT, E.B.; MARIN, J.; STONE, P.J. (1966). Experiments in Induction. Nueva York, Estados Unidos: Academic Press, 1966.

IGLESIAS FERNÁNDEZ, Carlos Ángel (1998). *Definición de una metodología para el desarrollo de sistemas Multiagente.* Tesis doctoral [en línea]. Madrid, España: Universidad Politécnica de Madrid, Departamento de Ingeniería de Sistemas Telemáticos, 1998. 322 p. [Fecha de consulta: 15/11/2009].
Disponible en: http://www.gsi.dit.upm.es/tesis/pdf/tesiscif.pdf

KOHONEN, T. (2001). Self-Organizing Maps, 1ra ed. Springer series in informarion sciences. Helsinki University of Technology Neural Networks Research Centre 286-310. Pitman, London.

KOLODNER, J. (1993). Case-Based Reasoning. San Mateo: Morgan Kaufmann.

MARTÍNEZ, Natalia; GARCÍA, María M.; GARCÍA, Zoila Z.; FERREIRA, Gheisa. (2009) "El paradigma del razonamiento basado en casos en el ámbito de los sistemas de enseñanza/aprendizaje inteligentes" [artículo en línea]. EDUTEC, Revista Electrónica de Tecnología Educativa. Núm. 30/ Noviembre 2009. [Fecha de consulta: 10/02/2010].
Disponible en:
http://edutec.rediris.es/Revelec2/revelec30/articulos_n30_pdf/Edutec-e30_Martinez_Garcia_Ferreira.pdf
ISSN 1135-9250.

MINSKY, M. L. (1954). *Neural Nets and the Brain-Model Problem (SNARC).* Tesis de doctorado. Nueva Jersey, Estados Unidos: Universidad de Princeton. 1954.

OVALLE, Demetrio A.; JIMÉNEZ, Jovani A. (2007) "Mecanismo de planificación instruccional usando razonamiento basado en casos en ambientes inteligentes distribuidos de aprendizaje" [en línea]. Colombia: Universidad Nacional de Colombia, sede Medellín. 2007. [Fecha de consulta: 15/11/2009].
Disponible en: http://www.colombiaaprende.edu.co/html/mediateca/1607/articles-108458_archivo.pdf

OVALLE, Demetrio A.; JIMÉNEZ, Jovani A. (2005) "Sistemas de Enseñanza / Aprendizaje basados en Agentes Inteligentes Pedagógicos" [en línea]. Colombia: Universidad Nacional de Colombia, sede Medellín. Noviembre 2005. [Fecha de consulta: 15/11/2009].
Disponible en: http://pisis.unalmed.edu.co/avances/archivos/ediciones/2005/ovalle_etal05.pdf

PEÑA, Clara I.; MARZO, Jose L.; DE LA ROSA, Josep Ll.; FABREGAT, Ramón (2002). Un sistema de tutoría inteligente adaptativo considerando estilos de aprendizaje. En: Congreso Iberoamericano, 4 Simposio Internacional de Informática Educativa, 7 Taller Internacional de Software Educativo (6^0: 2002: Galicia: España) [en línea]. [Fecha de consulta: 30/11/2009].
Disponible en: http://www.scientificcommons.org/839207

Pérez Pinto, Rolando (2006). *Evaluación de la metodología PASSI en un caso de estudio*. Tesis de grado (Ingeniería Informática). Ciudad de La Habana, Cuba: Instituto Superior Politécnico "José Antonio Echeverría", Facultad de Ingeniería Industrial, 2006.

RUIZ OLABUÉNAGA, J. E (1989). La técnica Delphi. La descodificación de la vida cotidiana. Métodos de investigación cualitativa. 1989, 171-179 p.

RUSSELL, S. J. y NORVIG, P. (2003). Artificial Intelligence: A Modern Approach. 2da ed. Prentice Hall, 2003.

ROSÉS A., Francesc. (2004). Home Page. Abril 2004.
Disponible en: http://www.froses.com/Assets/Files/Articles/Hibernate_Introduccion_es.pdf

SALGUEIRO, Fernando (2005). *Sistemas Inteligentes para el Modelado del Tutor.* Tesis de grado (Ingeniería Informática) [en línea]. Buenos Aires, Argentina: Universidad de Buenos Aires, Facultad de Ingeniería, 2005. 195 p. [Fecha de consulta: 03/11/2009].
Disponible en: http://laboratorios.fi.uba.ar/lsi/z-tesis.htm

SLAGLE, J. R. (1963). A heuristic program that solves symbolic integration problems in freshman calculus. Journal of the Association for Computing Machinery. Feldman. 1963.

