UNIVERSIDAD AMAZÓNICA DE PANDO
ÁREA DE CIENCIAS Y TECNOLOGÍA
CARRERA DE INGENIERIA INFORMATICA

TESIS DE GRADO

MODELADO DEL TUTOR MEDIANTE AGENTES INTELIGENTES

Postulante : **Univ. Francisco Lera Arce**

Tutor : **Lic. Javier Patty Magne**

Asesor : **Lic. Eduardo A. Zubieta Copeticon**

Cobija - Pando - Bolivia
2010

DEDICATORIA

Este trabajo está dedicado a mis papas Francisco y Amparo, por ser un apoyo incondicional y un ejemplo a seguir. Ustedes me han brindado una y otra vez esa fuerza y esa energía para salir victorioso ante cualquier obstáculo de mi vida. ¡Gracias por ser mis Padres, los quiero mucho!

AGRADECIMIENTOS

A Dios por darme fortaleza, serenidad y tesón para no desvanecer en los momentos más difíciles.

A mis hermanos Keila, Ariel y Kenia con quienes he contado siempre en los momentos dulces y amargos que nos ha tocado pasar. Junto a ustedes aprendí a vivir del lado mejor del camino de la vida donde no hay espinas porque siempre han estado conmigo.

A Jenny por aguantarme estos meses de estrés, trabajos hasta la madrugada, y brindarme su amor incondicional.

Al ser que viene en camino que lo espero con todo el amor y ternura, que despierta en mí las ganas de salir adelante y triunfar.

A los Docentes quienes me brindaron su ayuda para superarme intelectualmente y estimularme a seguir adelante.

A mis amigos que con su confianza me daban ánimo para culminar este trabajo que parecía interminable.

A todos aquellos que con el comentario más pequeño me dieron una luz sobre el camino a seguir.

ÍNDICE

CAPÍTULO 1 – INTRODUCCIÓN

CAPÍTULO 2 – MARCO TEORICO

ÍNDICE DE FIGURAS

CAPITULO 2 – MARCO TEÓRICO

CAPITULO 3 – SOLUCION PROPUESTA

ÍNDICE DE TABLAS

CAPITULO 2 – MARCO TEÓRICO

RESUMEN

Esta investigación surge motivado, por la necesidad de encontrar nuevas alternativas, que vengan a coadyuvar el proceso de enseñanza – aprendizaje. Por lo que es un punto de partida en lo que se refiere los Sistemas de Tutoría Inteligente.

En este contexto, se busca modelar un tutor que permita definir y aplicar una estrategia pedagógica para la enseñanza de la asignatura Algoritmos y Programación I del Programa de Ingeniería Informática del Área de Ciencias y Tecnología de la Universidad Amazónica de Pando. Se pretende dar los lineamientos que deben regir el comportamiento del tutor para adaptarse a las necesidades del estudiante según sus estilos de aprendizaje.

CAPÍTULO 1 – INTRODUCCIÓN

Algunas tecnologías como los Agentes Inteligentes, las Redes Neuronales, los Sistemas Expertos y los Sistemas Tutores Inteligentes, que hace unos años solo existían en forma teórica o en los ámbitos universitarios, se utilizan a diario en aplicaciones de uso relativamente sencillo que se encuentran en la etapa de producción y no son solo casos de estudio en los laboratorios universitarios. Además, estas tecnologías están disponibles para las grandes empresas y los centros de alta tecnología, así como también para aplicaciones pequeñas y medianas, que pueden utilizarlas a diario.

En los últimos años las tecnologías han evolucionado muy rápidamente y la Inteligencia Artificial (IA) surge como una de las ramas de estudio más recientes y promisorias en el campo de las Ciencias de la Computación. Ello hace que existan muchas aplicaciones poco exploradas lo cual torna a este campo en un área interesante para los investigadores, estudiantes y administradores que puedan utilizar en forma directa los resultados de las investigaciones.

1.1 ANTECEDENTES

Comenzando con las primeras implementaciones de los sistemas que seguían el modelo de instrucción asistida por computadora (CAI), en la década de los setenta, aparecen implementaciones como el sistema Scholar que lleva a la práctica la nueva propuesta de Carbonell (Carbonell et al., 1970). Esta es la primera implementación práctica donde el sistema recibe un feedback (retro alimetacion) de los alumnos e intenta inferir el estado cognitivo de los mismos.

Luego, con el advenimiento de la IA y la idea de que el sistema no realice la tarea de enseñar de una única forma, aparecieron los tutores Why y Sophie de Collins y Brown (Stevens et al., 1977); también muy limitados en su funcionalidad, pero ellos diferenciaron

los Intelligent CAI de los Sistemas de Tutoría Inteligente que hasta esa época se los consideraba sinónimos (Villareal Goulart et al., 2001).

El Proyecto CircSim

Actualmente hay que destacar el proyecto CircSim, Sistema Tutor Inteligente (STI) creado en conjunto por el Departamento de Ciencias de la Computación del Illinois Institute of Technology y el Departamento de Fisiología del Rush College of Medicine. Este tutor es el más avanzado actualmente y su implementación se utiliza en el Rush College of Medicine para complementar las clases teóricas sobre problemas cardiovasculares (Kim, 1989, 2000; Cho, 2000; Hume, 1995; Shah, 1997; Hume et al., 1992).

STI Orientados a la enseñanza de lenguajes de programación.

Se han desarrollado varios tutores y asesores inteligentes para apoyar la enseñanza de los lenguajes de programación (Castillo et al., 1998). Se implementaron y se observa utilizan distintas arquitecturas base, que se detallan a continuación con sus características más sobresalientes.

- El primer sistema tutor inteligente orientado a la enseñanza de programación es Meno (Wolfm, 1984), estaba específicamente orientado a la enseñanza de programación en el lenguaje Pascal.

- El Proust posee una innovación que es el "diagnóstico de intenciones" [Johnson, 1986]. Esto significa que el asesor es capaz de deducir las metas del usuario y compararlas con el código que analiza, lo que permite detectar errores no triviales en la lógica de programación. Esta característica se extendió luego a otros dominios, fuera del asesor Proust, por su gran utilidad.

- El Coach (Cognitive Adaptive Computer Help) (Selker, 1989). En él la asistencia al estudiante se ofrece en forma de un ambiente interactivo de ayuda, donde el estudiante proporciona un programa LISP como entrada y el asesor le ofrece información relevante (Castillo et al., 1998).

- El ACTP (Russell et al., 2003) y es uno de los precursores en el uso de modelado cognitivo en los Sistemas Tutores Inteligentes (STI) (Russell et al., 2003), pero no llega a ser un asesor inteligente completo. Utiliza "plantillas de comparación", que son librerías de código común, para encontrar los errores que cometen los estudiantes a la hora de escribir código fuente en Lisp. Este sistema podía detectar pequeños, pero comunes errores en código Lisp, con lo cual mejoró el conocimiento global de los estudiantes.

- El Tutor Capra de Fernández (Fernández, 1989) es otro tutor inteligente utilizado para la enseñanza de programadores novatos; fue desarrollado dentro del marco de la investigación del diagnóstico y detección de errores en entornos de programación. Este tutor propone lidiar con esta problemática antes mencionada, mediante un tutor inteligente que posea las siguientes características:

    - Clasificación heurística de los errores.
    - Análisis del flujo de datos.
    - Ejecución Simbólica

1.2 PLANTEAMIENTO DEL PROBLEMA

Este tema de investigación surge motivado por la necesidad de encontrar diferentes formas alternativas para la enseñanza de la asignatura Algoritmos y Programación I en la Carrera Ingeniería Informática de la Universidad Amazónica de Pando.

Desde su creación la Carrera de Ingeniería Informática, el porcentaje de alumnos que reprueban la asignatura de Algoritmos y Programación I en muy alto, siendo causa principal de la deserción del Programa. A lo largo de últimos años, con la mejora de los ambientes y la aplicación de diversas estrategias didácticas usando medios audiovisuales, foros de discusión, grupos de aprendizaje, se ha observado que si bien se evidencian

mejoras, las mismas apuntan a los grupos de estudiantes que normalmente tienen menores dificultades.

Por este motivo, se pensó en principio, en el desarrollo de un Sistema Tutor Inteligente (STI) (utilizando la Tecnología de Agentes) que realizase la tarea de enseñanza adaptando diferentes modalidades o estrategias de enseñanza. Esta podría ser una alternativa útil sobre todo para aquellos estudiantes que requieren un mayor grado de enseñanza del tipo uno a uno (Perkins et al., 1995), es decir personalizado. Por lo que la investigación se centra primordialmente en proponer un modelo que cumpla dichas características, y que sirva como punto de partida para futuras investigaciones.

1.2.1 DEFININCION DEL PROBLEMA

¿Cómo diseñar un modelo de tutor inteligente, mediante la utilización de Agentes Inteligentes, para la asignatura de Algoritmos y Programación I del programa de Ingeniería Informática de la Universidad Amazónica de Pando, que sea capaz de adaptarse a las necesidades y preferencias de los alumnos, según sus estilos de aprendizaje?

1.3 OBJETIVOS

1.3.1 Objetivo General

Diseñar un modelo de tutor inteligente, mediante la utilización de Agentes Inteligentes, para la asignatura de Algoritmos y Programación I del programa de Ingeniería Informática de la Universidad Amazónica de Pando, que sea capaz de adaptarse a las necesidades y preferencias de los alumnos, según sus estilos de aprendizaje.

1.3.2 Objetivos Específicos

- Determinar el estado actual de las investigaciones en el tema de la enseñanza humano e inteligente, relevantes para la tarea que desempeña el módulo correspondiente al tutor en un Sistema Tutor Inteligente (STI).
    - Establecer los fundamentos que sustentan las teorías seleccionadas.
    - Seleccionar los componentes del módulo del tutor, mediante el uso de submódulos, que permitan guiar a los estudiantes en el proceso de aprendizaje de manera de poder responder a sus necesidades.

- Modelar el subsistema tutor enmarcado dentro de la arquitectura de los Sistemas Tutores Inteligentes (STI), con sus submódulos e interfaces, utilizando las herramientas que provee la Ingeniería de Software y la actual tecnología de Agentes.

- Validar el modelo planteado con datos característicos de los alumnos, tales como los estilos de aprendizaje y el rendimiento académico.

1.3 JUSTIFICACIÓN

CIENTIFICA

Desde las últimas décadas del siglo XX se ha mostrado el creciente interés y demanda por el desarrollo y aplicaciones en sistemas de tutoría e inteligencia artificial, utilizando una fusión de especialidades como sistemas expertos, robótica e ingeniería de software.

Este trabajo de investigación científica aplica conceptos extraídos de la inteligencia artificial, busca alcanzar una conceptualización en lo que se refiere a los Sistemas de Tutoría Inteligente y los Agentes Inteligentes, combinándolos para dar lugar a una herramienta alternativa, que permita desarrollar las habilidades de los estudiantes en la resolución de problemas por ordenador, así mismo brinda información de las aplicaciones que se pueden desarrollar en el área de Actores Inteligentes.

SOCIAL

Con la implantación a largo plazo de este tipo de sistemas, se puede coadyuvar en el proceso de enseñanza-aprendizaje en cursos con altos índices de alumnos. Así, se personaliza el ambiente de aprendizaje, sin requerir más recursos humanos y a la vez se pueden flexibilizar los horarios de estudio para los estudiantes, permitiendo que los alumnos que trabajan o que viven alejados de los establecimientos educativos puedan interactuar con el sistema según su propio ritmo de estudios.

TECNICA

Se requiere a nivel técnico, por lo tanto, la utilización de estrategias enmarcadas en el desarrollo de habilidades tendientes a la atenuación del problema, pero teniendo en cuenta que la introducción de un Modelo de Tutoría Inteligente, adaptable a las necesidades particulares de cada uno de los estudiantes, es una opción valida, sobre todo en el Programa de Ingeniería Informática, donde ya se ha efectuado un Rediseño de la Malla Curricular que da las pautas necesarias de las destrezas que debe alcanzar el estudiante.

1.5 HIPOTESIS

"El Modelo de Tutoría Inteligente, para la asignatura de Algoritmos y Programación I del programa de Ingeniería Informática de la Universidad Amazónica de Pando, utilizando Agentes Inteligentes, permite que su diseño se adapte a las preferencias del estudiante, tomando en cuenta sus estilos de aprendizaje".

1.6 METODOLOGÍA

Para la elección de la metodología de investigación, se ha tomado como punto de partida la estrecha relación que existe entre el análisis del problema, hipótesis y objetivos. Dado que toda investigación sigue en forma general un mismo método científico (problema, hipótesis, conclusión), este llega a tener distintas acepciones de acuerdo al abordaje que se le da a la investigación.

Método Hipotético – Deductivo, Utiliza una estrategia que mezcla la inducción y deducción. En realidad trata de enfatizar el hecho de que en el proceso de adquisición de nuevos conocimientos la ciencia actúa de ambas formas y las dos son partes de un único método. Independientemente de donde empiece el proceso, el investigador necesita tanto ir de los datos a la teoría como de la teoría a los datos. Así, desde una teoría se deduce una consecuencia contrastable en la realidad, se realizan una serie de observaciones que sirven para corroborar o modificar lo deducido de la teoría.

En el caso de no existir una teoría previa, se puede empezar realizando una observación a partir de la cual se haría una generalización en forma de ley. A partir de un conjunto de leyes se podrá elaborar una teoría de la que, a su vez, deducirá nuevas consecuencias, lo cual nos permitiría volver a realizar observaciones que servirían como contraste y así sucesivamente.

1.7 ALCANCES

El presente trabajo de investigación se enmarca dentro del campo de la Inteligencia Artificial, a su vez en lo que se refiere a la utilización de Agentes Inteligentes, y de cómo podrían desenvolverse en la educación, utilizando la arquitectura de los Sistemas de Tutoría Inteligente, abordando específicamente el tema de resolución de problemas con computadora, para desarrollar las habilidades del estudiante en cuanto a esta temática, dentro de la Asignatura de Algoritmos y Programación I del programa de Ingeniería Informática de la Universidad Amazónica de Pando.

1.8 APORTES

El caso particular de un Modelo de Tutoría Inteligente con un dominio del contenido de la Asignatura Algoritmos y Programación I, permitirá a los estudiantes utilizar el modelo planteado para su desarrollo, y por ende construcción de un sistema de enseñanza Inteligente para el primer semestre de la carrera de Ingeniería Informática, y de esta manera

mejorar sus habilidades y a la vez el rendimiento en las materias más avanzadas que utilicen la base de conocimientos que debe adquirirse en esta asignatura.

No es el objetivo de un Agente Inteligente (Tutor de Aprendizaje) reemplazar a un tutor humano, sino que su implementación permita coadyuvar al docente en situaciones donde se requiere de refuerzos en la enseñanza. De esta manera se pueden manejar de forma más eficiente los escasos recursos humanos disponibles, pudiendo el tutor humano hacerse cargo en forma personalizada sólo de un cierto número de tareas que el sistema no puede realizar, o es muy complejo de implementar.

Esta perspectiva de la enseñanza tiene al estudiante como el centro del proceso educativo, siendo éste quien regula sus aprendizajes. De esta manera se mueve el foco de atención del tutor o profesor y en particular del alumno que, según el modelo clásico, cumple con una tarea puramente pasiva (la del aprendizaje) modificando esta visión por la de un estudiante como de atención, donde son sus necesidades las que deben prevalecer.

Tomando en cuenta las consideraciones señaladas, el modelo planteado se pueda perfeccionar a largo plazo para así, aprovechando los recursos disponibles en la actualidad, permitir la construcción de un sistema que coadyuve en las tareas de los tutores humanos y que a la vez mejore la experiencia de aprendizaje desde la perspectiva del estudiante.

CAPÍTULO 2 – MARCO TEÓRICO

A finales del siglo XIX se realizaron estudios sistemáticos, basados en distintas ciencias, como la psicología, la educación, la sociología, la medicina, etc., para explicar el proceso de aprendizaje y el funcionamiento de la mente humana. Se crearon muchas teorías a partir de estas investigaciones, desde las fisiológicas, que explican el funcionamiento del cerebro humano en función de intercambio de neuroreceptores y diferencias de potencial hasta las filosóficas que intentan explicar el funcionamiento del cerebro humano a partir de los estímulos externos.

Se observa que cada autor da una definición de la inteligencia: desde la enunciación muy general de Maturama, que la ve como *"un tributo o propiedad distintiva de algunos organismos"* (Maturama, 1998), pasando por investigadores como Piaget (Piaget, 1989) que plantea que la inteligencia *"es la capacidad de adaptación de un organismo a una situación nueva"*.

Es aquí donde la investigación sobre Inteligencia Artificial (IA) intenta asimilar estas definiciones dentro de sus propias estructuras, incorporándolas en los Sistemas Tutores Inteligentes (STI) que contemplan el aprendizaje humano y la enseñanza con base pedagógica. Es por ello, que el objetivo de este capítulo es presentar un marco conceptual que dé lugar al Marco Teórico de la Inteligencia Artificial (AI), los Sistemas Tutores Inteligentes (STI) y los Agentes Inteligentes.

2.1 INTELIGENCIA ARTIFICIAL

A diferencia de la filosofía y la psicología, que tratan de entender cómo funciona la inteligencia en abstracto, la Inteligencia Artificial (IA) es un intento por descubrir y aplicar los aspectos de la inteligencia humana que pueden ser simulados mediante construcciones artificiales. Se observa que hasta en las etapas tempranas de su desarrollo, la Inteligencia

Artificial (IA) ha presentado productos sorprendentes en sus aplicaciones (Stuart et al., 1995).

Hoy en día, el campo de la Inteligencia Artificial (IA) enmarca varias subáreas tales como los sistemas expertos, la demostración automática de teoremas, el juego automático, el reconocimiento de la voz y de patrones, el procesamiento del lenguaje natural, la visión artificial, la robótica, las redes neuronales, etc. (Castillo et al., 1998). En la *Figura 2.1* se pueden ver los subcampos dentro de la Inteligencia Artificial (García-Martinez et al., 2003).

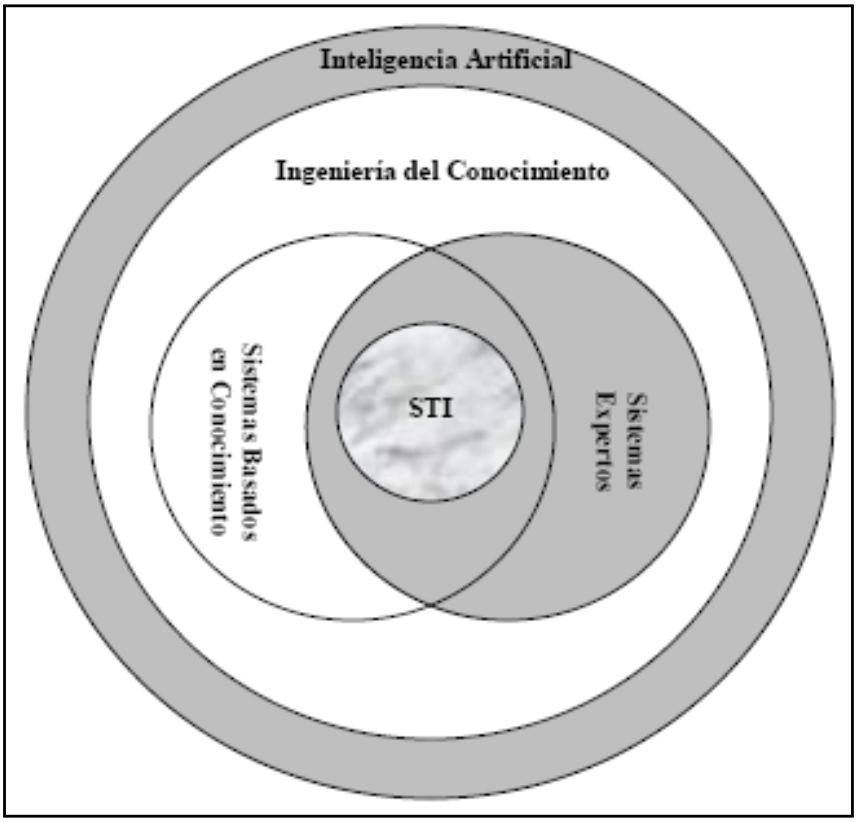


Figura2.1: *Relación entre la IA, la Ingeniería del Conocimiento (INCO) y otras áreas.*
Fuente: *(García-Martínez et al., 2003) Sistemas Inteligentes.*

La inteligencia artificial (IA) surge así como una disciplina cuyo objetivo es proveer técnicas para el desarrollo de programas capaces de simular la inteligencia que utilizan los humanos para solucionar problemas en una gran cantidad de dominios (Krishnamoorthy et

al, 1996), por lo que la IA provee un conjunto de formalismos que pueden representar los problemas, las herramientas y técnicas para resolverlos. Según diversos autores (Krishnamoorthy et al, 1996; Newell, 1969) las actividades esenciales asociadas con la inteligencia son:

- Responder de manera flexible a una gran variedad de situaciones.
- Dar sentido a los mensajes contradictorios y/o ambiguos.
- Reconocer la importancia relativa de los diferentes elementos de la situación problemática planteada.
- Encontrar similitudes entre situaciones, sin importar las diferencias que las separan.
- Encontrar las diferencias entre situaciones, sin importar lo similares que puedan parecer.

Por su amplio contenido, es difícil definir a la Inteligencia Artificial (IA), pero resulta de interés para este trabajo arribar a un concepto esclarecedor, por lo que se expondrán diferentes posiciones acerca de la IA. Si bien existen muchas definiciones de Inteligencia Artificial (IA) en las que cada autor la presenta de una manera ligeramente diferente, aquí se resumirán las más representativas, agrupadas como lo propone Stuart (Stuart et al., 1995) en dos categorías:

- Las que conciernen a los procesos de pensamiento y razonamiento.
- Las que conciernen al comportamiento

En general, estas definiciones miden el éxito de la implementación de herramientas de la Inteligencia Artificial (IA) de dos maneras diferentes:

- En términos de performance humana: Es decir, capacidad de resolución de problemas, capacidad de razonamiento abstracto, etc.
- En términos de "racionalidad" o concepto ideal de inteligencia: Como establece Stuart (Stuart et al., 1995) quien define a un sistema como racional "si hace lo

correcto", es decir, una acción a la que llega por medio de procesos lógicos medibles.

Aclaradas las posibles distinciones en las definiciones, se las enumerará cronológicamente:

(Bellman, 1978) la definió 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"*. Esta definición también es aplicada por Villareal Goulat (Villareal Goulat, 2001).

(Haugeland, 1985) la definió como: *"El nuevo y excitante esfuerzo de hacer pensar a las computadoras"…"computadoras con mente, en el sentido completo y literal de la frase".*

Según (Charniak & McDermott, 1985) es: *"El estudio de las facultades mentales a través del uso de modelos computacionales".*

Para (Kurzweil, 1990) es: *"El arte de crear máquinas que realicen funciones que requieran una cierta inteligencia cuando estas tareas son desempeñadas por personas".*

(Schalkoff, 1990) la definió como: *"Un campo de estudio que busca explicar y emular el comportamiento inteligente en términos de procesos computacionales".*

Para (Rich & Knight, 1991) es: *"El estudio para hacer a las computadoras realizar tareas, en las que por el momento los humanos son mejores".*

Según (Winston, 1992) es: *"El estudio de la computación para hacer posible el percibir, razonar y actuar".*

(Luger & Stubblefield, 1993) la definieron como: *"La rama de la ciencia de la computación que se encarga de la automatización del comportamiento inteligente".*

Por último, se considera a los dos pioneros de la investigación en Inteligencia Artificial, Barr y Feigenbaum, quienes la definen de la siguiente manera: *"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 en el comportamiento humano que se refiere a la comprensión del lenguaje, el aprendizaje, el razonamiento, la resolución de problemas, etc." (Barr et al., 1981).*

Todas estas definiciones mencionadas son válidas y cada una agrega un aspecto al amplio campo de estudio que es la Inteligencia Artificial (IA).

2.1.1 Historia de la Inteligencia Artificial

En este apartado se presenta una reseña histórica de la investigación en el campo de la Inteligencia Artificial (IA) cronológicamente ordenados de manera de obtener un panorama general de su evolución. Esta evolución permitirá observar cómo van surgiendo las nuevas herramientas y aplicaciones que facilitarán los diseños de los Agentes Inteligentes.

Stuart (Stuart et al., 1995) reconoce que los primeros trabajos en Inteligencia Artificial fueron realizados por Culloch y Pitts en 1943, quienes centraron sus investigaciones en tres áreas fundamentales:

- El conocimiento de la fisiología y las funciones básicas de las neuronas.
- El análisis formal de la lógica proposicional
- La teoría de computación de Turing.

Ellos propusieron la primera simulación de una neurona que funcionaba como un interruptor (encendido/apagado) con respecto al valor de las neuronas vecinas. Seis años después, en 1949 Hebb (Hebb, 1949) creó una regla simple para modificar el peso (o intensidad) de estas conexiones. Un año más tarde, Shannon (Shannon, 1950) y Turing en 1953 (Turing et al., 1953) escribieron programas que jugaban al ajedrez ejecutados en computadoras que seguían el paradigma de Von Newmann. Para esta misma época se creó

SNARC (Minsky, 1954), una red neuronal que simulaba 40 neuronas utilizando 3000 tubos de vacío y partes de un bombardero B-24.

Según Stuart (Stuart, 1995) en 1954 Newell y Simon ya tenían programas que podían utilizar razonamiento lógico, utilizando la teoría lógica. Simon declaró haber inventado un programa computacional capaz de pensar de manera no numérica. Pero fue, según Stuart (Stuart et al., 1995), en los finales de la década de los cincuenta en la que se adoptó el nombre que McCarthy le había dado a este campo de la computación por más de 20 años: *Inteligencia Artificial.*

Si bien en los comienzos de la década de los cincuenta el poder computacional era muy limitado, se desarrollaron teorías que hoy son básicas en el campo de la Inteligencia Artificial, sobre todo en el campo del pensamiento humano y los protocolos que las personas utilizan a lo largo de la resolución de un problema dado (Stuart et al., 1995).
En el año 1958, McCarthy en el MIT[1], definió el lenguaje de alto nivel denominado LISP, que se convertiría en el lenguaje dominante en el campo de la IA; si bien el uso de tiempo de CPU era difícil de encontrar, en ese mismo año otros investigadores en el MIT inventaron la política de *"time sharing"(tiempo compartido)*. En 1959 Gelernter construye el *Geometry Theorem Prover* (Gelernter, 1959), que como la lógica teórica, se utiliza para probar teoremas utilizando representaciones explícitas de axiomas. Stuart (Stuart et al., 1995) resalta que, luego de más de 45 años, muchos de los trabajos escritos entre 1958 y 1959 permanecen vigentes a la fecha de hoy. También en esta época, Friedberg (Friedberg et al., 1959), comenzó a experimentar con la idea de evolución en la computadora, ahora llamada *Algoritmos Genéticos* (AG).

Ya en la década de los sesenta, existían laboratorios de Inteligencia Artificial y robótica en las universidades más grandes del mundo (como el Massachussets Institute of Technology y Standford University, entre otras) y el programa de Slagle llamado SAINT creado en 1963 (Slagle, 1963) era capaz de solucionar problemas de cálculo que se le entregaban a los

[1] MIT es el acrónimo de Massachussets Institute of Technology (Instituto Tecnológico de Massachussets), institución dedicada a la ciencia, la ingeniería y la investigación, fundada en 1861 por el geólogo William Barton Rogers

estudiantes del primer año. Para 1968 Bertram con su SIR (Semantic Information Retrieval) era capaz de contestar preguntas formuladas directamente en inglés (utilizando un subconjunto de palabras de este idioma). Luego fue mejorado por un programa que entendía el lenguaje natural, en 1972, por Winograd (Winograd, 1972). En 1873 Woods construyó el sistema LUNAR (Woods, 1973), que permitía a los geólogos realizar preguntas, en inglés, acerca de muestras de rocas traídas por el programa espacial norteamericano *Apollo* desde la luna. Este fue el primer programa de lenguaje natural en ser realmente utilizado.

El trabajo de Winograd (Winograd, 1972) mostró cómo una gran cantidad de elementos pueden unirse colectivamente para representar un concepto individual, incrementando el paralelismo y haciendo más robustas las aplicaciones (Stuart et al., 1995). Recién en esta época, todos los trabajos expuestos respondían de manera satisfactoria a un pequeño conjunto de elementos de prueba, pero fallaban a la hora de trabajar con el universo completo de posibilidades: esto se debe a que la IA representaba los hechos de un problema de una manera determinada y establecía una serie de pasos a seguir para resolverlos. Antes de que la teoría NP[2] estuviera completamente desarrollada se suponía que la escalabilidad de los problemas era solo cuestión de mejorar el hardware.

En 1979 se publicó el informe Lighthill (Lighthill, 1973) que contenía grandes críticas a la IA y por el cual el gobierno británico suspendió el apoyo a la investigación en el área de la IA. A este informe se le suma que las posibles aplicaciones de los algoritmos desarrollados hasta la fecha era muy pocas, poniendo el ejemplo de que un Perceptron (Rosenblatt, 1958) (originalmente implementado como una simulación en una IBM® 704) de únicamente 2 entradas, solo podía representar un conjunto muy limitado de situaciones, y no fue hasta 1980 en que se desarrolló el algoritmo de Backpropagation (Werbos, 1990) para redes complejas multicapa.

[2] Un problema del tipo NP (nondeterministic polynomial time) es aquel que es verificable en un tiempo polinómico por una máquina no determinística de Turing, la cual puede seguir varios caminos computacionales simultáneamente (con la restricción de que éstos no pueden comunicarse entre sí). Khachian demostró en 1979, que los problemas de la programación lineal eran del tipo P (polynomial time) y no NP. (Borwein, et al, 1987].

En este período Buchanan desarrolla el programa Dendral (Buchanan et al., 1969) que realizaba la tarea de resolver la estructura molecular de una sustancia a partir de la información que provenía de un espectrómetro de masa. Este programa, según Stuart (Stuart et al., 1995) funcionaba de forma aceptable para moléculas complejas, probando con esto que las técnicas de la IA pueden ser aplicadas a dominios reales y no solamente a casos de laboratorio. Este fue el primer programa utilizado que contenía información sobre el dominio y las reglas para resolverlas, además se lo puede ver como uno de los precursores de los KBES[3]. Una de las primeras aplicaciones útiles fue MyCin de Feigenbaum, Buchanan, y Shortliffe (Buchanan et al., 1984), para el diagnóstico de las infecciones sanguíneas. Constaba de aproximadamente 450 reglas y se comportaba tan bien como un experto, y considerablemente mejor que un doctor recién recibido. Es una de las primeras aplicaciones que integra las incertezas dentro del dominio como una forma de emular la complejidad del dominio médico, ya que varias enfermedades pueden tener los mismos síntomas, o una enfermedad puede casi no presentar síntomas mesurables.

A su vez, el sistema de razonamiento llamado Prospector creado por Duda en 1979 (Duda et al., 1980) generó una gran publicidad cuando recomendó una exploración en profundidad de un sitio geológico, dando por resultado el descubrimiento de uno de los más grandes depósitos de molibdeno. También se realizaron grandes avances en el lenguaje natural en los que participaron Schank y Abelson en 1977, Schank y Riesbeck en 1981 y Dyer en 1983 A su vez, se intentó describir la organización de la memoria del cuerpo humano, por Rieger, 1976 y Kolodner, 1983 entre otros.

A partir del comienzo de la década de los ochenta, la Inteligencia Artificial dejó los laboratorios para convertirse en una industria. Un ejemplo de ello es el Sistema Experto (SE) “R1” utilizado por la Digital Equipment Corporation. Con este sistema la organización conformaba órdenes y pedidos de los nuevos sistemas de computadora, logrando ahorros para 1986 de cuarenta millones de dólares al año.

[3] Knowledge Based Expert Systems (*Base de conocimiento de los sistemas expertos)*.

También, a partir de la década de los ochenta, se anunció el proyecto de la "quinta generación de computadoras", ejecutable en 10 años en el cual las máquinas interpretarían el lenguaje Prolog (o un lenguaje similar) como si fuese lenguaje de máquina, pudiendo realizar millones de inferencias por segundo. Esto impulsó el interés por la IA, haciéndola avanzar sobre pasos firmes. Además, condujo al rediseño de algunos algoritmos, como el de Backpropagation (Werbos, 1990), el procesamiento distribuido, planteado por Rumelhart y McClelland en 1986 (Rumelhart et al., 1986) refuerza el impulso de la industria de la Inteligencia Artificial (IA).

Más tarde, comenzó el período que Stuart (Stuart et al., 1995) ubica de la década de los ochenta a la actualidad, donde no se desarrollaron nuevas teorías revolucionarias, pero se intentó refinar las teorías ya existentes para su correcto funcionamiento. Por ejemplo Tate (1977) y Chapman (1987), realizaron una síntesis de un programa planificador dentro de un framework[4] para facilitar el trabajo, el que fue utilizado incluso para programar las misiones espaciales. También se vio un resurgimiento del razonamiento probabilístico dentro de los sistemas inteligentes, como plantea Pearl (1978) y defiende Cheeseman (Cheeseman, 1985).

A principios de la década de los noventa, se vislumbró la aplicación nuevo paradigma de programación. Se deja atrás la programación estructurada con la que se armaron las primeras redes neuronales (Rosenblatt, 1958), quedando atrás la programación orientada a objetos (OOP), que en el caso de la IA se ve representada por la teoría de los marcos o "frames" (Minsky, 1974) y se encamina hacia la programación por agentes. En 1987 se creó SOAR de Laird, Newell y Rosenbloom (Laird. et al, 1987) que fue una de las primeras implementaciones de agentes que se utilizaba para procesar las entradas de un sistema de sensores, pero en una visión más general se comportaba como una arquitectura de resolución de problemas basada en reglas.

[4] Un framework es una estructura extensible para describir un conjunto de conceptos, métodos y tecnologías necesarias para un diseño completo y posterior manufactura de software, es una interface común integrada para el proceso de diseño, implementación, conversión de datos, etc. que facilita la producción de, en este caso, la planificación de las misiones espaciales.

A finales de la década de los noventa e inicio del siglo XXI, con el advenimiento del mayor poder computacional y de las grandes capacidades de almacenamiento de datos con costos de hardware más reducidos, se terminó por aceptar a la Inteligencia Artificial (IA) como un campo práctico y no solo teórico, en el cual los desarrollos aplicables son posibles más allá de los laboratorios. Esto llevó a continuar la tendencia de menor investigación en nuevas teorías, refinamiento de las teorías actuales, y la generación de miles de productos que se utilizan hoy en día, a nivel masivo como el actual "clippo", el asistente de Microsoft Word, que utiliza una red bayesiana para satisfacer las necesidades del usuario del procesador de texto.

2.2 SISTEMAS TUTORES INTELIGENTES

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 (CAI), 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 CAI tradicionales, permiten una mayor individualización en la instrucción, llegando más lejos, a través del entendimiento de las metas y creencias del estudiante.

