LINEAMIENTOS PARA EL DISEÑO DE SISTEMAS TUTORES INTELIGENTES PARA APRENDIZAJE EN EL DOMINIO DE SALUD

Tesis de Maestría

Ing. Carolina González Serrano

Director
Dr. Ing. Juan Carlos Vidal Rojas

Asesor
Dr. Ing. Juan Carlos Burguillo Rial

Universidad del Cauca
Maestría en Ingeniería, Área Telemática
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telemática
Popayán, Octubre de 2005

INDICE

INDICE DE TABLAS

INDICE DE ECUACIONES

INDICE DE FIGURAS

PREFACIO

Los Sistemas Tutores Inteligentes constituyen un grupo de aplicaciones de enseñanza que promueven un aprendizaje individual y flexible basado en el conocimiento y comportamiento del usuario. Hasta ahora estos sistemas han demostrado su efectividad en diversos dominios. Sin embargo su construcción implica un complejo e intenso trabajo de ingeniería del conocimiento, lo que impide un uso más general y aprovechado.

La característica clave de un sistema tutor inteligente es la capacidad de adaptarse al alumno, en este sentido la componente clave de dicho sistema es el denominado modelo del alumno, donde se almacena la información relativa al estudiante. Dicha información se genera a partir del comportamiento que el alumno muestra durante la interacción con el sistema, es decir, debe ser inferida por el propio sistema a partir de la información que tenga disponible: datos previos sobre el alumno, respuestas a preguntas que se le vayan planteando, patrón de comportamiento durante el proceso de aprendizaje, etc. El proceso que consiste en inferir a partir de los datos observables el estado cognitivo del alumno se denomina diagnóstico, y es sin duda el proceso más complicado dentro de un sistema tutor inteligente, dado que además de la dificultad que supone conlleva tratamiento de información que en muchos casos es incierta o imprecisa.

El objetivo del presente trabajo ha sido la definición de un marco de referencia que permita el desarrollo de Sistemas de Tutoría Inteligentes que puedan ser utilizados en diferentes dominios. Para ello se profundizó en el estudio de diversas técnicas de Inteligencia Artificial que tuviesen un fundamento teórico consistente, con el objeto de mejorar la precisión del modelo del alumno, pero poniendo especial énfasis en simplificar su uso de modo que no suponga una carga excesiva de trabajo adicional a la tarea ya de por si considerable de desarrollar un Sistema Tutor Inteligente.

Estructura de la Memoria

Esta memoria se estructura en 6 capítulos que van cubriendo el objetivo planteado. En el segundo capítulo estudiamos el origen y evolución de los Sistemas Tutores Inteligentes desde sus orígenes (enseñanza asistida por computador). Posteriormente, describimos la arquitectura básica y componentes de un Sistema Tutor Inteligente. Nos centramos después en lo que para nosotros es la cuestión clave para dotar a estos sistemas de inteligencia: el modelado del alumno.

En el tercer capítulo discutimos la aplicación de diversas técnicas de razonamiento aproximado que se han desarrollado en Inteligencia Artificial para el modelado del alumno. En primer lugar revisamos brevemente los enfoques más significativos de factores de certeza, la teoría de Dempster-Shafer, la lógica difusa y el modelo probabilístico basado en redes bayesianas.

En el cuarto capítulo se hace una revisión de la teoría de sistemas de múltiples agentes. Se estudia en detalle la aplicación de agentes inteligentes a sistemas de enseñanza, y se establece una clasificación entre los sistemas encontrados.

En el quinto capitulo describimos la arquitectura del sistema y sus componentes. Se presenta una descripción detallada de la arquitectura propuesta la cual toma como referencia las ideas en las que se fundamentan los Sistemas Tutores Inteligentes, y presenta una nueva aproximación que involucra el uso de Razonamiento Basado en

Casos para la implementación del ITS. La arquitectura propuesta está compuesta por un sistema de razonamiento basado en casos, que encapsula un sistema multi-agente. El sistema híbrido así definido constituye un mecanismo autónomo integrado por un conjunto de componentes que siguen el ciclo de vida del sistema CBR y donde cada fase del CBR es implementada usando sistemas de agentes.

Por último el capitulo seis se dedica a la descripción del prototipo de Sistema Tutor Inteligente para aprendizaje en salud utilizado para la validación de la arquitectura propuesta. El sistema tutor se basa en la creación de un modelo de usuario que representa las preferencias del mismo y los progresos que realiza en el proceso de adquisición de conocimientos. Su principal característica es la adaptabilidad al usuario.

Agradecimientos

Resulta difícil corresponder con pocas palabras al gran apoyo recibido por aquellas personas, que junto con mi trabajo, han hecho posible el discurrir de estas líneas. En primer lugar, quisiera agradecer a mi familia su cariño y el apoyo incondicional que en todo momento me han sabido hacer llegar. A mi pequeñito Diego Felipe y a mi esposo Diego Mauricio quién en todo momento ha confiado en mí y me ha dado ánimos para continuar trabajando.

En segundo lugar, y al margen de su labor como director, agradecer a Juan Carlos Vidal la oportunidad que en su momento me dio de trabajar a su lado, la confianza depositada en mí, pero sobre todo agradecerle sus consejos y tiempo. Resulta inevitable a estas alturas de párrafo no citar a Juan Carlos Burguillo, la persona de la que he aprendido tanto en tan poco tiempo. A los dos quiero expresarles en forma conjunta mi más sincera gratitud, por la enorme satisfacción que ha sido el trabajar a su lado y compartido tantos y tan buenos momentos.

En tercer lugar, quisiera agradecer la ayuda recibida de mis compañeros del Departamento de Sistemas de la Universidad del Cauca y del Departamento de Telemática e Informática de la Universidad de Vigo, que de una u otra forma han colaborado desde los comienzos de este trabajo. No pondré nombres por temor a dejar a alguien fuera, pero todos y cada uno de ellos conocen al igual que yo, las razones por las que les tengo que estar enormemente agradecida.

1. INTRODUCCIÓN

1.1. EL CONTEXTO DE SISTEMAS INTELIGENTES PARA APRENDIZAJE

Las nuevas tecnologías han aportado al campo de la educación aspectos innovadores que suponen una mejora cualitativa en las formas de enseñar y aprender. Ello obedece a una mayor preocupación tanto de las instituciones como de los investigadores a la hora de mejorar las prestaciones de docentes y alumnos en los procesos de aprendizaje.

La inteligencia artificial (*Artificial Intelligence – AI*) [1] es un campo de estudio que busca explicar y emular inteligencia, desarrollándola en términos de procesos computacionales que, si son utilizados correctamente por un programa, puede exhibir un comportamiento inteligente. El propósito de la AI no es la creación de agentes u hombres artificiales, como en la ciencia ficción. Tampoco intenta crear hombres o artefactos con sus mismas sensibilizaciones. Las propuestas sobre su desarrollo apuntan a la creación de modelos para soluciones inteligentes de problemas en dominios específicos. El propósito en realidad es la creación de sistemas inteligentes, empleando este concepto, no en toda su extensión, sino en lo referente a lo cognitivo [2].

En los últimos años la Inteligencia Artificial se ha utilizado en la búsqueda de nuevos métodos de enseñanza/aprendizaje [3]. Estos enfoques no pretenden sustituir al tutor humano, sino ayudar al alumno en el proceso de aprendizaje. En este sentido los Sistemas Inteligentes de Tutoría (ITSs) corresponden a una de las principales aplicaciones de la AI [4].

Los programas basados en técnicas de Inteligencia Artificial suelen adoptar la forma de tutoriales en los que el estudiante puede tomar la iniciativa [5] . Las diferencias más notables con respecto a los programas tutoriales convencionales, se deben a la forma en que se concibe su diseño. Un programa tutorial tradicional trata de inducir en el estudiante la respuesta correcta mediante una serie de estímulos que han sido cuidadosamente planificados. En cambio un programa tutor inteligente intenta simular alguna de las capacidades cognitivas del estudiante y utilizar los resultados de esta simulación como base de las decisiones pedagógicas a tomar [6]. El control de la iniciativa, en un tutorial convencional, corresponde totalmente a la computadora, mientras que en el inteligente hay situaciones en las que puede corresponder al estudiante.