STUART, J. R.; NORVIG, P.; CANNY, J. F.; MALIK, J. M.; DOUGLAS, D. E. (1995). "Artificial Intelligence A Modern Approach". Prentice Hall, Englewood Cliffs, New Jersey 07632.

TURING, Alan. (1950). Computing machinery and intelligence. Mind, vol. 59. No 236, p.433-460.

VANLEHN, K. (1988). "Student Modeling". M. Polson. Foundations of Intelligent Tutoring systems. Hillsdale. N.J. Lawrence Erlbaum Associates, pp 55-78

VILLAREAL FARAH, G. (2003) "Agentes Inteligentes en educación" [artículo en línea]. EDUTEC, Revista Electrónica de Tecnología Educativa. Núm. 26/ Abril 2003. [Fecha de consulta: 15/11/2009].
Disponible en: http://edutec.rediris.es/Revelec2/revelec16/villarreal.pdf
ISSN 1135-9250.

WENGER, Etienne. (1987). "Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge". 1a. ed. Estados Unidos: Morgan Kaufmann, 1987. 486p.
ISBN: 0934613265.

WOLF, B. (1984). "Context Dependent Planning in a Machine Tutor". Ph.D. Dissertation, University of Massachusetts, Amherst, Massachusetts.

Bibliografía

GELERNTER, H. (1959). Realization of the geometry theorem proving machine. En: Conferencia Internacional sobre Procesamiento de la Información (1959: UNESCO: Paris) pp. 273-282.

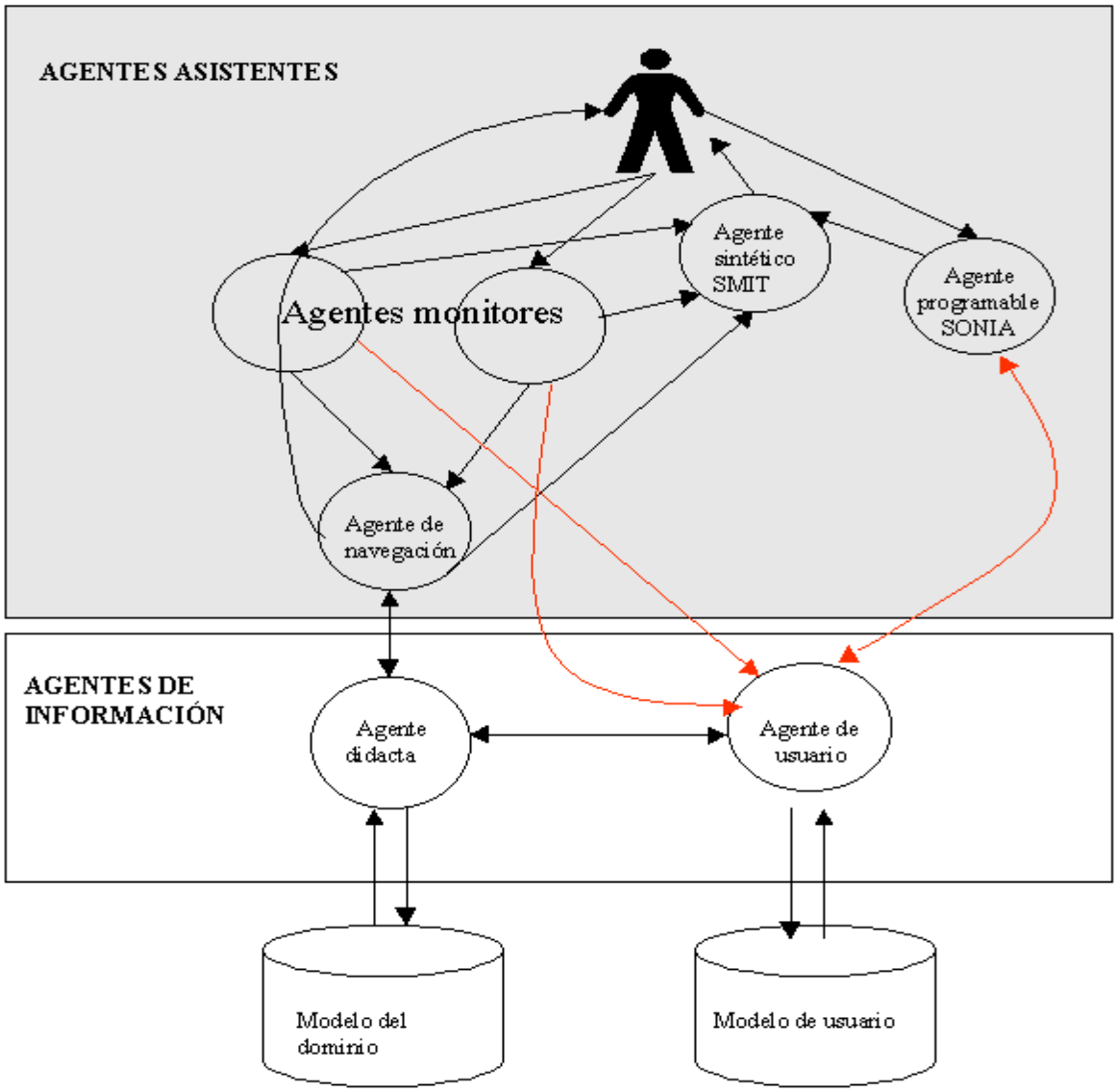
HUAPAYA, Constanza Raquel. (2009). *Sistemas Tutoriales Inteligentes. Un análisis crítico.* Tesis de postgrado [en línea]. Argentina: Universidad Nacional de La Plata, Facultad de Informática, 2009. 74 p. [Fecha de consulta: 02/02/2010].
Disponible en:
http://postgrado.info.unlp.edu.ar/Carrera/Especializaciones/Tecnologia%20Informatica%20Aplicada%20en%20Educacion/Trabajo%20Final%20Integrador/Huapaya.pdf