- Se usan técnicas de Inteligencia Artificial para planeación, optimización y búsquedas, dejando 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 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."*

Villareal (Villareal, 2003) plantea que los Sistemas Tutores Inteligentes (STI) simulan a un tutor autoritario que posee una estrategia de enseñanza de los conceptos del dominio del tipo uno a uno. Además es un experto en un dominio de conocimiento determinado y actúa como guía, tutor o entrenador. Este tutor debe poder adaptarse a las necesidades, que surgen a lo largo de la interacción en una sesión de tutelado, del estudiante alumno.

2.2.1 Arquitectura y Componentes

Los Sistemas Tutores Inteligentes (STI) tienen como principal objetivo impartir la enseñanza de un contenido dado un dominio en la forma más adecuada a las necesidades individuales del alumno. Estos sistemas se basan en una arquitectura compuesta por tres grandes módulos: el módulo del tutor, el módulo del alumno y el módulo del dominio (Villareal et al., 2001). Podría agregarse un cuarto módulo denominado el módulo de evaluación y, un quinto denominado módulo de interface y el ecosistema propuesto por Cataldi (Cataldi, 2004).

1. ***Módulo del Alumno:*** Este módulo debe representar el estado inicial del alumno y sus características individuales, entre ellas una de las más importantes es el conocimiento individual instantáneo sobre el dominio (Villareal Goulart et al., 2001). Guardia Robles (Guardia Robles, 1993) lo define como: *"El modelo del estudiante, que refleja cuánto conoce el estudiante sobre el dominio, así como las experiencias cognitivas y de aprendizaje que ha llevado, del cual puede obtenerse un diagnóstico."*

2. ***Módulo del Tutor:*** Este módulo posee el conocimiento sobre las estrategias y tácticas de enseñanza para poder seleccionarlas en función de las características del alumno, que están almacenadas en el módulo del alumno (Villareal Goulart et al., 2001). Pero debe ir más allá de la experiencia en el dominio, ya que debe ofrecer a cada estudiante un método de enseñanza de acuerdo con sus necesidades.

3. ***Módulo del Dominio:*** Este módulo posee el conocimiento de la materia formado por las reglas de producción, estereotipos, etc. De aquí el módulo tutor obtiene el conocimiento que debe enseñar (Villareal Goulart et al., 2001). Definido como *"El modelo experto o del dominio, el cual versa sobre la materia o curso que se impartirá" (Guardia Robles, 1993).*

4. ***Módulo de Evaluación:*** Se encarga de realizar una evaluación general del sistema y generar estadísticas acerca de los avances de los estudiantes; pudiendo efectuar el

diagnóstico evolutivo luego de cada uno de los estados considerados, de este modo podría también predecir el comportamiento en los eventos futuros. La evaluación de los estudiantes debe ser constante y durante la carga del proceso, con instancias de autoevaluación. También se deberán generar informes a utilizar para evaluar al sistema como método apto de enseñanza.

5. ***Módulo de Interface:*** Es la interface de interacción entre el STI y el alumno real, que se encarga de presentar el material del dominio y cualquier otro elemento didáctico de la manera correcta (Villareal Goulart et al., 2001). *"La interface, que permite a los usuarios interactuar con el sistema. Se distinguen tres tipos específicos de usuarios: el Estudiante, el Instructor, y el Desarrollador del sistema."*(Guardia Robles, 1993). Para su diseño pueden seguirse los criterios ergonómicos basados en el estándar ISO 9241 para *Human Computer Interface* (HCI) u otros similares.

Los primeros tres módulos conforman la arquitectura clásica propuesta por Carbonell (Carbonell, 1970) y también funcional de los STI (Villareal Goulart et al., 2001) se pueden ver en la *Figura 2.2.* Esta postura presentó grandes avances en el modelado de ambientes educativos, ya que separó el dominio de la forma en la que éste es utilizado (Villareal Goulart et al., 2001).

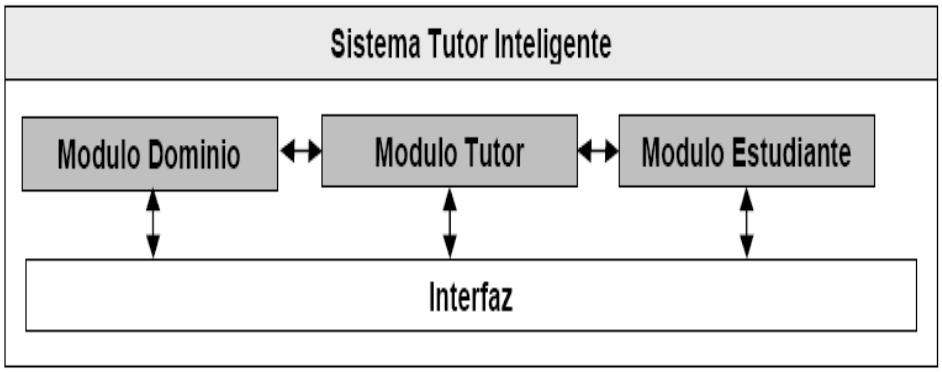


Figura 2.2: *Estructura clásica de un Sistema Tutor Inteligente propuesta por Carbonell.*
Fuente: *(Carbonell, 1970) Al in CAI: An artificial intelligence approach to computer assisted instruction.*

Existen visiones no clásicas de los Sistemas Tutores Inteligentes (STI), como lo es la propuesta de arquitectura general en Clancey (Clancey, 1991) la cual puede resumirse de la siguiente manera:

- ***Son dos sistemas expertos:*** Uno encargado de diagnosticar el estado actual del alumno y otro encargado de decidir la mejor forma de retroalimentarlo en el momento actual, cumpliendo el rol de asesor.

- ***Poseen un sistema experto de diagnóstico:*** Con un algoritmo de decisión sobre la mejor estrategia de retroalimentación, basado en una representación del conocimiento sobre el estado cognitivo del alumno.

2.2.2 Módulo del Alumno

El diseño del modelo del estudiante se centra, según Barr (Barr et al., 1983) alrededor de las preguntas: *¿Qué es lo que se desea que el estudiante sepa acerca del mecanismo de resolución de una problemática? Qué tipos de conocimientos debe tener un estudiante para poder resolver un problema de operación o reparación del mecanismo?.* De manera que el módulo del estudiante deberá tener conocimientos acerca de (Sierra et al., 2003):

- Los componentes del mecanismo para la resolución de un problema.
- La operación de los componentes del mecanismo.
- La interrelación entre los componentes del mecanismo.
- La operación del mecanismo.

Si un estudiante elige examinar un componente en particular, entonces se asume que el estudiante conoce algo acerca del componente. Dado el contexto del problema, la selección de un componente es de algún modo una confirmación o no, de que el estudiante comprende lo que el componente hace y cómo se relaciona con otros componentes en la operación del mecanismo. Es decir que cada vez que el estudiante chequea, manipula o

examina un componente, indica de algún modo lo que él conoce o desconoce acerca de la operación del mecanismo.

2.2.3 Módulo del Dominio

Para poder construir el modelo de un proceso, debe ser posible descomponerlo en sus partes constitutivas. Es decir, el mecanismo a ser modelado debe tener partes identificables en las cuales pueda ser descompuesto.

Por lo tanto para el caso del módulo del dominio, se debe descomponer el conocimiento de los temas para los cuales el sistema deberá ser experto. Entre ellos se encuentra el conocimiento dependiente del dominio, compuesto por las definiciones, los conceptos fundamentales y las agrupaciones de conceptos que forman los temas. También existe el conocimiento independiente del dominio, el cual se compondrá de los diferentes parámetros del sistema que se requieren para el correcto funcionamiento del mismo y por último se encuentran los elementos didácticos que se utilizan para mejorar y facilitar la experiencia del proceso enseñanza/aprendizaje.

2.2.4 Interface

La interface se encarga de presentar el contenido de la sesión de tutoría en forma homogénea a lo largo de toda la curricula, presentando en tiempo y forma los elementos pedagógicos que utiliza cada sesión, como pueden ser ejercitación, material multimedia, texto, etc. Si bien la interface existe en todos los Sistemas Tutor Inteligente, no se la considera un módulo fundamental, ya que se centra más en el aspecto gráfico y no en los contenidos. Se puede generar una interface que responda a la adecuación del usuario, donde este establezca parámetros, como la letra, el tamaño de la letra, el color de fondo, etc., para crear un ambiente de trabajo donde se sienta más cómodo y mejorando la sesión educativa. El diseño es muy importante para la interface, ya que depende en gran parte de los usuarios.

2.2.5 Módulos Externos: Evaluación y Registros Históricos

El sistema debe ser capaz de validarse, es decir, debe ser capaz de analizar el desempeño que tuvo en las interacción con los alumnos y reorganizarse si su desempeño fue inferior al esperado. Esta reorganización puede darse de forma automática, en Sistemas Tutores Inteligentes (STI) más complejos, o en forma manual, modificando los componentes que, dado el resultado de la evaluación, demostraron ser problemáticos o estar pobremente adaptados a las situaciones a las que el sistema está expuesto.

También el sistema debe ser capaz de brindar los resultados de esta evaluación de forma de informes, listados y datos procesados, para poder, por ejemplo, clusterizar los alumnos en grupos definidos, detectar anormalidades o situaciones incorrectas, inferir las causas y validar los datos obtenidos con el sistema en cursos pilotos, sin la interacción del sistema, en los cuales se han tomados los mismos datos, para poder verificar la mejora que resulta de la introducción del sistema tutor inteligente en el ámbito educativo.

El sistema debe ser capaz de generar históricos de acuerdo a la solicitud del usuario y debe poder mostrarlo de diferentes maneras. Estos módulos externos garantizan que el funcionamiento del sistema pueda ser validado para nuevos casos de aplicación.

2.2.6 Módulo del Tutor

La idea principal de este módulo es la de utilizar la computadora como *"herramienta de aprendizaje"*, buscando automatizar el proceso de enseñanza (Villareal Goulart *et al.*, 2001). Esto motivó los primeros esfuerzos en la década del sesenta en brindar soporte a la los docentes. Desde aquella época hasta la actualidad se han realizado cambios en el paradigma utilizado en la educación, partiendo de un estilo fuertemente conductista hasta el día de hoy donde el concepto de *"impartir"* conocimientos es reemplazado por el de *"descubrir"* o *"construir"* el conocimiento de forma significativa.

Por lo tanto, el software reproducía la forma de enseñanza que prevalecía en el sistema educacional de esa época. Una parte del contenido del dominio era presentada en una o más pantallas (orientadas solo a caracteres, sin gráficos u otros elementos didácticos) y la interacción se limitaba a presionar la tecla *Enter* para cambar de pantalla. Esta forma de enseñanza recibió el nombre de *Computer Assisted Instruction* (CAI).

Según Wenger (Wegner, 1987), la visión clásica del papel del módulo tutor *es tomar las decisiones pedagógicas en función de las interacciones con el alumno.* Estas decisiones derivan de reglas o estructuras de conocimiento que representan el conocimiento del tutor con respecto al dominio y están representadas en forma definida en el sistema.

En los tutores clásicos, el criterio de control varía conforme a otro de optimización, y suele ser una decisión de implementación, o sea, no existe cambio en la forma en la que se controla la interacción. Algunos sistemas pueden monitorear las actividades de los alumnos, adaptando sus acciones a las respuestas de los alumnos, pero nunca cediendo el control de la interacción. Un ejemplo de este sistema es el *CircSim* (Kim, 1989; Kim, 2000; Cho, 2000; Hume, 1995; Shah, 1997; Hume et al., 1992).

Otra visión es la de García Martínez, compartida por Sierra (Sierra *et al.*, 2003), donde el módulo instruccional o módulo del tutor (Carbonell, 1970) es una representación de los métodos que se usarán en el tutor inteligente para proveer información al estudiante. Este modelo es complejo pues está pensado para dirigir al estudiante en su proceso de aprendizaje y efectuar automáticamente ajustes en esta dirección a medida que el estudiante progresa.

Para Sierra (Sierra et al., 2003) *"En un sentido práctico, se tiene el siguiente problema a resolver cuando se construye el módulo tutor de un Sistema Tutor Inteligente. El estudiante está manipulando el modelo del dominio o mecanismo y el modelo de estudiante está realizando inferencias en base a estas manipulaciones. El tutor debe entonces hacer uso de esta información a efectos de proveer al estudiante con información que sea útil para éste".* En su forma más general, a efectos de poder definir correctamente la operación del

módulo tutor, debe ser posible responder a las siguientes preguntas: *¿Cuándo es necesario instruir? ¿Qué tipo de instrucción debe darse?* Los pasos metodológicos propuestos para el diseño del modelo del tutor son los siguientes (Sierra et al., 2003):

1. Analizar del modelo del estudiante a efectos de definir claramente cuáles son las acciones que este puede llevar a cabo.
2. Interpretar adecuadamente las acciones definidas en el Paso 1 en función del tipo de conocimiento que el estudiante debería poseer para llevar a cabo dichas acciones en forma correcta.
3. En base a los diferentes tipos de conocimiento identificados en el paso 2, determinar las estrategias de instrucción más apropiadas a fin que el estudiante incorpore significativamente estos conocimientos a su estructura cognitiva.

En la década del setenta Carbonell (Carbonell, 1970) presentó una nueva postura, que llevó a la práctica a través del sistema Scholar, donde tomó en consideración la forma en la que el tutor estructuraba el contenido del dominio en el aula. El profesor recibe el *"feedback"* de los alumnos e intenta inferir el estado cognitivo actual del alumno y a partir de esto el modifica su comportamiento (a través de estrategias de enseñanza) para que el alumno obtenga el mayor beneficio.

El ***Scholar*** es el primer programa para la enseñanza que utiliza una representación del conocimiento basada en Inteligencia Artificial (IA) (Castillo et al., 1998). Su principal objetivo es enseñar la geografía de América del Sur a estudiantes de nivel primario. Es el primer sistema en reconocer la posibilidad de separar el conocimiento de la materia a estudiar del resto de la aplicación. Consta de una interface de línea de comandos tradicional, donde *Scholar* plantea preguntas sobre el tema y el alumno las contesta. Se evalúa la respuesta y se genera una retroalimentación. Cuenta también con un reloj para indicarle al estudiante que está tardando demasiado (Carbonell, 1970).

Este enfoque pretende buscar una instrucción más personalizada, al contrario de los CAI´s, en la que estos sistemas no realizan la tarea de enseñar en una única forma. El sistema

propuesto por Carbonell utilizaba redes semánticas[5] para mantener un dialogo con el alumno en formato textual utilizando un subconjunto de palabras del lenguaje natural. Luego aparecieron trabajos como Why o Sophie de Collins y Brown (Stevens et al., 1977).

Hasta ese entonces los términos CAI (o Intelligent CAI) y los Sistemas Tutores Inteligentes (STI) eran considerados sinónimos (Villareal Goulart et al., 2001), actualmente estos tipos se encuentran el área de IA-Education (Giraffa et al., 1999).

A través del análisis de la aplicación Scholar, Collins (Stevens et al., 1977) presentó el proyecto Why, donde utilizó el método socrático de enseñanza siguiendo los lineamientos propuestos por Carbonell (Carbonell, 1970). En este caso el tutor socrático enseñaba a través de la exposición indirecta, mediante las cuestiones sucesivas que se requieren para formular los principios principales para poder analizar y evaluar las hipótesis, descubrir las contradicciones y finalmente realizar las inferencias correctas.

En la década de los ochenta, el creciente interés en los modelos didácticos computacionales diversificados, amplió el campo en el conocimiento y entendimiento del tutor humano, cuya complejidad no había sido tratada computacionalmente. El tutor Quadratic (O'Shea, 1981) podía modificar sus propias estrategias, aunque era solamente una modificación de parámetros en lugar de un cambio efectivo de estrategias.

Luego, se desarrolló el sistema *Predicate Logic Advisory Tool* denominado Plato que enseñaba las matemáticas a los niños de nivel primario. Plato generaba problemas y presentaba "apuntes" al alumno. Uno de sus módulos estaba constituido por el juego *"How the West was won"*, que consistía en usar operaciones matemáticas, para avanzar; el juego incluye a West, el asesor inteligente que apoya al estudiante aprendizaje los conceptos matemáticos asociados al juego. Se lo puede ver como el primer entorno de aprendizaje completo (Castillo et al., 1998).

[5] Una red semántica es una estructura para la representación de conocimiento definida como un conjunto de nodos interconectados por arcos rotulados. Las redes de este tipo sólo captan las definiciones de los conceptos, también y de manera inherente proporcionan uniones entre otros conceptos (Chaiben, 1996].

El paradigma utilizado es *"Issues and Examples"* (temas pendientes y ejemplos). En él, se enumeraron todos los tipos de combinaciones de operadores, a los que se llaman "temas pendientes", es decir, lo que al alumno le resta por aprender (esta es la información que debe ser utilizada en el módulo tutor). Luego, se esperaba el momento oportuno para mostrarle los ejemplos de cómo aplicar el tema pendiente y el beneficio que esto representa. Este paradigma es aplicable, y se pueden construir STI en forma general estableciendo los conceptos a enseñar (los *"temas pendientes"*); y reconocedores de patrones para saber si el estudiante los domina o no (Castillo et al., 1998).

En la década de los noventa las contribuciones en áreas, como las redes de computadoras y los sistemas distribuidos y de comunicación proporcionaron nuevas herramientas que se podían aplicar a los STI, como por ejemplo: e-mail, los foros, Internet en general, con información On-Line; además de los recursos didácticos importantes, como videos, imágenes, videoconferencias, etc. (Boff et al., 2000; Fernández et al., 2000; Zeve et al., 2000).

Estas nuevas herramientas permitieron revalorizar el aprendizaje guiado por computadora. A principios del dos mil Konzen (Konzen, 1995) dio el primer paso para determinar el perfil del alumno, esto influenciará la exposición del contenido a través de un cuestionario que se presenta al alumno. También Kawasaki propone un modelo siguiendo distintos principios pedagógicos (Kawasaki, 2000) y en la actualidad existen implementaciones del módulo del estudiante usando de redes Bayesianas para el modelado.

La *Tabla 2.1* resume las principales características de los primeros Sistemas Tutores Inteligentes que aparecieron, desde principios de la década del setenta hasta mediados de la década de los noventa.

STI	Autor	Año	Dominio	Características
Scholar	Carbonell	1970	Geografía	Lenguaje natural
Why	Stevens, Collins	1977	Meteorología	Dialogo socrático
Sophie	Brown, Burton	1977	Electrónica	Interface NLP
Wusor	Goldstein	1979	Estrategia	Estructura superpuesta
Guidon	Clancey	1981	Tutor *Mycin*	Primer "glass-box"
West	Burton	1981	Estrategia	Entrenamiento y ejemplos
Buggy	Brown	1981	Aritmética	Conocimiento incorrecto
Debuggy	Burton, VanLehn	1982	Aritmética	Diagnostico off-line
Steamer	Stevens, Hollan	1983	Diseño de calderas	Simulación, modelo mental
LMS	Sleeman	1984	Algebra	Reglas MAL
Meno	Wolf	1984	Programación	Gerenciamiento de discurso
Proust	Jhonson	1984	Programación	Diagnostico de intenciones
ACTP	Anderson	1984	Tutor de Lisp	Modelado cognitivo
Sierra	VanLehn	1987	Aritmética	Predicción de errores
Sherlock	Lesgold, Katz	1991	Electrónica	Aprendizaje cognitivo
CircSim	Evens, Michael, Rovick	1996	Medicina	Realmente utilizado

Tabla 2.1: *Características de los STI entre los 70' y 90'.*
Fuente: *(Kinshuk, 1996) Effectiveness of Intelligent Tutoring Tools Interfaces in relation to Student, Learning Topic and Curriculum Characteristics.*

2.2.6.1 Técnicas Educativas Generales

La idea de utilizar la tecnología y particularmente la computación en el campo de la enseñanza tanto en las escuelas como en universidades no es nueva. La Instrucción Asistida por Computadora (CAI) hoy se ha integrado, alentada por el abaratamiento de los costos de hardware y el acceso a Internet (Nakabayashi, 1997; Alpert et al., 1999; Yang *et al.*, 2002). Durante el proceso de aprendizaje, se quiere promover la interacción entre el estudiante y el docente (en este caso personificado por el Sistema Tutor Inteligente) por lo que sería adecuado un sistema que permita *"múltiples estrategias"*. Esto le brindaría robustez y el nivel de adaptación a las características cambiantes de los estudiantes.

Castillo (Castillo *et al.*, 1998) establece que la pedagogía influye en el desarrollo de sistemas de enseñanza, los que deben basarse en el modelo del estudiante, así como las estrategias para enseñar efectivamente, en teorías de instrucción, por lo tanto abarca tanto al

modelo del estudiante y al modelo del tutor (estrategias de enseñanza). Castillo (Castillo *et al.*, 1998) sostiene que: *"La psicología interviene en el establecimiento de modelos cognitivos, necesarios para el entendimiento y aplicación apropiadas de los modelos de enseñanza citados"* a fin de dar cuenta que para enseñar se debe conocer como los sujetos conocen.

La necesidad de que la instrucción asistida por computadora se transforme en un apoyo efectivo para el tutelado, requiere de una base teórica que respalde el desarrollo de los sistemas para que estos sean efectivos en el proceso de enseñanza.

2.2.6.2 Teorías de Aprendizaje

Existen diferentes teorías que explican el proceso del aprendizaje, que se pueden agrupar en:

- ***Instruccional:*** Es donde el profesor o tutor es el centro y el encargado de diseñar el contenido, relegando al alumno a la función de *"aprender"* el conocimiento que el profesor le comunica. Este proceso de aprendizaje es altamente guiado por el profesor y no requiere la iniciativa personal del alumno.

- ***Aprendizaje por descubrimiento:*** Es donde el profesor tiene una tarea de *"explorador"*, entregándole al alumno las herramientas necesarias para que este descubra el conocimiento a su propio ritmo. Este proceso de aprendizaje no es guiado y requiere de iniciativa por parte del alumno, ya que es este el encargado de *"descubrir"* el conocimiento por sus propios medios, guiado por el tutor para que se puedan lograr los objetivos de la sesión pedagógica planteados, pero dando lugar a nuevos objetivos que surgen de la experimentación del alumno.

Los procesos de aprendizaje fueron tema de estudio desde siempre. Las primeras reflexiones las expone Platón cuando en su libro La República (Platón, 2001), explica el mito de las cavernas y aclarando que: *"el mundo que se conoce es la proyección de*

nuestras ideas innatas". A esto se lo conoce como pensamiento racionalista. Luego, Aristóteles, rechaza la teoría de las ideas innatas y establece que "*el conocimiento procede de los sentidos que dotan a la mente de imágenes que se asocian entre si según tres leyes: contigüidad, similitud y contraste, iniciando así el pensamiento empirista"*. Estas teorías son reflotadas nuevamente, posteriormente, por Santo Tomás de Aquino.

Las primeras aplicaciones de la CAI se dieron desde la visión conductista. Skinner año muy influenciado por las investigaciones de Pavlov[6] que define al conductismo no como una forma de estudiar la conducta sino como una filosofía de la ciencia dedicada al objeto y a los métodos de la psicología. Skinner plantea que la conducta de los organismos puede ser explicada a través de las contingencias ambientales, y los procesos internos de naturaleza mental no tienen ningún poder causal-explicativo (Hernández Rojas, 1998). En la actualidad el conductivismo ha evolucionado a modelos cognitivistas más flexibles y adecuados que ponen en primer plano al estudiante en el proceso educativo. A continuación se resumen los modelos más representativos y se resaltan sus características principales.

A) Teoría de la Instrucción de Gagné

En esta teoría, el aprendizaje se define como un proceso que permite a un organismo vivo modificar sus comportamientos en forma rápida y permanente; por tanto, el aprendizaje se verifica cuando existe un cambio de comportamiento, relativamente estable. Esto supone cuatro elementos: un aprendiz, una situación o entorno que permite el aprendizaje, un comportamiento explícito del aprendiz y un cambio interno. Esta teoría establece que hay varios tipos de aprendizajes y que por lo tanto son necesarios diferentes modos de instrucción. Estos tipos son: información verbal, destrezas intelectuales: discriminar, formar conceptos, aprender reglas, estrategias cognitivas, actitudes y destrezas motoras.

[6] Pavlov, médico ruso, le presentó la comida a los perros de su laboratorio y la introducía con el sonido de una campana, por medio de mediciones, luego un tiempo comprobó que el sonido de la campana provocaba salivación en los caninos, aunque el sonido no viniese acompañado de comida. Luego de años evaluando los resultados experimentales, estos darían lugar a la teoría conocida como *Condicionamiento Clásico* de Pavlov, que aportaría, mas adelante, las bases para el Conductismo, escuela psicológica que pretende explicar y predecir la conducta y que aporta luz sobre muchos de los aprendizajes no sólo en los perros, también en los humanos.

Para cada tipo de aprendizaje son necesarias diferentes condiciones internas y externas. Una vez que plantea estos factores que afectan al aprendizaje, Gagné puede plantear al proceso de aprendizaje como un sistema, donde dichas entradas afectan la salida; y por lo tanto, el profesor puede mejorar la salida controlando las entradas. Gagné sugiere agregar el concepto de jerarquía como base para su descripción del funcionamiento del sistema.

B) Teoría de Equilibración de Piaget

Para Piaget (Piaget, 1989) existen dos tipos de aprendizaje, uno el aprendizaje en sentido estricto, a través del cual se consigue información específica y otro, el aprendizaje en sentido amplio, que consiste en el progreso de las estructuras de cognitivas. El aprendizaje se produce cuando se presentara un desequilibrio o conflicto cognitivo que da lugar a dos procesos: la asimilación y la acomodación.

La asimilación es la integración de elementos exteriores a estructuras en evolución o ya acabadas en el organismo (Piaget, 1989). El individuo interpreta la información del medio de acuerdo a sus conceptos disponibles. A través del proceso de acomodación, el sujeto adapta sus conocimientos a la realidad en tanto y en cuanto contrasta con ella lo asimilado; por otra parte se produce el cambio de estructuras cognitivas del individuo en dos sentidos, el primero como consecuencia de la suma de nuevos conceptos y el segundo como consecuencia de una reinterpretación de lo ya sabido a la luz de los nuevos conocimientos.

La teoría de Piaget reduce todo el aprendizaje a adquisiciones espontáneas y necesarias. Esto es un fallo puesto que la mayor parte de los conceptos en realidad no son necesarios y además, no pueden adquirirse sin la intervención de la cultura y la instrucción, es decir no son espontáneos.

C) Teoría socio - cultural de Vigotsky

Vigotsky (Vigotsky, 1926) plantea que *"estudiar las conducta del hombre sin lo psíquico, como pretende la psicología, es tan imposible como estudiar lo psíquico sin la conducta.*

Por tanto no hay lugar para ciencias distintas. El estado actual de las dos ramas del saber plantea insistentemente la cuestión de tal necesidad y fecundidad de la completa fusión de ambas ciencias".

El aprendizaje sucede a partir de la interacción social, puede establecerse que: el individuo reconstruye su saber debido a que se entremezclan procesos de construcción personal y procesos en colaboración con los otros que intervinieron. En este el maestro debe ser un agente facilitador y mediador de la cultura. En la *Tabla 2.2* se puede ver el paralelismo entre la concepción de Piaget y la de Vigotsky.

Teoría de Piaget	Teoría de Vigotsky
El conocimiento es un proceso de interacción entre el sujeto y el medio entendido físico únicamente	El conocimiento es un proceso de interacción entre el sujeto y el medio entendido social y culturalmente
El ser humano al nacer es un individuo biológico	El ser humano al nacer es un individuo social
En el desarrollo del ser humano hay un proceso de socialización	En el desarrollo del ser humano hay un proceso de diferenciación social
La potencialidad cognoscitiva del sujeto depende de la etapa del desarrollo en la que se encuentre.	La potencialidad cognitiva del sujeto depende de la calidad de la interacción social y de la ZDP del sujeto
El ser humano al nacer se encuentra en un estado de desorganización que deberá ir organizando a lo largo de las etapas del desarrollo de su vida	El ser humano al nacer tiene una percepción organizada puesto que está dotado para dirigirla a estímulos humanos y para establecer interacciones sociales

Tabla 2.2: *Paralelismo entre la concepción de Piaget y la de Vigotsky.*
Fuente: *(Vigotsky, 1978)Mind in society, Harvard University Press Cambridge M.A.*

D) Teoría del Andamiaje de Bruner

Esta teoría está relacionada con la de Piaget, ya que enfatiza el aprendizaje por descubrimiento. El alumno descubre, pues esto le causa mayor satisfacción y obtiene una mayor retención. Llega un paso más adelante la teoría de Piaget, desarrollando un proceso de instrucción donde el profesor debe crear un ambiente que favorezca lograr el descubrimiento (Bruner, 1990). Propone que los contenidos a enseñar se presenten como un conjunto de problemas y relaciones que el alumno debe resolver a fin de que se interese y resulte significativo el aprendizaje (Guardia Robles, 1993).

Bruner (Bruner, 1990) explica cómo el estudiante llega a dominar los contenidos que el docente le propone construyendo el concepto de andamiaje, estrechamente ligado al de zona de desarrollo próximo de Vigotsky. Según esta idea, en el proceso interacción y diálogo en que se basa la enseñanza, el experto tiende un conjunto de andamios o ayudas por medio de las cuales el alumno elabora las construcciones necesarias para aprender los contenidos. El andamiaje debe contemplar los rasgos particulares del alumno. Y se irán regulando en la medida en que vayan progresando en sus habilidades.

Así, algunos requerirán apoyos como explicaciones, modelos, etc., mientras que otros necesitarán apoyos más complejos. Progresivamente, el maestro ajustará el sistema de ayudas y apoyos necesarios conforme se desarrollen las habilidades de los alumnos, y así poder ir cediendo el control y el manejo de los contenidos y formas de discurso a aprender.

E) Teoría del Aprendizaje Significativo de Ausubel (Ausubel et al., 1983)

Trata de la adquisición de aprendizaje significativo, por lo tanto no considera el aprendizaje mecánico (de memoria). Para considerar que un aprendizaje es significativo, requiere tener un sentido para ser incorporado al conjunto de conocimientos que el sujeto posee (el concepto aprendido debe relacionarse con conceptos previamente adquiridos). También considera al aprendizaje como receptivo, ya que es el profesor quien establece los contenidos para facilitar la organización del nuevo contenido en la estructura mental del alumno.

Ausubel establece que para que se produzca la reestructuración es necesaria una instrucción que presente la información organizada y explícita que desequilibre las estructuras existentes. Un aprendizaje es significativo si puede incorporarse a las estructuras de conocimiento, es decir los nuevos conceptos adquieren significado para el sujeto dentro de sus estructuras cognitivas. Un aprendizaje es memorístico cuando los conceptos aprendidos no están relacionados entre sí y carecen de significado para la estructura del sujeto.

En general un aprendizaje significativo es más eficaz que el memorístico puesto que: produce retención más duradera, facilita nuevos aprendizajes y produce cambios que persisten más allá del olvido de los detalles concretos. Otro aporte de Ausubel es el concepto de los organizadores previos, sobre los cuales el alumno se apoya, de tal forma que éstos hacen las veces de estructura o andamio entre los conocimientos a adquirir y los que ya poseía.

F) Inteligencias Múltiples de Howard Gardner.

Esta teoría de la inteligencia humana, desarrollada por el psicólogo Howard Gardner (Gardner, 1993), establece que hay al menos ocho formas en las que las personas perciben y entienden el mundo. Gardner llama a cada una de estas formas inteligencias (conjunto de habilidades que permiten resolver problemas de la realidad).

Gardner (Gardner, 2003) define una inteligencia como *"La capacidad de resolver problemas, o de crear productos, que sean valiosos en uno o más ambientes culturales"*. Este autor fundamenta su estructura en pruebas biológicas y antropológicas y, más específicamente, en bases neurológicas, evolucionistas y transculturales. No obstante, el autor aclara que es un acercamiento que no establece las fuentes de tales capacidades o los medios para medir éstas. Gardner (Gardner, 1993) presenta una clasificación de inteligencias, aunque sugiere que no es terminante:

- Verbal-lingüística.
- Lógico-matemático.
- Visual-espacial.
- Kinestésica.
- Musical-Rítmica.
- Interpersonal.
- Intrapersonal.
- Naturalista.

Para Gardner (Gardner, 1993): *"El objetivo de la escuela debe ser el de desarrollar las inteligencias y ayudar a la gente a alcanzar los fines vocacionales y aficiones que se adecuen a su particular espectro de inteligencias"*. Este objetivo se consigue con una

escuela centrada en los individuos, donde la evaluación de las capacidades y las tendencias individuales es continua.

2.2.6.3 Técnicas Educativas Aplicables

Una vez presentadas las técnicas educativas más importantes, se debe analizar cuáles de ellas son aplicables a la enseñanza de lenguajes de programación con paradigma lineal estructurado y dentro de ellas, cuales se pueden implementar en un asesor inteligente o un sistema tutor inteligente.

Cualquiera de las teorías de enseñanza explicadas en los apartados anteriores resulta útil para entender el proceso de aprendizaje de lenguajes de programación. La selección de un marco teórico adecuado solo puede justificarse por medio de razones pedagógicas, correspondientes al tipo de alumnos con el que se va a enfrentar al sistema y a las técnicas de programación o características disponibles para el futuro sistema (Perkins, 1995).

A continuación se estudiará la aplicabilidad de cada uno de los modelos anteriormente detallados a un Sistema Tutor Inteligente (STI) aplicado a la enseñanza de la programación.

Como no es el propósito de esta tesis evaluar los distintos modelos de enseñanza y de aprendizaje, por lo que se seleccionó la Teoría Uno (Perkins, 1995) que no es un modelo, ni un método de enseñanza, sino un conjunto de recomendaciones compatibles con cualquier teoría. Básicamente, ella estipula que: “La gente aprende más cuando tiene una oportunidad razonable y una motivación para hacerlo” y para aplicarla se deben reunir las siguientes condiciones (Perkins, 1995):

Información clara a través de descripción, ejemplos de los objetivos, conocimientos requeridos y de los resultados esperados.
Práctica reflexiva, es decir, oportunidad para el alumno de ocuparse activa y reflexivamente de aquello que deba aprender.

Realimentación informativa a través de consejos claros y precisos para que el alumno mejore el rendimiento y pueda proceder de manera más eficaz.

Fuerte motivación intrínseca y extrínseca mediante actividades ampliamente recompensadas, sea porque son muy interesantes y atractivas en sí mismas o porque permiten obtener otros logros que importan al alumno.

Por lo tanto se tomará la Teoría Uno planteada por Perkins (Perkins, 1995) como la base teórica del Sistema Tutor Inteligente (STI), de la que se toman las bases de las distintas formas de enseñanza disponibles. Perkins (Perkins, 1995) plantea que si se combinan las condiciones que estipula la Teoría Uno con cada uno de los programas de estudio, se obtienen los métodos respectivos. En otras palabras, la Teoría Uno se "personifica" de distintas maneras según el programa del momento. Las cuatro formas principales se enumeran a continuación y se explican más detalladamente en las secciones siguientes:

- La instrucción didáctica (o magistral).
- El entrenamiento.
- La enseñanza socrática.
- Otras.