En la actualidad, y a pesar de las grandes actividades de investigación y desarrollo en el campo de los ITSs, no hay muchos sistemas en uso. Esto debido a que la mayoría de los tutores existentes modelan el conocimiento por medio de sistemas de producción de reglas [7], los cuales poseen grandes desventajas como: (1) la dificultad en el proceso de adquisición del conocimiento, (2) el costoso desarrollo y mantenimiento, (3) el manejo de grandes volúmenes de información, (4) la poca adaptación al alumno, y (5) la complejidad en la generación de estrategias de enseñanza.

En este sentido, la combinación de técnicas de Inteligencia Artificial como sistemas multi-agente (*multi-agent systems – MAS*) [8] y razonamiento basado en casos (*case-based reasoning – CBR*) [9] en el diseño de ITSs, presenta un poderoso potencial que permitirá tratar estas deficiencias, ya que será posible la generación de entornos

efectivos de aprendizaje que posean propiedades de adaptación según cambios en el entorno.

1.1.1. Razonamiento Basado en Casos

Los sistemas expertos, o sistemas basados en el conocimiento (*Knowledge Based Systems - KBS*) constituyen una de las ramas de la Inteligencia Artificial (*Artificial Intelligence – IA*). Sin embargo, el desarrollo de estos sistemas se ha encontrado con varios problemas importantes como se menciona en [10].

Kolodner [11][12] propuso en la década de los 80 un modelo revolucionario, que pretendía atenuar los inconvenientes presentados con los KBS: un sistema de razonamiento basado en casos. Estos sistemas se caracterizaban, entre otros aspectos, por superar algunos de los problemas que afectaban a los sistemas expertos. El razonamiento basado en casos se utiliza como paradigma para la resolución de problemas, implicando un enfoque fundamentalmente diferente al de la mayoría de los sistemas de IA. En lugar de basarse exclusivamente en el conocimiento general del dominio de un problema, o establecer asociaciones a través de un conjunto de relaciones generalizadas entre descriptores de problemas y conclusiones, se utiliza el conocimiento específico de experiencias previas en situaciones concretas. Desde entonces, los sistemas CBR se han aplicado con éxito a un amplio espectro de problemas [13][14]. El aprendizaje basado en casos consiste en adquirir conocimiento a partir de experiencias precedentes o casos. También se conoce como razonamiento basado en casos por el hecho de que este tipo de aprendizaje no se concibe sin el proceso de razonamiento que conlleva la obtención de una nueva experiencia. Es un método para resolver problemas recordando situaciones similares, reutilizando la información y el conocimiento sobre dichas situaciones [13].

Los sistemas CBR están especialmente indicados cuando las reglas que definen un sistema de conocimiento son difíciles de obtener, o el número y complejidad de las mismas es demasiado grande para ser representado con un sistema experto. Esta tecnología ha sido utilizada satisfactoriamente en disciplinas como el derecho, la medicina y sistemas de diagnóstico con grandes bases de datos. Una descripción detallada de CBR y su aplicación en ITSs es revisada en detalle en el capítulo 3.

Se pueden encontrar propuestas en la literatura de un gran número de definiciones del concepto de agente, sin que ninguna de ellas haya sido plenamente aceptada por la comunidad científica. Podemos definir entonces de una forma general un agente inteligente como una entidad computacional, que actúa de forma autónoma, ejecuta sus acciones con algún nivel de pro actividad y posee algunos atributos claves, tales como: cooperación, aprendizaje y movilidad [8].

En la mayoría de las ocasiones, los agentes no son desarrollados de forma independiente sino como entidades que constituyen un sistema. A este sistema se le denomina multi-agente [15]. En este caso, los agentes deben o pueden interactuar entre ellos. Las interacciones más habituales como son: (i) informar o consultar a otros agentes permiten a los agentes “hablar” entre ellos, (ii) tener en cuenta lo que realiza cada uno, y (iii) razonar acerca del rol asumido por los diferentes agentes que integran el sistema.

Algunos dominios de aplicación de sistemas multi-agente son: sistemas de gestión de workflow, gestión de redes, control de tráfico aéreo, reingeniería de procesos de

negocio, data mining, recuperación de información, comercio electrónico, educación, correo electrónico, libreras digitales, entre otros [16].

Con la aparición de los sistemas multi-agente, se pueden obtener nuevos beneficios como son: (a) orientación a actividades conjuntas y de cooperación, (b) mecanismos cooperativos de resolución de conflictos, (c) mecanismos de negociación entre agentes, (d) establecimiento cooperativo de compromisos y planificación de actividades, (e) establecimiento de modelos de conocimiento, comunicación y adquisición que son extraídos del entorno cooperativo que caracteriza el MAS. Las ventajas del trabajo con sistemas multi-agente en ITSs son discutidas en el capitulo 4.

1.2. MOTIVACION DE LA INVESTIGACIÓN

En la actualidad, se hace cada vez más notable la incorporación de elementos tecnológicos relacionados con la informática como complemento para el proceso de enseñanza y aprendizaje tradicional. La incorporación de estos elementos tecnológicos permite incrementar la eficiencia a la hora de transmitir y adquirir conocimiento, aumentar su disponibilidad, permitir el manejo de grandes volúmenes de información y facilitar su actualización.

Entre los complementos tecnológicos más conocidos al proceso de enseñanza tradicional se encuentran las aplicaciones informáticas educativas [17]. Este tipo de aplicaciones han ido creciendo en popularidad de forma paralela al auge de las nuevas tecnologías e Internet en la sociedad. Prueba de ello es el hecho que en la propia red se pueden encontrar numerosos ejemplos de este tipo de aplicaciones.

Los tipos de aplicaciones educativas existentes son numerosos y variados, e incluyen desde simples recopilaciones de documentos indexados y enlazados mediante hipertexto (*como la mayoría de los tutoriales que se pueden encontrar en Internet*) hasta vistosas aplicaciones que se apoyan en las capacidades de la multimedia y en diversos recursos gráficos para mostrar los contenidos al estudiante. La diferencia entre estos dos tipos de aplicaciones radica principalmente en la vistosidad con la que se muestran los contenidos pero, en los dos casos, el procedimiento de enseñanza de dichos contenidos sigue un esquema pasivo y lineal muy parecido a lo que sería leer un libro, y aunque la presencia de la animación multimedia pueda sorprender inicialmente, puede llegar a cansar provocando la desmotivación y el abandono por parte del estudiante.

Son numerosos los especialistas de la pedagogía y la informática que se encuentran investigando actualmente en el desarrollo de ITS como sistemas flexibles, interactivos y adaptativos que utilizan técnicas estadísticas y de inteligencia artificial, como el razonamiento basado en casos, redes bayesianas, teoría de la probabilidad, factores de certidumbre, lógica difusa, etc. para mejorar el proceso de enseñanza aprendizaje asistido por computador [18][19].

La característica distintiva de un Sistema Tutor Inteligente es su capacidad de adaptación al alumno, adaptación que se puede realizar a varios niveles: en el nivel en que se presenta el material o las ayudas, en la dificultad de los problemas propuestos o en la selección de la estrategia instructora más adecuada según sus capacidades, habilidades y estilos de aprendizaje preferidos. La importancia de adaptar la enseñanza a cada alumno es analizada en estudios como el que aparece en Bloom [20], que avalan la instrucción individualizada como la forma más efectiva de aprendizaje. En este trabajo, Bloom concluyó que cuando se usan los métodos convencionales de enseñanza (un profesor para treinta alumnos, con exámenes periódicos) las calificaciones obtenidas por los alumnos tienen una distribución normal,

con media entre 50 y 60% pero con una desviación típica grande. Si el profesor adapta sus lecciones para intentar evitar los errores que sus alumnos cometen en los exámenes, las medias se mueven hasta un 84% y la desviación típica disminuye considerablemente. El cambio más dramático ocurre cuando los alumnos reciben enseñanza individualizada. La media llega a ser del 98%, con una desviación típica que es la mitad de la desviación típica de los alumnos que recibieron una enseñanza convencional. Estos resultados constituyen un argumento más a favor del uso de los Sistemas Tutores Inteligentes, ya que demuestran los buenos resultados de la enseñanza individualizada. Por tanto, si la característica clave de los Sistemas Tutores Inteligentes es su capacidad de adaptarse a cada alumno que utiliza el sistema, el problema de obtener toda la información posible acerca del alumno se convierte en el problema principal a la hora de diseñar un tutor inteligente. En efecto, es necesario que en cada momento el ITS disponga de una representación del estado actual del conocimiento del alumno, con objeto de poder seleccionar el material al nivel adecuado de detalle, proponer el problema apropiado o seleccionar la estrategia tutorial más efectiva en ese momento. El modelo del alumno es la componente del ITS que representa el estado actual del conocimiento del alumno, y el proceso que manipula esta estructura se llama diagnóstico. Ambas componentes deben diseñarse juntas, y este problema de diseño es el que se conoce como el problema del modelado del alumno.