IGLESIAS, C. A.; GARIJO, C.; GONZALEZ, J. (2001) "Metodologías orientadas a agentes". Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial. Vol. 2. Núm. 6. p 12-23.
ISSN: 1137-3601.

MORENO, Mailyn; FERNÁNDEZ, Zaili R.; LORENZO, Mary L.; ROSETE, Alejandro; DELGADO, Martha D.; HADFEG, Yahima. (2007). "Integración de metodologías orientadas a agentes." Revista cubana de ciencias informáticas. Vol. 1. Núm. 4/Diciembre 2007. p 4-19.
ISSN: 1994-1536

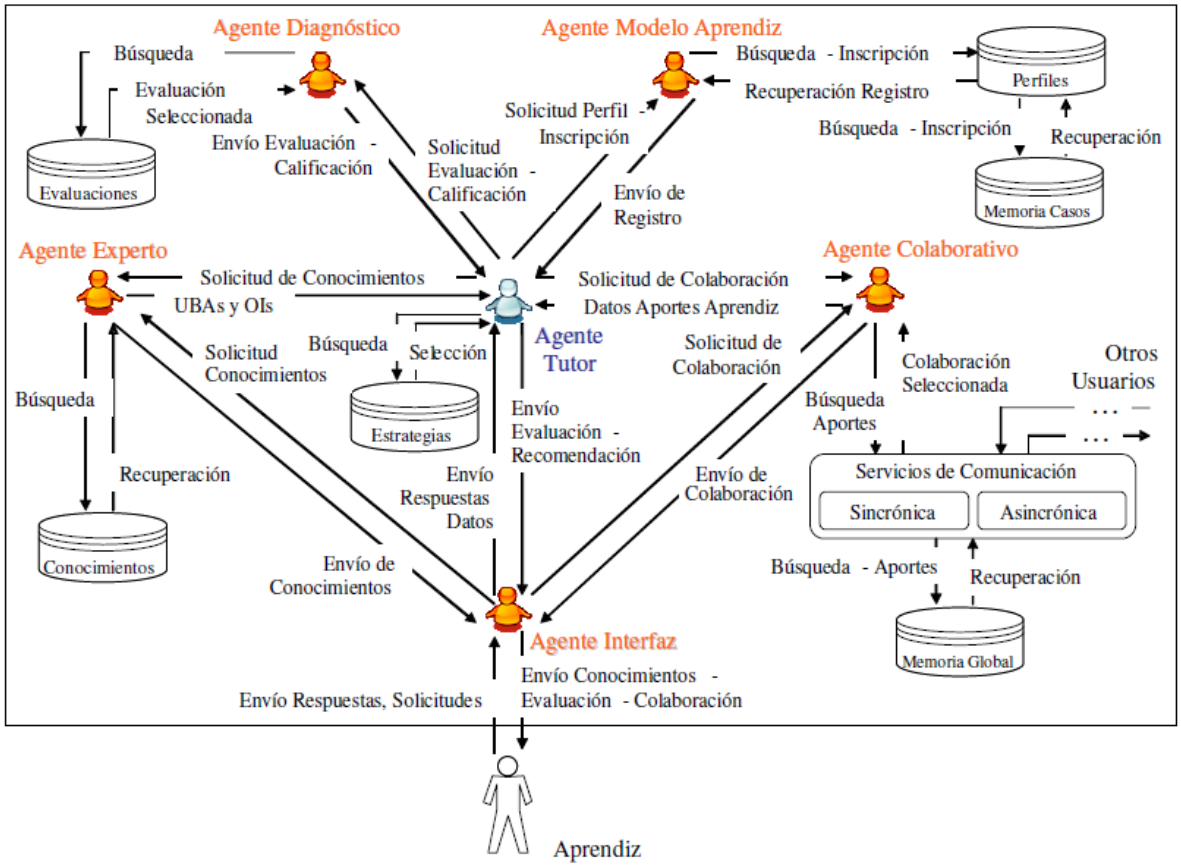
Anexos

Anexo 1: Arquitectura del Sistema Multi-Agente MAS-PLANG.



Fuente: Tomado de (Peña y colegas, 2002).

Anexo 2: Arquitectura del Sistema Multi-Agente ALLEGRO.



Fuente: Tomado de (Ovalle y Jiménez, 2007).

Anexo 3: Lista de chequeo

Modelo de Requerimientos del Sistema

Claridad
¿En la Descripción de Entorno están representados los actores internos y externos y sus relaciones con el sistema?
¿Todos los casos de uso y agentes identificados tienen un nombre sugerente y entendible?
¿Las relaciones entre los CU y actores del Diagrama de Descripción de Dominio satisfacen la lógica del sistema?
¿Fueron lógicamente agrupados los CU en la identificación de los agentes?
¿Los roles identificados para cada agente tienen correspondencia con las funcionalidades descritas del agente?
¿La secuencia de los mensajes entre los roles y actores en cada escenario presentan un orden lógico?
Completitud
¿Están abarcadas todas las funcionalidades del sistema en los CU?
¿Todos los Casos de Uso están identificados con un agente?