A) La Instrucción Didáctica.

Según Perkins (Perkins, 1995) es la presentación clara y correcta de la información por parte de los maestros y de los textos. Su objetivo se centra especialmente en la explicación donde se exponen *"los qué"* y *"los por qué"* de un determinado tema. Es fundamental que se aclaren los componentes básicos de una buena explicación, ya que en la instrucción didáctica el alumno queda relegado a un papel de sujeto pasivo y el tutor o maestro queda en primer plano de proveedor único de los saberes y conocimientos. Algunas de las características que debe reunir una buena explicación en la práctica educativa pueden enumerarse a continuación:

- ***Identificación de objetivos para los alumnos:*** Este paso quizás es uno de los más importantes ya que establece cuál es el objetivo de la sesión pedagógica que se está llevando a cabo y qué es lo que se pretende obtener como resultado para los estudiantes una vez finalizada dicha sesión.

- ***Supervisar y señalar el avance hacia los objetivos:*** Se debe verificar a través de preguntas clave para la lección que los conocimientos que se han explicado han sido correctamente interpretados por el estudiante y que este puede seguir adelante hacia los objetivos de la lección sin correr el riesgo de que concepciones erróneas puedan llegar a modificar la interpretación por parte del alumno de conocimientos posteriores.

- ***Mostrar numerosos ejemplos sobre los conceptos analizados:*** Los ejemplos son la herramienta pedagógica más útil a la hora de verificar si los estudiantes realmente entendieron el concepto explicado, así como también para incorporar a la estructura cognitiva dicho concepto, yendo desde una abstracción mayor, como lo es el concepto, hasta instancias particulares de dicho concepto.

- ***Legitimar un nuevo concepto o procedimiento mediante principios ya conocidos por los alumnos:*** Un elemento pedagógico realmente útil es la relación entre conceptos conocidos por el estudiante con el concepto que se está intentando explicar. Perkins (Perkins, 1995) da un ejemplo típico de tal cosa para ilustrar esta característica de la instrucción didáctica:

    - **Maestro:** *"¿El concepto de nicho es realmente útil para hablar de los sistemas ecológicos? Bien, examinemos esta cuestión. Pensemos en otras situaciones en las que hablamos de funciones dentro de un sistema; por ejemplo, las funciones de las personas en una empresa o en la escuela."*

B) El Entrenamiento.

Existe un vínculo entre el entrenamiento y la instrucción didáctica y por eso se lo presenta en segundo lugar. Sin una instrucción didáctica que presente cierta base de información

clara sobre los conceptos nuevos, los estudiantes carecerían de los conocimientos básicos para realizar las ejercitaciones prácticas. Surge entonces una aparente contradicción, que es, dada la claridad informativa de la instrucción didáctica, cuál pasa a ser la función del maestro, si en una forma ideal, la instrucción didáctica aclaró todas las dudas posibles sobre el concepto explicado. El entrenamiento ofrece una respuesta, ya que hace hincapié en dos de las condiciones planteadas por la Teoría Uno: la práctica reflexiva y la realimentación informativa. Las principales actividades del docente entrenador consiste en:

- Asignar prácticas.
- Alentar a los alumnos a reflexionar sobre lo que están haciendo.
- Ofrecer la realimentación.

Al mismo tiempo, se espera que el docente entrenador suministre la información en forma clara mientras que es la relación entre el entrenador y sus alumnos la que debe fomentar mecanismos de motivación de estos últimos y garantizar así una de las cuestiones fundamentales planteadas por la Teoría Uno que afirma que *"La gente aprende más cuando tiene una oportunidad razonable y una motivación para hacerlo".*

Perkins (Perkins, 1995) plantea una analogía muy clara con respecto al fútbol: *"El entrenador observa desde afuera el desempeño de los deportistas y les da consejos. Elogia los puntos fuertes, detecta los débiles, hace observar ciertos principios, ofrece guía e inspiración y decide qué tipo de prácticas se deben enfatizar"*. Y es esta función tan importante la que se extrapola y se aplica en cualquier dominio, sea este fútbol, clase de matemática o literatura.

C) La Enseñanza Socrática.

Ambos métodos vistos con anterioridad (la instrucción didáctica y el entrenamiento) poseen un aspecto regulativo, pues su función consiste en moldear y guiar las actividades de los alumnos y como se explicó anteriormente se relega al estudiante a un papel secundario,

teniendo como única función la de asimilar los conocimientos que se le presentan por el maestro tutor o el maestro entrenador.

En este marco surge el método socrático, a través del cual se logra que los estudiantes trabajen de una manera más flexible, pero no en forma libre, sino que continuamente estén recibiendo el apoyo en sus investigaciones por parte de un maestro o tutor, pero sin el esquema más rígido en el que el tutor les dice todo el tiempo lo que tienen que hacer. Perkins (Perkins, 1995) plantea que de esta forma el estudiante debe aprender no solo las respuestas sino que también el *"arte de las preguntas"*. La enseñanza se realiza cuando el maestro socrático plantea un enigma conceptual e incita a investigar el asunto de manera libre por parte de los estudiantes, haciendo preguntas del tipo:

- *¿Qué piensan al respecto?*
- *¿Qué posición se podría tomar?*
- *¿Qué definiciones necesitamos?*

Luego de la investigación inicial se proponen ideas, criterios y definiciones, donde el maestro actúa como incitador y moderador en la conversación: presta ayuda cuando las paradojas estancan el proceso de aprendizaje y genera contraejemplos y potenciales contradicciones cuando percibe en los estudiantes una satisfacción prematura. De esta manera el aprendizaje es "guiado" por los tutores hasta alcanzar el objetivo, ya que sin una guía eficaz los estudiantes pueden perderse por las ramas, pero esto no se hace de una manera rígida como en la instrucción didáctica y el entrenamiento, ya que el método socrático espera que sea el mismo estudiante el que realice los descubrimientos por sí mismo. El científico cognitivo Collins (Perkins, 1995) analizó los pasos fundamentales del método socrático que se describen a continuación:

- Se seleccionan ejemplos positivos y negativos para ilustrar las cualidades pertinentes al tema en consideración.
- Se varían los casos sistemáticamente a fin de centrar la atención en datos específicos.

- Se emplean contraejemplos para poner en tela de juicio las conclusiones del alumno.
- Se proponen casos hipotéticos para que el alumno reflexione sobre situaciones afines que podrían no ocurrir naturalmente.
- Se utilizan estrategias de identificación de hipótesis a fin de forzar la articulación de una hipótesis específica de trabajo.
- Se emplean estrategias de evaluación de hipótesis para fomentar la evaluación crítica de predicciones e hipótesis.
- Se promueve la identificación de otras predicciones que podrían explicar el fenómeno en cuestión.
- Se utilizan estrategias capciosas para inducir al alumno a hacer predicciones incorrectas y formulaciones prematuras.
- Se procura que el alumno deduzca las consecuencias hasta llegar a una contradicción para que aprenda a construir teorías válidas y consistentes.
- Se cuestionan las respuestas provenientes de autoridades tales como el maestro y el manual a fin de promover el pensamiento independiente.

Uno de los conceptos claves de la Teoría Uno, es que el maestro socrático *no provea abundancia de datos, pero controle la claridad y la calidad de la información suministrada por los alumnos haciéndoles preguntas*. Cuando los alumnos discuten entre sí sobre una determinada cuestión, el maestro socrático les exige una práctica continua de reflexión. Provee además, realimentación inmediata por medio de estímulos y críticas, e incita a que todos los participantes de la conversación hagan lo mismo.

D) Otros Métodos de Enseñanza.

Los protocolos pedagógicos propuestos por la Teoría Uno no son los únicos acercamientos que proveen las características requeridas por la Teoría Uno: *"información clara, práctica reflexiva, realimentación informativa y fuerte motivación intrínseca y extrínseca"*. Existen otras aproximaciones como pueda serlo el aprendizaje cooperativo y la colaboración entre pares, que vale la pena resaltar en el marco de los asesores inteligentes, donde el sistema

debe reaccionar como un *"estudiante virtual"* que interactúa con los estudiantes usuarios del sistema para resolver los distintos problemas que se le presentan y para lograr definiciones básicas para los conceptos que se pretenden enseñar en una sesión pedagógica a través del "andamiaje" (Ausubel et al., 1983; Guardia Robles, 1993) como lo hará un ayudante alumno.

Perkins (Perkins, 1995) sostiene que los niños aprenden mucho mejor en grupos cooperativos bien configurados que en soledad. Por lo general, las agrupaciones cooperativas pueden ayudar a lograr determinados fines, sin la planificación cuidadosa que requieren los métodos como la instrucción didáctica y el entrenamiento. Los resultados obtenidos son satisfactorios por que el aprendizaje cooperativo exige que todos los participantes se hagan responsables del desempeño del grupo.

El aprendizaje cooperativo y la colaboración entre pares pueden utilizar la dinámica de grupos para promover el aprendizaje reflexivo, donde los estudiantes piensan y discuten los conceptos que se les intenta enseñar y los problemas que se deben resolver. En el aprendizaje cooperativo y colaborativo la predisposición hacia la resolución de la tarea brinda la motivación del contacto social para mantener a los estudiantes interesados en sus actividades académicas.

Entonces parecería que en el caso del aprendizaje cooperativo y colaborativo el docente no posee un rol que deba cumplir y por lo tanto este método de enseñanza no sería aplicable en los Sistemas Tutores Inteligentes, pero un análisis más profundo revela que los docentes tienen las siguientes tareas (Perkins, 1995):

- El maestro "entrena" a los alumnos para que juzguen con criterios propios adecuados a la circunstancia planteada.
- El maestro debe "aumentar" el interés intrínseco de los problemas que el mismo presenta.

- El maestro puede “pedir a los alumnos que se califiquen mutuamente”, pero ofreciendo una fuerte realimentación con respecto a qué tipo de respuestas tienen más o menos sentido y por qué.

Esta teoría sobre la motivación intrínseca supera a la Teoría Uno por una razón muy sencilla: la Teoría Uno no dice nada sobre las interacciones entre la motivación intrínseca y la motivación extrínseca; sin embargo, hay muchas interacciones importantes por las que hay que preocuparse cuando se quiere brindar una fuerte motivación intrínseca.

También podrían tenerse en cuenta la valoración de las inteligencias múltiples, desarrolladas por el psicólogo Howard Gardner (Gardner, 1993), alegando que las concepciones convencionales de la inteligencia humana basadas en el coeficiente intelectual son demasiado monolíticas. Según Gardner, la inteligencia humana posee siete dimensiones diferentes o siete inteligencias y a cada una de ellas le corresponde un determinado sistema simbólico (ver sección 2.3.6.3 – Teorías de Aprendizaje).

Por lo tanto se podría desarrollar un protocolo pedagógico específico para cada una de estas inteligencias múltiples, donde, por ejemplo para un individuo con una inteligencia del tipo lógico-matemática, los problemas se le presenten en forma de notación formal matemática, mientras que para el resto de las inteligencias se les presente en lenguaje natural, para maximizar la comprensión del problema en cada uno de los casos.

Gardner (Gardner, 1993) señala que la práctica educativa convencional se centra fundamentalmente en la inteligencia lingüística y matemática, pero dado el carácter múltiple de la inteligencia humana se debería dar cabida a las diversas habilidades de las personas. Esto puede incluirse en el Sistema Tutor Inteligente (STI) a través de distintos protocolos pedagógicos que se adecuen de la mejor manera a cada uno de los estudiantes.

2.2.7 Modelos de Implementación de Sistemas Tutores Inteligentes (STI)

Las técnicas convencionales de programación para desarrollar Sistemas Tutores Inteligentes (STI), se basan en el planteo de los módulos básicos, que interactúan entre sí y donde cada uno de ellos posee la información necesaria para poder cumplir con su tarea. Esta definición de módulos dentro de un sistema tutor inteligente es útil a la hora de identificar cada una de las partes de éste, pero es casi imposible, hasta el momento, que en las implementaciones de los Sistemas Tutores Inteligentes estos módulos estén definidos completamente como lo indica la teoría que los soporta, ya que siempre existe un solapamiento con el dominio del problema a resolver con todos los demás módulos.

La mayoría de los Sistemas Tutores Inteligentes (STI) no presenta el nivel esperado de *"inteligencia"* (Bolan Frigo et al., 2004). Independientemente de las técnicas de programación utilizadas, la dificultad de modelar cómo funciona la mente humana impide que muchos de los Sistemas Tutores Inteligentes alcancen sus objetivos. Si bien existen versiones que aseguran que el modelado de Sistemas Tutores Inteligentes (STI) con el paradigma de programación multiagentes es el mejor camino para construir el modelo pedagógico, como la de Bolan Frigo, Pozzebon y Bittencour (Bolan Frigo et al., 2004), no existen pruebas sólidas de que esto sea así, ya que existen problemas de comunicación entre los agentes que dificultan su cooperación.

Además, tanto el paradigma de agentes como el orientado a objetos comparten el pasaje de mensajes para realizar la comunicación y el empleo de la herencia y la agregación para definir su estructura. Ahora, las principales diferencias entre estos paradigmas estriban en que los mensajes de los agentes tienen un tipo predeterminado, en el uso de actos comunicativos y en que el estado mental del agente se basa en sus creencias, intenciones, acuerdos, deseos y demás motivaciones (módulo BDI) (Iglesias et al., 2001).

Una de las ventajas principales, según Iglesias, Garijo y Gonzalez (Iglesias et al., 2001) es la popularidad de las metodologías orientadas a objetos, para las que existen muchas metodologías para implementar este paradigma, como es el caso de OMT (Object Modeling

Technique), OOSE (Object Oriented Software Engineering), OOD (Object Oriented Design), RDD (Responsibility Driving Design) y UML (Unified Modeling Language) entre otras. La experiencia adquirida con la utilización de estas metodologías desde hace por lo menos una década puede dificultar la adopción de las metodologías de agentes por la *"inercia"* que se genera en la industria, hasta tanto se disponga de una propia.

Por lo tanto, 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 las estructura compuesta por Carbonell (Carbonell, 1970) o alguna de sus modificaciones posteriores (Salgueiro et al., 2004; Costa et al., 2004).

2.3 AGENTES INTELIGENTES

Básicamente, un agente es una entidad de software que exhibe un comportamiento autónomo y un sistema multi-agente es un conjunto de agentes que tienen la capacidad de interactuar en un entorno común. Así, agentes en un entorno con otros agentes poseen capacidades como la comunicación, negociación, y coordinación. Características que podemos considerar opcionales son encontradas en varios tipos de agentes, como la movilidad y, la necesidad de interacción con usuarios y el consiguiente aprendizaje de su comportamiento.

2.3.1 ¿Qué es un Agente?

Resulta tan embarazosa como la pregunta ¿qué es la inteligencia? El problema es que, aunque el término sea ampliamente utilizado por mucha gente, desafía los intentos de establecer una definición única y universalmente aceptada. Esto no debe ser necesariamente un problema: si se desarrollan con éxito muchas aplicaciones interesantes y útiles, entonces apenas importa que no se establezca un acuerdo sobre detalles terminológicos potencialmente triviales. Sin embargo, si no intentamos definirlo, existe el peligro de que la palabra "agente" pueda convertirse en un término mal empleado, confundiendo a la

comunidad de investigadores. Se pueden encontrar una gran cantidad de definiciones. Se presentan algunas para examinarlas y compararlas. Desde una perspectiva amplia y poco limitada:

"Los agentes inteligentes son programas que realizan tareas interactivas, dirigidas por la petición y los deseos de los usuarios. Poseen un grado de autonomía e independencia, en virtud del cual pueden realizar una serie de tareas sin que las personas u otros agentes les dirijan en cada paso que dan en su camino."

El Agente AIMA (Russell y Norving 1995)

"Un agente es cualquier cosa que pueda ser vista percibiendo su entorno a través de sensores y actuado hacia el entorno a través de unos efectores."

El Agente Maes (Maes 1995)

"Los agentes autónomos son sistemas computacionales que habitan en entornos dinámicos complejos, percibiendo y actuando autónomamente en ese entorno, y realizan un conjunto de metas o tareas para las que han sido diseñados."

El Agente KidSim (Smith, Cyper y Spohrer)

"Definiremos un agente como una entidad software persistente dedicada a un propósito específico. La 'persistencia' distingue a los agentes de las subrutinas; los agentes tienen sus propias ideas sobre cómo ejecutar tareas, sobre sus agendas. Con 'propósito específico' se distinguen los agentes de las aplicaciones multifunción, que son típicamente más pequeños."

El Agente Hayes-Roth (Hayes-Roth 1995)

"Los agentes inteligentes realizan continuamente tres funciones: percibir condiciones dinámicas en el entorno, actuar afectando a las condiciones del entorno, y razonar para interpretar lo percibido; resuelven problemas, muestran interfaces y determinan acciones"

El Agente IBM[7]

"Los agentes inteligentes son entidades software que llevan a cabo un conjunto de operaciones en beneficio de un usuario u otro programa con algún grado de independencia o autonomía, y haciendo esto, emplean algún conocimiento o representación de las metas y deseos del usuario"

El Agente Wooldridge y Jennings

"Es un hardware o más comúnmente un sistema software basado en computador que disfruta de las siguientes propiedades:

- *autonomía: los agentes operan sin la intervención directa de personas u otros, y tienen algún tipo de control sobre sus actuaciones y estado interno.*
- *habilidad social: los agentes interactúan con otros agentes (posiblemente humanos) vía algún tipo de lenguaje de comunicación de agentes.*
- *reactividad: los agentes perciben el entorno o ambiente, (lo que representa la palabra físicamente, un usuario vía una interfaz de usuario, una colección de otros agentes, Internet, o quizás todos ellos combinados), y responde rápidamente a cambios que ocurren en dicho entorno.*
- *pro-actividad: los agentes no actúan simplemente en respuesta a su entorno, sino que son capaces de exhibir 'comportamiento dirigido hacia el objetivo', tomando la iniciativa"*

El Agente FAQ:

- *Agentes autónomos: son programas que viajan de un sitio a otro, decidiendo ellos mismos cuándo moverse y lo que hacer. (ej. Telescript de General Magic). Es importante destacar que sólo pueden viajar entre servidores especiales.*
- *Agentes Inteligentes: son programas que ayudan a los usuarios a hacer cosas, tales como elegir un producto, dar una guía al usuario a través de un formulario rellenado, o incluso ayudar a los usuarios a hacer sus búsquedas.*

[7] http://activist.gpl.ibm.com:81/WhitePaper/ptc2.htm

- *User-Agent: es el nombre técnico para los programas que realizan tareas de red para los usuarios, tales como Web User-agents como Netscape Navigator o Microsoft Internet Explorer, y Email User-Agent como Qualcomm Eudora etc.*

En estas definiciones, la propiedad de autonomía tiene gran importancia, y se obvia el término inteligente. Los estudiosos del tema no se ponen de acuerdo, pues algunos valoran mucho la autonomía y otros en cambio piensan que ya existían sistemas autónomos; como el control de procesos y la monitorización, y que la propiedad que caracteriza a un agente es la inteligencia.

La cuestión de qué es un agente, está aún siendo debatida, corriendo el riesgo de que cualquier programa sea denominado agente. Se pueden distinguir dos nociones extremas de agentes:

- *Una noción débil de agente consiste en definir un agente como a una entidad que es capaz de intercambiar mensajes utilizando un lenguaje de comunicación de agentes (CIKM*8*, 1994). Esta definición es la más utilizada dentro de la ingeniería software basada en agentes, cuyo fin es conseguir la interoperabilidad entre aplicaciones a nivel semántico utilizando la emergente tecnología de agentes.*

- *Una noción más fuerte o restrictiva de agente es la enunciada por Shoham (Shoham et al, 1993) en su propuesta de programación orientada a agentes (AOP), donde un agente se define como una entidad cuyo estado es visto como un conjunto de componentes mentales, tales como creencias, capacidades, elecciones y acuerdos.*

Después de esbozar la definición de agente inteligente, podemos extenderla de manera natural a Sistema basado en agentes, que será el sistema que utiliza agentes para realizar su

[8] International Conference on Information and Knowledge Management (Conferencia Internacional en la Información y Dirección de Conocimiento).

función. Si este sistema utiliza varios agentes, y éstos colaboran juntos e interactúan para resolver un problema, estaremos ante un Sistema multiagente.

Debido a la controversia sobre la definición de agente (ya se ha visto varias definiciones), se ha optado por enumerar un conjunto de propiedades que lo caracterizan: autonomía, sociabilidad, reactividad, iniciativa, movilidad, veracidad, benevolencia y racionalidad. Un agente, sin embargo, no tiene porqué poseerlas todas.

Varios investigadores y grupos de investigación han definido el término de agente desde diferentes puntos de vista, esto ha influido a que en la actualidad existan diferentes definiciones de lo que es un agente. La dificultad se debe especialmente a que los agentes se pueden emplear en muchos dominios de aplicación, llevando consigo a que cada dominio asocie varios y diferentes atributos a la definición de agente. Por lo tanto es conveniente dar una corta definición de agente que no involucre las características que debe tener un agente inteligente.

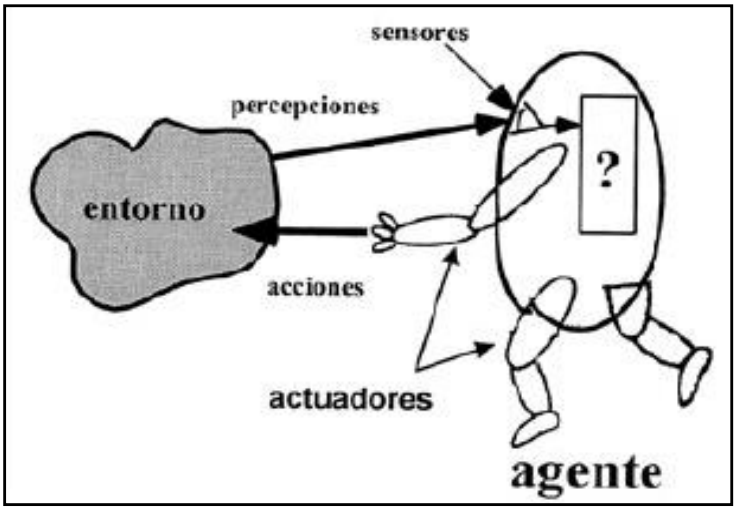


Figura 2.3: *Descripción del agente*
Fuente: *http://www.mipagina.cantv.net/vmendillo/Tesis/Agentes.html*

Un agente es un sistema que está en algún ambiente y que es capaz de tomar acciones autónomas de acuerdo al estado del ambiente para cumplir sus objetivos de diseño. Lo anterior no quiere decir que el agente tenga dominio completo del ambiente, por el

contrario en la mayoría de situaciones este es muy complejo y el agente solo tendrá un control parcial de este. Así mismo puede suceder que para un estado del ambiente muy similar, dos acciones diferentes tomadas por el agente produzcan efectos heterogéneos en el ambiente. Las acciones que puede tomar el agente se ven afectadas por las diferentes propiedades del ambiente.

2.3.2 CLASES DE AGENTES[9]

2.3.2.1 Agentes Colaborativos

Este tipo de agentes se enfatiza en la autonomía y las habilidades sociales con otros agentes para ejecutar las tareas de sus usuarios. La coordinación de los agentes se logra mediante la negociación para alcanzar acuerdos que sean aceptables para los agentes negociantes.

Los agentes colaborativos son capaces de actuar racionalmente y autónomamente en ambientes multi-agente y con restricciones de recursos. Otras características de estos agentes es que poseen habilidades sociales, son proactivos, benévolos, estáticos y veraces. Se pueden utilizar en la solución de algunos de los siguientes problemas:

- Para resolver problemas que son muy grandes para un agente centralizado.
- Para permitir la interconexión e interoperabilidad de sistemas de IA existentes como sistemas expertos, sistemas de soporte de decisión etc.
- Solucionar problemas que son inherentemente distribuidos.
- Proporcionar soluciones que simulen recursos de información distribuidos.
- Incrementar la modularidad, velocidad, confiabilidad, flexibilidad y reutilización en sistemas de información.

[9] MultiRobot Labs, Computer Science Department and Robotics Institute, Carnegie Mellon University pagina web vigente http://www-2.cs.cmu.edu/~multirobotlab/

2.3.2.2 Agentes de Interface

Los agentes de interfaz se enfatizan en la autonomía y la adaptabilidad para realizar tareas a sus usuarios. Este tipo de agentes básicamente presta soporte y asistencia a un usuario que está aprendiendo una nueva aplicación o nuevos conceptos. El agente puede aprender mediante alguna de las siguientes cuatro técnicas, observando y monitoreando la interfaz:

1. Por observación e imitación del usuario.
2. A través de una retroalimentación positiva o negativa del usuario.
3. Recibiendo instrucciones explicitas del usuario.
4. Asesorándose de otros agentes.

De esta manera el agente puede actuar como un asistente personal y autónomo del usuario, cooperando con él para terminar una cierta tarea.

2.3.2.3 Agentes móviles

Estos agentes se enfatizan en las habilidades sociales y la autonomía, a diferencia de los agentes cooperativos, estos son móviles.

Los agentes móviles son procesos de software que son capaces de transitar por una red, generalmente una WAN, interactuando con computadores alejados, reuniendo información para el usuario y volviendo a su origen cuando las tareas fijadas por el usuario se hayan completado. Las tareas que se pueden realizar son por ejemplo reservaciones de vuelos, manejo de una red de telecomunicaciones entre otras.

Los agentes móviles traen con si grandes beneficios aunque no son funcionales, esto quiere decir que una tarea que realiza un agente móvil puede ser realizada por un agente colaborativo, la diferencia radica en que para movilizar el agente se requiere de un costo muy alto de recursos. Algunas de las ventajas que se pueden obtener al usar agentes móviles son:

- *Reducen el costo de comunicación*, por ejemplo cuando en una ubicación hay un gran volumen de información que necesita ser examinada y transmitida, esto ocuparía una gran cantidad de recursos en la red y consumiría mucho tiempo. En este caso el agente móvil puede determinar la información relevante al usuario y transmitir un resumen comprimido de esta información.
- *Facilitar la coordinación*, es más sencillo coordinar un cierto número de requerimientos remotos e independientes al comparar solo los resultados localmente.
- *Reduce los recursos locales,* los agentes móviles pueden ejecutar sus tareas en computadores diferentes del local, de tal manera que no consuman recursos de procesamiento, memoria y almacenamiento en estos.
- *Computación asíncrona*, mientras que un agente móvil realiza su tarea el usuario puede ir realizando otra, de tal manera que después de un tiempo el resultado del agente móvil sea enviado al usuario.

2.3.2.4 Agentes de información

Los agentes de información nacieron debido a la gran cantidad de herramientas que surgieron para el manejo y recuperación de información. Los agentes de información tienen los roles de manejar, manipular, e integrar información de muchas fuentes de datos distribuidas.

La hipótesis fundamental de los agentes de información es que puedan mejorar de algún modo, pero no completamente el problema de la sobrecarga de información y en general el manejo de esta.

- ¿Cómo formular, describir, descomponer problemas y sintetizar resultados entre un grupo de agentes inteligentes?
- ¿Cómo permitir a los agentes comunicarse e interactuar?
- ¿Qué lenguajes de comunicación y protocolos se pueden usar?

- ¿Qué arquitectura es la más adecuada para construir Sistemas multi-agente prácticos?
- ¿Qué lenguajes y herramientas de desarrollo se pueden utilizar?
- ¿Cómo construir herramientas para soportar las metodologías de desarrollo?, etc.

Un Sistema Multi-Agente (SMA) es aquel que consiste de un número de agentes que interactúan entre sí. En un caso general, los agentes actúan en nombre de los usuarios, con diferentes objetivos y motivaciones. Para interactuar exitosamente, los agentes deben ser capaces de cooperar, coordinarse, y negociar con otros, tal como lo hacen los humanos.

2.3.3 Taxonomía[10] de los Agentes

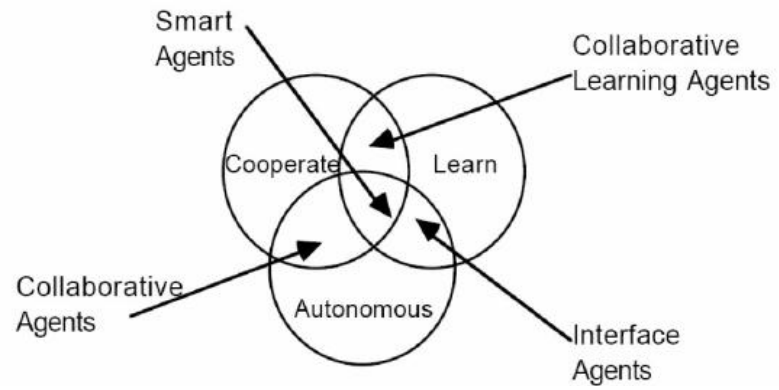


Figura 2.4: *Taxonomía de agentes*
Fuente: *http://es.wikipedia.org/wiki/Taxonomía*

"Se puede hacer una taxonomía más detallada de los agentes basándose en los siguientes criterios: inteligencia, autonomía, racionalidad, movilidad, comunicación y agencia. Vamos a empezar introduciendo estos términos[11] ":

Inteligencia: Es bastante difícil de definir con precisión. Cuando se aplica a los agentes, se podría definir la inteligencia como algunas capacidades de alto nivel, como la capacidad del agente para razonar sobre su conocimiento, aprender e inferir nuevos conocimientos, y

[10] La taxonomía (del griego ταξις, taxis, *"ordenamiento"*, y νομος, nomos, *"norma"* o *"regla"*) es, en su sentido más general, la ciencia de la clasificación (http://es.wikipedia.org/wiki/Taxonomía).
[11] Tecnología de agentes y sus aplicaciones Revista de Telecomunicaciones de Alcatel, Primer trimestre de 1999 http://www.mipagina.cantv.net/vmendillo/Tesis/Agentes.htm

planificar las acciones consecuentes para alcanzar sus objetivos. En este caso, la inteligencia tiene el mismo sentido que el término *"inteligencia artificial"*.

Autonomía: Existen diferentes formas de entender el término *"autonomía"*. En el sentido más común (absoluto), la autonomía se refiere a lo impredecible del comportamiento del agente: cuanto más impredecible sea, más autonomía aparecerá. Sin embargo, la autonomía absoluta no se aplica muy bien a los agentes ya que, de acuerdo con nuestra mínima definición, un agente tiene que servir para algunos propósitos que se la limitan. Un tipo más útil de autonomía es la *"autonomía social"*, que se refiere a la autonomía del agente cuando se pone en una sociedad de agentes. No obstante, unas relaciones más sociales, como la cooperación, coordinación y compromiso, limitan al agente. Finalmente, la ejecución de la autonomía se refiere a la capacidad del agente para elegir y llevar a cabo la acción adecuada en el momento oportuno (técnicas de planificación). De hecho, al diseñar un agente, se tiene que encontrar un fino equilibrio entre autonomía y servidumbre. Un nivel inferior puede consistir en un agente que represente a un usuario, como el bien conocido PDA (Asistente Personal Digital).

Racionalidad: Se aplica esencialmente a los agentes autónomos e inteligentes. Los agentes racionales tienden a elegir y realizar acciones que maximicen su esperada utilidad en función de sus percepciones y de nuestro propio conocimiento. La racionalidad también implica que la acción elegida sea consistente con los deseos y creencias del agente.

Movilidad: Es la capacidad que tiene un agente para iniciar su ejecución en cualquier sitio, e irse a otra posición (llevándose datos y códigos) donde continuar su ejecución. De hecho, la *"movilidad"* tiene dividida a toda la comunidad de agentes en dos escuelas: la de los que piensan que la movilidad no es un aspecto esencial (la comunidad de multi-agentes DAI, esencialmente académicos) y la de los que claman que los agentes móviles son el futuro de los agentes (comunidad de programadores orientados a objetos). Como sucede en todo, la verdad está en un punto intermedio. Sí parece claro que algo que puede ser hecho con los agentes móviles, puede a priori hacerse con técnicas de programación convencionales, se trata de un hecho que la programación basada en agentes móviles trae un nuevo paradigma

que realza la flexibilidad y eficacia del diseño y ejecución de las aplicaciones distribuidas, y reduce el ancho de banda de red requerido.

Comunicación: Es un aspecto crucial para un agente, que, por definición, tiene que comunicarse durante su vida bien con la entidad en nombre de la que actúa, bien con los otros agentes con los cuales necesita colaborar. El nivel de interacción/comunicación depende en gran medida del nivel de conocimiento del agente, que se mueve entre *"datos"*, *"información"* y *"conocimiento"*. Los datos se corresponden con el nivel más básico, que puede ser el contenido de una variable, un archivo, etc. La información está mucho más estructurada y puede consistir en la descripción de parte de un equipo de telecomunicación, un documento XML (eXtensible Markup Language) (XML) como un formulario de pedido (en aplicaciones de agentes de comercio electrónico), o un correo electrónico (con datos asociados).

El conocimiento consiste en información estructurada y reglas lógicas. Se pueden asociar con cada nivel de información los lenguajes de comunicación y protocolos adecuados. Por ejemplo, un agente puede usar FTP (Protocolo de Transferencia de Archivos) para enviar un archivo, o llamar al método de otro agente para obtener o transmitir información.

El HTTP (HyperText Transfer Protocol) se puede utilizar para transmitir descripciones XML. Finalmente, lenguajes basados en la Speech Acts Theory, como KQML/KIF (Knowledge Query and Manipulation Language/Knowledge Interchange Format) (KQML, KIF), el lenguaje de comunicación de agentes FIPA (Formation for Intelligent Physical Agents) pueden ser usados para intercambiar información.

Sociabilidad: los agentes son capaces de interaccionar con otros agentes (humanos o no) a través de un lenguaje de comunicación entre agentes;

Reactividad: los agentes son capaces de percibir estímulos de su entorno y reaccionar a dichos estímulos;

Pro actividad, iniciativa: los agentes no son sólo entidades que reaccionan a un estímulo, sino que tienen un carácter emprendedor, y pueden actuar guiados por sus objetivos;

Veracidad: asunción de que un agente no comunica información falsa a propósito;

Benevolencia: asunción de que un agente está dispuesto a ayudar a otros agentes si esto no entra en conflicto con sus propios objetivos.

2.3.3.1 Propiedades de los Ambientes[12]

Los ambientes pueden tener diferentes propiedades que hacen que las percepciones y acciones de un agente cambien, a continuación se verán dichas propiedades:

Accesibles Vs no accesibles

¿Es posible explorar toda la información necesaria?

Si el aparato sensorial de un agente le permite tener acceso al estado total de un ambiente, se dice que éste es accesible a tal agente. Un agente es realmente accesible si los sensores detectan todos los aspectos relevantes a la elección de una acción. Los ambientes accesibles son cómodos, puesto que no es necesario que el agente mantenga un estado interno para estar al tanto de lo que sucede en el mundo.

Deterministas Vs. No deterministas

¿La evolución del entorno se sigue de un cómputo o hay actores que responden de forma no previsible?