2. SISTEMAS INTELIGENTES EN EL ÁMBITO DE LA EDUCACIÓN

2.1. INTRODUCCIÓN

La idea de aprovechar herramientas informáticas en la enseñanza se remonta a los años 50. Pero no será hasta los 80 cuando la enseñanza asistida por computador recobre un especial interés gracias a las técnicas de la ***Inteligencia Artificial***. En aquella época surgen los denominados ***Sistemas Tutores Inteligentes*** con la vocación clara de desarrollar procesos de enseñanza adaptados a los diferentes usuarios/estudiantes.

Con la situación actual del mercado de las tecnologías de la información, el abaratamiento de los computadores, el gran impulso de las comunicaciones e Internet, el desarrollo de sistemas multimedia y la cada vez mayor aceptación de herramientas informáticas por parte de la sociedad, estamos quizás ante una situación inmejorable para abordar la demanda formativa y educativa, ofreciendo ***Ambientes de Aprendizaje Inteligentes*** *(Intelligent Learning Environments- ILE)* como herramientas de apoyo a la enseñanza/aprendizaje.

Con los ambientes de aprendizaje inteligentes se pretende ayudar, colaborar y/o favorecer los procesos de aprendizaje como parte integrada en los modelos de enseñanza más actualizados. Es decir, su creación se enfoca más como una herramienta complementaria de la enseñanza/aprendizaje que permite aumentar la calidad de la instrucción, que como una herramienta que sustituye en sí todo un sistema clásico de enseñanza/aprendizaje. La utilización de ILEs implica por parte del docente una nueva y más amplia visión de sus actividades de enseñanza.

En este capítulo estudiaremos la evolución de los Sistemas Tutores Inteligentes desde sus orígenes (*Computer Assisted Instruction - CAI*) hasta hoy. Posteriormente, describiremos la arquitectura básica y componentes de un ITS. Nos centraremos después en lo que para nosotros es el aspecto clave para dotar a estos sistemas de inteligencia: el modelado del alumno (*Student Model -SM*).

2.2. PERSPECTIVA HISTÓRICA: DE LA ENSEÑANZA ASISTIDA POR COMPUTADOR A LOS SISTEMAS TUTORES INTELIGENTES

En los años 50 aparecieron los primeros sistemas de enseñanza, los llamados programas lineales. Estos programas se caracterizaban por mostrar el conocimiento de una manera lineal. Es decir, ningún factor podía cambiar el orden de enseñanza establecido en su momento por el programador. Esta actuación de los sistemas tenía su origen en la teoría conductista, defendida en su momento por Skinner [21]. Dicha teoría propugnaba que las personas funcionan por estímulos y que a igual estímulo corresponde igual respuesta. Según esto, no se debía permitir cometer errores a los alumnos, ya que éstos les darían un refuerzo negativo. Por lo tanto, en el desarrollo de una sesión de enseñanza no se tiene en cuenta para nada la aptitud del alumno.

Los sistemas CAI (*Computer Assisted Instruction – CAI*) [22] tenían principalmente dos usos distintos: a) como libro de texto electrónico, en el que el alumno podía leer el material relativo a la asignatura que intentaba aprender, y b) como lugar donde practicar los conocimientos aprendidos, resolviendo una serie de problemas propuestos y recibiendo cierto tipo de ayudas durante este proceso de resolución.

Un programa de CAI tiene integradas rutinas para reaccionar ante las respuestas del alumno, corrigiéndole y dándole ayuda si son incorrectas, o avanzando en el currículum si son correctas. En general, el comportamiento de un sistema CAI se ajusta a la siguiente descripción: el sistema presenta cierto material que debe ser aprendido y seguidamente propone un problema al alumno que representa cierta parte del currículum. El alumno responde, y su respuesta es evaluada comparándola con la respuesta correcta. El computador informa al alumno de si la respuesta era correcta o no, y elige el nuevo material para ser presentado o el nuevo problema a proponer. Si la respuesta es incorrecta, el computador puede presentar de nuevo la materia, proponer un problema más sencillo, o adoptar cualquier otra estrategia para ayudar al alumno. La selección de dicha estrategia conlleva normalmente un intento de identificar el origen del error, es decir, de encontrar qué es lo que el alumno no ha entendido o aprendido correctamente, y tratar este error específicamente. El principal problema de los sistemas tradicionales de CAI era la poca flexibilidad que ofrecían. En efecto, tanto la presentación del material como la elección del problema adecuado, la corrección del mismo o la selección de la estrategia instructora eran procedimientos predefinidos en el sistema, y por tanto iguales para todos los alumnos que lo utilizaran. Debido a esta limitación, muchos educadores los consideraban como sofisticados libros de texto electrónicos [23], y empezaron a surgir los primeros intentos de hacer que estos sistemas tuvieran cierta capacidad de adaptación a los alumnos, surgiendo así los primeros Sistemas Inteligentes de Enseñanza Asistida por Computador (ICAI), que Wenger [23] no considera diferentes de los CAI, sino mejoras o refinamientos de los mismos ya que tecnológicamente se basan en los mismos modelos.

Los sucesores de los programas lineales en el campo de la enseñanza asistida por computador, fueron los programas ramificados [24]. Estos tenían un número fijo de temas, al igual que los programas lineales, sin embargo se diferenciaban por la capacidad de actuar según la respuesta del alumno. La mejora ofrecida por estos sistemas se consiguió gracias a la técnica de Pattern-matching y al diseño de lenguajes de autor. En cuanto a la técnica de Pattern-matching, ésta permitía tratar las respuestas de los alumnos como aceptables o parcialmente aceptables, en lugar de totalmente correctas o incorrectas como exige la propuesta de Skinner. Por su parte, el material de enseñanza obtenido en los programas lineales era en general demasiado grande e intratable por medios clásicos. Por ello se desarrollaron los "lenguajes de autor" para permitir crear material de enseñanza de forma tratable por el sistema [25].

A partir de los años 70 y en paralelo al desarrollo de los sistemas CAI comenzaron a aparecer los primeros ITS, que usaban formalismos y técnicas propias de la Inteligencia Artificial para definir el conocimiento que se quería transmitir. Una primera especificación de los requisitos que debe cumplir un ITS es la realizada por Hartley y Sleeman [26]. Para ellos, un ITS debe tener: a) conocimiento del dominio (modelo experto), b) conocimiento del alumno (modelo del alumno), y c) conocimiento de estrategias instructoras (tutor). Probablemente, esta especificación básica es el resultado de analizar qué tipos de conocimiento deben considerarse en un ITS: qué enseñar, cómo enseñarlo y qué es lo que el alumno sabe. Normalmente, cada una de estas áreas de conocimiento se almacena y mantiene en módulos diferentes, dotando al ITS de una deseable modularidad que permite, al menos teóricamente, que partes del mismo puedan ser utilizadas en otro ITS o el desarrollo de sistemas genéricos de autor que asistan en la implementación de ITSs. Vemos por tanto que la definición básica, que curiosamente no se ha visto alterada en casi treinta años de investigación sobre ITS, influye también directamente en la arquitectura de los mismos.

Así, aunque para muchos investigadores los términos ICAI y ITS son equivalentes, para nosotros esta separación entre el conocimiento acerca del dominio, acerca del alumno y acerca de estrategias pedagógicas marca la diferencia entre ambos. En

efecto, el desarrollo de los ITS supone un cambio radical en la concepción de los sistemas de ayuda a la enseñanza: de programar decisiones a programar conocimiento. La figura 1 muestra la evolución de los sistemas de enseñanza.

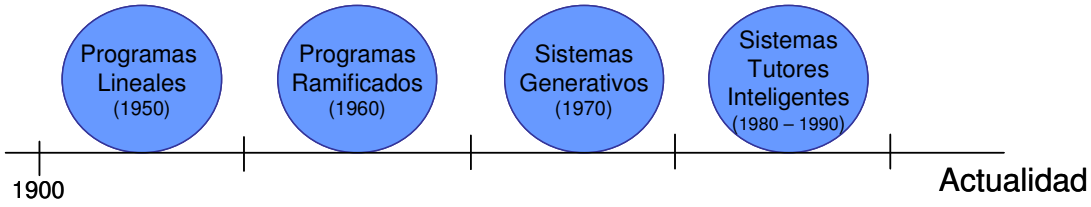


Figura 1. Evolución de los sistemas de enseñanza tradicionales

Aunque no existe una definición precisa de lo que es un ITS, para el trabajo de esta tesis hemos considerado las definiciones propuestas por [27][28].