¿En los diagramas de secuencia de los escenarios identificados se modela el ciclo de vida completo de cada agente?
¿Se confeccionó un Diagrama de Actividades para cada agente?
Consistencia
¿Todos las relaciones entre los CU de un mismo agente están representadas en el Diagrama de Identificación de Agentes?
¿Todas las relaciones entre los CU de distintos agentes están representadas como comunicaciones en el Diagrama de Identificación de Agentes?
¿Cada agente tiene identificadas las tareas que le permiten alcanzar el propósito de los escenarios en que está involucrado?

Modelo de Sociedades de Agentes

Claridad
¿Las ontologías identificadas presentan una estructura lógica y coherente?
Completitud
¿Las ontologías identificadas representan todo el conocimiento manejado por los agentes?
¿Todas las comunicaciones entre los agentes en el Diagrama de Descripción de Ontología de Comunicación tienen identificados la ontología, el lenguaje y protocolo?
Consistencia
¿Cada agente tiene identificada al menos una ontología?
¿Existe correspondencia entre las comunicaciones que representan entre los diagramas de Descripción de Ontología de Comunicación, de Descripción de Roles, de Identificación de Agentes y de Identificación de Roles?
¿Se corresponden los roles por agente en la descripción de roles con los roles identificados en la fase de Identificación de roles?
¿Se corresponden los métodos de los agentes con las actividades identificadas a cada uno de ellos?
¿Se utilizan protocolos estándares en las comunicaciones entre los agentes, y en caso de que se cree un nuevo protocolo éste es descrito?