Si el estado siguiente de un ambiente se determina completamente mediante el estado actual y las acciones escogidas por los agentes, se dice que el ambiente es determinista. En principio, un agente no tiene por qué preocuparse sobre la incertidumbre en un ambiente accesible y determinista. Pero si el ambiente es inaccesible, entonces podría parecer que es no determinista. Lo anterior es especialmente válido cuando el ambiente es complejo, dificultando el estar al tanto de todos los aspectos inaccesibles. Por ello, es más conveniente

[12] http://www.depi.itchihuahua.edu.mx/apacheco/ai/agentes.htm

calificar el que un ambiente sea determinista o no determinista considerando el punto de vista del agente.

Episódicos vs. No episódicos

¿La acción del agente se produce bajo demanda o el agente ha de ser proactivo?

En un ambiente episódico, la experiencia del agente se divide en *"episodios"*. Cada episodio consta de un agente que percibe y actúa. La calidad de su actuación dependerá del episodio mismo, dado que los episodios subsecuentes no dependerán de las acciones producidas en episodios anteriores. Los ambientes episódicos son más sencillos puesto que el agente no tiene que pensar por adelantado.

Estáticos vs. Dinámicos

¿Cambia el entorno aunque no entre en acción el agente?

Si existe la posibilidad de que el ambiente sufra modificaciones mientras el agente se encuentra deliberando, se dice que tal ambiente se comporta en forma dinámica en relación con el agente, de lo contrario, se dice que es estático. Es más fácil trabajar con ambientes estáticos puesto que el agente no tiene que observar lo que sucede en el mundo al mismo tiempo que decide sobre el curso de una acción, ni tampoco tiene que preocuparse por el paso del tiempo. Si el ambiente no cambia con el paso del tiempo, pero si se modifica la calificación asignada al desempeño de un agente, se dice que el agente es semidinámico.

Discretos vs. Continuos

¿El modelo del ambiente es continuo o discreto (simbólico)?

Si existe una cantidad limitada de percepciones y acciones distintas y claramente discernibles, se dice que el ambiente es discreto. El ajedrez es discreto: existe una cantidad fija de posibles jugadas en cada ronda. La conducción de un taxi es continua: la velocidad y la ubicación del taxi y de los demás vehículos se extiende a través de un rango de valores continuos.

2.3.4 Arquitectura

Bajo el concepto de arquitectura de agente subyace la descripción particular de los elementos que constituirán un agente concreto y como estos elementos interactúan entre sí. En este punto comentaremos primera una breve clasificación de agentes para a continuación presentar diferentes arquitecturas que hacen uso del concepto de agente en un sentido más o menos amplio. Algunas de ellas además están pensadas para el funcionamiento conjunto de agentes desarrollando lo que se conoce como arquitecturas multi-agente. Se ha estudiado fundamentalmente aquellas que pretenden ser aplicadas a entornos de tiempo real o al menos con capacidad reactiva.

Una primera división de las arquitecturas considera el acceso a los sensores y actuadores. En concreto si todas las capas de la arquitectura tienen acceso a dichos componentes se considera una arquitectura horizontal. Por el contrario, en las arquitecturas verticales, sólo la capa inferior tiene acceso a ellos.

Otra posible clasificación atendería al tipo de procesamiento empleado, distinguiendo en este caso las arquitecturas deliberativas, las reactivas y las híbridas.

Las arquitecturas deliberativas son aquellas que manejan símbolos físicos. Se indica que estas arquitecturas deben ser capaces de describir objetivos y como alcanzarlos. Normalmente hacen uso de técnicas de planificación para determinar qué pasos realizar. Una subdivisión de este grupo, contemplaría las arquitecturas intencionales, aquellas en que los agentes pueden razonar sobre sus creencias e intenciones, y las sociales donde cada agente es capaz de mantener un modelo de otros agentes y razonar sobre ello. Normalmente presentan una estructura horizontal.

Las arquitecturas reactivas por el contrario no presentan un modelo de razonamiento simbólico complejo, sino un modelo estimulo-respuesta. En este caso las arquitecturas son verticales, donde las capas superiores especializadas manejan directamente los datos, inhibiendo las capas inferiores y obteniendo las acciones a realizar ante dichas entradas.

Finalmente las arquitecturas híbridas presentan varios subsistemas, unos deliberativos para resolver tareas que requieren un modelo simbólico y otros reactivos para responder ante estímulos que no requieren deliberación.

Una metodología particular para construir agentes, especifica cómo puede descomponerse el agente construyendo un conjunto de componentes modulares y cómo deben realizarse estos módulos para que interactúen. El conjunto total de módulos y sus interacciones tienen que dar una respuesta a la pregunta de cómo los datos de los sensores y el estado interno actual del agente determinan las acciones y el futuro estado interno del agente. Una arquitectura comprende técnicas y algoritmos que soportan esta metodología.

tipo	*aproximación*	*tipo componente*	*estructura de subordinación*	*estructura de Acoplamiento*	*constitución*
Modular horizontal	Funcional horizontal	módulo	Jerárquica	Fija (progresiva)	predefinida
Pizarra	funcional	tarea	jerárquica(meta)		variable predefinida
Subsunción	Funcional vertical	tarea primitiva	jerárquica	fija	predefinida
Tareas competitivas	Funcional vertical	tarea + acciones primitivas	jerárquica (competitiva)	variable	predefinida
Reglas producción	funcional	Regla	jerárquica	evolutiva	predefinida
Conexionista	Funcional vertical	neurona formal	igualitaria	fija (por peso)	predefinida
Sistema dinámico	Funcional vertical	Estímulo/respuesta	igualitaria	fija (progresiva)	emergente
Multiagente	objeto / funcional	Agente	igualitaria	variable	emergente

Figura 2.4: *Arquitecturas de agentes*

Fuente: *(Perez, 2007) Metodologias para la construccion de sistemas multiagentes.*

2.3.4.1 Arquitectura Abstracta de Agentes

Asuma que un ambiente se puede encontrar en cualquiera de un conjunto finito de estados instantáneos discretos:

$$E = \{e, e', \ldots\}.$$

Se asume que un agente tiene un repertorio de posibles acciones disponibles, que transforman el estado del ambiente:

$$Ac = \{\alpha, \alpha', \ldots\}$$

Una ejecución, r, de un agente sobre un ambiente es una secuencia de estados y acciones:

$$r : e_0 \xrightarrow{\alpha_0} e_1 \xrightarrow{\alpha_1} e_2 \xrightarrow{\alpha_2} e_3 \xrightarrow{\alpha_3} \cdots \xrightarrow{\alpha_{u-1}} e_u$$

2.3.4.1.1 Funciones de Transformación de Estados

Una función de transformación de Estados representa la conducta del ambiente:

$$\tau : \mathcal{R}^{Ac} \rightarrow \wp(E)$$

Tome en cuenta que los ambientes son:

- Dependientes de la historia
- No deterministas

Si τ(r)=ø, no existen estados sucesores posibles para r (el sistema ha terminado su ejecución).

Formalmente, un ambiente ***Env*** es una tripleta, $Env = < E, e_0, \tau >$, donde: E es el conjunto de estados ambientales $e_0 \in E$ Es el estado inicial τ Es una función de transformación de estados.

2.3.4.1.2 Agentes y acciones

Un agente representa una "función" que transforma ejecuciones en acciones:

$$Ag : \mathcal{R}^{E} \rightarrow Ac$$

Un agente toma decisiones acerca de que acción realizar basándose en el historial del sistema hasta la fecha.

2.3.4.1.3 Sistemas de Agentes

Un sistema es un par que contiene un agente y su ambiente.

Todo sistema tendrá asociado un conjunto de posibles ejecuciones.

El conjunto de ejecuciones de un agente Ag en el ambiente Env se denota

$$R(Ag, Env)$$

Formalmente, una secuencia $(e_0, \alpha_0, e_1, \alpha_1, e_2, \ldots)$

Representa una ejecución de un agente Ag en el ambiente $Env =< E, e_0, \tau >$ si: e_0 es el estado inicial de Env

$$\alpha_0 = Ag(e_0)$$

2.3.5 SMA (Sistemas Multi Agente)

Para resolver tal tipo de problemas, el estudio de los SMA ha estado fuertemente influenciado e inspirado por varias áreas científicas:

- Economía
- Filosofía
- Teoría de Juegos
- Lógica
- Ecología
- Ciencias Sociales
- etc.

Hasta el momento, tenemos la noción de agentes autónomos individuales. Un Sistema Multi-agente (SMA) contiene cierto número de agentes que:

- Interactúan a través de la comunicación
- Son capaces de actuar en el ambiente
- Tienen diferentes “esferas” de influencia

- Están conectados por relaciones en la organización

2.3.5.1 Construcción de SMA

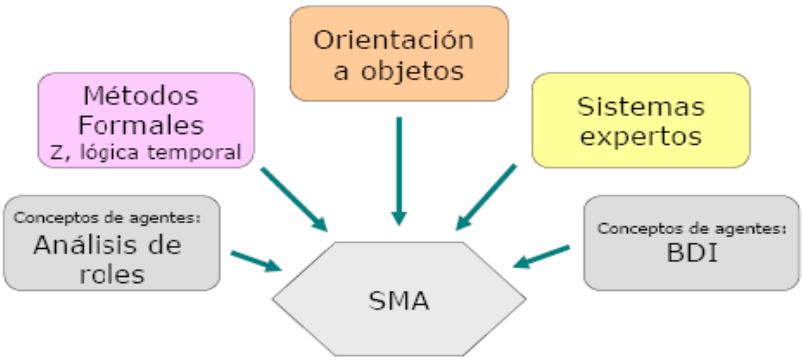


Figura 2.5: *Componentes de SMA*

Fuente: *(Perez, 2007) Metodologias para la construccion de sistemas multiagentes.*

La construcción de SMA integra tecnologías de distintas áreas de conocimiento: técnicas de ingeniería del software para estructurar el proceso de desarrollo; técnicas de inteligencia artificial para dotar a los programas de capacidad para tratar situaciones imprevistas y tomar decisiones, y programación concurrente y distribuida para tratar la coordinación de tareas ejecutadas en diferentes máquinas bajo diferentes políticas de planificación. Debido a esta combinación de tecnologías, el desarrollo de SMA se complica.

Existen plataformas de desarrollo que dan soluciones parciales al modelado de comportamiento y a la coordinación de agentes. El rango de estas soluciones va desde proporcionar servicios básicos (gestión de agentes, librerías de algoritmos, localización de agentes o movilidad), como JADE, Grasshopper o ABLE, hasta entornos de desarrollo donde se parametrizan armazones (framework) software, como ZEUS o AgenTool. Aunque facilitan el proceso, las plataformas de desarrollo quedan incompletas sin un proceso de desarrollo de software especializado para agentes que haga similar la creación de SMA a la producción de software convencional.

Las técnicas convencionales de ingeniería (Unified Process), CommonKADS no tienen en cuenta las necesidades de especificación de los SMA, como la especificación de

planificación de tareas, intercambio de información con lenguajes de comunicación orientados a agentes, movilidad del código o motivación de los componentes del sistema. Por ello, se plantean nuevas metodologías basadas en agentes (BDI, Vowel Engineering, MAS-CommonKADS, GAIA. Estas metodologías parten de un modelo, informal en la mayoría de casos, de cómo debe ser un SMA y dan guías para su construcción.

En las primeras metodologías, las guías consistían en una lista breve de pasos a seguir. Las más modernas, aunque han progresado en la integración con la ingeniería del software clásica, aún no muestran la madurez que se puede encontrar en metodologías convencionales como el Unified Process. El motivo principal es que siguen faltando herramientas de soporte y un lenguaje para la especificación del SMA que permitan trabajar de forma similar a como se trabaja en Rational Rose, TogetherJ o Paradigm+. Existe una gran cantidad de metodologías y cada día van creciendo en paridad a grupos de investigación que las proponen; De entre las metodologías existentes, se ha seleccionado un conjunto utilizando tres criterios: utilización de diferentes vistas para la especificación del sistema, incorporar la idea de proceso de desarrollo, e integrar técnicas de ingeniería y teoría de agentes.

De acuerdo con estos criterios, se han identificado tres metodologías. La primera es MAS-CommonKADS y posterior su estudio evolutivo hacia Masina que, debido a su origen CommonKADS, se trata de una metodología orientada al desarrollo utilizando experiencia en sistemas expertos. La segunda BDI que ha influido notablemente en la forma de concebir el control de los agentes. También, se encuentran las metodologías soportadas por herramientas: ZEUS. Por último, INGENIAS, creada a partir del trabajo de MESSAGE.

2.3.5.1.1 Lenguajes de agentes y el diseño de Sistemas Multi-Agente

Los lenguajes de agentes parten, en su mayoría, de modelos operacionales que definen la semántica de sus instrucciones. Estos modelos se suelen describir informalmente, aunque también existen trabajos que primero parten de un modelo operacional expresado como arquitectura software que luego es formalizado, como es el caso de AgentSpeak (L).

El trabajo más referenciado en lenguajes de agentes es Agent0 que acuñó el término programación orientada a agentes. Agent0 propone un nuevo paradigma de programación en el que la entidad principal es el agente. En Agent0, un agente es una entidad cuyo estado se ve como un conjunto de componentes mentales tales como creencias, habilidades, elecciones y compromisos. Con estas entidades y un conjunto de primitivas, como enviar mensaje, comprometerse o solicitar la ejecución de una tarea, se elabora un lenguaje de descripción de agentes. Se hallan otros lenguajes que siguen el modelo de Shoham como CASA o PLACA. Estos lenguajes añaden principalmente la capacidad de planificar acciones de los agentes en el SMA.

Desde un punto de vista metodológico, el principal problema de estos lenguajes es su aplicación a desarrollos de complejidad media. Los lenguajes de agentes pueden verse como lenguajes de implementación de más alto nivel que otros más convencionales como C++ o JAVA, pero con una particularidad: son muy pobres en mecanismos de abstracción y encapsulación. Los programas o especificaciones generados con ellos tienen como unidad principal de encapsulación el agente y de abstracción procedimental la tarea, lo cual limita su aplicación. Así pues, a los problemas de desarrollar un sistema directamente con un lenguaje de implementación se une la falta de medios para realizar esta tarea incrementalmente. Entonces, la necesidad de metodologías es doblemente justificable. Por un lado, para cubrir las deficiencias de los lenguajes de agentes en cuanto a mecanismos de encapsulación y abstracción.

En este sentido, las metodologías actuales definen elementos que les sirven para agrupar las distintas funcionalidades asociadas a un agente o a un grupo de agentes. Así aparecen conceptos como rol o servicio. Y por otro lado, para facilitar la comprensión de sistemas complejos tal y como ocurre con los lenguajes convencionales de implementación.

2.3.5.1.2 Plataformas para el Diseño de Sistemas Multi-Agente

El desarrollo de SMA hoy en día es más proclive a la utilización de plataformas de desarrollo que a la aplicación de lenguajes de agentes. Esto se debe en gran parte al nivel de conocimientos necesarios que generalmente implica programar con un lenguaje de agentes. Por ello, han proliferado por un lado armazones software de SMA adaptables a diferentes dominios de aplicación y por otras plataformas de desarrollo de ámbito genérico que son implementaciones de estándares de agentes. Aunque el desarrollo con los armazones es más sencillo, hoy en día predominan los segundos.

Las plataformas de desarrollo más extendidas son JADE Y Grasshopper. JADE es la implementación oficial del estándar FIPA, y soporta todos los servicios básicos de infraestructura especificados en FIPA (comunicaciones, movilidad, gestión de agentes y localización de agentes), a los que añade algunas utilidades gráficas para facilitar la administración de las plataformas y la depuración de los mensajes intercambiados por agentes en tiempo de ejecución.

Grasshopper es la implementación del estándar MASIF, que soporta la movilidad de agentes en un entorno distribuido utilizando comunicación y servicios CORBA. En JADE y Grasshopper existe una arquitectura básica de agente que hay que utilizar para acceder a los servicios de la plataforma correspondiente.

El diseño de agentes con estas plataformas significa atenerse a unos estándares de comunicación y de gestión de agentes. El resto, como la especificación del control del agente, su inteligencia o las relaciones entre las tareas del sistema, se deja al criterio del desarrollador. La aportación de una metodología a desarrollos basados en este tipo de plataformas consistiría en organizar el proceso de generación del SMA y en proporcionar elementos para que el diseñador pueda describir estos aspectos teniendo en cuenta las restricciones de la plataforma destino.

En cuanto a los armazones software, la aportación de las metodologías varía según el caso. ZEUS, por ejemplo, presenta un entorno de desarrollo con el que se programa visualmente el SMA. Las posibilidades de configuración del SMA a través de este entorno son muy variadas. Entre otros, hay que suministrar una ontología, reglas de comportamiento, planes de ejecución de tareas y mensajes a enviar a otros agentes.

Debido a esta versatilidad en su configuración, el diseño con ZEUS se hace tan complejo como programar con lenguajes de agentes. Aunque se facilita el desarrollo de aspectos como la coordinación, quedan pendientes decisiones como qué hay que coordinar y para qué. Por ello, la aportación de una metodología en este caso consistiría en proporcionar un ciclo de desarrollo, actividades y elementos conceptuales que ayudasen al descubrimiento incremental de los elementos requeridos por ZEUS. Un ejemplo, aunque limitado, de metodología para ZEUS se tiene de mano de sus propios desarrolladores (Collis y Ndumu), donde se aplica la idea de rol para guiar el descubrimiento de funciones del sistema.

ABLE (Agent Building and Learning Environment) aunque también permite el prototipado rápido de SMA, no llega al nivel de ZEUS. ABLE es una herramienta de IBM para la construcción de sistemas de agentes inteligentes donde todos sus elementos, incluso los agentes, se construyen por composición de AbleBeans, una extensión de los JavaBeans.

Son de interés un conjunto de AbleBeans especializados que implementan sistemas de aprendizaje estadísticos (mapas auto organizativos, redes neuronales, mapas de conexión) y control simbólico (razonamiento por encadenamiento hacia delante y lógica de predicados). ABLE incorpora también un entorno visual de desarrollo donde se interconectan AbleBeans. El interés de esta plataforma es que soluciona visualmente la construcción y comunicación de los agentes.

ABLE no llega a detallar cómo se obtiene la funcionalidad de los agentes y cómo deben actuar en cada situación. La aportación de una metodología a ABLE sería más amplia que en el caso de ZEUS. Se trataría de dar medios para detallar aspectos de control del agente teniendo en cuenta los elementos de control que vienen predefinidos en ABLE. Sería útil,

por ejemplo, la idea de casos de uso (Jacobson, Booch y Rumbaugh 00) para identificar conjuntos de funciones a proporcionar por el sistema y escenarios para describir cómo se espera que se realice el proceso de aprendizaje de los agentes.

	ESTADO MENTAL	EVOLUCION DE ESTADO MENTAL	CONTROL	RESPONSABILIDAD
JADE	No existe	Codificado por el usuario dentro de las clases de comportamiento. Se permiten añadir/quitar comportamientos.	Ejecución de las instacias de clases de comportamiento	Asignación de tareas directa. Las tareas se codifican dentro de las clases de comportamiento.
RETSINA	Un conjuto de objetivos a satisfacer	Satisfacción de objetivos	Actuación de un Planificador para extraer y nenerar planes de tareas que satisfagan los objetivos actuales. Un secuenciador (sheduler) se encarga de elegir las que se ejecutan	Asignación de tareas directa. Las tareas permitidas así como su secuenciación (planes), se codifican en tiempo de diseño.
GRASSHOPPER	A definir por el diseñador	A definir por el diseñador	A definir por el diseñador	A definir por el diseñador
ZEUS	Sigue una codificación BDI. Los objetivos se definen durante el diseño. Se codifican ontologias que sirven para exprear reglas de comportamiento y que constituyen l equivalente a las creencias.	Modificación de las entidades del estado mental definidas por el diseñador. Actuación del planificadro lleva a reconducir la estrategia del agente. Generación automática de compromisos según las relaciones entre los agentes.	Reglas que asocian entidades mentales a tareas. El motor de coordinación y el planificador deciden qué hacer a continuación en función de: compromisos adquiridos, hora rios de finalización de tareas, reglas codificadas, estado de las tareas actuales.	Asignación de tareas dentro del IDE. Relación entre agentes para determinar con quién se puede hacer contratos.
ABLE	Codificación simbólicas (motores de reglas) y no simbólicas (redes neuronales, mapas de conexionado).	Las especificas de cada paradigma. El motor de reglas incorpora gestión de predicados. Las redes neuronales, algoritmos de aprendizaje.	Dependiente del paradigma.	A codificar directamente.
DESIR	El equivalente a estado mental es lo que llaman estados de información que es el modelo parcial de un conjunto de predicados proposicionales.	Mediante la modificación del valor de verdad de los predicados.	Definición de reglas que modifican el estado actual de las tareas y que son responsables del flujo de entrada y salida de datos de una tarea a otra	Las tareas se asignan directamente

Tabla 2.3: *Plataformas para diseño de Sistemas Multi-Agente*
Fuente: *(Perez, 2007) Metodologias para la construccion de sistemas multiagentes.*

En general la implementación de aplicaciones con sistemas multi-agente ha sido realizada por expertos donde los agentes, la plataforma de comunicación y el protocolo de interacción han sido programados en forma ad-hoc para cada sistema. A pesar que estas

aplicaciones se basan en los mismos conceptos teóricos de agentes y sistemas multi-agente, al momento de implementarlas se han utilizado diferentes lenguajes de programación de propósito general, y cada aplicación ha resuelto el mismo problema de diferente manera. Por ejemplo, el protocolo de comunicación entre agentes ha sido resuelto de diferentes maneras en la mayoría de las aplicaciones.

Las plataformas de desarrollo de sistemas multi-agente coinciden en algunas características básicas, pero sin embargo, poseen capacidades diferentes y están orientadas a diferentes escenarios. A pesar de que el uso de una plataforma acelerará el desarrollo de la aplicación, el aprendizaje y uso de estas plataformas no es trivial, y requerirá de un tiempo extra considerable que se suma al del desarrollo de la aplicación. Por lo tanto, elegir la plataforma adecuada resulta fundamental.

	JACK	MADKit	ZEUS
ACL soportado		KQML	KQML
Arquitectura base	BDI	Agente/grupo/rol	BDI[1]
Tipo de agentes soportados	Cualquiera	Cualquiera	Deliberativos Colaborativos
Lenguajes soportados para implementación de agentes	Jack	Java [15] Jess [16] Python [19] Scheme [22] BeanShell [1]	Java [15]
Movilidad de código	No detalla	No detalla	No detalla
Disponibilidad	on-line	on-line	on-line
Licencia	gratis 30 días	GPL/LGPL	Mozilla public
Interface	GUI	GUI	GUI
Instalación	Simple	Simple	Simple
Documentación	Muy completa	Pobre	Pobre
Ayuda	Manual	On-line	Manual

Tabla 2.4: *Comparación JACK MADKit ZEUS*
Fuente: *(Perez, 2007) Metodologias para la construccion de sistemas multiagentes.*

2.3.5.1.3 Metodologías Existentes para diseño de Sistemas Multi-Agente

Estas metodologías proporcionan medios para construir Sistemas Multi-Agente de forma disciplinada y repetible. En este trabajo se presenta un resumen de lo que proponen las metodologías actuales.

Ante la imposibilidad de revisar todas y cada una de las existentes, se ha optado por seleccionar un conjunto significativo cercano a las prácticas de ingeniería del software. La selección atiende a la presencia de un proceso de desarrollo, el combinar diferentes vistas para describir el sistema, e incorporar elementos asociados al área de los agentes.

Vowel Engineering

Se trata de la metodología seguida en el grupo MAGMA. El término vowel engineering viene de que el sistema final depende de la ordenación y agrupamiento de cuatro vocales A (por agentes), E (por entorno), I (por interacciones) y O (por organización).

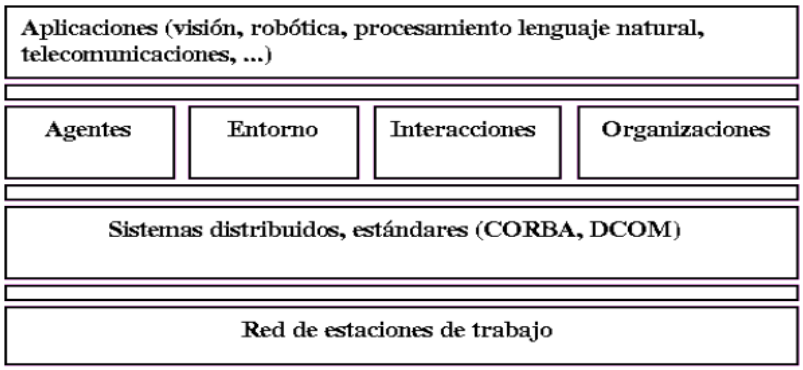


Figura 2.6: Arquitectura de capas de Vowel Engineering
Fuente: *(Perez, 2007) Metodologias para la construccion de sistemas multiagentes.*

A favor de esta metodología está la visión del modelado de sistemas como composición de elementos. Esta composición se define con un lenguaje apropiado para el problema como es un lenguaje de descripción de arquitecturas, Unicon en este caso. Como consecuencia de esta forma de desarrollo, hay que señalar que facilita la reutilización de código. Metodológicamente es mejorable.

MaSE[13]

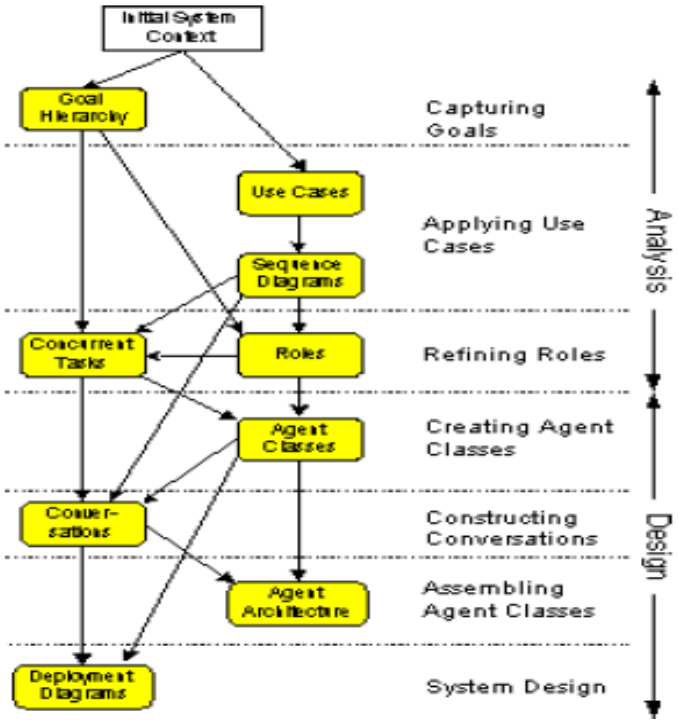


Figura 2.7: *MaSE*
Fuente: http://macr.cis.ksu.edu/projects/mase.htm.

MaSE (Multi-agent systems Software Engineering) se concibe como una abstracción del paradigma orientado a objetos donde los agentes son especializaciones de objetos. En lugar de simples objetos, con métodos que pueden invocarse desde otros objetos, los agentes se coordinan unos con otros vía conversaciones y actúan pro activamente para alcanzar metas individuales y del sistema.

GAIA[14]

GAIA es una metodología para el diseño de sistemas basados en agentes cuyo objetivo es obtener un sistema que maximice alguna medida de calidad global (no se llega a detallar cuál). GAIA pretende ayudar al analista a ir sistemáticamente desde unos requisitos iníciales a un diseño que, según los autores, esté lo suficientemente detallado como para ser

[13] http://macr.cis.ksu.edu/projects/mase.htm
[14] www.ecs.soton.ac.uk/~nrj/download-files/jaamas2000.pdf
www.citeseer.ist.psu.edu/wooldridge00gaia.htm

implementado directamente. En GAIA se entiende que el objetivo del análisis es conseguir comprender el sistema y su estructura sin referenciar ningún aspecto de implementación. Esto se consigue a través de la idea de organización. Una organización en GAIA es una colección de roles, los cuales mantienen ciertas relaciones con otros y toman parte en patrones institucionalizados de interacción con otros roles.

MESSAGE[15]

MESSAGE trata de integrar resultados de las anteriores. Propone el análisis y diseño del SMA desde cinco puntos de vista para capturar los diferentes aspectos de un SMA:

- Organización, que captura la estructura global del sistema;
- Tareas/Objetivos, que determina qué hace el SMA y sus agentes constituyentes en términos de los objetivos que persiguen y las tareas implicadas en el proceso;
- Agente, que contiene una descripción detallada y extensa de cada agente y rol dentro del SMA;
- Dominio que actúa como repositorio de información (para entidades y relaciones) concernientes al dominio del problema;
- Interacción, que trata las interacciones a distintos niveles de abstracción.

A favor de MESSAGE hay que destacar que ha sido la primera metodología en utilizar una herramienta para soporte del proceso de especificación de SMA de forma visual, como en UML. En cuanto a la implementación, MESSAGE provee guías en cuanto a posibles arquitecturas y componentes a utilizar en esta etapa. Basándose en estas guías y los modelos de análisis y diseño, se realizó manualmente la implementación, lo cual hizo que se detectaran incorrecciones en las definiciones iníciales de los modelos. Esta experiencia es la base de la crítica realizada con anterioridad a ZEUS.

[15] http://www.eurescom.de/~public-webspace/P900-series/P907/D1/

Modelos / Metodologías	Agentes	Organización	Cooperación	Coordinación	Interacción	Funcional	Escenarios	Roles	Dominio	Conceptual	Creencias	Objetivos	Planes	Objetos	Capacidad	Tarea	Sistema	Diseño
Burmeister [1]	X	X	X					X			X	X	X					
BDI [8]	X				X						X	X	X					
MASB [9]	X				X		X	X		X				X				
AOSE [6]		X						X										
MASE [5]	X			X	X													
GAIA [11]	X				X	X		X										
ZEUS [4]	X	X						X										
MESSAGE [2]	X	X			X				X			X						
OO-METHOD [3]					X	X								X				
CoMoMas [10]	X		X												X	X	X	X
MAS-CommonKADS [7]	X	X		X	X										X	X		

Tabla 2.5: *Metodologías "Relación de los modelos que contempla cada una de las Metodologías"*
Fuente: *(Pérez, 2007) Metodologías para la construcción de sistemas multiagentes.*

2.3.6 Comparativa entre Agentes, Objetos y sistemas expertos

AGENTES Vs OBJETOS

Los métodos de producción de software convencionales que parten de un modelo conceptual orientado a objetos están muy arraigados y es poco apreciable las ventajas que los agentes conceden.

AGENTES	OBJETOS
Autonomía de decisión	Ejecuta modelos invocados
Flujo de control propio	Flujo de control del llamante
Encapsula la activación del comportamiento	Encapsula estado y comportamiento
Estado mental: objetivos, creencias	Estado: Valor de las variables
Comportamiento: como decir lo que hacer	Comportamiento: salida a partir de una entrada
Interacciones: actos de habla (intencionalidad)	Mensajes invocan procedimiento
Organización: relaciones sociales	Asociaciones entre objetos

Tabla 2.6: *Agentes Vs Objetos*
Fuente: *(Pérez, 2007) Metodologías para la construcción de sistemas multiagentes.*

AGENTES Vs SISTEMAS EXPERTOS

Un sistema experto (SE) es una rama de la Inteligencia Artificial y es aquel que imita las actividades de un humano para resolver problemas de distinto índole (no necesariamente tiene que ser de Inteligencia Artificial). También se dice que un SE se basa en el conocimiento declarativo (hechos sobre objetos, situaciones) y el conocimiento de control (información sobre el seguimiento de una acción), mientras que los agentes evocan unas grandes ventajas.

AGENTES	SISTEMAS EXPERTOS
Interactúan con el entorno	Sistemas cerrados
Distribución de la toma de decisiones: Comportamiento emergente	Sistemas de decisión centralizados
Mayor grado de interacción con el usuario	Interacción con el usuario bajo petición del usuario
Interacción con otros agentes	Normalmente carecen de habilidades sociales (cooperación).

Tabla 2.7: *Agentes Vs Sistemas Expertos*
Fuente: *(Pérez, 2007) Metodologías para la construcción de sistemas multiagentes.*

2.3.7 ESTANDARES ACEPTADOS

2.3.7.1 Estándares

- FIPA (www.fipa.org)
  - JADE http://jade.cselt.it
  - LEAP (para PDAs) http://leap.crm-paris.com
  - FIPA OS http://fipa-os.sourceforge.net
  - JACK http://www.agent-software.com
  - ZEUS (BT) http://193.113.209.147/projects/agents/zeus/
- Agentes móviles (MASIF, FIPA)
  - Grasshopper http://www.grasshopper.de
  - Aglets (IBM) http://sourceforge.net/projects/aglets/

- No estándares
  - ABLE (IBM) http://www.alphaworks.ibm.com/tech/able

2.3.7.2 Organizaciones de Estandarización

- OMG (Object Management Group)
  - Mobile Agent System Interoperability Facilities (MASIF)
- KSE (Knowledge Sharing Effort)
  - Knowledge Querying and Manipulation Language (KQML)
  - Knowledge Interchange Format (KIF)
- FIPA (Foundation for Intelligent Physical Agents)
  - Especificaciones de arquitectura, infraestructura y aplicaciones
- Agent Society
  - Arquitectura y protocolos de comunicación genéricos

2.4 METODOLOGIA INGENIAS[16]

Parte del trabajo de MESSAGE, es una metodología que ha tenido un gran impacto en la comunidad dedicada al estudio de los agente software. INGENIAS profundiza en los elementos mostrados en el método de especificación, en el proceso de desarrollo, además de incorporar nuevas herramientas de soporte y ejemplos de desarrollo. INGENIAS, como MESSAGE, define un conjunto de meta-modelos (una descripción de alto nivel de qué elementos tiene un modelo) con 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. Estos meta-modelos se construyen mediante un lenguaje de meta-modelado, el GOPRR (Graph, Object, Property, Relationship, and Role). En la construcción de estos meta-modelos se integran resultados de investigación en forma de entidades y relaciones entre entidades.

[16] MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR : Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002

La instanciación de estos meta-modelos produce diagramas, los modelos, similares a los que se usa en UML, con la diferencia de que estos diagramas se han creado exclusivamente para definir el sistema multi-agente. El proceso de instanciación de los meta-modelos no es trivial. Existen muchas entidades y relaciones a identificar, además de dependencias entre distintos modelos. Por ello, INGENIAS define un conjunto de actividades cuya ejecución termina en un conjunto de modelos. Estas actividades a su vez se organizan siguiendo un paradigma de ingeniería del software, el Proceso Unificado.

La ejecución de actividades para producir modelos se basa en la herramienta INGENIAS IDE, una herramienta para modelado visual. Esta herramienta almacena la especificación del sistema utilizando XML.