"Un Sistema Tutor Inteligente, es un sistema software que utiliza técnicas de Inteligencia Artificial (IA) para representar el conocimiento e interactúa con los estudiantes para enseñárselo. Un ITS consiste básicamente de tres modelos que se comunican e interactúan entre sí. Estos modelos representan el conocimiento del dominio, el estudiante, y el tutor. "

2.3. ARQUITECTURA DE LOS SISTEMAS TUTORES INTELIGENTES

Un ITS es capaz de guiar al alumno a lo largo de un dominio en particular del conocimiento, resolviendo durante el proceso tareas tales como la elaboración de una estrategia de tutorización, la generación de ejercicios a la medida de las necesidades del alumno, la resolución pedagógica de estos ejercicios, así como la explicación de la solución. Estas tareas se organizan en distintos módulos, siendo los componentes claves del ITS tradicional: ***un modelo del alumno, un modelo pedagógico, un modelo didáctico y una interfaz con la que interactúa el usuario***.

Dependiendo de la arquitectura del sistema, estos módulos se pueden encontrar organizados en diferentes formas. Pueden estar distribuidos y subdivididos en partes más pequeñas, funcionando como entidades, semi o completamente autónomas, que se comunican entre sí y actúan racionalmente de acuerdo a sus percepciones del exterior y el estado de su conocimiento. En la figura 2 se presenta la arquitectura básica de un ITS tradicional.

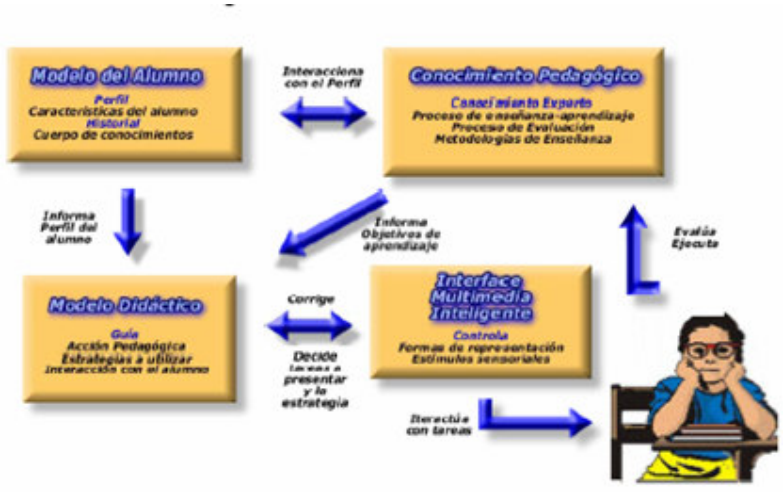


Figura 2. Arquitectura de Sistema Tutor Inteligente

2.3.1. Componentes fundamentales de un ITS

La arquitectura básica de un ITS consiste en un módulo experto, un módulo del alumno y un módulo tutor, que operan de forma interactiva y se comunican a través de un módulo central que se suele denominar módulo entorno. El módulo experto contiene el conocimiento acerca de la materia que se pretende enseñar, el módulo del alumno guarda toda la información relativa al mismo que se genera durante la interacción con el sistema, y el módulo tutor controla los planes y decisiones pedagógicas. Finalmente, el módulo entorno gestiona la interacción de las otras componentes del sistema y controla el interfaz hombre-máquina. La figura 2 representa la arquitectura básica de los ITSs, siendo los modelos del dominio, pedagógico y del estudiante, los específicos de un sistema de enseñanza inteligente.

2.3.1.1. Módulo Experto

El módulo experto de un ITS proporciona los conocimientos del dominio, que satisfacen dos propósitos diferentes. En primer lugar, presentar la materia de la forma adecuada para que el alumno adquiera las habilidades y conceptos. Esto incluye la capacidad de generar preguntas, explicaciones, respuestas y tareas para el alumno. En segundo lugar, el módulo experto debe ser capaz de resolver los problemas generados y de corregir las soluciones presentadas, incluso de aceptar aquellas soluciones válidas que han sido obtenidas por medios distintos. Además, debe poder explicar sus razonamientos en un lenguaje comprensible para el alumno.

Al diseñar el módulo experto, es importante considerar qué tipo de conocimiento se está modelando. Fundamentalmente, podemos dividir el conocimiento en tres tipos: declarativo, de procedimientos y cualitativo. El conocimiento de procedimientos (*procedural knowledge*) es conocimiento acerca de cómo llevar a cabo cierta tarea, y por tanto suele ser específico de cada dominio en particular. Un ejemplo de representación de dicho conocimiento (el más extendido) es una base de conocimientos junto con un conjunto de reglas, al estilo de los sistemas expertos basados en reglas, que es el enfoque utilizado en varios sistemas; como por ejemplo en el Tutor de Geometría de Anderson [29] y en el sistema BUGGY [30]. El

conocimiento declarativo (*declarative knowledge*) es un conjunto de hechos que se organizan de forma adecuada para razonar sobre ellos. Un ejemplo de dominio de conocimiento declarativo es la Geografía, y una forma común de representarlo es una red semántica, como en el sistema SCHOLAR [31], en la que los nodos representan hechos y los enlaces representan relaciones jerárquicas. Esta estructura permite definir procedimientos de inferencia flexibles que operan sobre la base de conocimientos. El tercer tipo, el conocimiento cualitativo, es quizás el más difícil de modelar. Se usa para modelar relaciones espaciales y procesos dinámicos. El razonamiento causal es una parte del conocimiento cualitativo que resulta de gran importancia en los sistemas de diagnóstico de averías. El razonamiento sobre la estructura causal de un dispositivo se usa para determinar potenciales problemas. Las Redes Bayesianas [32] parecen la estructura más adecuada para modelar este tipo de conocimiento (puesto que en una Red Bayesiana los enlaces que conectan los nodos modelan relaciones de tipo causal) y son la alternativa elegida en HYDRIVE[33]. Anderson [34] agrupa los modelos expertos en tres categorías: los modelos de caja negra, los modelos de caja de cristal y los modelos cognitivos.

Los modelos expertos de caja negra son capaces de resolver problemas sobre el dominio. Las soluciones a dichos problemas [35] se usan como ejemplo para los alumnos y para determinar si las soluciones presentadas por éstos son o no correctas. Sin embargo, los cálculos internos que se realizan o bien no están disponibles o bien están expresados en términos que el alumno no puede comprender. Un ejemplo típico es un programa que juega a las damas buscando entre las millones de jugadas que se crean a partir de los movimientos posibles. El objetivo del sistema no puede ser enseñar al alumno esta estrategia de resolución de problemas, puesto que no es así como juegan los humanos. Sin embargo, las soluciones así generadas aún resultan útiles para el proceso de enseñanza. El modelo de caja negra es, como su nombre indica, totalmente opaco para el alumno. Un modelo más transparente al usuario es el llamado modelo de caja de cristal. En este modelo, cada paso en el razonamiento puede ser revisado e interpretado. Para construir un modelo de caja de cristal, se debe utilizar la misma metodología que la usada en un sistema experto. El experto humano en el dominio y el ingeniero de conocimiento trabajan juntos para definir el espacio, identificar y formalizar los conceptos claves, diseñar un sistema en el que implementar el conocimiento y probar y refinar este sistema. De este modo, el módulo experto que se obtiene parece más adecuado para enseñar al alumno, puesto que una componente de este módulo es una representación de la forma en que un humano razona para resolver el problema.

Un ejemplo clásico de los modelos mencionados anteriormente, es el sistema GUIDON [36] que reutiliza el módulo experto del sistema MYCIN [37] para enseñar conocimientos relativos a enfermedades infecciosas. La lección más importante que podemos sacar del desarrollo de este sistema es que al construir el módulo experto no sólo debemos pensar en el conocimiento de la materia, sino también en la forma en la que lo vamos a representar. Las exhaustivas búsquedas hacia atrás que hacía MYCIN para determinar la enfermedad a partir de los síntomas no son representativas del modo de razonamiento humano, y muchas de las reglas MYCIN eran demasiado complejas para ser enseñadas a un alumno.