Modelo de Implementación de Agentes

Completitud
¿Están representados todos los agentes con sus respectivas tareas?
Consistencia
¿El diagrama de Estructura del Sistema Multi-Agente y agentes individuales muestra representación acertada que constituye el punto de partida para la futura implementación?

Modelo de Despliegue

Completitud
¿Todos los agentes están ubicados en alguna unidad de procesamiento?
Consistencia
¿Se especifica el medio a través del cual se comunican las unidades de procesamiento?

Anexo 4: Análisis de los resultados de la validación de expertos.

Tabla de frecuencias absolutas:							
No	Elementos	C1	C2	C3	C4	C5	Total
1	E1	0	7	0	0	0	7
2	E2	0	7	0	0	0	7
3	E3	0	6	1	0	0	7
4	E4	5	2	0	0	0	7
5	E5	0	6	1	0	0	7
6	E6	1	5	1	0	0	7
7	E7	2	3	2	0	0	7
8	E8	2	3	2	0	0	7
9	E9	3	4	0	0	0	7
10	E10	2	5	0	0	0	7
11	E11	1	6	0	0	0	7

Tabla de frecuencias absolutas acumuladas:						
No	Elementos	C1	C2	C3	C4	C5
1	E1	0	7	7	7	7
2	E2	0	7	7	7	7
3	E3	0	6	7	7	7
4	E4	5	7	7	7	7
5	E5	0	6	7	7	7
6	E6	1	6	7	7	7
7	E7	2	5	7	7	7
8	E8	2	5	7	7	7
9	E9	3	7	7	7	7
10	E10	2	7	7	7	7
11	E11	1	7	7	7	7

Tabla de frecuencias relativas acumuladas:						
No	Elementos	C1	C2	C3	C4	C5
1	E1	0,0001	0,9999	0,9999	0,9999	0,9999
2	E2	0,0001	0,9999	0,9999	0,9999	0,9999
3	E3	0,0001	0,86	0,9999	0,9999	0,9999
4	E4	0,71	0,9999	0,9999	0,9999	0,9999
5	E5	0,0001	0,86	0,9999	0,9999	0,9999
6	E6	0,14	0,86	0,9999	0,9999	0,9999
7	E7	0,29	0,71	0,9999	0,9999	0,9999
8	E8	0,29	0,71	0,9999	0,9999	0,9999
9	E9	0,43	0,9999	0,9999	0,9999	0,9999
10	E10	0,29	0,9999	0,9999	0,9999	0,9999
11	E11	0,14	0,9999	0,9999	0,9999	0,9999

Puntos de corte:									
No	Aspectos	C1	C2	C3	C4	Suma	P	N-P	
1	I1	-3,72	3,72	3,72	3,72	7,44	1,86	-0,22	Bastante Adecuado
2	I2	-3,72	3,72	3,72	3,72	7,44	1,86	-0,22	Muy Adecuado
3	I3	-3,72	1,07	3,72	3,72	4,79	1,20	0,44	Bastante Adecuado
4	I4	0,57	3,72	3,72	3,72	11,72	2,93	-1,29	Bastante Adecuado
5	I5	-3,72	1,07	3,72	3,72	4,79	1,20	0,44	Bastante Adecuado
6	I6	-1,07	1,07	3,72	3,72	7,44	1,86	-0,22	Bastante Adecuado
7	I7	-0,57	0,57	3,72	3,72	7,44	1,86	-0,22	Bastante Adecuado
8	I8	-0,57	0,57	3,72	3,72	7,44	1,86	-0,22	Bastante Adecuado
9	I9	-0,18	3,72	3,72	3,72	10,98	2,74	-1,11	Bastante Adecuado
10	I10	-0,57	3,72	3,72	3,72	10,59	2,65	-2,65	Muy Adecuado
11	I11	-1,07	3,72	3,72	3,72	10,09	2,52	-0,88	Bastante Adecuado
Suma		-18,32	26,65	40,91	40,91	90,14			
P.de corte		-1,67	2,42	3,72	3,72				N = 1,64