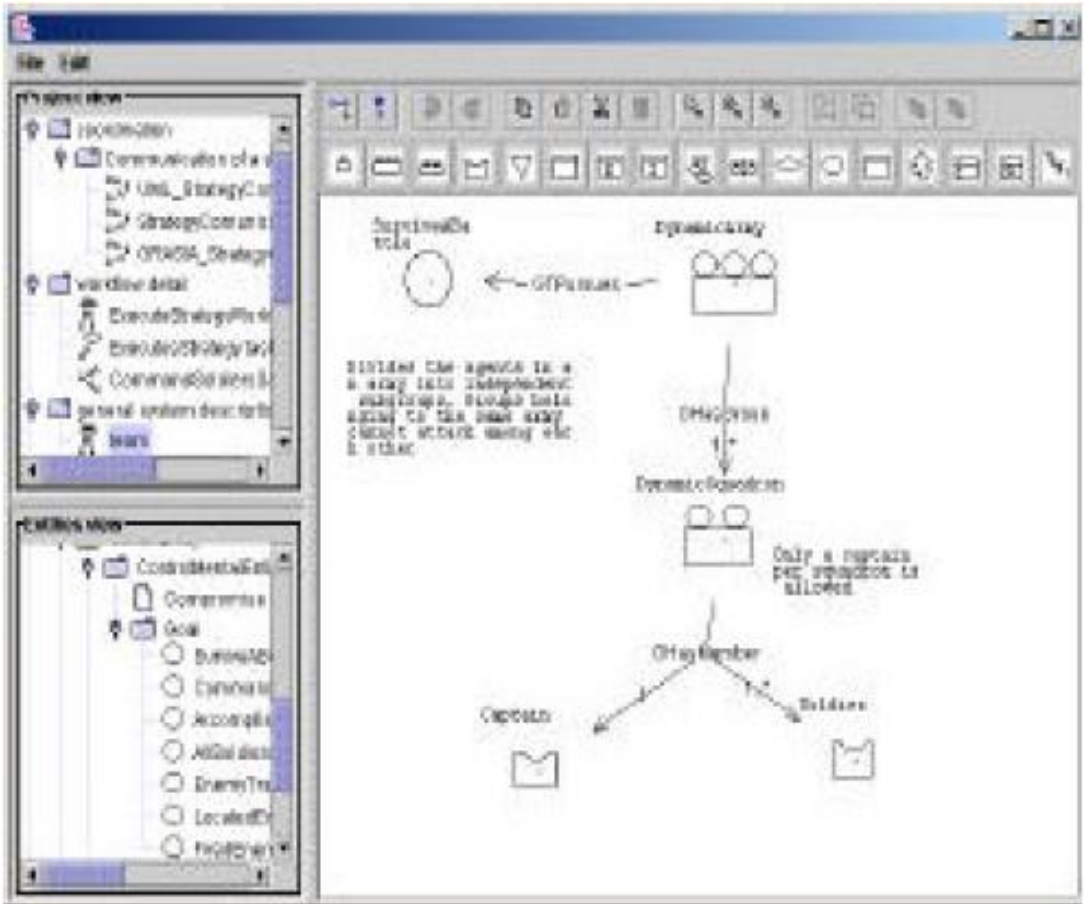


Figura 2.8: *pantalla principal de ingenias*
Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

Desde la especificación en XML, se plantea el procesarla:

- Para generar código. Genera código rellenando plantillas de código con información de la especificación.

- Para generar la documentación. De una forma similar a la generación de código, se parte de plantillas de documentación que se completan utilizando información de los modelos Tanto la herramienta de análisis/diseño como el editor se pueden encontrar en http://ingenias.sourceforge.net. Desde esta web también se tiene acceso a ejemplos de especificación siguiendo la metodología.

2.4.1 Nomenclatura

El nombre asociado a las relaciones obedece a unas reglas sencillas mostradas. Se trata de hacer que el nombre de la relación sea precedido por un conjunto de letras que denote su procedencia, como el flujo de trabajo (WF), meta-modelo de agente (A), interacción (I), unidad de interacción (UI), modelos de tareas y objetivos (GT), relaciones sociales (AGO), organización (O) o el entorno (E).

IdentificadorRelacion::= RelacionFlujoTrabajo | RelacionAgente |
RelacionInteraccion | RelacionUnidadInteraccion |
RelacionMetaTarea| RelacionSocial |
RelacionOrganizacion | RelacionEntorno
RelacionFlujoTrabajo::= **WF** Identificador
RelacionAgente ::= **A** Identificador
RelacionInteraccion::= **I** Identificador
RelacionUnidadInteraccion ::= **UI** Identificador
RelacionMetaTarea::= **GT** Relacion
RelacionSocial ::= **AGO** Relacion
RelacionOrganizacion ::= **O** Relacion
RelacionEntorno ::= **E** Relacion

Figura 2.9: *Expresión BNF para la los nombres de las relaciones*
Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

2.4.1.1 Entidades básicas

La metodología proporciona una jerarquía de conceptos básicos para el desarrollo de un SMA así como una notación para representar estos conceptos. La jerarquía comienza con la Entidad MAS GRASIA y la Relación MAS GRASIA, que reciben este nombre por el grupo de investigación en que han sido desarrolladas (Grupo de Agentes Software del departamento de Sistemas Informáticos y programación o GRASIA).

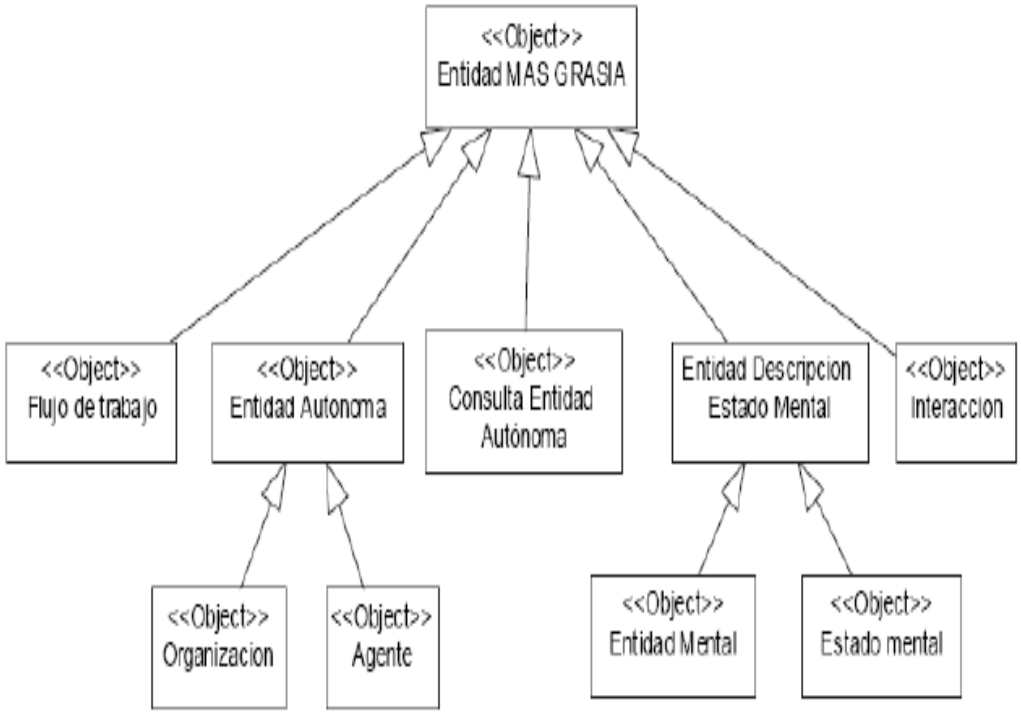
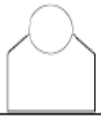
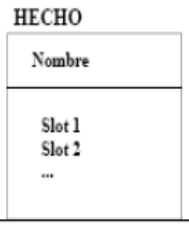


Figura 2.10: *Entidades básicas de la metodología*
Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

2.4.1.2 Notación

Los diagramas correspondientes al nivel del modelo (M2) que se presentan a lo largo de este trabajo utilizan una notación desarrollada ad hoc con la herramienta METAEDIT+. Y posteriormente exportados a la herramienta de trabajo INGENIAS IDE Para entender los diagramas se debe revisar antes la notación.

	**Objetivo**. Se etiqueta con el nombre del objetivo
	**Rol**. Se etiqueta con el nombre del rol.
	**Procesador de estado mental**. Se etiqueta con el nombre del procesador.
	**Gestor de estado mental**. Se etiqueta con el nombre del gestor.
	**Agente**. Se etiqueta con el nombre del agente.

	**Hecho.** Se etiqueta con el nombre del hecho y los nombres de los slots identificados.
	**Creencia.** Se etiqueta con el nombre de la evidencia e información acerca de qué es lo que se está aceptando como cierto.
	**Evento.** Se etiqueta con el nombre del evento y los nombres de los slots identificados.

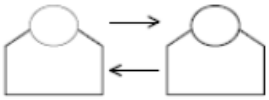
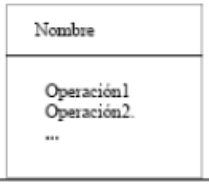
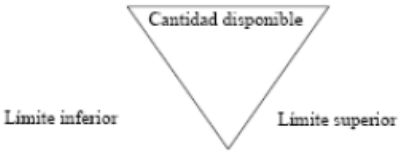
	**Grupo.** Se etiqueta con el nombre del grupo.
	**Organización**. Se etiqueta con el nombre de la organización.
	**Flujo de trabajo.** Se etiqueta con el nombre del flujo.
	Interacción. Se etiqueta con el nombre de la interacción y su naturaleza, como coordinación, planificación o negociación.
	**Consulta de entidades autónomas**. Se etiqueta con nombres concretos de agentes existentes o expresiones que denotan agentes existentes.
	**Unidad de interacción.** Se etiqueta con el nombre de la unidad y el acto del habla al que hace referencia, como *request*, *inform*, o *not-understood*.
	**Tarea.** Se etiqueta con el nombre de la tarea.
	**Aplicación.** Se etiqueta con el nombre de la aplicación y las operaciones soportadas.
	**Recurso.** Se etiqueta con el nombre del recurso, la cantidad disponible del mismo, el límite inferior y superior admisibles. Por debajo o encima de estos límites, el recurso se deshabilita.

Tabla 2.8: *Notación empleada en la representación de los modelos*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

2.4.2 Meta-Modelos del Sistema Multi Agente

Un meta-modelo define las primitivas y las propiedades sintácticas y semánticas de un modelo. A diferencia de otros enfoques más formales, como Z, los meta-modelos están

orientados a la generación de representaciones visuales de aspectos concretos del sistema de forma incremental y flexible. Los modelos crecen incorporando más detalle gracias a que no es necesario que se instancien absolutamente todos los elementos del meta-modelo para tener un modelo. Como ha demostrado UML, construido también con meta-modelos, este tipo de notación facilita enormemente el desarrollo de sistemas.

Otra ventaja de utilizar meta-modelos es que las especificaciones generadas de SMA son lo suficientemente estructuradas para ser procesadas de forma automática. Así, se puede plantear la verificación automática de la construcción de los modelos para ver si cumplen restricciones identificadas por el desarrollador, generar documentación del sistema en diferentes formatos de diferentes partes de los modelos, e incluso establecer la generación automática de código desde la información recogida en los modelos.

El meta-modelado se comprende a partir de una estructura de cuatro niveles. En esta estructura, el nivel M1 establece las primitivas de meta-modelado, esto es, el lenguaje de meta-modelado. En el nivel M2, se definen los meta-modelos en sí con las primitivas de M1. En M3 se aplican los meta-modelos para generar instancias, que se denominan modelos. Por último, la instanciación de los modelos, lleva al nivel donde se tiene la información que se manejará en el sistema.

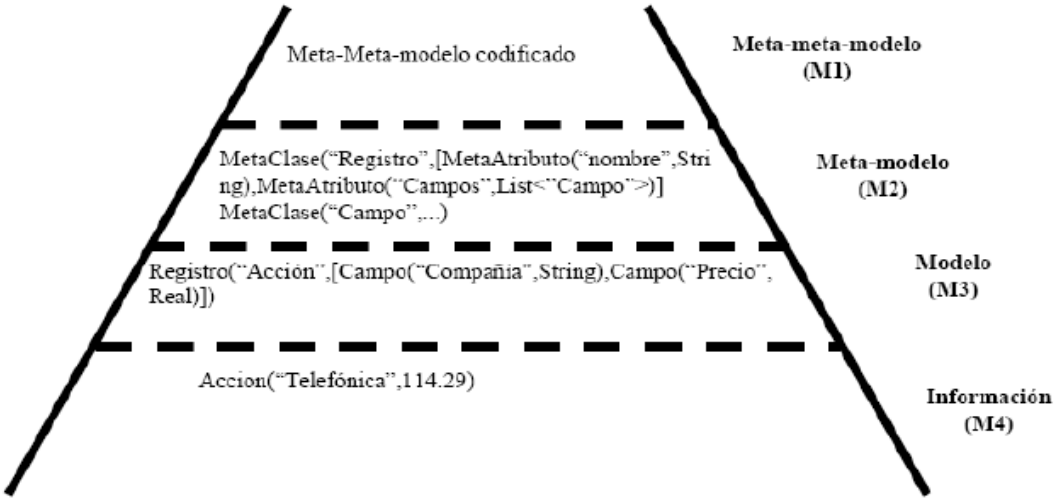


Figura 2.11: *Estructura de cuatro niveles utilizada en el meta-modelado*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

2.4.2.1 Modelo de organización

La organización define una estructura donde van a existir los agentes, recursos, tareas y objetivos

Estructura. Descomposición de la organización en:

- Grupos
- Flujos de trabajo
- Interrelación de tareas en flujos de trabajo
- Relaciones entre agentes respecto a las tareas
- Recursos disponibles y asignación
- Relaciones sociales
- Relaciones de poder (p.ej. subordinación) y cliente/servidor entre agentes
- Relaciones entre grupos
- Relaciones entre organizaciones
- Funcionalidad
- Propósito
- Tareas que debe realizar

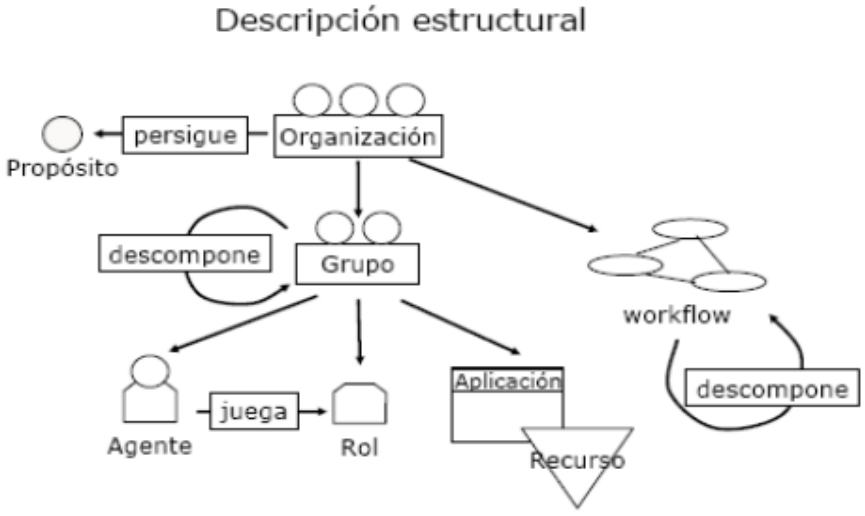


Figura 2.12: *Descripción estructural*
Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

Descripción Social

En concreto, aquí se incluyen relaciones de subordinación, de prestación de un servicio y de cliente de un servicio. Estas relaciones se definen teniendo como extremos posibles organizaciones, agentes, roles y grupos.

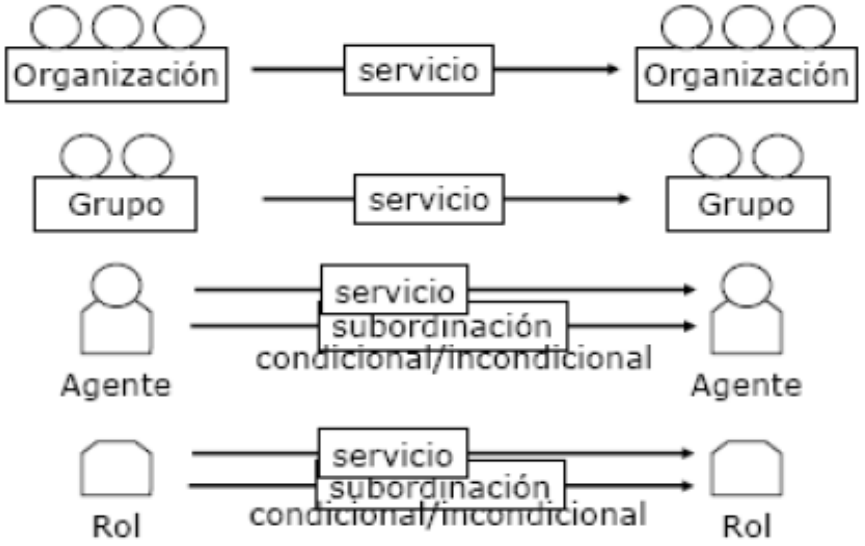


Figura 2.13: *Descripción social*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

Descripción Funcional

El objetivo del flujo de trabajo es establecer cómo se asignan los recursos, qué pasos(tareas) son necesarios para la consecución de un objetivo, y quiénes son los responsables de ejecutarlas. Según el WorkFlow Management Coalition (WfMC), un flujo de trabajo (Workflow Management Coalition 99) es la automatización de un proceso de negocio, en su totalidad o parcialmente, durante el cual los documentos, información o tareas son pasados de un participante a otro, de acuerdo con un conjunto de reglas procedimentales. En el flujo de trabajo, se habla de actividades en lugar de tareas7, aunque en este contexto se pueden emplear indistintamente.

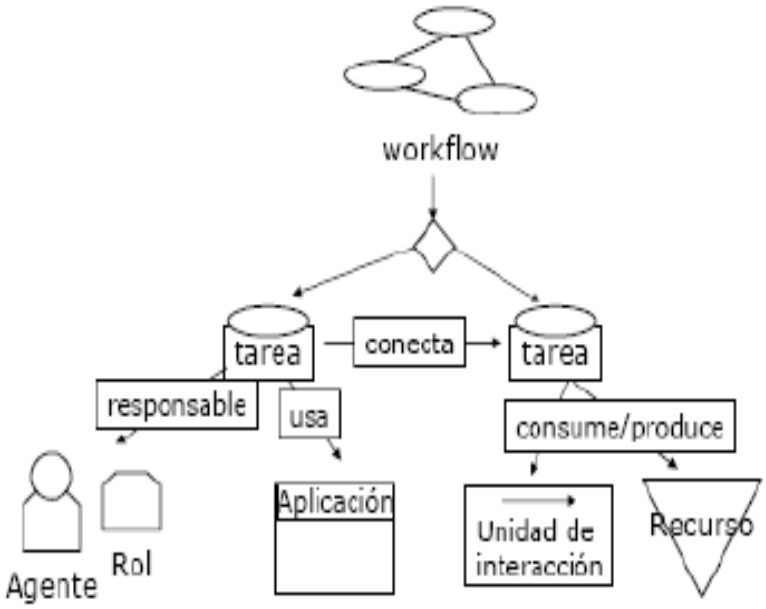
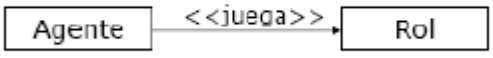


Figura 2.14: *Descripción funcional*
Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

2.4.2.2 Modelo de agente

El meta-modelo de agente se usa para describir agentes particulares excluyendo las interacciones con otros agentes. Este meta-modelo se centra en la funcionalidad del agente y en el diseño de su control. En este sentido, proporciona información acerca de los siguientes aspectos:

- Descripción de agentes particulares
- Funcionalidad del agente: Responsabilidades
- Qué tareas sabe ejecutar
- Qué objetivos se compromete a alcanzar



- Comportamiento: Control del agente
- Estado mental

- Agregación de entidades mentales: objetivos, creencias, compromisos, hechos
- Gestión de estado mental
- Creación, destrucción, modificación de las entidades del estado mental
- Mecanismo de decisión: procesador de estado mental
- Reglas, planificación, etc.

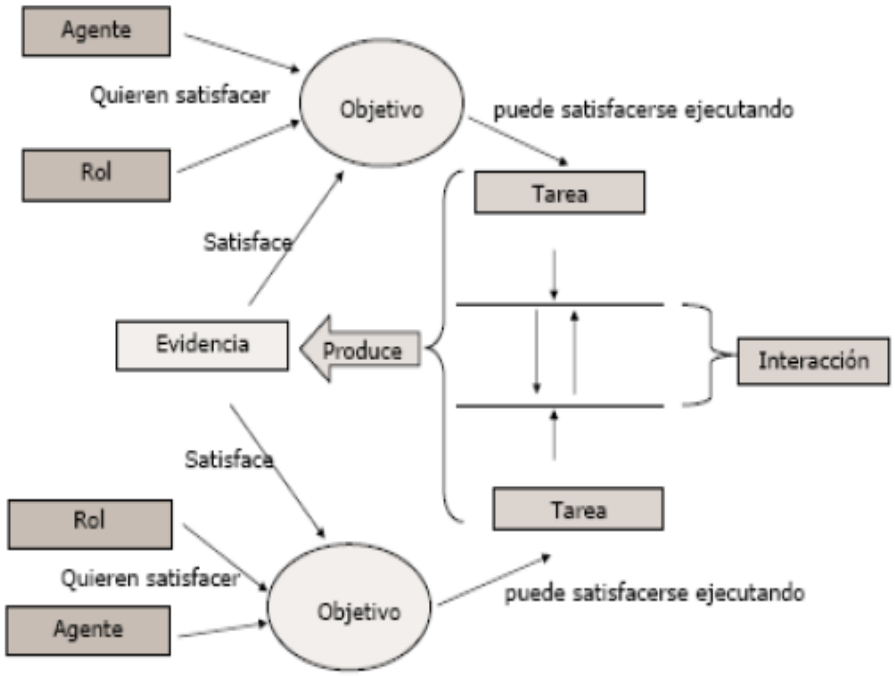


Figura 2.15: *Elementos del modelo de agente*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

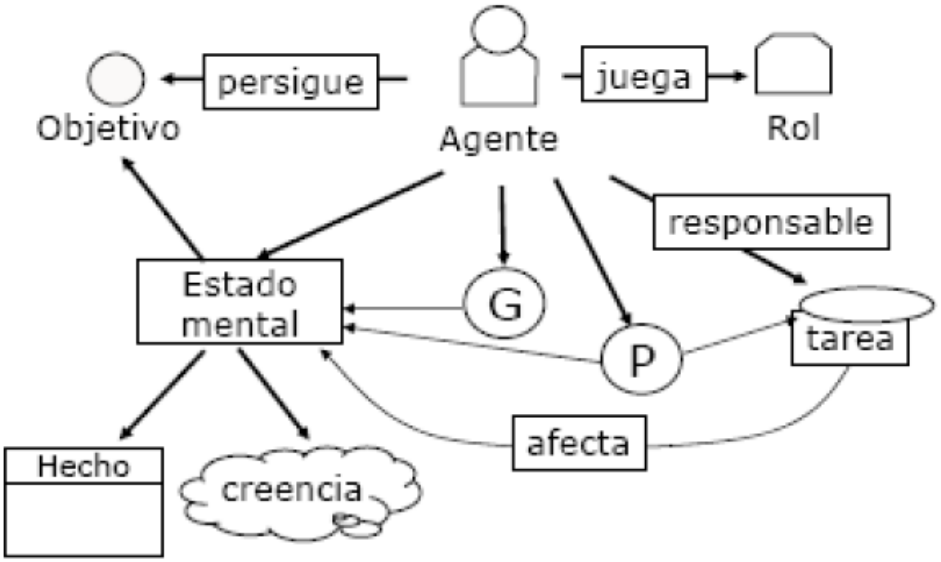


Figura 2.16: *Diagrama del modelo de agente*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

Planificador clásico

Las tareas transforman entidades mentales para alcanzar objetivos del agente

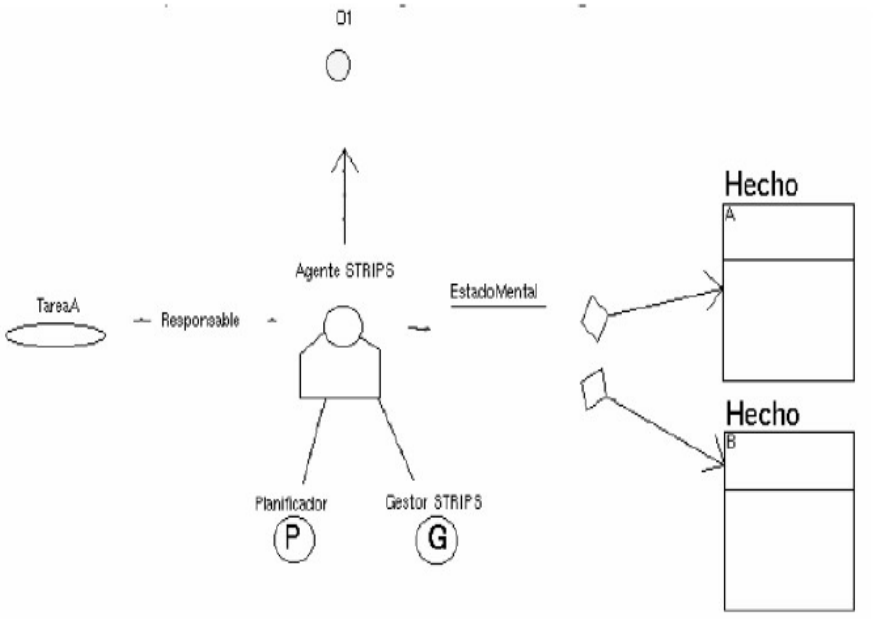


Figura 2.17: *Agente planificador*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

2.4.2.3 Modelo de objetivos y tareas

- Qué consecuencias tiene la ejecución de tareas y por qué se deberían ejecutar
- Justifica la ejecución de tareas basándose en objetivos
- Que a su vez se van modificando tras su ejecución
- Objetivo: Situación deseada
- Conjunto de estados que el agente quiere lograr, mantener, o evitar
- Una función de utilidad que maximizar
- Responde a ¿por qué?
- Tarea: Transiciones de estado
- Conduce a la consecución de objetivos
- Responde a ¿cómo?

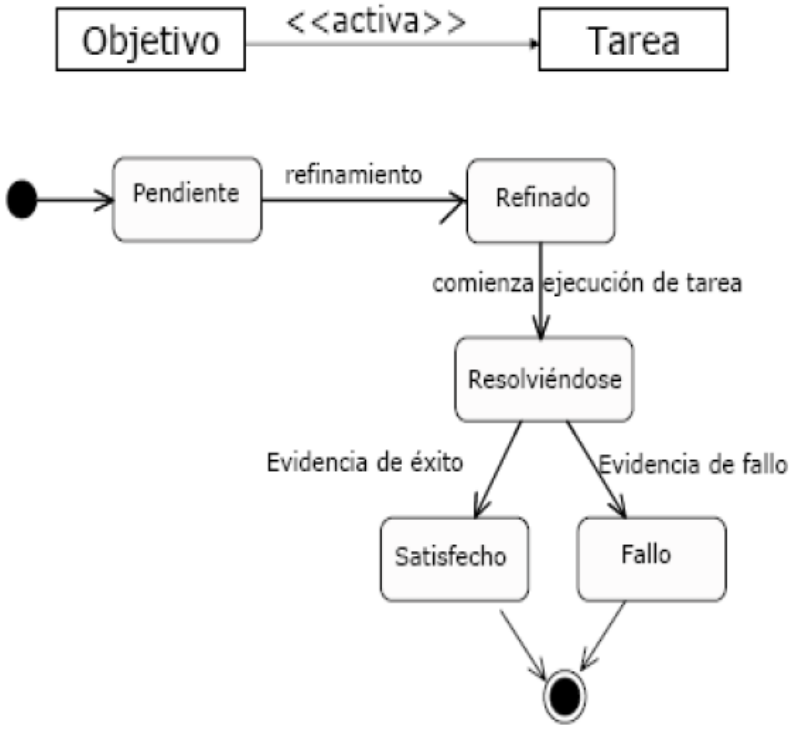


Figura 2.18: *Ciclo de vida de un objetivo*
Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

Descomposición de objetivos

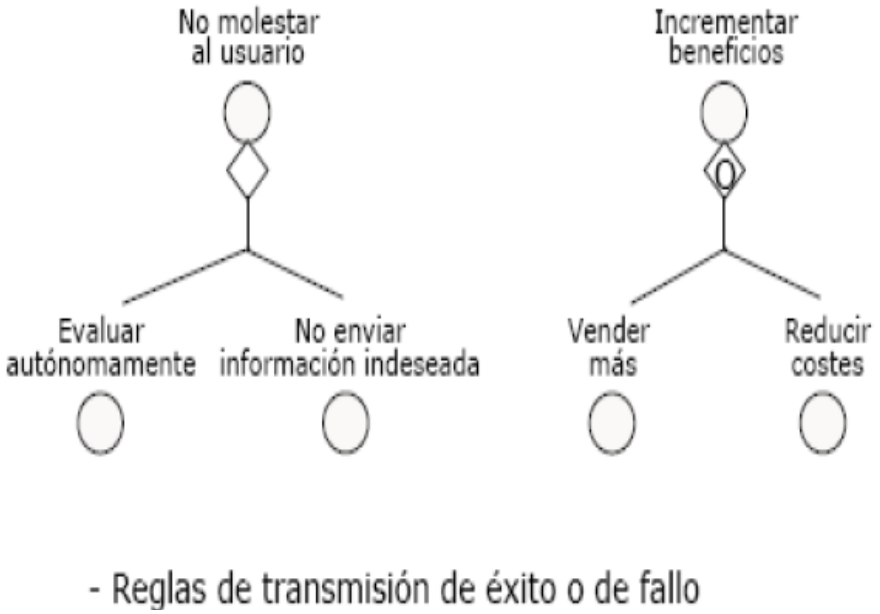


Figura 2.19: *Árboles Y/O*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

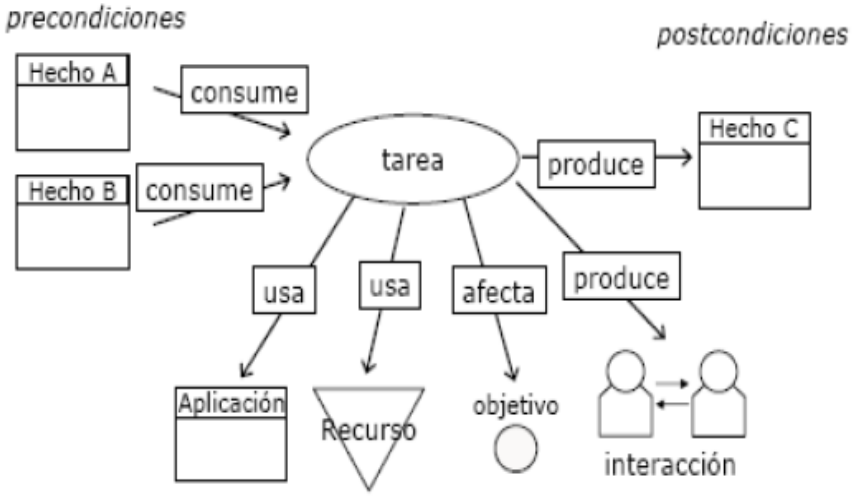


Figura 2.20: *Elementos de definición de tareas*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

Relaciones entre tareas y objetivos

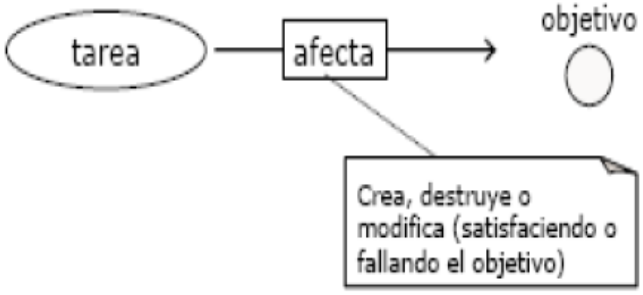


Figura 2.21: *Una tarea afecta a un objetivo*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

Ejemplo de tareas

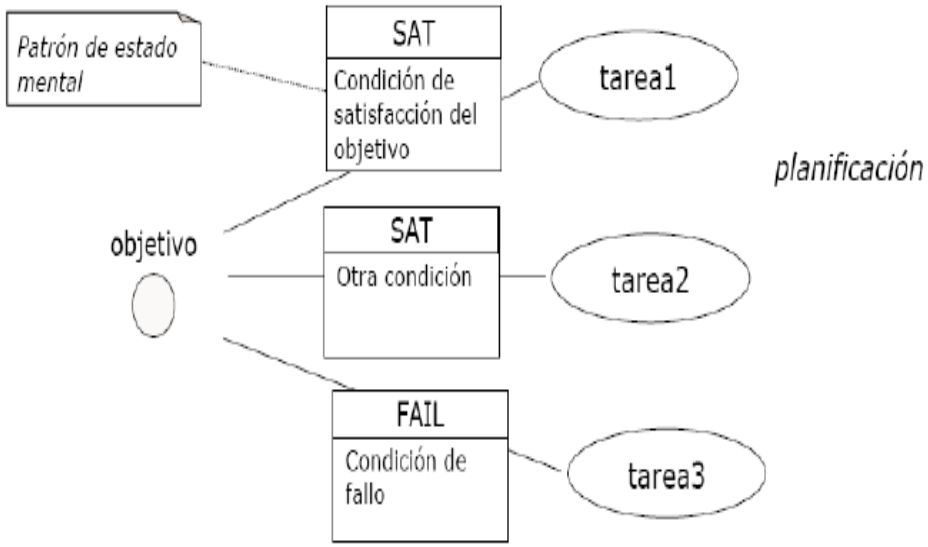


Figura 2.22: *Ejemplo de Tareas*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

2.4.2.4 Modelo de interacciones

Intercambio de conocimiento o peticiones (intencionalidad) entre agentes. Define las interacciones entre los agentes o entre agentes y humanos, se definen a alto nivel, en diseño se detalla el protocolo de interacción, se puede usar el concepto de protocolo de interacción de Agent UML o los protocolos de Gaia

Definición de interacciones

Qué actores participan en la interacción, cada actor debe mostrar la razón por la que participa, roles iniciador y colaboradores, definición de unidades de interacción, mensajes, actos de habla

Orden de las unidades de interacción

Protocolos: contract net, FIPA request, específicos. Diagramas de protocolos AUML

Acciones ejecutadas en la interacción

Criterios para decidir cuándo ejecutar una tarea, consecuencias de la ejecución de una tarea.