Por último, tenemos los llamados modelos cognitivos [35], que simulan cómo usa un humano el conocimiento que queremos enseñar. El objetivo de este modelo es descomponer el conocimiento en componentes plenas de significado, y usar este conocimiento de una forma similar a la humana. Este tipo de modelo experto puede comunicarse con un alumno de una forma mucho más extensa que los modelos anteriores. Sin embargo, la construcción de modelos cognitivos es un proceso muy complicado y que consume mucho tiempo, y se plantea la necesidad de determinar

qué componentes psicológicas son esenciales para modelar el aprendizaje y cuáles pueden ser sacrificadas a cambio de una menor complejidad computacional. Dentro de las características más importantes que se han tenido en cuenta en el desarrollo de los sistemas tutores médicos presentados anteriormente, podemos mencionar: (1) la necesidad de representar conocimiento implícito, (2) la importancia de soportar el proceso de abstracción-refinamiento de clasificación heurística, y (3) los desafíos de crear representaciones del conocimiento de suficiente tamaño, complejidad y validez para soportar el aprendizaje por parte del estudiante.

2.3.1.2. Módulo del Alumno

El módulo del alumno en un ITS representa el conocimiento que tiene el alumno del dominio que intentamos enseñarle. Las acciones del alumno son interpretadas en un intento de reconstruir el estado de conocimiento que le llevó a realizar esas acciones. Esta información puede entonces compararse con la contenida en el módulo experto, y ser usada para tomar decisiones didácticas que guiarán al alumno y organizarán sus actividades. El modelo del alumno es una representación cualitativa aproximada, posiblemente parcial, del conocimiento del alumno sobre cierto dominio, o tema/habilidad de dicho dominio que puede explicar total o parcialmente aspectos específicos del comportamiento del alumno. Decir que el modelo del alumno es una representación cualitativa significa que no es ni numérica ni física, sino que describe los objetos y los procesos en términos de relaciones espaciales, temporales y causales. Decir que el modelo del alumno es aproximado y posiblemente parcial significa que nos interesa más la utilidad computacional que la fidelidad cognitiva [38]. Un modelo del alumno más preciso sólo es mejor si el esfuerzo computacional realizado para aumentar su precisión o completitud no es excesivo comparado con la ganancia pedagógica obtenida.

El modelado del alumno es el elemento central en el diseño y desarrollo de un ITS. Un buen punto de partida para entender la importancia del mismo puede ser, precisar el significado del término inteligente en el contexto de los Sistemas Tutores.

Revisando la literatura, encontramos que en VanhLehn afirma en [39] que no existía una definición aceptada de este término en este contexto. Entre los años 1993 y 1995 Shute [40] realizó un estudio para intentar buscar un consenso en este aspecto entre los investigadores más reconocidos en el campo. Para ello les pidió que resumieran en dos o tres líneas sus ideas acerca de qué significaba la "**I**" en el acrónimo ITS. Las respuestas fueron variadas e interesantes, y la conclusión casi unánime fue, en palabras de Shute:

"Como vemos en esta muestra no aleatoria de respuestas sobre lo que es la inteligencia en un ITS, casi todo el mundo coincide en que el elemento más crítico es el diagnóstico cognitivo (o modelado del alumno). La siguiente característica más citada es la adaptación en la asistencia. Y aunque algunos sostienen que la asistencia forma parte de la "T" de ITS, nuestra postura es que las dos componentes, (diagnóstico y asistencia), trabajando conjuntamente, constituyen la inteligencia de un ITS".

Las cuestiones fundamentales en el modelado del alumno son:

- ***Selección de la estructura*** que se usará para representar el modelo del alumno. Esta información puede almacenarse de muchas formas distintas: en un vector, en una red semántica, en una red bayesiana, en forma de afirmaciones, etc.

- ***Inicialización del modelo del alumno***. La estructura elegida para representar el conocimiento del alumno debe inicializarse cuando la interacción con el sistema comienza. Para ello disponemos de varias opciones: utilizar la información disponible acerca del alumno, pedirle que se clasifique en clases o estereotipos de alumnos previamente definidos, realizar tests previos, etc.

- ***Diagnóstico.*** Una vez que el modelo del alumno se ha inicializado comienza la interacción con el sistema. El procedimiento de diagnóstico elegido actualizará el modelo del alumno tras sus interacciones con el sistema, utilizando dos fuentes de información principalmente: a) el modelo del alumno actual y b) su comportamiento en el proceso interactivo de enseñanza, comportamiento que puede medirse en función de distintas variables que es preciso definir previamente (soluciones a problemas, respuestas a preguntas, tiempo empleado en lectura de pantallas, etc.).

El modelo del alumno puede ser utilizado con diferentes propósitos:

a) Para determinar si el alumno está preparado para continuar con el siguiente tema del currículum, y para elegir este tema.

b) Para generar explicaciones que el alumno pueda entender (al nivel de detalle adecuado a sus conocimientos actuales).

c) Para ofrecer consejos y ayudas sin que el estudiante lo solicite. En este sentido, es importante que el tutor no interrumpa a los alumnos con demasiada frecuencia, y que les permita aprender de sus errores.

d) Para generar problemas al nivel adecuado. La generación dinámica de problemas es otra área que se apoya fuertemente en el modelo del alumno.

Una vez identificados los puntos débiles del alumno, se genera un problema que el módulo experto resuelve paralelamente para ser capaz de diagnosticar la solución del alumno. Así, a cada alumno que interactúe con el sistema se le presentará una colección diferente de problemas, adecuada a su nivel de conocimiento.

e) Para seleccionar la estrategia tutorial más apropiada dado el nivel de conocimiento actual.

Básicamente, los tipos de modelo del alumno que se han utilizado son:

• ***Modelo de superposición*** (*overlay model*). En este enfoque se considera que el conocimiento del alumno es un subconjunto propio del conocimiento del experto. Este enfoque supone que todas las diferencias entre el comportamiento del alumno y el del experto se explican como una falta de conocimiento del alumno. El modelo funciona bien cuando el principal objetivo del sistema instructor es transmitir el conocimiento experto al alumno. El mayor problema de dichos modelos es que no consideran que el alumno pueda poseer conocimiento que el experto no posee, y por tanto son incapaces de reaccionar ante esta situación. Esta carencia motivó la aparición de otros modelos [42].

• ***Modelo diferencial*** *(differential model)*. Es una modificación del modelo de superposición. Este modelo divide el conocimiento del alumno en dos categorías: conocimiento que el alumno debería poseer y conocimiento que no puede esperarse que el alumno tenga. Así, a diferencia del modelo de superposición, el modelo diferencial reconoce y trata de representar explícitamente tanto el conocimiento del

alumno como las diferencias alumno/experto. Puede considerarse como un modelo de superposición, pero en lugar de sobre el conocimiento del experto, sobre un subconjunto de éste [42].

• ***Modelo de perturbación*** *(perturbation model).* Mientras que el modelo de superposición representa el conocimiento del alumno en términos del conocimiento "correcto", el modelo de perturbación lo combina con una representación del conocimiento incorrecto. De este modo, no se considera al alumno como un "subconjunto" del experto, sino que el conocimiento del alumno puede ser potencialmente diferente en calidad y cantidad al del experto. La técnica más frecuente para implementar un modelo de perturbación es representar el conocimiento experto y añadirle los errores que más frecuentemente cometen los alumnos. El modelo del alumno es entonces un modelo de superposición sobre este conjunto de conocimiento aumentado (que incluye conocimientos correctos e incorrectos). En la literatura aparecen dos tipos de errores: errores de concepto (*misconceptions*) y fallos o erratas (*bugs*). La colección de errores que se incluye en un modelo de perturbación se llama *biblioteca o catálogo de errores*. Esta biblioteca puede construirse de dos formas diferentes: mediante un análisis empírico (enumeración) o generando los errores a partir de un conjunto de errores de concepto subyacentes (técnicas generativas). Aunque la información adicional en un modelo de perturbación proporciona nuevas explicaciones del comportamiento del alumno, introduce también nuevos problemas: el esfuerzo necesario para construir y mantener el modelo del alumno es mucho mayor [42].

• ***Modelo basado en restricciones****.* Este modelo es una modificación del modelo de superposición propuesto por [43] e implementado con éxito en el tutor de SQL de Mitrovic [44]. El dominio de conocimiento se representa mediante una serie de restricciones sobre el estado de los problemas, y el modelo del alumno es simplemente una lista de las restricciones que ha violado en el proceso de resolución del problema. La principal ventaja de este enfoque es su robustez y flexibilidad.