Definición del contexto de la interacción

Objetivos que persigue la interacción, estado mental de los participantes

Modelo de control

Mecanismos de coordinación, comunicaciones entre agentes, actos del habla, determina un conjunto de primitivas con las que se comunican los agentes:

request: solicitar la ejecución de una acción
inform: modificar la información que almacena un agente
not-understood: no se ha comprendido el mensaje

Necesita de un lenguaje de contenido:

- XML
- SL0

Una ontología

A qué se refieren los elementos que aparecen en el mensaje, un protocolo *Fipa-request:* solicitar de un agente la ejecución de una tarea Protocolo

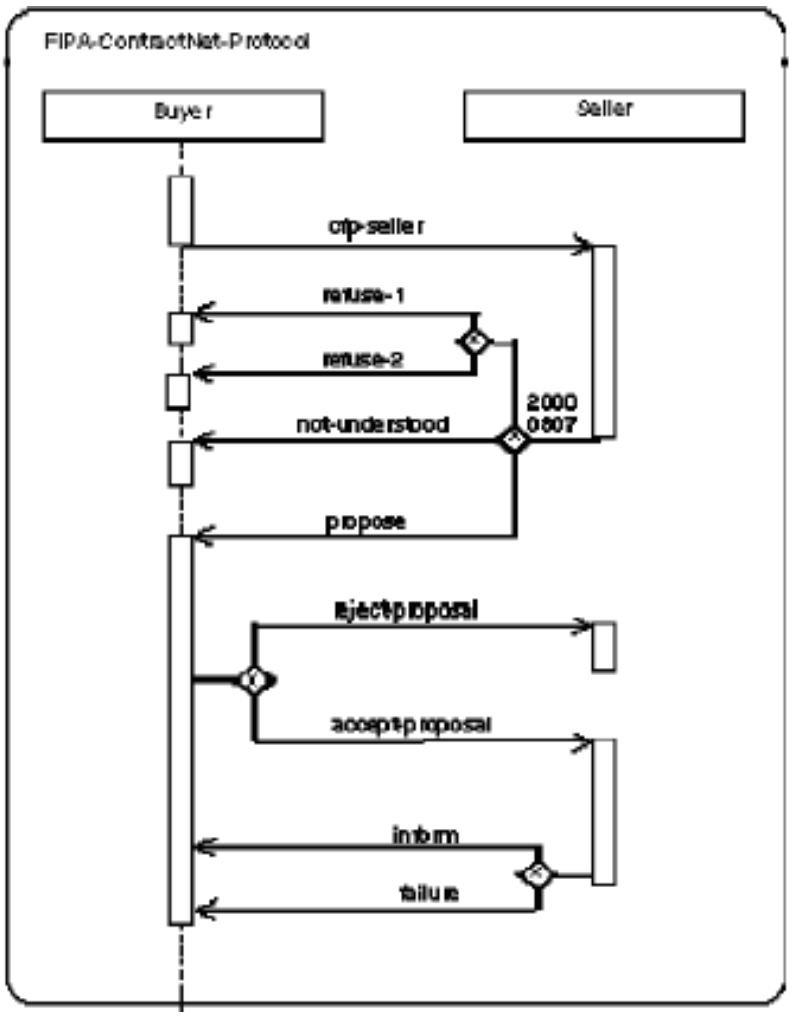


Figura 2.23: *Protocolos FIPA_CONTRCTNET*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

2.4.2.5 Modelo de entorno

Define las entidades del entorno del SMA con las que vaya a interactuar

Recursos

Elementos consumibles o no consumibles:

Descriptores de ficheros, hilos de ejecución, memoria, dispositivos de E/S, sockets, ancho de banda, etc.

Aplicaciones

Uso más complejo, por medio de alguna interfaz, se pueden ver como objetos o como agentes

Agentes

Satisfacen el principio de racionalidad

Tipos de entorno

Accesible/Inaccesible

Capacidad para percibir todo el entorno

Determinista/No determinista

Dado un estado y una acción ejecutada, se puede predecir el siguiente estado

Episódico/No episódico

La experiencia del agente se puede segmentar en episodios independientes

Estático/Dinámico

El mundo no cambia mientras el agente delibera

Continuo/Discreto

Existe un conjunto finito de variables a observar y un conjunto finito de acciones posibles

Formas de modelar el entorno

Representar el mundo que rodea al agente

Tarea extremadamente difícil

Enfoque pragmático y redes neuronales

Discretizar el entorno utilizando un conjunto finito de variables observables

Categorizar el tipo de entidades relevantes del entorno

Restringir la interacción (percepción y actuación) con estas entidades:

Recursos

Aplicaciones

Agentes

Ejemplo de entorno

Entorno para un asistente de ficheros en un PC

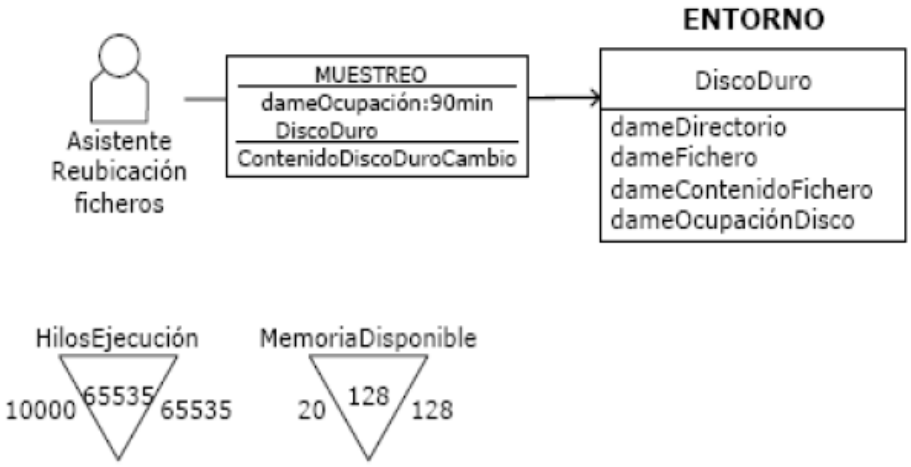


Figura 2.24: *Entorno de asistente de ficheros de un pc*

Fuente: *MODELADO DE SISTEMAS MULTI-AGENTE, TESIS DOCTORAL AUTOR: Doctor Jorge J. Gómez Sanz, Universidad Complutense de Madrid Junio 2002*

Son muchos diagramas, entidades y relaciones Por ello se define:

- Un proceso de desarrollo
- Compuesto de actividades
- Y que determina entregas a realizar
- Un entorno de desarrollo que facilite la implementación

Y damos

- Ejemplos de modelado
- Una tesis doctoral que describe la notación y el proceso
- Otras tesis próximamente (extensiones)

CAPÍTULO 3 – SOLUCIÓN PROPUESTA

3.1 ESTABLECIMIENTO Y JUSTIFICACIÓN DE LA SOLUCIÓN

Para obtener un Sistemas Tutores Inteligentes (STI) verdaderos, con módulos intercambiables, se debe partir del modelo tripartito propuesto por Carbonell (Carbonell, 1970) sin solapamiento de funciones.

De este modo se obtendrán módulos intercambiables que pueden interactuar entre sí. Se piensa entonces en módulos independientes con interfaces completamente definidas, de modo tal que puedan intercambiarse, por ejemplo el dominio, sin necesidad de modificar el resto de la estructura del Sistema Tutor Inteligente (STI).

Para el caso en que se plantea la solución, se ha tomado como dominio de aplicación el correspondiente al área de Algoritmos y Programación I. El módulo del tutor debe diseñarse de tal modo que pueda brindar una solución adecuada a los problemas que presentan los estudiantes.

Algunos investigadores presentan una estructura teórica (Carbonell, 1970; Salgueiro et al., 2004; Costa et al., 2004) que difiere de la real implementada en los Sistemas Tutores Inteligentes (STI). La diferencia entre la descripción teórica y la implementada se debe a que no están bien diferenciadas las distintas interfaces.

Se observa que muchos de los conocimientos particulares del dominio (pertenecientes al módulo de dominio) se encuentran dentro de los módulos del tutor y del estudiante tal como se observa en la Figura 3.1 existiendo zonas de solapamiento entre módulos.

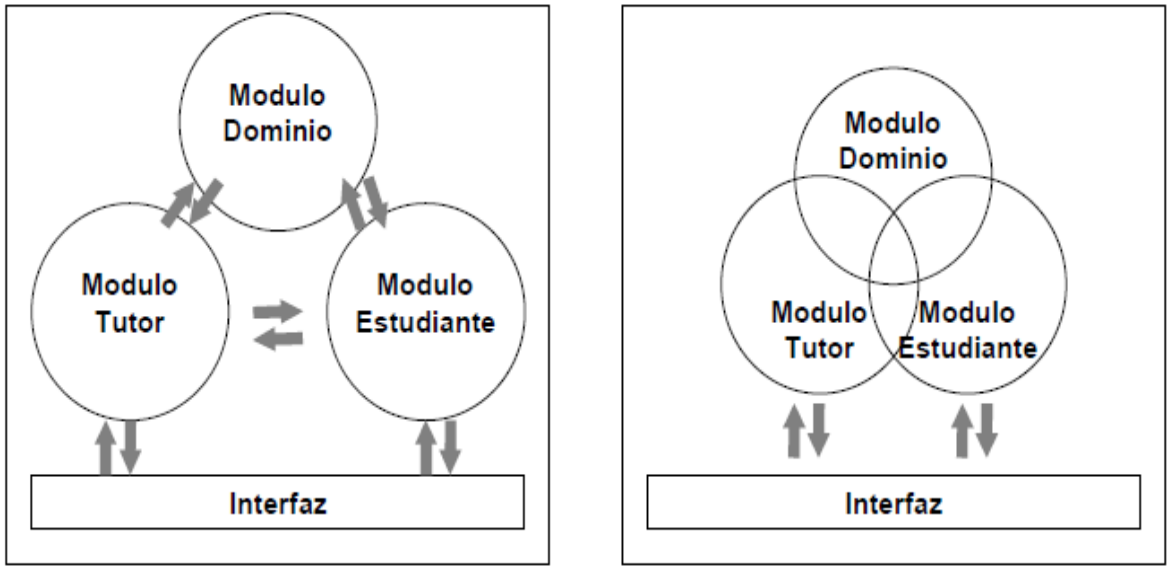


***Figura 3.1**: Diferencias entre el esquema teórico de un Sistema Tutor Inteligente y su versión fina implementada.*
***Fuente:** (Carbonell, 1970)*

Existen también otros problemas en las implementaciones de los Sistemas Tutores Inteligentes (STI), ya que muchas funciones están relacionadas tanto al módulo tutor como al módulo del estudiante. Por ello, se debe identificar qué módulo será el encargado de realizar cada una de las funciones del Sistema Tutor Inteligente (STI) a fin de que éste quede definido en su totalidad. De este modo se lo podría desarrollar a través de módulos completamente intercambiables e independientes del dominio de la aplicación.

La definición de cada una de las tareas particulares de los módulos, permite realizar agrupamientos de tareas en cada uno de ellos. A estos agrupamientos se los definirá como submódulos pudiéndose realizar un análisis similar al de los grandes bloques del sistema tutor inteligente, definiendo sus funcionalidades básicas y sus interfaces, para permitir la independencia de los submódulos.

Un entorno de desarrollo (framework) para la implementación de los Sistemas Tutores Inteligentes (STI), debe permitir la reusabilidad de código, como plantean El Sheikh y Sticklen (El Sheikh et al., 1998) entre otros. De acuerdo a lo planteado en la descripción del problema, en el modelo planteado el estudiante usuario debe comprender un tópico en

particular y el sistema tutor inteligente debe adaptarse de acuerdo a sus necesidades y preferencias. Es decir, se trata de un modelo de tutor adaptable.

Además de las necesidades de un mayor análisis sobre cada uno de los módulos planteados por Carbonell (Carbonell, 1790) para desarrollar Sistemas Tutores Inteligentes efectivos, éstos deben centrarse en las necesidades reales de los estudiantes usuarios.

Esto significa contar con los protocolos pedagógicos más adecuados de acuerdo a las necesidades y las preferencias de cada usuario del sistema en particular que tenga en cuenta la diferencia entre los distintos tipos de conocimientos a explicar: el conocimiento declarativo, que incluye los hechos, conceptos y vocabulario y el conocimiento procedural que incluye los pasos, las formulas y los algoritmos a utilizar en la resolución de problemas. Es decir, cumpliendo con la hipótesis principal: *"Un Sistema Tutor Inteligente (STI) que se adapte a las preferencias del estudiante este obtendrá mejores resultados".*

3.2 SUBMÓDULOS E INTERFACES

Para el análisis se partirá del diagrama básico de los Sistemas Tutores Inteligentes (STI) planteado por Carbonell (Carbonell, 1970) y Guardia Robles (Guardia, 1993), luego se ampliará ese esquema a los submódulos se ha detectado como faltantes (Salgueiro *et al*., 2004) cuya falta ha sido comunicada por Cataldi (Cataldi *et al*., 2004), por Costa (Costa *et al*., 2004) y por García-Martínez (García-Martínez *et al*., 2004) planteando de este modo las necesidades básicas de cada submódulo y cómo estos submódulos deben articularse entre sí para poder contribuir con partes completamente independientes.

A pesar de los avances tecnológicos, el modelado de los Sistemas Tutores Inteligentes (STI) sigue siendo una tarea compleja, ya que más allá de considerar los tres módulos básicos de la arquitectura tripartita propuesta por Carbonell (Carbonell, 1970) se deben tener en cuenta las interfaces, las comunicaciones entre los módulos y la gestión de la información en los mismos.

A la hora de modelar un STI se deben tomar en consideración las características del dominio (o sea del contenido), del comportamiento observable del alumno (modelos del alumno) y del conjunto de estrategias que serán abordadas por el módulo del tutor en búsqueda de una enseñanza personalizada (modelos de tutor).

Si bien el desarrollo de esta tesis solo abarca con profundidad el modelado del tutor, no se deben dejar de lado los otros módulos, ya que los datos que recibe o envía el módulo del tutor deben proceder de (o dirigirse a) alguna parte, ya que no todos provienen directamente del usuario.

Estos datos solo pueden provenir de las interfaces con los otros módulos, porque en caso contrario habría solapamiento entre los módulos. De este modo, mantener la estructura teórica no es suficiente para que quede definido el sistema sin superposiciones.

Hay que considerar el listado de funcionalidades de los módulos para que no se realicen dos veces las mismas tareas. La división de los módulos en funcionalidades puede evitar el solapamiento, como se verá en este apartado.

3.2.1 Componentes básicos de los Módulos.

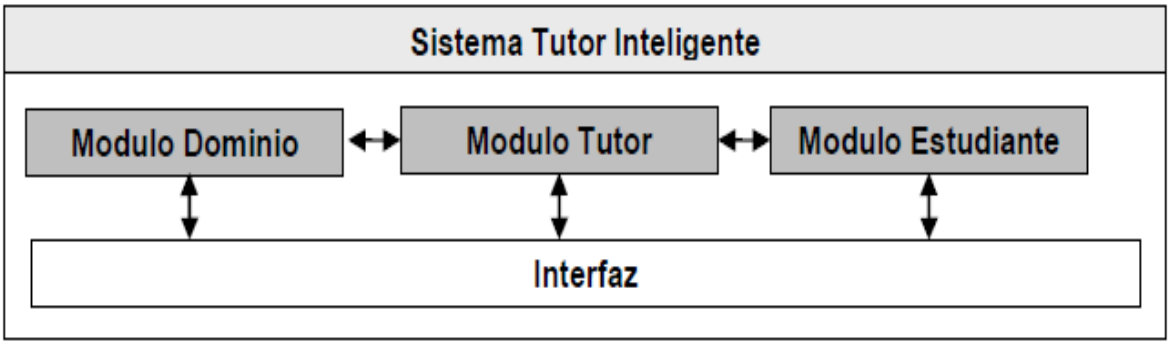


Figura 3.2: *Estructura clásica de un Sistema Tutor Inteligente propuesta por Carbonell*
Fuente: *(Carbonell, 1970).*

Figura 3.3: *Estructura de un STI, con la identificación de submódulos y un módulo evaluador.*
Fuente: *Elaboración Propia*

En la *figura 3.3* se observa la estructura básica de un Sistema Tutor Inteligente en forma teórica. Luego, la estructura del módulo del tutor se puede complementar con la del módulo del estudiante planteado por Costa (Costa et al.; 2005) y con el módulo de evaluación planteado por Cánepa Paulín (Cánepa Paulín et al.; 2004) en su plan de tesis en función de la metodología propuesta por Cataldi (Cataldi et al.; 2004).

3.3 ANÁLISIS DE LOS SUB - MÓDULOS

A continuación se describen cada uno de los submódulo que se proponen para el módulo del tutor. Se presentan las interfaces y las tareas específicas de cada uno de los submódulo, comparándolos con los desarrollos actuales en los Sistemas Tutores Inteligentes (STI) y se los ubica dentro de la estructura global del módulo del tutor.

3.3.1 Sub-Módulo Protocolos Pedagógicos

En la Figura se puede ver el esquema general del submódulo de los protocolos pedagógicos, con sus dos componentes básicos: el analizador de perfil y la base de protocolos pedagógicos disponibles. Este submódulo interactúa, con el resto de los submódulos del módulo del tutor y además es el que realiza las peticiones de los datos al módulo del estudiante, para saber cuál es el perfil de éste y su estado de conocimientos.

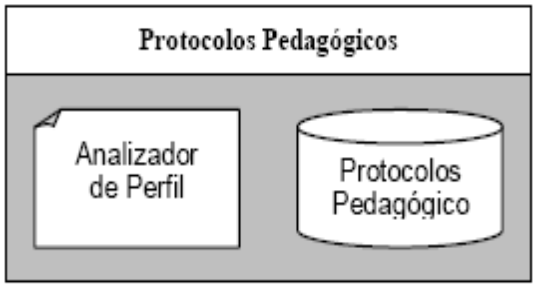


Figura 3.4: *Sub-módulo Protocolos Pedagógicos*
Fuente: *Elaboración propia*

El sub-módulo de protocolos pedagógicos consta de un analizador de perfil, para determinar, a partir de los datos almacenados en el módulo del estudiante, cuál será el mejor protocolo pedagógico para utilizar en la siguiente sesión pedagógica.
El sistema cuenta con una base de protocolos pedagógicos donde su uso estará subordinado a la disponibilidad de los contenidos en el módulo de conocimientos, pero la lección siempre se puede generar para alguno de los protocolos, o de una manera incompleta para los protocolos que requieran más recursos didácticos.

3.3.1.1. Agente Analizador de Perfil

El Agente Analizador de Perfil debe encontrar el mejor protocolo pedagógico disponible en el Sistema Tutor Inteligente (STI) que se adecue a las preferencias del estudiante para una determinada sesión pedagógica. La eficacia resultante en la elección de protocolo estará influida por la precisión en el método de selección utilizado.

Para obtener datos acerca del modo en que cada alumno aprende, se utilizarán las planillas de estilo de aprendizaje que son herramientas para la toma de datos. Se ha determinado la validez[17] y confiabilidad[18] de este instrumento a través de su aplicación por diversos investigadores desde la fecha de su creación (Felder, 1990; Figueroa, 2004). Partiendo de los datos que proveen las planillas se determinará el protocolo pedagógico de preferencia para cada estudiante en particular.

El objetivo de las planillas de estilos de aprendizaje es obtener datos acerca de la forma en que el alumno aprende normalmente de un modo más natural. En el Marco Teórico, sección *2.2.6.1– Técnicas Educativas Generales*, se analizaron los diferentes estilos de aprendizaje.

Se seleccionó la planilla de estilos de aprendizaje de Felder (Felder, 1998; Figueroa *et al*, 2004) entre las opciones disponibles, ya que la misma había sido utilizada con buenos resultados en estudios acerca de los estilos de aprendizaje de los estudiantes de Ingeniería Química e Ingeniería Informática. A su vez esta planilla es una herramienta validada, que permite obtener información sólida para dar de sustento a una metodología que abarque más con vistas a la aplicación en el ámbito específico de la Ingeniería Informática. La planilla completa de estilos de aprendizaje se puede ver en el Anexo 1 de esta tesis.

Se pretende administrar el cuestionario a los alumnos, con esos datos se propone utilizar las herramientas que proporciona la Inteligencia Artificial (AI), tales como las Agentes Inteligentes a fin de obtener la relación de las preferencias de los estudiantes con los protocolos pedagógicos.

A partir de una muestra significativa estadísticamente[19] de estudiantes de los cuales se tendrían las planillas de estilos de aprendizaje completas, se verá como los estilos de aprendizaje permiten el agrupamiento de acuerdo a las técnicas de enseñanza o protocolos

[17] Validez significa que la definición se ajusta al concepto y ésta debe referirse justamente a ese concepto y no a algo similar, para asegurar que se está midiendo justamente lo que pretende y no otra cosa.

[18] Confiabilidad se refiere a que si se repite la medición o el registro de información, el resultado será siempre el mismo, independientemente del investigador.

[19] Con un error de generalización menor al 3% para una población de 800 alumnos, el tamaño de la muestra es de 120 sujetos, como se muestra en el apartado 5.2.1.

pedagógicos. Esto permitirá correlacionar la preferencia del alumno con la disponibilidad del protocolo pedagógico más adecuado existente en el sistema.

Como la selección del protocolo pedagógico es uno de los elementos a determinar se desea agrupar los estudiantes en familias con características comunes, aunque desconocidas de antemano. Esto se puede llevar a cabo utilizando un Agente Inteligente que realice una "*clusterización*" o agrupamiento según determinadas características comunes del conjunto de individuos original.

El agente tiene un diseño de manera que utiliza aprendizaje no supervisado, donde las salidas que proporciona no poseen un valor esperado durante la etapa selección, por lo tanto los "*clusters*"[20] resultantes al aplicar la tarea del Agente a los datos de entrada, si bien poseen características en común, estas no pueden verse a simple vista.

En la *Figura 4.5* se puede ver activa solamente el conjunto *Mc* que mejor se adapta al patrón de entrada *x*. Existen también otros conjuntos *Mi* que se concuerdan aceptablemente con *x*, en la zona de *Mc* (mostrar la salida que proporciona el agente)

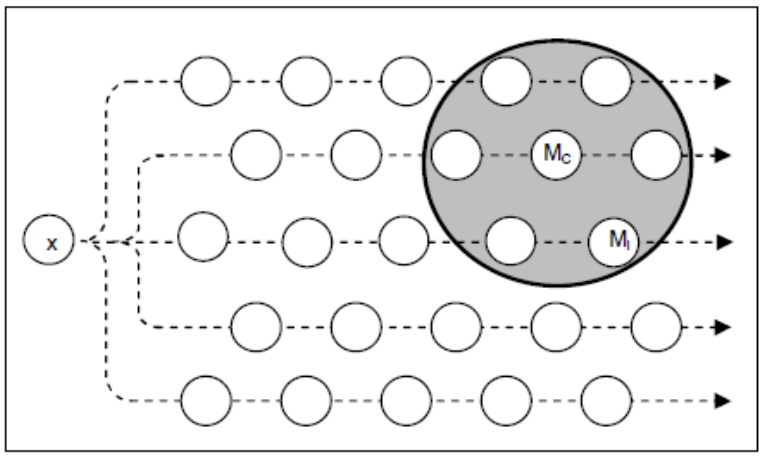


Figura 3.5: *Conjunto auto organizado. Cuando se ingresa un mensaje "x", este se envía a todos los conjuntos M_i del modelo.*
Fuente: *Elaboración propia*

[20] Un clúster es una familia de datos o atributos que poseen características en común

Una vez finalizado el proceso de formación del mapa topológico obteniendo los clúster o familias con características comunes, se agrega a cada uno de los individuos pertenecientes a la población original el identificador del grupo generado por el Agente para continuar con el análisis.

Una vez obtenidos los agrupamientos resultantes realizado por el Agente. Se utilizará un algoritmo de inducción para hallar las reglas que caracterizan a cada uno de estos grupos. En este caso se utilizarán algoritmos que pertenecen a la familia *Top-Down Induction Trees* (TDIT) (o en castellano *Árboles Inductivos de Arriba Hacia Abajo*).

Si bien existen varios algoritmos que realizan estas funciones, uno de los más completos es el C4.5 de Quinlan (Quinlan, 1993), que es una extensión del algoritmo ID3 (*Induction Decision Trees*) propuesto también por Quinlan (Quinlan, 1986).

Su objetivo es generar un árbol de decisión y luego las reglas de inferencia que caracterizan a dicho árbol. En este caso particular, el C4.5 tomará como entradas los datos de los estudiantes ya categorizados y como salida se obtendrán reglas del tipo:

Si Precondición$_1$, Precondición$_2$, …, Precondición$_n$ **Entonces** Cluster$_k$

Se utiliza el C4.5 en lugar del ID3 ya que el primero permite trabajar con valores continuos para los atributos, separando los posibles resultados en dos ramas: una para aquellos $Ai \leq Q$ y otra para $Ai>Q$, (donde *Ai* es un atributo y *Q* es un valor límite para el cual todos los elementos de la regla menores a Q se abren en un subárbol de reglas y los mayores en otro). El algoritmo genera un árbol de decisión a partir de los datos mediante particiones realizadas recursivamente.

El árbol se construye mediante la estrategia de *depth-first*(profundidad primero) considerando todas las divisiones posibles del conjunto de datos. Se selecciona, luego, la división del conjunto de datos que resulta en la mayor ganancia de información. En el caso

particular de que existan atributos discretos, se considerará una división por cada uno de sus *n* resultados.

C4.5, construye reglas que abarcan todos los datos, cubriendo todos los ejemplos positivos y ninguno de los negativos. En el caso de existir un ejemplo negativo a una de las reglas, se continúa avanzando en profundidad en el árbol de atributos.

Además, el algoritmo puede aceptar entradas incompletas que soluciona mediante pre o post procesamiento, donde se reduce la cantidad de reglas generadas mediante *pruning*[21] (truncamiento o poda), sin afectar la precisión de las reglas obtenidas.

El resultado será un conjunto de reglas que describen a cada uno de los clusters, pudiendo identificar las características de cada una de las familias de elementos encontradas. En la *Figura 3.7* se puede ver un resultado gráfico posible como salida del algoritmo de inducción C4.5.

Para clasificar a un individuo dentro de los grupos establecidos se debe partir del nodo denominado *Root* y se debe evaluar cada uno de los atributos hasta llegar a un nodo terminal que contiene la clasificación final.

[21] En el caso particular del algoritmo C4.5, el mecanismo de pruning funciona de la siguiente manera: una vez creado el árbol de decisión sobre un conjunto de instancias de entrenamiento, se realizan ajustes al árbol y se evalúa esta versión modificada contra un conjunto de instancias de validación. El algoritmo solo mantiene los cambios si el árbol modificado o *podado* (pruned) se comporta de manera superior al original. El uso de un conjunto de instancias de validación ayuda a eliminar el *overfitting*en la información por medio de la evaluación de las inconsistencias y las coincidencias de este conjunto contra el conjunto de instancias de entrenamiento. El error se reduce podando los nodos internos del árbol, reetiquetándolos con la nueva clasificación y descartando todos sus nodos descendientes. Esto obviamente limita el *overfitting*, pero es un proceso de costo computacional elevado.

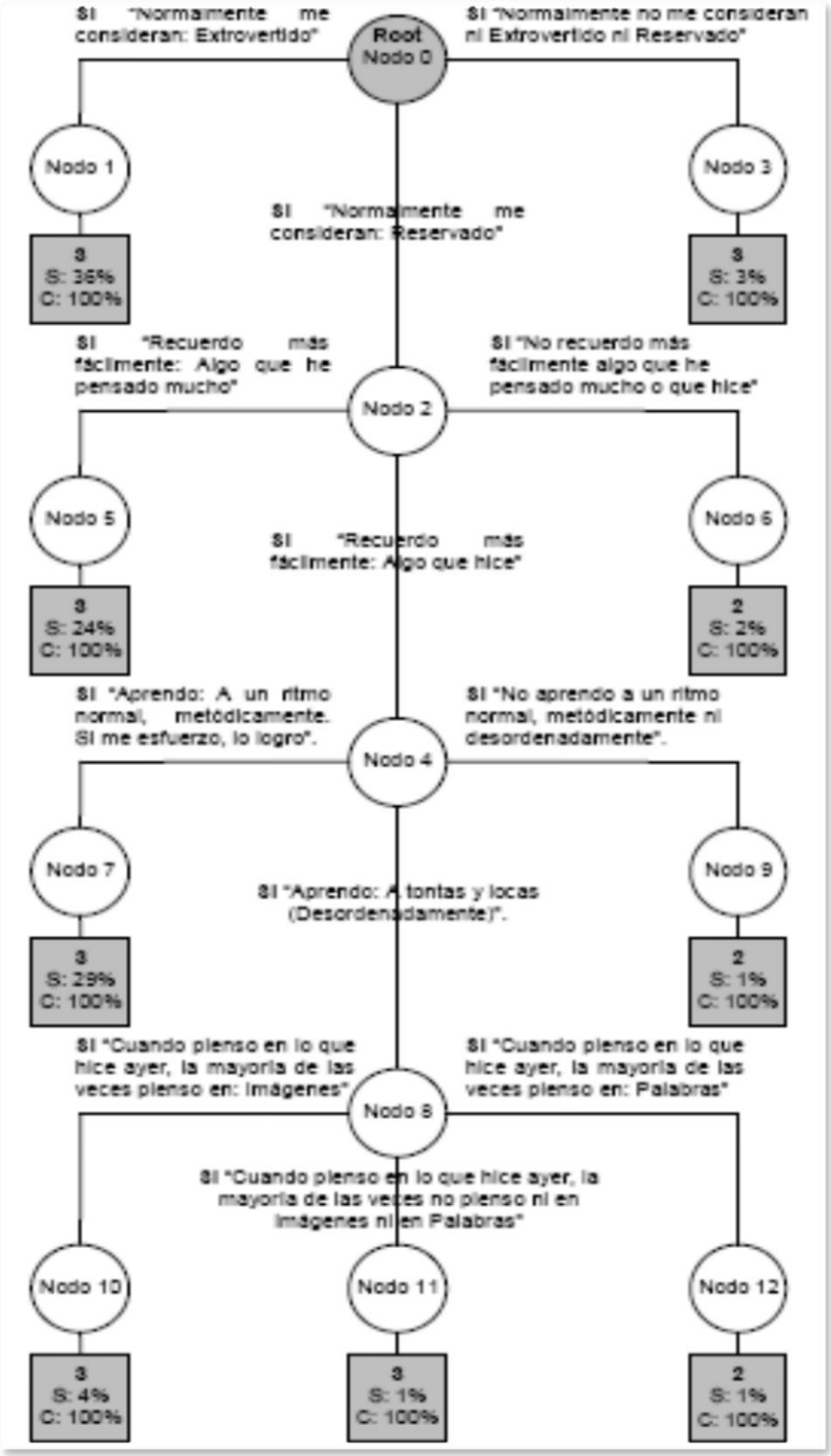


Figura 3.6: *Árbol de decisión generado por el algoritmo C4.5 a partir de la clasificación obtenida por el Agente, donde S representa el soporte y C es la confianza de la regla*

Una vez obtenida la menor cantidad posible de reglas por medio de pruning (poda) para evitar el overfitting[22] (sobre ajuste), se está en condiciones de pasar a otra etapa del análisis en la que, por medio de un proceso de inferencia, se relaciona las reglas descriptivas de los distintos clusters con los protocolos pedagógicos disponibles.

Para llevar a cabo la inferencia es necesario contar con datos adicionales del desempeño de los mismos estudiantes en el curso en estudio con diferentes protocolos de enseñanza. En la *Figura 3.7* se puede ver el esquema que representa al proceso de selección en forma global, donde se parte de una población de estudiantes de los cuales se posee sus preferencias con respecto a los estilos de aprendizaje a través de las planillas, se los agrupa usando un Agente Inteligente quien tiene la labor de arma una tabla de estudiantes clasificados, donde para todos los atributos que describen a la planilla se le asigna un clúster C_j predicho por el Agente y el algoritmo C4.5 se encarga de generar las reglas que describen a los clusters, relacionando el clúster C_j no con todos los atributos, como en la tabla de estudiantes clasificados, sino con un conjunto de reglas del estilo *Si* Pre_1 *y* Pre_2 *y … Pre*$_m$ *entonces* C_j.

Una vez identificadas las reglas que describen a cada uno de los clusters generados por el Agente, se realiza una inferencia, planteando una serie de hipótesis, para poder relacionar la relación entre los protocolos pedagógicos con los estilos de aprendizaje y agregar esta característica dentro del submódulo correspondiente de un STI.

[22] En el caso en que los datos relevados sean demasiado homogéneos (una varianza muy pequeña) el algoritmo de inducción podría encontrar un modelo demasiado optimista, que satisface demasiado la tendencia de los datos tomados, pero que a la vez es pobre para hacer predicciones para los datos que tiene una varianza grande, los cuales pueden ser los más representativos del dominio.

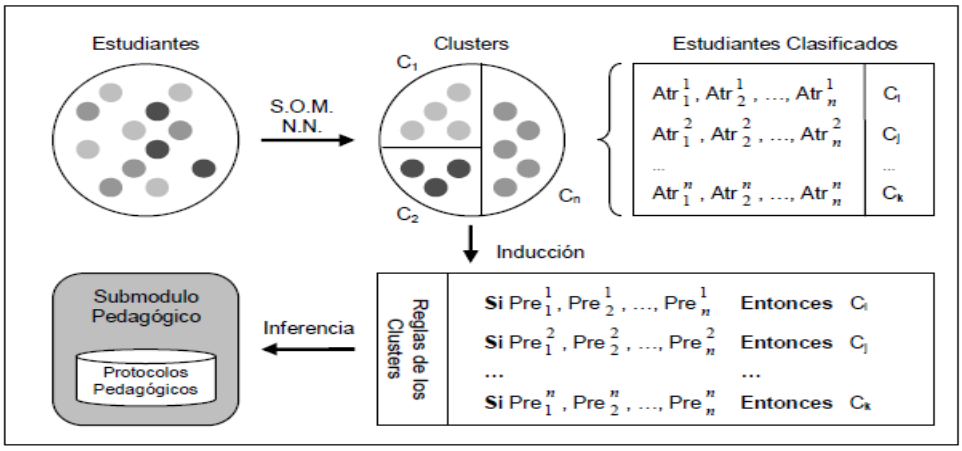


Figura 3.7: *Esquema básico para la inferencia de los distintos patrones para categorizar los estilos pedagógicos de los alumnos.*
Fuente: *Elaboración propia*

3.1.1.2 – Inferencia del Protocolo Pedagógico

En esta etapa se busca relacionar los agrupamientos generados por S.O.M. con los protocolos pedagógicos.

Para hallar la relación entre el estilo de aprendizaje y el protocolo pedagógico que corresponde a cada grupo se tomaran los protocolos básicos que describe Perkins (Perkins, 1995) en la *Teoría Uno*:

- **La instrucción didáctica o Magistral:** Satisface una necesidad que surge en el marco de la instrucción: la de expandir el repertorio de conocimientos del alumnado.

- **El entrenamiento**: Satisface la necesidad de asegurar una práctica efectiva.

- **La enseñanza socrática**: El docente ayuda al alumno a comprender ciertos conceptos por sí mismo y darle la oportunidad de investigar y de aprender cómo hacerlo.

Por lo tanto la investigación se orienta en la búsqueda de la relación entre la predilección de los alumnos de aprender de manera más eficiente y los protocolos pedagógicos utilizados por los tutores humanos (profesores); Para ello, como parámetro orientador se cuenta con las notas de las evaluaciones parciales para establecer dicha relación.

Para realizar la inferencia y relacionar los clusters obtenidos por medio de la utilización de los Agentes Analizador de Perfil con los protocolos pedagógicos se tomará un supuesto de dos cursos (A y B) en los que se ha dictado la materia de Programación I. El único cambio fundamental entre ambos se centró en la forma de enseñanza del docente, es decir, en el protocolo pedagógico utilizado para dictar las clases.

Desde este marco de referencia, se evalúan dos cursos según el control de variables planteado por García en el libro *El Paradigma del Pensamiento del Profesor* (García, 1995). Las variables planteadas para los cursos de referencia son las siguientes[23]:

- Contenidos de la materia similares.
- Horarios Similares (dentro de un turno, como puede ser mañana, tarde, noche).
- Bibliografía utilizada como referencias similares.
- Ingreso aleatorio de los estudiantes, sin preferencia definida por algún curso.
- Cantidad y formación similar en los ayudantes y Jefes de trabajos prácticos.
- Herramientas didácticas similares (como el uso de cañón y foros de discusión).
- Modo en el dictado de la clase, donde cada uno de los docentes presenta las clases en función del protocolo pedagógico que le resulta más natural de llevar a cabo entre las opciones posibles que se definen en la *Teoría Uno* y que se analizan en esta investigación, independientemente de las necesidades o preferencias de los estudiantes individualizados.

[23] Se ha observado que en el aprendizaje, no solo impactan las variables antes mencionadas, sino que también en el marco global de una clase para un grupo de personas, con uno o varios tutores humanos, sólo puede explicarse utilizando un protocolo pedagógico que debido al número de tutores limitados, no pueden individualizarse las necesidades de cada uno de los estudiantes, como casos particulares. Esta limitación no la poseen los Sistemas Tutores Inteligentes (STI), sino que por el contrario, éstos planean las sesiones en forma independiente para cada uno de los estudiantes usuarios, con objetivos definidos en función de las necesidades instantáneas de los mismos.

A partir del análisis de García, se observa para este estudio que la única que cambia es la denominada "*Modo de dictado de clase*", es decir, el protocolo pedagógico para cada curso.

Se tomarán como base para realizar la inferencia los datos correspondientes a las calificaciones en los primeros parciales, o las primeras oportunidades de evaluación (que pueden incluir eventualmente los exámenes de recuperación), ya que es el único estado del que se poseen datos completos y donde el estudiante es precisamente un novato en el tema.

En la etapa del final del curso, el estudiante ya ha puesto a prueba sus conocimientos durante el examen parcial y conoce, por lo menos, la mecánica de algunos ejercicios y la estructura general de las evaluaciones.

Para llevar a cabo la inferencia, se planteará la siguiente hipótesis: *Es posible relacionarlos estilos de aprendizaje con los protocolos pedagógicos*. De esta se desprenden dos hipótesis más particulares:

1. *La composición de estilos de aprendizaje (necesidades y preferencias de los estudiantes) de cada estudiante determina el estilo de enseñanza (o protocolo pedagógico) más adecuado.*

2. *Aquellos estudiantes en los que el estilo de enseñanza no coincide con su preferencia, presentan dificultades en la aprobación de la asignatura.*

A partir de la segunda hipótesis se desprende que para los alumnos aprobados, el protocolo mayoritario preferido por los estudiantes deberá ser el que coincida con el utilizado en clase, mientras que para los reprobados, el protocolo mayoritario deberá encontrarse invertido25[24].

[24] Si un alumno pertenece al protocolo mayoritario (predicho por S.O.M) de los reprobados, es porque su preferencia de protocolo pedagógico no coincide con el del curso en el que está inscripto y debería haberse anotado en otro.

Para validar esta afirmación el Agente Analizador de Perfil efectúa su función de acuerdo a las siguientes características:

- Se seleccionaron los aprobados del curso con profesor que dicta en estilo socrático más la mayoría de los reprobados del curso con profesor que dicta en modo magistral y el agente considera la salida como protocolo socrático.

- Se seleccionaron los aprobados del curso con profesor que dicta en estilo magistral más los reprobados del curso con profesor que dicta en modo socrático y el agente considera la salida como protocolo magistral.

Para realizar el entrenamiento de esta manera se debe suprimir el "*ruido*" que aportan los grupos que quedan fuera del análisis (los que aprobaron con cualquier protocolo, que se denominarán *indiferentes* y los que reprobaron por falta de estudio u otras razones) y se espera que el error de la herramienta sea menor que el porcentaje de elementos que quedan fuera del análisis. Por lo tanto, para cada clúster generado se analizará:

- Estudiantes Aprobados
  - Con protocolo Correcto → Es mayoritario
  - Indiferentes y mal clasificados → Es minoritario
- Estudiantes Reprobados
  - Con protocolo Invertido → Es mayoritario
  - Falta de estudio y otras causas → Es minoritario

En este punto la problemática se reduce a la *Figura 3.8* donde el profesor, divide al universo de estudiantes en dos conjuntos disjuntos: los aprobados y los reprobados basándose en su rendimiento académico, mientras que el Agente divide al universo de estudiantes en dos conjuntos disjuntos, el clúster 1 y clúster 2, basándose en el estilo de aprendizaje.

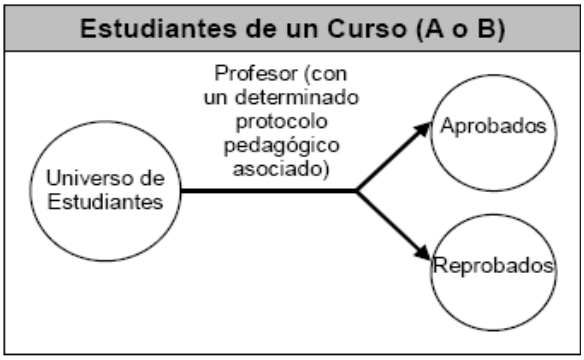
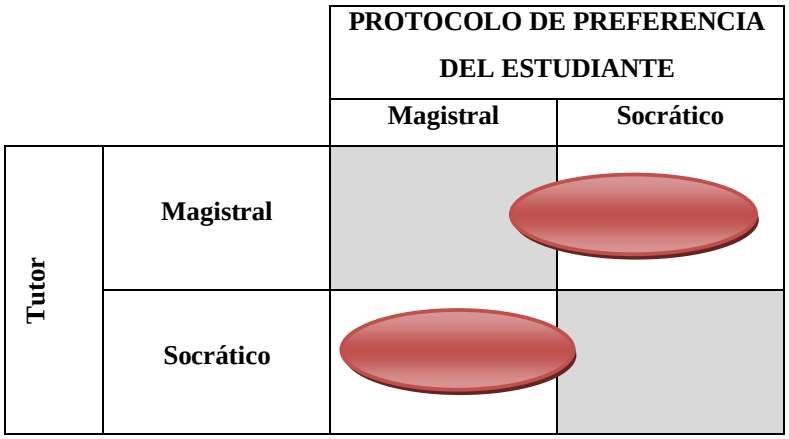


Figura 3.8: *Estudiante siendo evaluado por el docente*
Fuente: *Elaboración propia*

Ahora se busca relacionar las formas de enseñanza y los estilos de aprendizaje, tomando como base para el análisis a los estudiantes reprobados. En la *Figura 3.9* se observa, tomando como base un ejemplo en donde solo existen dos protocolos pedagógicos y dos preferencias en el conjunto de estudiantes, que los estudiantes cuya preferencia coincide con la forma o estilo de enseñanza no tienen problemas para aprobar. Existen dos subconjuntos (en rojo) de estudiantes cuya preferencia no coincide con la forma de enseñanza y son los que reprueban, ya que se encuentran "*mal ubicados*".



Problemática:

Se analizan los estudiantes desaprobados que prefieren el protocolo pedagógico invertido al utilizado al recibirlas clases. Estos son los que, según las hipótesis, están mal clasificados.

Figura 3.9: *Esquema general de la inferencia, para relacionar los clusters con los protocolos pedagógicos.*
Fuente: *Elaboración propia*

Siendo la hipótesis: *Los alumnos reprobados que no pertenecen al clúster mayoritario predicho por el Agente Inteligente deben tener la preferencia de un protocolo pedagógico diferente (invertido en este caso) al del profesor con que cursaron la materia.*

Por otra parte, se muestra la idea de la hipótesis, donde los alumnos reprobados, que no pertenecen al clúster mayoritario, deben tener una preferencia de protocolo pedagógico diferente al protocolo pedagógico recibido en las clases.

Existen además dos casos que no se encuadran en las ideas precedentes y que son:

- **Los estudiantes aprobados:** Que no han recibido lecciones utilizando el protocolo pedagógico de su preferencia y los que sí lo han hecho, que puede denominarse *indiferentes* al protocolo. Estos casos caen fuera del análisis, ya que es posible obtener resultados satisfactorios recibiendo lecciones con protocolos pedagógicos diferentes al de preferencia del estudiante. En este sentido Perkins (1995) sólo señala que la tendencia a fallar es mayor cuando el estudiante no recibe lecciones con el protocolo de su preferencia, pero no hace referencia alguna a los estudiantes que superan los exámenes con un protocolo diferente.

- **Los estudiantes reprobados**: Que han recibido lecciones utilizando el protocolo pedagógico de su preferencia: éstos casos se pueden deber a diversos motivos, como falta de estudio, falencias en la forma de estudiar (falta de una metodología de estudio), etc., que no modifican la inferencia realizada en este caso.

Por lo tanto, si se analizan los datos provistos por los docentes de ambos cursos con respecto a las categorizaciones realizadas por el Agente Inteligente, se puede obtener cuáles el porcentaje de alumnos que estarían *mal ubicados* en los cursos, y que se evidencian a través de los resultados *reprobados* obtenidos en las evaluaciones.

Para que el resultado obtenido sea satisfactorio, el Agente Inteligente deberá tener un error de clasificación menor al porcentaje de elementos que quedaron fuera del análisis, de esta manera esta herramienta será útil para la clasificación de las preferencias de enseñanza (protocolo pedagógico) de los estudiantes a partir de sus estilos de aprendizaje.

De este modo, no solo el submódulo analizador de perfil entrega el mejor protocolo pedagógico, sino que debería dar una escala o *ranking* de estos protocolos en orden descendente con respecto de su preferencia para el estudiante seleccionado. Luego, lo único que se requiere es recorrer todos los protocolos pedagógicos incluidos en el sistema, en el orden que se indica en este ranking y comprobar que el material necesario para llevar a cabo la lección utilizando ese protocolo en particular esté disponible en el módulo del dominio. Es necesario evaluar las opciones alternativas de protocolos pedagógicos ya que el módulo del dominio puede no contener almacenados los datos suficientes para completar satisfactoriamente una sesión pedagógica en todos los protocolos que se encuentran en el sistema.

3.3.2 Sub-Modulo Planeador de la Lección

Este submódulo interactúa con varios componentes del Sistema Tutor Inteligente para ejercer el control entre la interacción entre las preguntas y respuestas entre el sistema y el estudiante. Cuando se procesan las respuestas a una serie de preguntas realizadas por el sistema, el planeador de la lección debe decidir qué acción tomar a continuación, siguiendo las pautas básicas del protocolo pedagógico seleccionado, por lo tanto el planeador requiere conocimientos en cada punto de la lección para poder comunicarse con el usuario, esto se logra a través de reglas o *"guiones"*, las cuales se obtienen de los expertos por las técnicas de educción de conocimientos. Este proceso formal se lleva acabo utilizando la metodología para generar sistemas expertos.

Por otro lado se requiere mantener el estado de la interacción entre el sistema y el estudiante usuario, para ello se mantiene el stack de objetivos, donde se almacenan, en la forma de estructura del tipo *LIFO*[25]. El generador de contenidos se encargará de conocer el contexto de las respuestas del estudiante con respecto al valor tope del *stack* de objetivos (o a los primeros *n* valores del *stack*).

[25] LIFO es el acrónimo en inglés de *Last In – FirstOut*, o en castellano el último en entrar es el primero en salir. Es la metodología básica para la operación de una estructura del tipo *pila*.

3.3.2.1 Agente Generador de la Lección.

Una vez seleccionado el protocolo pedagógico por el Agente Analizador de Perfil, se deben generar los contenidos de la sesión a ser presentados al alumno usuario del sistema. Además el trabajo del Generador de la Lección es el de manejar la interacción entre el sistema y el alumno usuario.

Luego de la selección del protocolo pedagógico, el paso que sigue es encontrar cuál es el concepto a explicar en la lección, esto se realiza comparando el mapa de conocimientos que se obtiene del módulo del dominio y los conocimientos que el sistema supone que el estudiante posee que se obtiene del módulo del estudiante.

Se deben "*superponer*" los conocimientos del estudiante sobre el mapa de conocimiento que se obtiene del módulo de dominio para encontrar los conocimientos faltantes, y de acuerdo al tiempo disponible o una función que destaque la importancia parcial de cada uno de los temas, se agregaran al *stack* de objetivos de la lección.

El Agente Generador de contenidos debe ahora encargarse de presentar los contenidos en orden y procesar las respuestas de los estudiantes, agregando conceptos mal clasificados del alumno por el sistema en el *stack* de objetivos de la lección e informando de esta discrepancia entre la realidad y lo supuesto por el sistema en el módulo del estudiante.

Cuando el sistema detecta que el estudiante ha adquirido un nuevo concepto, o que se ha corregido un concepto erróneo, es el generador de contenidos el encargado de actualizar el estado del *stack* de objetivos y si este se encontrase vacío, dará por finalizada la sesión tutelar.

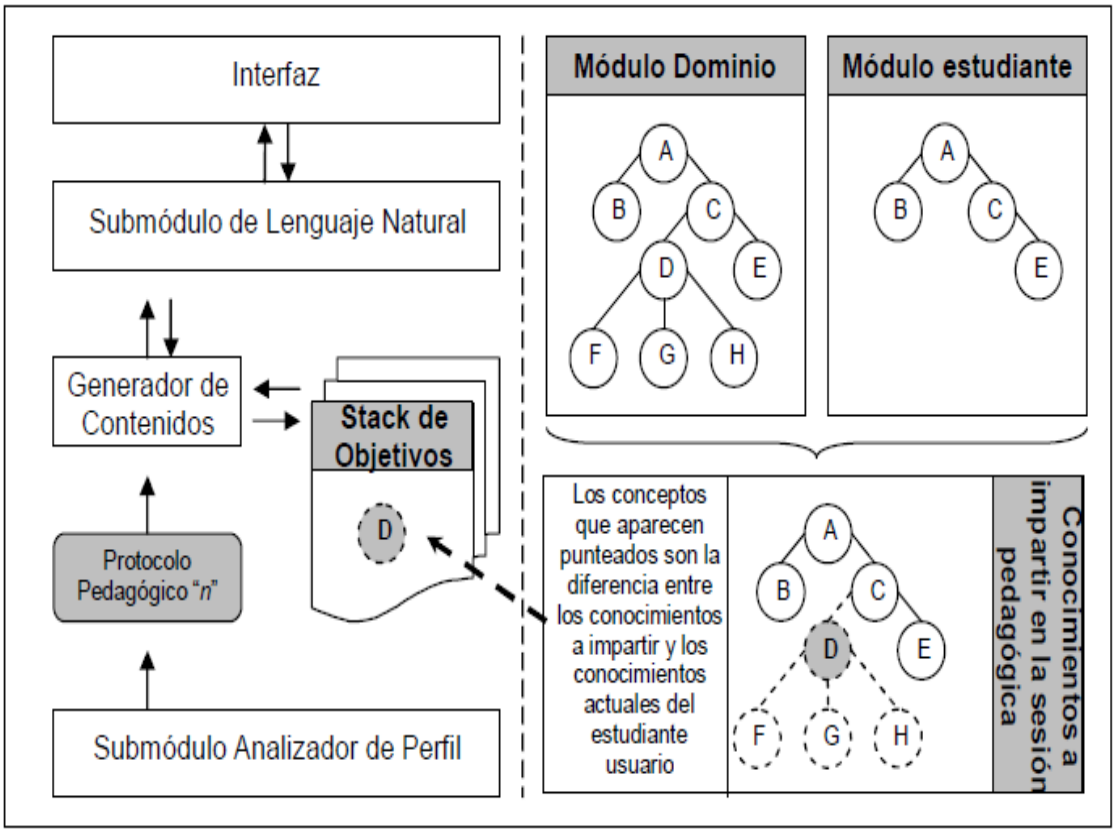


Figura 3.10: *Esquema general de los pasos a seguir luego de la obtención del protocolo pedagógico por el submódulo analizador de perfil y de la creación de los contenidos de la lección.*
Fuente: *Elaboración Propia*

Una vez que las salidas son procesadas por el generador del lenguaje natural, estas se envían directamente a la interface del sistema. Cuando el estudiante responde las preguntas del sistema, es decir, continúa la interacción planteada por el sistema, todos los datos son pasados desde la interface del sistema al módulo de lenguaje natural y de este, parte la información procesada para convertirse en la entrada del generador de la lección, el cual la evaluara y provocara las respuestas pertinentes. El esquema general del funcionamiento del generador de la lección puede verse en la *Figura 3.10* donde gestiona sobre el módulo del estudiante los conocimientos del alumno y sobre el módulo del dominio los conocimientos a explicar, los superpone y detecta los conocimientos faltantes en la estructura del estudiante y seleccionando uno y utilizando el protocolo pedagógico resultante del submódulo analizador de perfil se pasa a generar los contenidos necesarios.

Este módulo es el responsable de controlar la interacción entre el tutor y el estudiante, por lo tanto debe decidir cómo responder ante el estudiante dado un problema en particular (Woolf, 1984). La lección debe ser planeada especialmente y debe ser lo suficientemente flexible y coherente como para respetar la interacción. El ítem de la flexibilidad es central en el planeamiento de la lección, ya que en tutores más antiguos, como el tutor *Meno*(Woolf, 1984), el STI era capaz de seleccionar discursos almacenados con anterioridad, pero la falta de control global sobre la lección no permitían flexibilidad con respecto a las interacciones del alumno (el grado de la flexibilidad de la interacción está dada por el tamaño de los discursos almacenados: cuanto más largo el discurso almacenado, menor el grado de flexibilidad de la interacción).

Por lo tanto se pueden redefinir las funciones básicas del módulo generador de contenidos cómo decidir la forma de presentar al estudiante la información, el orden en la que esta información se presentará y cómo se responderá a las interacciones del estudiante usuario. Se pueden plantear los objetivos principales de este módulo como las mencionadas por WooWoo (WooWoo, 1991):

- Decidir cuándo presentar los contenidos de la lección, interactuando con los otros módulos del sistema tutor inteligente en general y con los submódulos del módulo tutor en particular, como el generador de lenguaje natural, el evaluador, los protocolos pedagógicos, etc.

- Corregir los conceptos erróneos del estudiante para un problema dado y generar el feedback en todas las respuestas del usuario hacia el sistema durante la sesión pedagógica.

- Si el estudiante usuario no logra dar con la respuesta correcta, el tutor debe proveer los medios para que el estudiante se encauce nuevamente en el camino correcto para solucionar el problema. Es aquí donde se insertaran los conceptos de pistas que aporta Hume (Hume, 1995; Hume, 1996). Es en este paso donde esta función difiere

levemente con lo planteado por WooWoo, ya que se utilizara el *stack* de objetivos para conocer el estado de la lección.

- La forma en la que el discurso de la sesión se encuentra en el sistema, debe estar en forma explícita, en lugar de almacenado en forma procedural como en el tutor Meno (Woolf, 1984). Esto debe permitir modificar las estrategias de enseñanza a medida que la lección progresa en el tiempo. Se pueden utilizar las técnicas de *"guiones"* para codificar las acciones posibles en el discurso dentro de cada uno de los temas o unidades didácticas. Así se pueden generar los contenidos de una manera sencilla y lo suficientemente flexible como para obtener un mayor rendimiento de una interacción entre el sistema y el estudiante usuario.

- El sistema debe responder las preguntas del usuario con la respuesta más apropiada dentro de todas sus opciones (Solo se están evaluando las interacciones que son iniciadas por el sistema y no por el estudiante).

3.3.2.2 Determinación de Objetivos de la Lección

La determinación de los objetivos de la lección es una tarea esencial a llevar a cabo por los Sistemas Tutores Inteligentes, ya que debe generar objetivos globales, coherentes y consistentes para que estos sean impartidos a cada estudiante en particular. (WooWoo, 1991). Dado que cada uno de los conceptos almacenados en el módulo del dominio para la asignatura están relacionados entre sí, la secuencia en la que éstos se presentan afectará de manera decisiva la performance del sistema en su conjunto y las necesidades del estudiante. La determinación de los objetivos dejará planteada la estrategia del sistema para una sesión pedagógica en particular.

WooWoo (WooWoo, 1991) plantea la determinación de objetivos utilizando como base una "*tabla de predicción*" que almacena los resultados en el módulo del estudiante. Estas ideas son implementadas en el tutor *CircSim* (Kim, 1989; Kim, 2000; Cho, 2000; Hume, 1995;

Shah, 1997; Hume *et al*., 1992) y pueden generar un planeamiento tanto de tipo dinámico como de tipo jerárquico.

Por lo tanto, la determinación de los objetivos de la lección no solo requiere el uso de conocimientos almacenados en el módulo del estudiante, para determinar lo que éste conoce y lo que desconoce, sino que también requiere meta información sobre los conceptos a impartir. La meta información es información adicional acerca del conocimiento y se utiliza en el caso del módulo del dominio para determinar las dependencias entre los distintos conceptos, el tiempo que puede tardarse en impartirlos y la importancia de los mismos (si son un concepto aislado, si son subtemas o si son temas principales o capítulos). Con la meta información se puede armar el mapa de conocimientos del alumno como el conjunto básico de conocimientos necesarios para finalizar con la curricula de la asignatura.

Por otra parte no es el objetivo de una lección pedagógica obtener las dependencias del módulo del dominio e impartirlas todas en una única "*mega*" sesión pedagógica, sino que la idea de la flexibilidad de un Sistema Tutor Inteligente también reside en que este sea capaz de evaluar los conocimientos adquiridos por el estudiante y actuar conforme a ello. Es por esto que se requiere contrastar los datos del módulo del dominio con el mapa de conocimientos instantáneo del estudiante que se obtiene en el módulo del estudiante para encontrar cuales son los conceptos posibles y disponibles para impartirle al estudiante. Un esquema del procedimiento a seguir puede verse en la *Figura 3.11*.

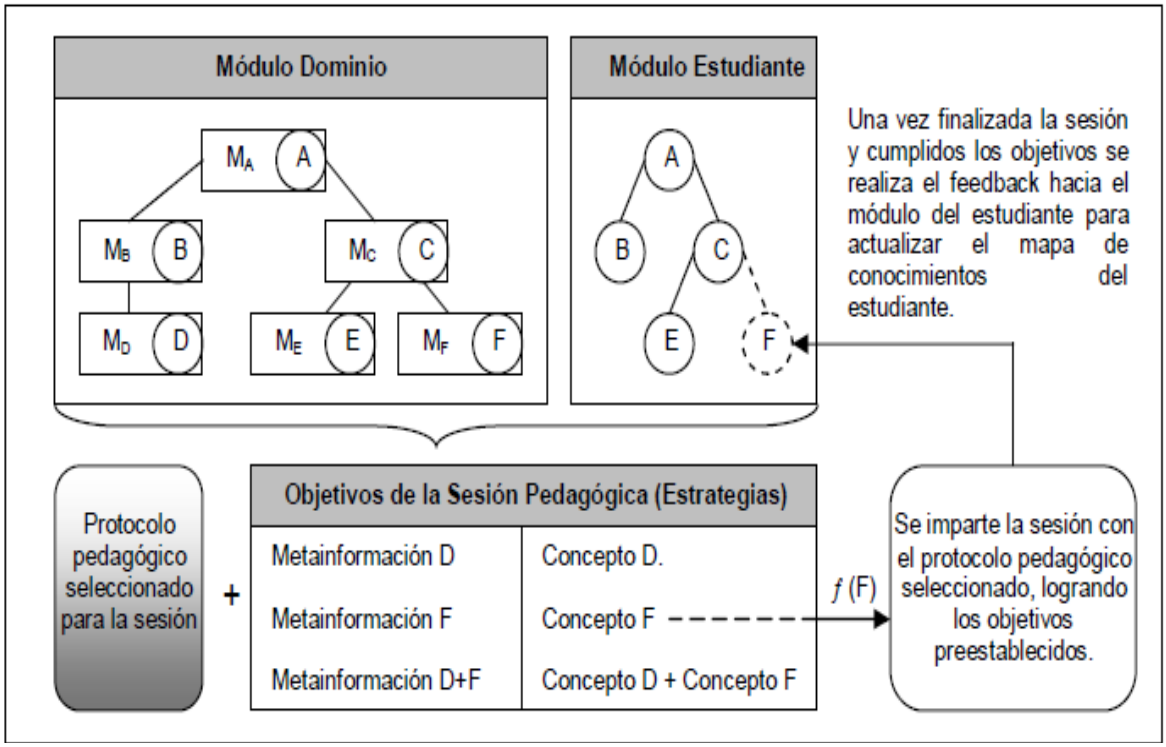


Figura 3.11: *Esquema general para la selección de objetivos para la sesión pedagógica y retroalimentación luego de finalizar dicha sesión.*
Fuente: *Elaboración propia*

Una vez que se determinan todas las combinaciones válidas entre los conocimientos faltantes en el mapa de conocimientos del estudiante y los conceptos almacenados en el módulo de dominio debe existir una función para determinar cuál de todas las combinaciones será la seleccionada para la sesión en particular. Se debe tener en cuenta que el número de opciones puede ser relativamente grande y éstas requieren un análisis más complejo, el cual no es el objetivo de esta tesis explicar.

Existen distintas opciones para implementar la función que determinará cuál de todas las opciones será la seleccionada para una determinada sesión pedagógica. Por ejemplo, en el tutor *CircSim*(Kim, 1989; Kim, 2000; Cho, 2000; Hume, 1995; Shah, 1997; Hume*et al*., 1992) se selecciona el objetivo de la sesión por medio de un conjunto de reglas que toman como base los conocimientos del estudiante (WooWoo, 1991), estas reglas son tomadas de sesiones de educación de conocimiento con tutores humanos, pero fallan nuevamente en la separación de los módulos, ya que los conocimientos resultantes a enseñar deben provenir del módulo de dominio y no estar escritos en forma "*Hard-Coded*" dentro del módulo del

estudiante. Otra opción es, por ejemplo, la que se utiliza en el tutor Meno (Woolf, 1984) donde se va avanzando linealmente por cada uno de los temas de la curricula.

Independientemente el método que se utilice para la creación de la función para determinar el objetivo de la lección, los parámetros sobre los que se puede realizar la evaluación están relacionados con los meta conocimientos que almacenan los conceptos en el módulo del dominio. Por ejemplo, se pueden citar los siguientes criterios como evaluación de cada una de las opciones: duración, profundidad, relación con la curricula docente, etcétera.

El funcionamiento del modelo con sus diversos componentes se lo puede ver en la *figura 3.12*, el modelado del tutor utilizando la metodología Ingenias se la puede apreciar en el Anexo 2.

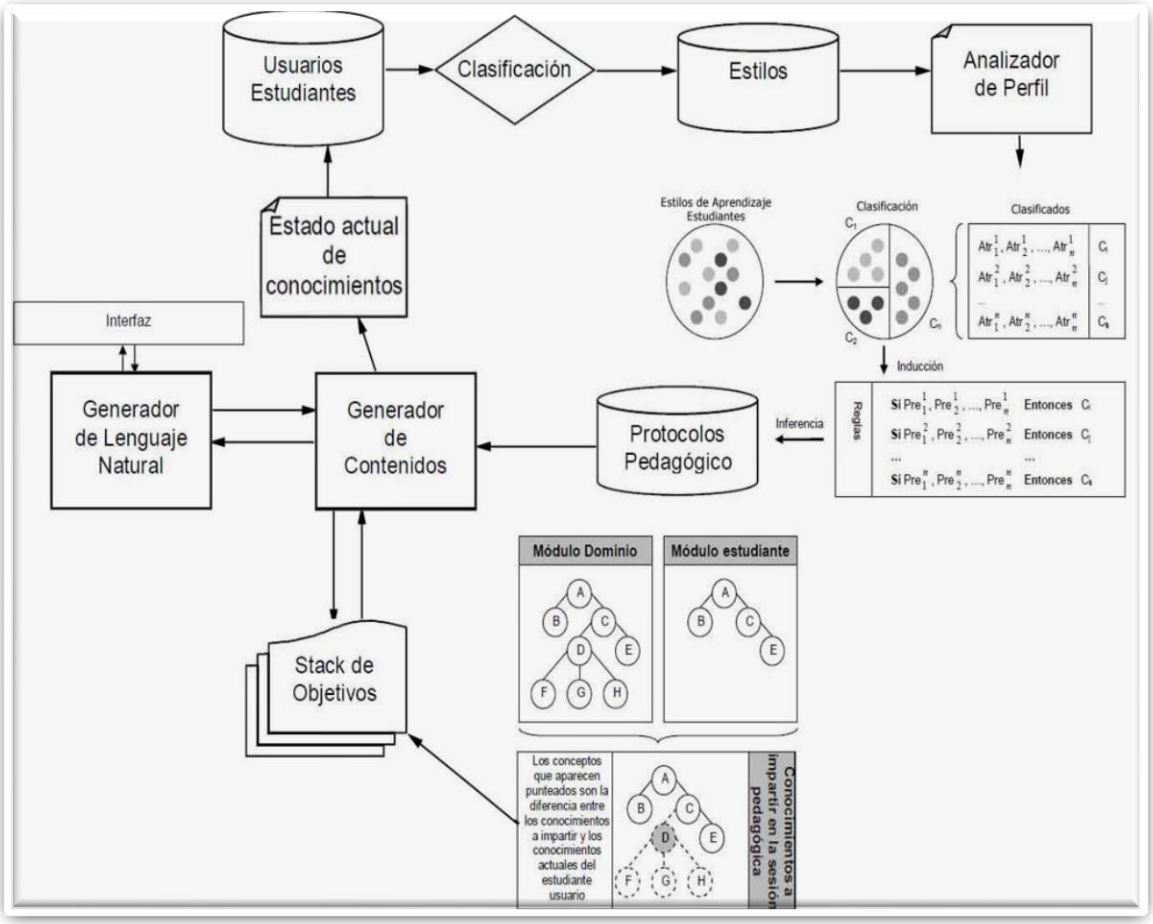


Figura 3.12: *Modelo propuesto del funcionamiento del módulo del tutor.*
Fuente: *Elaboración propia.*

Capítulo 4 – Conclusiones

4.1 CONTRIBUCIÓN REALIZADAS

Se comenzó determinando el estado actual de las investigaciones sobre el tema de Tutorizado inteligente, partiendo como base de trabajos generales, desde el histórico Acholar de Carbonell, hasta el moderno *CircSim*de Evens para analizar lasfuncionalidades básicas, los puntos fuertes y las debilidades de los Sistemas TutoresInteligentes.

Luego se continuó analizando los casos concretos de Sistemas Tutores Inteligentes que comparten el campo de aplicación de esta tesis, como el *Meno Tutor* de Wolf, el Proust de Jonhson y el tutor *Coach* de Selker para determinar el estado de desarrollo de los mismos. De esta manera quedan establecidos los fundamentos básicos que sustentan a los distintos tipos de Sistemas Tutores Inteligentes, así como también el modelo tripartito de módulos propuesto por Carbonell y extensamente utilizado en el ámbito delos STI.

Se procedió luego al análisis de las interfaces, sub-módulos y componentes fundamentales para detectar los fallos en las implementaciones previas y alterar una estructura teórica con más de 35 años para soportar las nuevas tecnologías.

Se modelaron las partes por separado y se las integro dentro de la arquitectura planteada de Sistema Tutor Inteligente, utilizando en el caso del sub-módulo de protocolos pedagógicos las herramientas que brinda la Inteligencia Artificial (IA) para modelar el comportamiento del sistema como un tutor humano. Se utilizaron Agentes Inteligentes y demostraron, en el paso posterior del modelo, de ser aptos para la tarea propuesta, ya que según su arquitectura pueden clasificar a los distintos estudiantes de una manera satisfactoria y funcionar como el pilar del módulo para la selección del protocolo pedagógico que más se adapte a las necesidades del estudiante usuario.

De esta manera queda cumplido el objetivo general, ya que para el modelo planteado, que es capaz de adaptarse a las necesidades y preferencias de los alumnos, según sus estilos de aprendizaje. Además, cumple con la condición de ser uninstrumento adicional para realizar Tutorizado "*uno a uno*" entre el estudiante y el tutor virtual.

Queda cumplido también el objetivo de este trabajo de proveer una herramienta adicional como punto de partida en futuras investigaciones que a largo plazo se convierta en una ayuda para los tutores humanos, quienes pueden relegar algunas de sus tareas que, ya sea por falta de tiempo o recursos, no pueden cumplir de la forma más satisfactoria para el estudiante, mientras que provee un soporte secundario para los alumnos que quieren complementar sus conocimientos o regular su propio ritmo de aprendizaje.

Con el objetivo del trabajo cumplido, se provee al campo de los Sistemas Tutores Inteligentes de una nueva herramienta, para facilitar la selección del protocolo pedagógico adecuado, redundando esto en una ganancia, no solo para el desempeño del STI en sí mismo, sino en el estudiante, que es el componente humano fundamental que hace útil al sistema y le brinda identidad. Así se pretende realizar un aporte y mejorar el desempeño académico de los distintos estudiantes y por ende su calidad de vida.

H1) La composición de estilos de aprendizaje (necesidades y preferencias de los estudiantes) de cada estudiante determina el estilo de enseñanza (o protocolo pedagógico) más adecuado.

La hipótesis 1 se valida ya que se ha encontrado una herramienta que es capaz de determinar el estilo de enseñanza partiendo de la planilla de estilos de aprendizaje.

H2) Aquellos estudiantes en los que el estilo de enseñanza no coincide con supreferencia, presentan dificultades en la aprobación de la asignatura.

Los estudiantes mayoritarios reprobados (75%) tienen que ser nuevamente clasificados, por el Agente Inteligente, como pertenecientes a otro protocolo pedagógico que difiere del utilizado en las clases tomadas.

Las conclusiones a las que se ha llegado con esta investigación se las puede resumir de la siguiente manera:

- La herramienta más adecuada para la toma de información es la planilla de estilos de aprendizaje se ajusta a lasnecesidades para relevar información sobre las distintas preferencias de los estudiantes con respecto a sus estilos de aprendizaje y a su vez suformato permite un post-procesamiento como el realizado en este trabajo.