Este modelo es robusto ya que no depende de la estrategia que haya seguido el alumno para resolver el problema, y por tanto puede modelar a alumnos que tengan patrones de comportamiento inconsistentes, es decir, que utilicen estrategias diferentes para problemas diferentes. Además, el modelo es suficientemente flexible para reconocer soluciones innovadoras como correctas. Por último, debemos distinguir entre dos tipos diferentes de modelado del alumno, que Anderson, Corbett, Koedinger y Pelletier denominan traza del conocimiento y traza del modelo (*knowledge tracing y model tracing*) [45]. La traza del conocimiento consiste en determinar qué sabe el alumno, incluyendo tanto el conocimiento correcto sobre el dominio como sus errores. La traza del modelo pretende analizar el procedimiento de resolución de problemas que utiliza el alumno. La traza del modelo resulta útil en sistemas que intentan dar respuesta a peticiones de ayuda del alumno y ofrecerle pistas e información cuando no sabe seguir resolviendo el problema. De hecho, para poder ayudar al alumno el sistema necesita ser capaz de analizar y criticar la solución en curso y tener una idea de que línea de razonamiento está siguiendo. Por otro lado, la traza del conocimiento resulta útil para la evaluación del alumno y la toma de decisiones pedagógicas, como qué material/problema debe ser propuesto a continuación.

En [46] se trata un aspecto de gran interés que puede ser considerado durante el proceso de modelado del alumno el cual se refiere a las características de los equipos de los estudiantes.

2.3.1.3. Módulo del Tutor

Representaciones explícitas del conocimiento pedagógico permiten a los sistemas tutores adaptar y mejorar sus estrategias en el tiempo. Un sistema tutor debe ser capaz de a) controlar el currículo (selección de material y orden de presentación), b) responder a las preguntas de los alumnos y c) saber cuándo un alumno necesita ayuda y determinar qué tipo de ayuda necesita. Para ello se definen las estrategias instructoras, que a nivel global afectan a la ordenación en la presentación de contenidos y a nivel local a las decisiones sobre cuándo y cómo intervenir para proporcionar ayuda, explicaciones, enseñanza, preguntas o correcciones.

El módulo tutor controla las actividades e interacción instructora. Diferentes niveles de control determinan diferentes estrategias: en el nivel de control máximo (*monitoring*) el sistema adapta las acciones a las necesidades del alumno, llevando siempre el control. En el nivel intermedio (*mixed-initiative dialogue*) el alumno y el sistema comparten el control mediante el intercambio de preguntas y respuestas. En el otro extremo tenemos las actividades autorizadas (*guided-discovery learning*), en las que la intervención del sistema se reduce a modificar el entorno. Al elegir entre estas estrategias de control cada dominio y cada alumno deben ser evaluados de forma independiente. De este modo, las estrategias de control pueden guardarse en el módulo de instrucción y ser seleccionadas de forma que también el tipo de instrucción que recibe el alumno sea individualizada, como por ejemplo ocurre en el sistema ALLEN [47]. Los dos tutores más frecuentes son los tutores *expositivos* y los tutores de *procedimientos*. Cada uno de ellos se asocia con el tipo de dominio que se pretende enseñar: en los tutores expositivos se transmite conocimiento sobre hechos, y las inferencias que se realizan van encaminadas a mantener el enfoque y la coherencia. Los tutores de procedimientos enseñan habilidades y procedimientos, y tienen la tarea adicional de ordenar las habilidades que componen la habilidad principal que se desea que el alumno adquiera y de elegir los ejercicios y ejemplos.

2.3.1.4. Módulo del Entorno

El módulo entorno específica y da soporte a las actividades del alumno y a los métodos que se usan para realizar dichas actividades. Los entornos deben ser fáciles de utilizar y atractivos, de forma que el alumno pierda el mínimo tiempo posible en aprender a utilizar el entorno y pueda centrar toda su atención en el proceso de aprendizaje de la materia.

Burton define en [48] los seis aspectos clave en el diseño del entorno:

a) *aspectos del dominio* que se desean representar;
b) *nivel de abstracción* de la representación;
c) *fidelidad* de la representación;
d) *orden en la presentación* de contenidos;
e) *herramientas* de corrección y ayuda y
f) *nivel de control* que ejercerá la herramienta.

Es curioso ver que, aún cuando actualmente se cuenta con herramientas mucho más potentes para el diseño de interfaz, estas seis características pueden seguir considerándose básicas a la hora de planificar el desarrollo del interfaz. Sin duda alguna, el desarrollo de Internet ha influenciado enormemente al diseño de entornos e interfaz en los Sistemas Tutores Inteligentes. En efecto, las posibilidades se han visto multiplicadas con la aparición de las capacidades hipermedia y multimedia [49]. Hay ya

muchos Sistemas Tutores Inteligentes accesibles en Internet que aprovechan las ventajas que la red de redes puede ofrecer, como por ejemplo el tutor de LISP ELM-ART [50] y el tutor de Programación Lineal ILESA [51].

Así, los ITS se caracterizan por representar separadamente la materia que se enseña (modelo del dominio) y las estrategias para enseñarla (modelo pedagógico). Por otro lado, caracterizan al alumno (a través de un modelo del estudiante) para procurar una enseñanza individualizada. Además, de una manera cada vez más necesaria y al igual que cualquier software que se comunica con usuarios, el interfaz de comunicación corresponde con un módulo bien planificado, de fácil manipulación, y que favorece el proceso de comunicación tutor-alumno.

2.4. EL MODELADO DEL ALUMNO

El proceso de construir y mantener un modelo del alumno se basa en inferir a partir de sus interacciones con el sistema (respuestas a las preguntas planteadas, pantallas visitadas, etc.) cuál es su estado de conocimiento. Aparte de lo complicado que puede resultar realizar este tipo de inferencias, hay varias fuentes de incertidumbre que pueden dificultarlo aún más. En efecto, la información que pueda proporcionar el comportamiento del alumno es incierta, dada la gran cantidad de factores que pueden influir en él. Una respuesta incorrecta puede deberse a muchas causas diferentes, como errores de concepto, falta de conocimiento, deficiencias en la adquisición de habilidades, pero también a errores en los cálculos o incluso a un fallo al elegir la respuesta correcta. De la misma forma, una respuesta correcta puede demostrar que el alumno ha alcanzado cierto nivel de conocimiento, pero también puede deberse a haber acertado por casualidad, como puede ocurrir sobre todo cuando se plantean preguntas tipo test. Además, si el objetivo del sistema es la enseñanza no basta sólo con poder clasificar una respuesta como correcta o incorrecta sino que también es importante saber por qué esa pregunta fue respondida correcta o incorrectamente, ya que de otro modo será imposible seleccionar la estrategia instructora más adecuada para la situación actual del alumno. En Inteligencia Artificial se han desarrollado varias teorías para razonamiento aproximado. En esta sección revisaremos brevemente los enfoques más significativos para modelado del alumno.

2.4.1. Técnicas de razonamiento aproximado

En este apartado presentaremos de una forma muy breve las diferentes técnicas de razonamiento aproximado que se han aplicado al problema de modelado del alumno. Con esta presentación no se pretende hacer una descripción exhaustiva ni un análisis detallado de dichas técnicas, sino más bien presentar de forma introductoria los aspectos básicos de cada teoría para después poder analizar las distintas aplicaciones que se han hecho al modelado del alumno.

2.4.1.1. Sistemas basados en reglas y factores de certeza

Quizás la primera teoría que se aplicó con éxito para el problema de tratamiento de la incertidumbre en IA fue el modelo de los factores de certeza, tal como se desarrolló para el sistema MYCIN [37], un sistema experto que diagnostica enfermedades infecciosas. En este modelo la información se estructura en hechos y reglas (afirmaciones de la forma SI-ENTONCES). Asociados a estos hechos y reglas aparecen los factores de certeza, que son números entre –1 y 1 que se usan para expresar el grado de creencia de dos formas distintas:

a) Para expresar el grado de creencia en una hipótesis, dada la evidencia disponible hasta el momento.

b) Para indicar el grado de creencia en una conclusión que se establece a partir de una premisa en una regla.