- Se pueden organizar en 2 grupos los datos y este número está dentro del rango delos protocolos pedagógicos predichos por la *Teoría Uno*. Estos grupospresentan características similares

- De todos los atributos analizados por la herramienta,Existen atributos que presentan una mayor ganancia de información y seha detectado que para las preguntas específicas de la planilla estos son:

    1. Edad
    2. Sexo
    3. Carrera

    Mientras que para los datos adicionales la mayor ganancia deinformación se encuentra en los atributos:

    1. Normalmente me consideran: Extrovertido / Reservado /Ninguna.
    2. Recuerdo más fácilmente: Algo que he pensado mucho / algo quehe hecho / ninguno.
    3. Aprendo: A un ritmo normal, metódicamente. Si me esfuerzo, lologro / A tontas y locas Desordenadamente) /Ninguna.

4. Cuando pienso en lo que hice ayer, la mayoría de las veces nopienso ni en Imágenes ni en Palabras En ambos casos, el orden en que se presentan es decreciente con respectoa la ganancia de información.

- Para utilizar estilos de aprendizaje y clasificarlos, toda la información necesaria estará disponible dentro del módulo del estudiante y no requiere interacción extra con la interfaz, y por lo tanto se mantiene la independencia de los módulos así como también la estructura propuesta por Carbonell. Si la información que se requiere para la identificación adecuada de los protocolos pedagógicos no se encuentra disponible en el sistema, esta debería ser ingresada directamente al módulo del tutor,rompiendo la independencia de los módulos.

- Existe una forma de relacionar las familias generadas por el Agente Inteligente y el protocolo pedagógico, utilizando la hipótesis siguiente:

  H1) La composición de estilos de aprendizaje (necesidades y preferencias de los estudiantes) de cada estudiante determina el estilo deenseñanza (o protocolo pedagógico) más adecuado.

  La hipótesis 1 se valida ya que se ha encontrado una herramienta que es capaz de determinar el estilo de enseñanza partiendo de la planilla deestilos de aprendizaje.

- La utilización de un Agente Inteligente permite armar un ranking de protocolos más adecuados.

- Dentro del modelo se utiliza las nuevas tecnologías, sobre todo las de los ambientes distribuidos, así como también las tecnologías que propone la Inteligencia Artificial (IA), en este caso los Agentes Inteligentes.

- El modelo propuesto aporta una solución viable para la implementación de un STI capaz debrindar un protocolo pedagógico que mejor se adapte a cada estudiante,

mejorando sus posibilidades de aprobación y por ende combatiendo el problema de el alto índice de desaprobados.

4.2 RECOMENDACIONES

Dentro de las recomendaciones y las futuras líneas de investigación, para ampliar el desarrollo de este trabajo es la de adaptar las ideas aquí planteadas a otras estructuras similares a la de los Sistemas Tutores Inteligentes (STI), como pueden ser lo los Asesores Inteligentes o mismos Sistemas Tutores Inteligentes que no poseen un modelo de interacción tradicional. De aquí también se desprende que las mismas ideas pueden ser adaptadas a modelos en los que no se aplica el marco de referencia tripartito planteado por Carbonell, adaptando y redistribuyendo los submódulos que desempeñan las tareas del tutor a las nuevas estructuras, que, si bien difieren de la estructura clásica, deberánseguir cumpliendo las mismas funciones y logrando los mismos objetivos.

Otro campo en el que la investigación puede seguir su curso es en el de agregado de nuevos módulos a la estructura clásica. Se mantienen los tres módulos básicos, pero se agregan módulos para ampliar los campos de aplicación del sistema, como puede ser el módulo evaluador o el módulo ambiental, propuesto por varios autores.

De esta manera, las interfaces que unen a los diferentes módulos añadidos deberán ser definidas para que la interacción con el módulo del tutor se realice de forma eficiente y sin la superposición de funcionalidades que tanto ha demorado el desarrollo de Sistemas Tutores Inteligentes.

BIBLIOGRAFÍA

Ausubel, D.; Novak, J.; Hanessian, H. (1983) *Psicología educativa: un punto de vista cognitivo*. México: Trillas.

Barr, A.; Feigenbaum, E. A. (1981). *The Handbook of Artificial Intelligence*. Volume I. William Kaufman, Los Altos, CA.

Bhatt, K.; Evens ,M.; Argamon, S. (2004): *Hedged Responses and Expressions of Affect in Human/Human and Human/Computer Tutorial Interactions*. Computer Science Department, Illinois Institute of Technology. Proceedings of COGSCI 2004, Chicago, IL

Boff, E.; Giraffa, L. M. (2000). *Construyendo un ambiente de enseñanza-aprendizaje cooperativo: una experiencia interdisciplinaria*. En SBIE 2000 – Simposio brasilero de informática educativa. Maceio, Halagaos. Anales. p. 112-119.

Bolan Frigo, L.; Pozzebon, E.; Bittencourt, G. (2004). *O Papel dos Agentes Inteligentes nos Sistemas Tutores Inteligentes.* World Congress on Engineering and Technology Education (WCETE'2004). Guarujá / Santos, SP. Proceedings of the World Congress on Engineering and Technology Education.

Brown, E.; Palincsar, B. (1989) *Guided, Cooperative learning and the individual knowledge acquisition*. En Resnick B. (comp.) Knowing, learning, and instruction. Hillsdale. N.J.

Bruner, J. (1990). *Actos de significado. Más allá de la revolución cognitiva*. Alianza. Madrid. 2002.

Bruno, O. (2003). *Resultados de investigación acerca de los problemas de aprendizaje de algoritmos*. UTN.

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

Burton, R. R.; Brown, J. S. (1981). An investigation of computer coaching for informal learning activities. In: Sleeman, D., Brown, J. (eds.): Intelligent Tutoring Systems, Capitulo 4, p. 79-98, London: Academic Press.

Carbonell, J. R. (1970). *AI in CAI: An artificial intelligence approach to computer assisted instruction.* IEEE transaction on Man Machine System. Volumen 11 número 4, p. 190-202.

Castillo, E.; Gutiérrez, J. M.; Hadi A. S. (1998). *Sistemas Expertos y Modelos de Redes Probabilísticas*. Monografías de la Academia Española de Ingeniería, Madrid.

Cataldi, Z. (2004). *Metodología para el diseño y evaluación de sistemas tutores inteligentes*. FI-UNLP. CAPIS. ITBA.

Cataldi, Z.; Figueroa, N.; Lage, F.; Denazis, J. (2003). *Analysis of the interactions between students taking part in problem based learning experiences using software for groupware in an initial course of computer science engineering career*. Full Paper Aceptado en FIE 2003. 35th ASEE/IEEE Frontiers in Education Conference. 5-8 de noviembre. Colorado.

Cho, B. (2000). *Dynamic Planning Models to Support Curriculum Planning and Multiple Tutoring Protocols in Intelligent Tutoring Systems.* Ph.D. tesis, Illinois Institute of Technology.

Duda, R. O.; Gaschnig, J. G.; Hart, P. E. (1980). *Model Design in the Prospector Consultant System for Mineral Exploration*. En Michie, D., editor, Expert Systems in the Microelectronic Age. Edinburgh University Press, Edinburgh, p. 153–167.

El Sheikh, E.; Sticklen, J. (1998). *A Framework for Developing Intelligent Tutoring Systems Incorporating Reusability*. IEA-98-AIE: 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Benicassim, Castellón, España, Springer-Verlag (Lecture Notes in Artificial Intelligence, vol. 1415).

Evens, M. W.; Spitkovsky, J.; Boyle, P.; Michael, J.; Rovick, A. A. (1993). *Synthesizing tutorial Dialogues*. Preoceedings of the 15th Annual Conference of the Cognitive Science Society.

Felder, R.; Soloman, B. (1998). *Learning styles and strategies*. Consultado el 21 de junio de 2004 en www.2.ncsu.edu/unity/lockers/users/f/felder/ .

Fernández, C. T.; Lima, M. D. De Alecar; Zuanásbar, D.; Adeche H. M. (2000) *ambiente de apoyo del desenvolvimiento de cursos hipermedia en la web*. SBIE 2000- Simposio brasilero de informática educativa.Maceio, Halagaos. Anales. p. 24-31.

Fernández, I. (1989) *Estrategias de Enseñanza en un Sistema Inteligente de Enseñanza Asistida por Ordenador.* Tesis Doctoral (Tercer Ciclo) de la Universidad del País Vasco, San Sebastián.

Figueroa, N.; Cataldi Z.;Lage F. J.; Salgueiro F. A.; Costa G.; Rendón J. (2004). *Nuevos enfoques para el estudio del desgranamiento universitario*. CAEDI 2004: IV Congreso Argentino de enseñanza de Ingeniería. 1, 2 y 3 de Septiembre, ITBA, Buenos Aires.

Figueroa, N.; Lage, F.; Cataldi, Z.; Denazis, J. (2003). *Evaluación de las experiencias para mejoramiento del proceso de aprendizaje en asignatura inicial de la carrera ingeniería informática*. International Conference on Engineering and Computer Education, ICECE 2003, paper 804. 16-19 de Marzo San Paulo.

García-Martínez, R.; Servente; M. y Pasquini (2003) *Sistemas Inteligentes*. Nueva Librería. ISBN 987-1104-05-7.

Gardner, H. (1993) *Las inteligencias múltiples. La teoría en la práctica*. Paidós. Barcelona.

Gardner, H. (2003). *Multiple Intelligences after twenty years.* American Educational Research Association.

Gelernter, H. (1959). *Realization of a geometry theorem proving machine*. In Proceedings of an International Conference on Information processing, pages 273- 282, Paris. UNESCO House.

Gertner, A. S; Conati; C.; VanLehn, K. (1998). *Learning Procedural help in Andes: Generating hints using a Bayesian network student model*. Research & Development. American Association for Artificial Intelligence.

Giraffa, L. M. M.; Viccari, R. M. (1999). *Intelligent tutoring system built using agent techniques*. Revista de educación, ciencia y cultura, Canoas: La Salle, p. 23-40. Grossberg, S. (1976). *Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors.* Biological Cybernetics 23, p 121-134.

Guardia Robles, B. (1993). *Asesores Inteligentes para apoyar el proceso de enseñanza de lenguajes de programación*. Tesis de grado. Asesor. ITESM: Instituto Tecnológico de Monterrey.

Hebb, D. O. (1949). *The Organization of Behavior*, NY: John Wiley & Sons. Hecht-Nielsen, R. (1988), *Applications of counterpropagation networks.* Neural Networks, 1, 131-139.

Hernández, C.; García, P. (1991). *Las teorías de la psicología educativa: análisis por dimensiones educativas,* México, Facultad de Psicología UNAM.

Hernandez Sampieri, C. (2001). *Metodología de la investigación*. Mc Graw Hil. México.

Herrmann, N. (1996) *The Whole brain business book*, McGraw-Hill.

Hume G., Michael, J; Rovick, A.; Evens, M. (1996), *Hinting as a tactic in one-on-one tutoring*. Journal of Learning Sciencies.

Hume G., Michael, J; Rovick, A.; Evens, M. (1997). The *Use of Hints by Human and Computer Tutors: The Consequences of the Tutoring* . Illinois Institute of Technology.

Hume, G. D. (1995). *Using Student Modelling to Determine When and How to Hint in an Intelligent Tutoring System* Ph.D., Illinois Institute of Technology.

Hume, G.; Evens, M. (1992) *Student modeling and the classification of errors cardiovascular intelligent tutoring system.* Proceedings of the 4th Midwest Artificial Intelligence and Cognitive Science Society Conference, Utica, IL.

Hunt, E. B. (1975). *Artificial Intelligence*. New York. Academic Press, EE.UU. Iglesias, C. A.; Garijo, C.; González, J. (2001). *Metodologías orientadas a agentes.*

Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial. Número 6, Volumen 2. p. 12-23.

Kawasaki, E. I.; Imar, N.; Fernández, C. T. (2000). *Un modelo de sistema de tutoría inteligente basado en principios pedagógicos para educación de adultos*. En SBIE 2000: Simposio brasilero de informática educativa. Maceio, Halagaos. Anales. p. 154-159.

Keefe, J. (1988). *Aprendiendo Perfiles de Aprendizaje*. Asociación. Nacional de Escuelas Secundarias.

Kim, J. H. (1989). *CIRCSIM-Tutor: An Intelligent Tutoring System for Circulatory Physiology*. Ph.D. tesis, Illinois Institute of Technology.

Kim, J. H. (2000) *Natural Language Analysis and Generation for Tutorial Dialogue.* Ph.D. tesis, Illinois Institute of Technology.

Konzen, A. A. (1995) *Uma estratégia de ensino híbrida para sistemas tutores inteligentes.* En: Simposio Brasilero de Informática en Educación, 1995, Florianópolis, Brasil. Imprensa Universitária da Universidade Federal de Santa Catarina, p.327–338.

Krishnamoorthy, C. S.; Rajeev, S. (1996). *Artificial Intelligence and Expert Systems for Engineers.* CRC Press, CRC Press LLC.
Laird, J. E., Newell, A., and Rosenbloom, P. S. (1987). *SOAR: an architecture for general intelligence*. Artificial Intelligence, 33(1): 1-64.

Madoz, C; DeGiusti, A. (1997) *Vinculacion de un curso interactuvi multimedial*. Congreso Argentino de Ciencias de la Computación. CACIC97. La Plata. UNLP.

Maturana, H. (1998) *Da Biologia à Psicologia* Artmed Editora Ltda..

Minsky, M. (1974). *A Framework for Representing Knowledge*. MIT-AI Laboratory Memo 306. Reimpreso en The Psychology of Computer Vision, P. Winston, McGraw-Hill, 1975. versión corta en J. Haugeland, Mind Design, MIT Press, 1981, and in Cognitive Science, Collins, Allan and Edward E. Smith. Morgan- Kaufmann, 1992.

Minsky, M. L. (1954). Neural *Nets and the Brain-Model Problem (SNARC)*. PhD thesis, Princeton University.

Myers, I., Briggs, K. (1962) *The Myers-Briggs Type Indicator*, Princeton Educational testing Services.

Newell, A. (1969). *Heuristic programming, in Ill-Structured Problems in Progress in Operations Research.* Ed. I. S. Aronofsky, John Wiley and Sons, New York.

Perkins, D. (1995) *La escuela inteligente*. Gedisa.

Piaget, J.(1989). *La construcción de lo real en el niño*. Crítica. Grijalbo.

Platon (2001), la Republica. Ed Tecnos. Reimpresion 2001.

Quinlan, J. R. 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann.

Rosenblatt, F. (1958), *The perceptron: A probabilistic model for information storage and organization in the brain*. Psychological Review, 65, p. 386-408.

Rovick, A.; Brenner L. (1983). *HEARTSIM: A Cardiovascular Simulation with Didactic Feedback.* The Physiologist, vol. 26 no. 4, p. 236-239.

Rumelhart, D. E.; Hinton, G. E.; Williams, R. J. (1986). *Learning internal representations by back-propagating errors in Parallel Distributed Processing*: *Explorations in the Microstructure of Cognition*. Eds. Cambridge, MA: MIT Press, vol. 1, p. 318-362.

Russell,S.J. y Norvig P. (2003). *Artificial Intelligence: A Modern Approach* (2nd Edition). Prentice Hall.

Self, J. (1998). *The defining characteristics of intelligent tutoring systems research: STIs care, precisely*. Computer Based Learning Unit University of Leeds, Leeds LS2 9JT, England. Luego publicado en el International Journal of Artificial intelligence in education (1999), Leeds, England.

Seu, Jai, Ru-Charn Chang, Jun Li, Evens, M.; Michael, J. and Rovick, a. (1991). *Language Differences in Face-to-Face and Keyboard-to-Keyboard tutoring Session*. Proceedings of the Cognitive Science Society.

Shah, F. (1997). *Recognizing and Responding to Student Plans in an Intelligent Tutoring System: CIRCSIM-Tutor* Ph.D. tesis, Illinois Institute of Technology.

Sierra, E., Hossian, A. y García Martínez, R. (2001). *Selección de Estrategias Instruccionales. Abordaje desde la Teoría del Conocimiento*. Revista del Instituto Tecnológico de Buenos Aires. Volumen 25. Páginas 24-36.

Sierra, E.; García-Martínez, R.; Cataldi, Z.; Britos, P.; Hossian, A. (2003). *Sistemas Tutoriales Inteligentes Centrados en Reparación de Mecanismos. Una Propuesta Metodológica de Diseño*. Revista Latinoamericana de Tecnología Educativa. Volumen 1 N° 1. Páginas 19-30. Facultad de Educación. Universidad de Extremadura.

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

Stevens, A.; Collins, A. (1977). *The goal structure of a Socratic tutor*. In Proceedings of the National ACM Conference. New York: ACM.

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, A. (1950). *Computing machinery and intelligence*. Mind, vol. LIX, no. 236, p. 433-460.

VanLehn, K (1988). *Student Modelling.* M. Polson. Foundations of Intelligent Tutoring systems. Hillsdale. N.J. Lawrence Erlbaum Associates, 55-78

Vigotsky, L. (1978) *Mind in society. The development on Higher psychological process*. Harvard University Press Cambridge M.A

Villareal Goulart R. R.; Giraffa M. L. (2001). *Arquitecturas de Sistemas Tutores Inteligentes* .

Wenger, E. (1987). *Artificial intelligence and tutoring systems. Computational and Cognitive Approaches to the Communication of Knowledge*. Los Altos C. A. Morgan and Kaufman.

Winograd, T. (1972). *Understanding natural language*. Cognitive Psychology, 3(l).Reprinted as a book by Academic Press.

Wolf, B. (1984). C*ontext Dependent Planning in a Machine Tutor.* Ph.D. Dissertation, University of Massachusetts, Amherst, Massachusetts.

Woo Woo, C. (1991). *Instructional planning in an Intelligent Tutoring System: Combining global lesson plan with local discourse control*. Degree of Doctor of Philosophy in Computer Science in the Graduate School of the Illinois Institute of Technology. Chicago, Illinois. December, 1991.

Woods, W. A. (1973). *Progress in natural language understanding: An application to lunar geology*. In AFIPS Conference Proceedings, volume 42, p. 441-450.

Yang A., Kinshuk & Patel A. (2002). *A Plug-able Web-based Intelligent Tutoring System.* In S. Wrycza (Ed.) Proceedings of the Xth European Conference on Information Systems (June 6-8, 2002, Gdańsk, Poland), Gdansk, Poland. Páginas 1422-1429.

Zeve C. M. D.; Sloczinski H.; Polonia E.; Nitz-Ke J. A.; Lima J. V. (2000) *Aprendizaje colaborativo: la utilización de cd-rom e Internet en un sistema integrado.* En SBIE 2000: Simposio brasilero de informática educativa.Maceio, Halagaos. Anales. p. 66-72.

ANEXOS

ESTILOS DE APRENDIZAJE DE LOS ALUMNOS DE INGENIERÍA

Carrera:.....Universidad:……..

Año de ingreso:……..Año que cursa:……..

Edad:…….Sexo: M F

1. Entiendo mejor un tema después de:
   - Probarlo/ejercitarlo
   - Pensarlo

2. Prefiero ser considerado como:
   - Realista
   - Innovador

3. Cuando pienso acerca de lo que hice ayer, la mayoría de las veces pienso en:
   - Imágenes
   - Palabras

4. En general, tiendo a:
   - Entender los detalles de un tema puesto que la estructura general puede ser confusa
   - Entender la estructura general de un tema puesto que los detalles son confusos

5. Cuando aprendo algo nuevo me ayuda:
   - Hablar acerca de ello
   - Pensar acerca de ello

6. Si fuera profesor, preferiría enseñar:
   - Sobre la base de hechos y situaciones reales
   - En base a teorías e ideas

7. Prefiero obtener nueva información en forma de:
   - Dibujos, diagramas, gráficos o mapas
   - Instrucciones escritas o información verbal

8. Una vez que entiendo:

- Todas la partes, yo entiendo el tema en su conjunto
- El tema en su conjunto, veo como encajan las partes de ese tema

9. Estudiando en grupo, con temas difíciles probablemente:
   - Camine, y contribuya con ideas
   - Este sentado, y escuche

10. Es más fácil para mí:
    - Aprender hechos.
    - Aprender conceptos.

11. En un libro con muchos dibujos y esquemas probablemente:
    - Mire los dibujos y esquemas cuidadosamente
    - Me focalice sobre el texto escrito

12. Cuando resuelvo un problema en matemáticas:
    - Trabajo a mi manera, para resolver un paso por vez
    - La mayoría de las veces solo veo la solución y tengo que esforzarme para darme cuenta de los pasos para llegar a ella

13. En el aula:
    - Suelo conocer a la mayoría de mis compañeros
    - Raramente conozco a la mayoría de mis compañeros

14. Cuando leo libros que no son de ficción, prefiero:
    - Algún libro que enseñe nuevos hechos o me diga cómo hacer algo
    - Libros que me den nuevas ideas acerca de las cuales pensar

15. Prefiero docentes que:
    - Pongan muchos dibujos o diagramas en el pizarrón
    - La mayor parte del tiempo estén explicando en forma verbal

16. Cuando analizo una historia o novela:
    - Pienso acerca de los hechos y trato de juntarlos para comprender el tema
    - Termino de leer para comprender el tema, y luego vuelvo atrás para encontrar los hechos que lo demuestran.

17. Cuando inicio la resolución de un problema en casa:
    - Inmediatamente busco la solución

- Primero trato de entender completamente el problema y luego buscar la solución

18. Prefiero la idea de:
    - Certeza
    - Teoría

19. Recuerdo mejor:
    - Lo que veo
    - Lo que escucho

20. Es más importante para mí que el docente:
    - Presente el material de estudio en pasos claros y secuenciales
    - Me dé un pantallazo general y presente material relacionado con otros temas

21. Prefiero estudiar:
    - En grupo
    - Solo

22. Probablemente sea considerado:
    - Cuidadoso con los detalles de mi trabajo
    - Creativo acerca de cómo hacer mi trabajo

23. Cuando recibo indicaciones para ir a un lugar nuevo, prefiero:
    - Un mapa
    - Instrucciones escritas

24. Aprendo:
    - A un ritmo normal, metódicamente. Si me esfuerzo, lo logro
    - A tontas y a locas (desordenadamente). Estaré totalmente confundido y de repente "click"

25. Preferiría primero:
    - Probar o experimentar (cosas) soluciones
    - Pensar la solución o como haré ciertas cosas

26. Cuando leo por placer, me gustan los escritores:
    - Que dicen claramente lo que piensan
    - Que expresan sus ideas en forma creativa e interesante

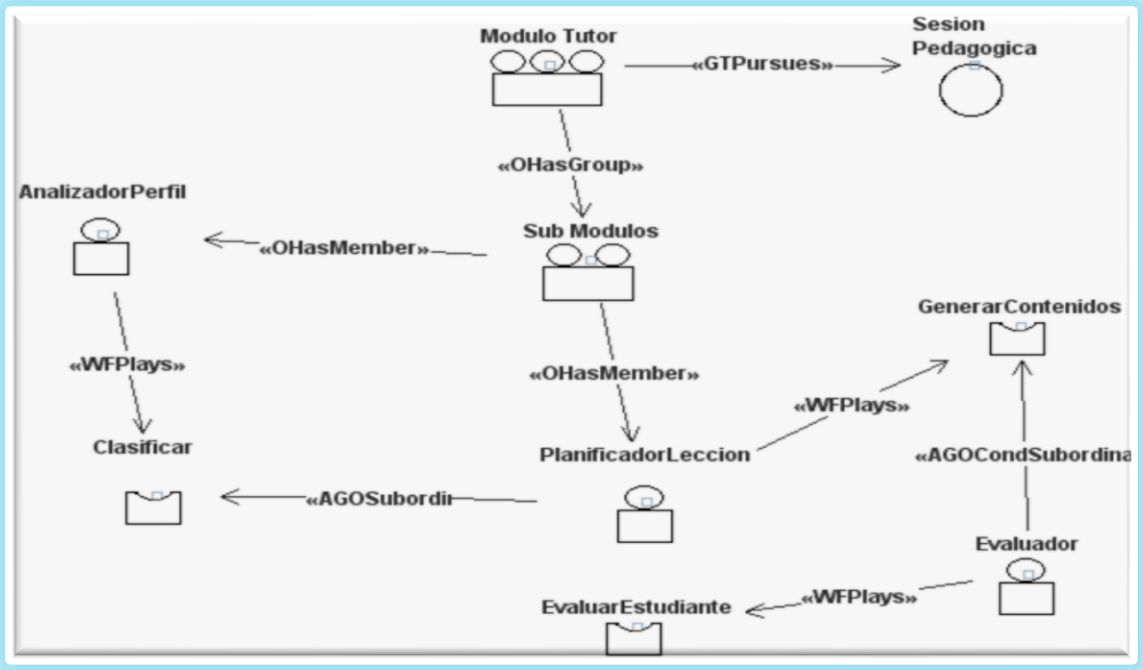
27. Cuando veo un diagrama o diseño en clase, es probable que recuerde:
    - El dibujo
    - Lo que el docente dijo acerca de ello

28. Cuando considero nueva información, probablemente:
    - Me focalice en los detalles, perdiendo de vista el esquema general.
    - Trate de entender el esquema general antes de adentrarme en los detalles

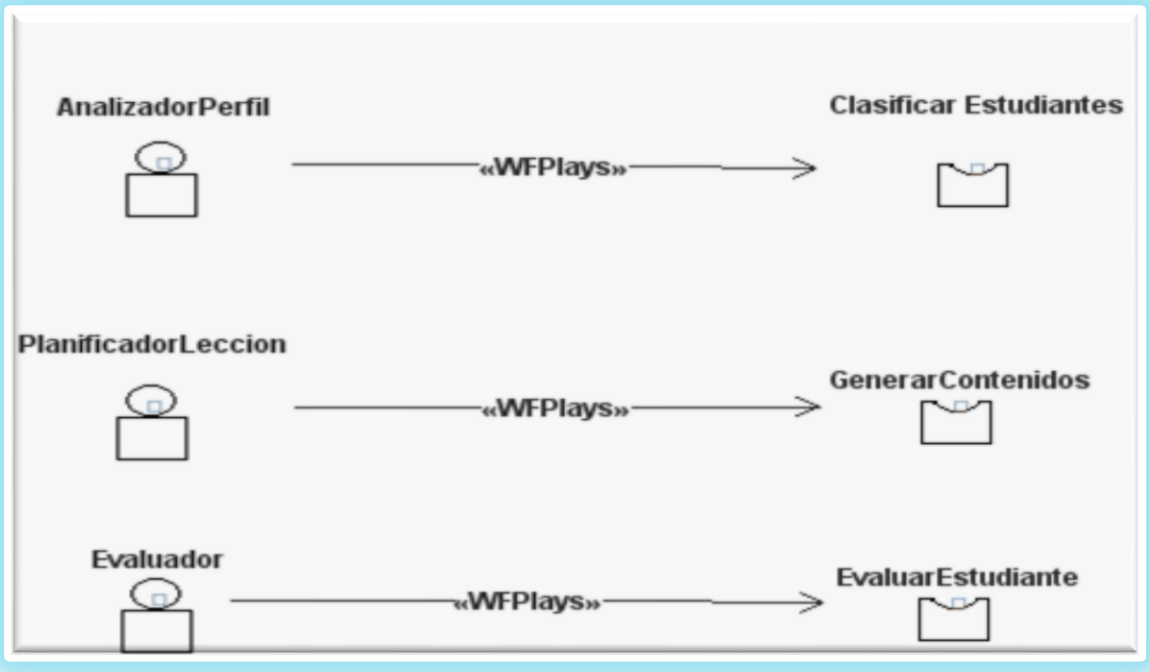
29. Recuerdo más fácilmente:
    - Algo que hice
    - Algo que he pensado

30. Cuando tengo que realizar una tarea, prefiero:
    - Conocer a fondo la manera que la llevare a cabo
    - Tratar nuevas maneras de realizarla

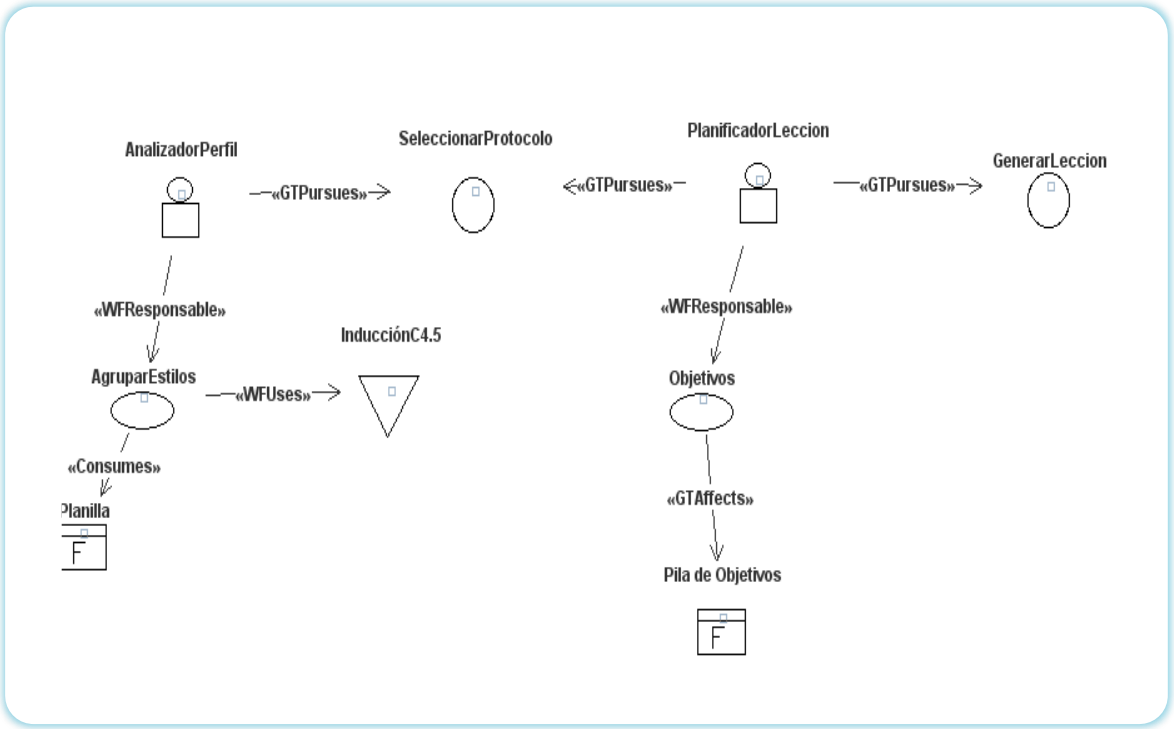
MODELO DE ORGANIZACIÓN



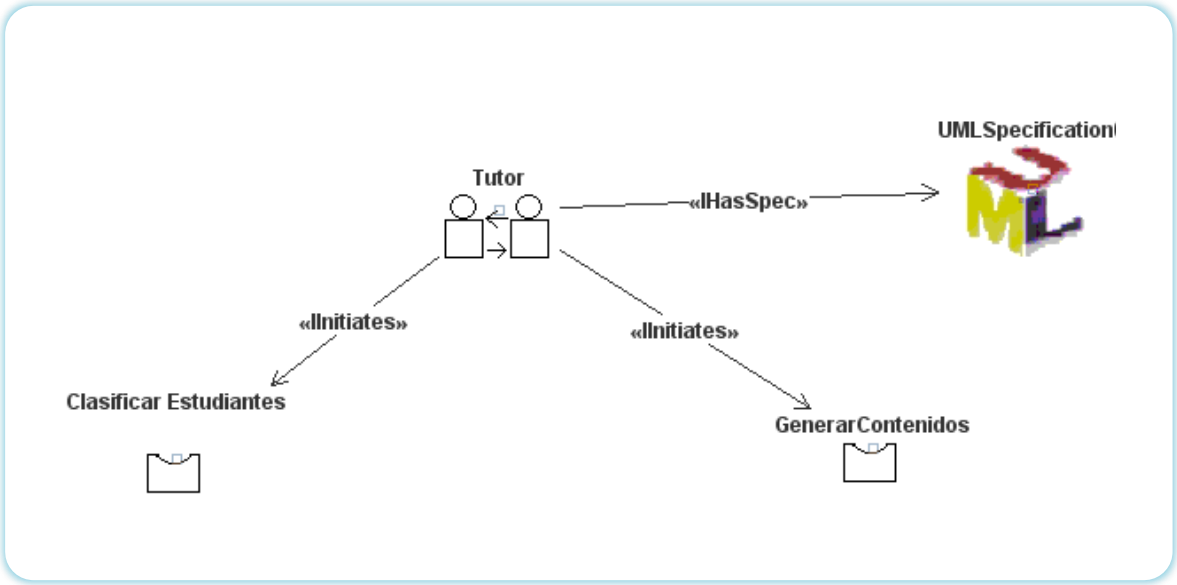
MODELO DE AGENTE



MODELO DE TAREAS



MODELO DE INTERACCIÓN



Cobija, 29 de marzo del 2010

Señor:
Lic. Javier Patty Magne
DOCENTE DE LA ASIGNATURA DE TALLER DE LICENCIATURA II
Presente.-

REF: CONFORMIDAD Y AVAL PARA LA PRESENTACIÓN DEL INFORME FINAL DE LA TESIS DE GRADO NIVEL LICENCIATURA DEL POSTULANTE UNIV. FRANCISCO LERA ARCE

De mi mayor consideración:

En calidad de Asesor realizando el seguimiento continuo del desarrollo de la Tesis de Grado del postulante **Univ. Francisco Lera Arce**, es que mediante la presente expreso ante usted, que el contenido de forma y fondo del Informe Final de la Tesis de Grado presentada, a merita el aval para que el postulante efectúe la presentación de su trabajo de licenciatura a objeto de optar al título de **Ingeniero en Sistemas Informáticos.**

Es cuanto informo para los fines consiguientes.

Atentamente.

Lic. Eduardo Alberto Zubieta Copeticon
ASESOR

Cobija, 29 de marzo del 2010

Señor:
Ing. José Balderrama Méndez
COORDINADOR DE PROGRAMA DE ING. INFORMÁTICA
Presente.-

REF: CONFORMIDAD Y AVAL PARA LA PRESENTACIÓN DEL INFORME FINAL DE LA TESIS DE GRADO NIVEL LICENCIATURA DEL POSTULANTE UNIV. FRANCISCO LERA ARCE

De mi mayor consideración:

En calidad de Tutor Colectivo de la Asignatura de Taller de Licenciatura II, se ha realizado el seguimiento continuo del desarrollo de la de la Tesis de Grado del postulante **Univ. Francisco Lera Arce** y habiéndose cumplido con todas los requisitos exigidos en el reglamento, es que mediante la presente expreso ante su autoridad que el contenido de forma y fondo del Informe Final de la Tesis de Grado presentada, a merita el aval para que el postulante efectúe la presentación de su trabajo de licenciatura a objeto de optar al título de **Ingeniero en Sistemas Informáticos.**

Es cuanto informo para los fines consiguientes.

Atentamente.

Lic. Javier Patty Magne
DOCENTE DE LA ASIGNATURA DE TALLER DE LICENCIATURA II