Un factor de creencia cercano a 1 implica que la evidencia disponible apoya fuertemente la hipótesis. Un factor de certeza cercano a –1 implica que la evidencia disponible apoya la negación de la hipótesis. Un factor de certeza de 0 indica que la evidencia disponible no apoya ni la hipótesis ni su negación. Un factor de certeza de una regla se usa para expresar la confianza en determinada agrupación antecendente-consecuente.

Hasta finales de los ochenta, los diseñadores de ITS sólo disponían de un número limitado de técnicas para tratar con la incertidumbre. Tenían que elegir entre técnicas carentes de fundamentos teóricos sólidos como MYCIN y técnicas generales que en realidad se ajustaban poco al tratamiento de los problemas de este dominio. Muchos investigadores prefirieron desarrollar sus propios heurísticos para resolver este problema, buscando enfoques robustos y fáciles de implementar. Incluso hoy en día, algunos investigadores, más preocupados por otros aspectos de sus ITS, implementan sus propios heurísticos para actualizar su modelo del alumno, como por ejemplo ocurre en el tutor Web de LISP basado en reglas ELM-ART [50], y en el tutor de derivación simbólica TUDER [52].

Los únicos sistemas que hemos encontrado en la literatura que usan el modelo de factores de certeza son los derivados de MYCIN: NEOMYCIN [53] y GUIDON [54] GUIDON es un intento de explorar la posibilidad de transformar sistemas expertos ya existentes en ITS. El sistema GUIDON se construyó a partir de MYCIN. Los objetivos perseguidos al desarrollar el sistema GUIDON fueron a) explorar la utilidad pedagógica de la base de conocimientos de un sistema experto, b) determinar qué conocimiento adicional requiere un sistema tutor, y c) expresar estrategias instructoras en términos independientes del dominio. Para ello, la base de conocimientos se mantuvo, añadiéndole nueva información, y se construyó un módulo tutor independiente, con lo cual GUIDON fue uno de los primeros sistemas en los que el conocimiento pedagógico aparecía separado del conocimiento del dominio. Aunque este enfoque no parece haber sido muy utilizado, queríamos presentarlo aquí por dos razones:

a) Aunque como ya hemos comentado el modelo basado en factores de certeza carece de fundamentos teóricos sólidos, la validación del sistema MYCIN demostró que funcionaba razonablemente bien. Evidentemente, unos resultados empíricos no son suficientes para validar el modelo de factores de certeza en general, pero al menos demuestran que el enfoque funciona bien para diagnóstico, que es una de las componentes clave en el problema del modelado del alumno.

b) El modelo de factores de certeza es muy fácil de utilizar e implementar, de forma que puede ser usado en una primera etapa para evaluar y validar los primeros prototipos del sistema, siendo siempre preferible a utilizar técnicas ad-hoc que pueden tener comportamientos imprevisibles al no haber sido debidamente evaluadas.

La principal ventaja de este enfoque es que los cálculos que hay que realizar para la propagación de la incertidumbre son muy fáciles de comprender, realizar e implementar. Aunque MYCIN tuvo mucho éxito en su dominio (diagnóstico médico), Heckerman demostró no sólo que el modelo contiene graves incoherencias, sino que es imposible construir un modelo coherente de factores de certeza [55].

2.4.1.2. Lógica Difusa

En la sección anterior hemos discutido la representación de la incertidumbre como grado de creencia. La lógica difusa [56] es otro enfoque para cuantificar grados de conocimiento, pero en un sentido diferente: se relaciona con la vaguedad y la imprecisión, que son elementos inherentes en el lenguaje natural. Por ejemplo, es habitual el uso de frases como "Juan es bastante bueno en Matemáticas, por tanto será capaz de resolver este problema que no es demasiado difícil". En esta sección introduciremos los conceptos básicos sobre lógica difusa. Una buena introducción a la lógica difusa y sus aplicaciones se encuentra en [57], y una descripción más completa puede ser revisada en [58]. Para representar la imprecisión, la lógica difusa utiliza los siguientes conceptos:

- ***Conjuntos difusos.*** Un conjunto difuso A es un conjunto cuya función característica o función de pertenencia m_A toma valores en el intervalo [0,1].

Los conjuntos difusos y las funciones de pertenencia difusas pueden utilizarse de dos formas diferentes:

a) Para estimar grados de pertenencia a un conjunto. Por ejemplo, si sabemos que sólo el 35% de los alumnos respondieron correctamente a la pregunta, ¿en qué grado es difícil la pregunta?

b) Para expresar posibilidades en una situación con información incompleta. Por ejemplo, si decimos que una pregunta es fácil, ¿cuántos alumnos la responderán correctamente? En este caso, podemos interpretar la función de pertenencia m como una distribución de posibilidad que indica preferencias en los valores que puede tomar esta variable.

Las operaciones sobre conjuntos difusos (unión, intersección, etc.) se definen como análogas a las operaciones correspondientes en conjuntos ordinarios.

- ***Variables difusas***. Una variable difusa A es una variable que toma como valores conjuntos difusos. En nuestro ejemplo, podemos definir una variable X = "grado de dificultad de una pregunta", pudiendo entonces X tomar cuatro valores posibles: Difícil, No demasiado difícil, Bastante fácil y Fácil.

- ***Relaciones difusas.*** Son conjuntos difusos definidos sobre el conjunto producto. Por ejemplo, podemos definir una relación difusa como "la dificultad de las preguntas X e Y es la misma" en términos del tanto por ciento de alumnos que dan respuesta correcta a cada una de las preguntas.

- ***Reglas difusas***. Relacionan dos o más afirmaciones difusas. Las reglas difusas se utilizan (como en otras técnicas de razonamiento no exacto) para determinar la creencia en la conclusión dado la evidencia disponible sobre la premisa de la regla.

Entre los sistemas desarrollados, basados en lógica difusa podemos encontrar el sistema KNOME [59] el cual realiza el modelado del usuario en el sistema UNIX CONSULTANT (UC), una herramienta de consulta en lenguaje natural para el sistema operativo UNIX. Durante la interacción con el usuario, KNOME crea y mantiene un modelo del usuario que usa para proporcionar ayuda al nivel de detalle adecuado según el conocimiento que posee el usuario. SPYROS [60] es un ITS sobre programación paralela. Como muchos otros sistemas tutores en dominios basados en procedimientos, SPYROS usa un conjunto de objetivos y planes estructurados en

forma de árbol para representar el conocimiento del dominio. Este árbol de objetivos y planes se completa con planes y objetivos incorrectos, para que el sistema tenga la capacidad de interpretar la solución del alumno.

MFD [61] es un tutor de matemáticas desarrollado para enseñar operaciones básicas con diferentes tipos de números (números enteros, fracciones, números mixtos y decimales). En MDF, cada tipo de problema se considera un tema, y hay relaciones de prerrequisito entre ellos. Cada tema tiene asociada una serie de habilidades, que son pasos en el proceso de resolución del problema. Por ejemplo, el tema *sumar fracciones* tiene asociadas las siguientes habilidades {*encontrar el mínimo común múltiplo, calcular fracciones equivalentes, sumar numeradores y simplificar fracciones*}. El modelo del alumno de MDF contiene dos tipos de información distintos: un *nivel de conocimiento* para cada tema y *factores generales* relativos a cada alumno, en concreto la capacidad de adquisición de nuevos conocimientos y la capacidad de recordar conocimientos antiguos, a los que llamaremos, respectivamente, factores de *adquisición* y *recuerdo.* Sin embargo, como reconocen los propios autores es un trabajo en una etapa temprana y aún hay muchas cuestiones que deben investigarse más a fondo:

- El sistema no usa ningún marco teórico sólido, por lo que no hay una comprensión formal de cómo funciona.

- En lugar de utilizar el vector de creencias se colapsa dicho vector a un único valor mediante una suma ponderada de sus componentes, en la que cada valor del vector se pondera utilizando el nivel correspondiente. Este número se usa como medida del conocimiento del alumno en el tema. Los autores consideran prioritario encontrar un mejor uso para este vector, sin embargo en una publicación posterior sobre el mismo sistema no se mencionan avances en este tema.

ALLEN es un ITS sobre Análisis de Circuitos. A los alumnos se les enseña en dos fases diferentes: una primera etapa de adquisición de conocimientos conceptuales, que conlleva el estudio de la teoría y ejemplos en un entorno basado en hipertextos, y una segunda etapa de adquisición de habilidades que mejora las habilidades del alumno mientras que éste resuelve problemas en los que debe aplicar la teoría aprendida. La interacción con el alumno en esta etapa de resolución de problemas es adaptativa, en el sentido de que se puede llevar a cabo bajo tres estrategias instructoras diferentes.

El sistema ML-MODELER [62] es el módulo del alumno de un sistema adaptativo para la enseñanza de Química, que modela dinámicamente el proceso de aprendizaje de un alumno y es capaz de proporcionar tutorización adaptativa. ML-MODELER compara la traza de la solución del alumno con la traza de la solución experta, genera hipótesis sobre los errores del alumno e infiere (utilizando razonamiento basado en casos) los métodos de aprendizaje que el alumno ha utilizado para alcanzar el estado actual de conocimiento. De esta forma, ML-MODELER es capaz de modelar no sólo qué errores y qué áreas conceptuales están siendo problemáticas para el alumno, sino también el posible uso incorrecto de técnicas de aprendizaje como analogía, generalización y especificación. La estructura usada para representar tanto el conocimiento experto como el conocimiento del alumno es una red conceptual (los autores la llaman MOP) que representa el problema, su solución y los conceptos usados para resolverlo. Cada red conceptual de un alumno representa un episodio de resolución de problemas que consiste en una red conceptual de características, hechos, conceptos y un conjunto de ecuaciones y procedimientos. El modelo del alumno consiste en su estado de conocimiento y sus mecanismos de aprendizaje y se representa también mediante una red conceptual que incluye los conceptos,

procedimientos y mecanismos de aprendizaje que ML-MODELER cree que está usando el alumno. La lógica difusa se utiliza en este sistema para seleccionar los heurísticos y conceptos que mejor explican el comportamiento del alumno.

En conclusión, la lógica difusa ha sido usada en el modelado del alumno tan sólo como una alternativa de bajo coste (en términos del esfuerzo de ingeniería del conocimiento requerido). Pero una aplicación más consistente, detallada y cuidadosa de estas técnicas podría producir mejores resultados y, como consecuencia, modelos del alumno más precisos.

2.4.1.3. La Teoría de Dempster-Shafer

La teoría de Dempster-Shafer [63] se diseñó con objeto de tratar la diferencia entre la incertidumbre y la ignorancia. La teoría de Dempster-Shafer supone que hay un conjunto exhaustivo fijo de elementos mutuamente excluyentes. Las tareas de diagnóstico en la teoría de Dempster-Shafer se realizan de forma incremental e iterativa. En este proceso, la evidencia adquirida en una iteración (M1) se combina con la adquirida en la iteración siguiente (M2) mediante las reglas de combinación de Dempster.

2.4.1.3.1. Inferencias por defecto en identificación de objetivos

El sistema descrito en [64] utiliza la teoría de Dempster-Shafer para la identificación de objetivos en una herramienta de consultoría que asesora a los alumnos sobre qué carrera elegir. Dada la información acerca de qué asignaturas ha elegido el alumno, el sistema intenta determinar cuál es la carrera que le interesa. Para ello, espera a tener varias observaciones que constituyan evidencia, integrando después estas observaciones usando la regla de combinación de Dempster-Shafer. Los criterios que el sistema usa son: a) el objetivo debe tener una credibilidad que exceda cierto umbral prefijado y b) la diferencia entre esta credibilidad y la siguiente mayor debe ser también superior a cierto umbral. Una vez que se determina el objetivo del alumno, se determina con certeza total, es decir, ningún hecho posterior puede cambiar esta creencia. Carberry utilizó este procedimiento basándose en evidencia psicológica que demuestra que es así como lo hacen los humanos cuando deben hacer inferencia en varias etapas (y no propagando la incertidumbre de una etapa a la siguiente).

2.4.1.3.2. El sistema PHI

PHI [65] es un sistema de ayuda inteligente para usuarios del correo electrónico. Usa la teoría de Dempster-Shafer para identificación de objetivos, procesando la evidencia existente sobre los objetivos que pueda tener un usuario de correo electrónico. Se distingue entre dos tipos diferentes de planes: básicos y abstractos. Por ejemplo, una observación puede sugerir que el usuario está intentando almacenar mensajes (plan abstracto) pero no si lo que planea hacer es editarlos o grabarlos (plan básico). Bauer [58] también usa la información sobre el usuario recogida en sesiones anteriores como evidencia para predecir cuáles son sus planes en la sesión actual, enfoque que parece funcionar muy bien cuando el número de sesiones es grande. Para evitar que el sistema se cree expectativas muy definitivas sesión ficticia en la cual el sistema no pudo hacer ninguna inferencia. Conforme el número de sesiones aumenta, el impacto de esta primera sesión ficticia va perdiendo importancia, imitando así el comportamiento de las redes bayesianas. En versiones más recientes, Bauer introduce una nueva forma de interpretar el comportamiento del usuario en sesiones anteriores: el sistema no sólo graba las acciones del usuario, sino también el contexto en el que fueron realizadas, de forma que el sistema también analiza cómo depende de la situación el plan elegido.

2.4.1.3.3. Esquemas de inferencias para modelos de errores jerárquicos

Una característica importante de la teoría de Dempster-Shafer es que trata con conjuntos de hipótesis, y por tanto con un conjunto de tamaño mucho mayor que si se trata con las hipótesis individuales. Este hecho puede conducir a problemas de complejidad computacional, por ejemplo: imaginemos que un alumno resuelve problemas de restas, y que se supone que tiene exactamente uno de los 36 errores catalogados en una librería. Si utilizamos la teoría de Dempster-Shafer para intentar determinar cuál de estos errores está cometiendo, tenemos que tratar con (236-1) subconjuntos no vacíos del conjunto de los 36 errores posibles, lo cual puede resultar demasiado costoso computacionalmente, especialmente si se realizan muchas observaciones. Para reducir esta complejidad computacional, se dividen los 36 errores en 3 clases básicas, donde cada clase contiene errores que producirían respuestas incorrectas en cada tipo particular de problema. Cuando el usuario da una respuesta incorrecta A a un problema P, se asigna una función de creencia a la clase básica de errores que producen respuestas incorrectas a P (no necesariamente la A) y a cada uno de los subconjuntos de un sólo elemento (una única hipótesis) cuyo error produciría la respuesta A a la pregunta P. Este procedimiento fue utilizado con respuestas incorrectas generadas artificialmente y el sistema era capaz de diagnosticar el tipo de error que las generaba. Pero, aún cuando parece trabajar mucho mejor que la aplicación directa de reglas erróneas en términos de la complejidad computacional, no parece claro si funcionaría bajo circunstancias más reales.

2.4.1.4. Redes Bayesianas

Una red bayesiana [66] es un grafo acíclico dirigido en el que los nodos son variables y los arcos representan relaciones de influencia causal entre ellos. Los parámetros usados para representar la incertidumbre son las probabilidades condicionadas de cada nodo dado los diferentes estados de sus padres.

Las redes bayesianas permiten hacer dos tipos de inferencia distintos:

- Inferencia abductiva: Sabiendo que el alumno ha resuelto correctamente una situación, ¿cuál es la probabilidad de que domine cierta parte del currículo?
- Inferencia predictiva: Sabiendo que el alumno domina cierta parte del currículo, ¿cuál es la probabilidad de que sea capaz de resolver cierto problema P?

La primera propuesta de usar redes bayesianas en el modelado del alumno aparece en [67]. En este artículo se discute la aplicación de dos modelos teóricos distintos al problema del modelado: la teoría del espacio de conocimiento y las redes bayesianas. Es aquí donde se pueden encontrar las primeras ideas acerca de cómo construir y usar tales modelos. Desde entonces se han desarrollado varios sistemas en los que las redes bayesianas se han utilizado con éxito para construir y actualizar el modelo del alumno.

Los sistemas OLAE [68], POLA [69] y ANDES [70] son el resultado de una década (la de los noventa) de investigación del equipo liderado por Kurt Vahn Lehn en la Universidad de Pittsburgh. POLA es el módulo de diagnóstico del alumno en ANDES (Sistema Instructor Inteligente para Física Newtoniana), y representa una mejora respecto a OLAE, puesto que permite construir el modelo del alumno con la técnica de traza del modelo. Por tanto, describiremos primero el sistema OLAE, y después el sistema POLA.

OLAE es una herramienta que recopila información sobre alumnos que resuelven problemas a nivel introductorio de física, analiza esos datos con métodos probabilísticos (redes bayesianas) y determina lo que sabe el alumno. OLAE genera automáticamente para cada problema una red bayesiana que relaciona el conocimiento (representado en forma de reglas de primer orden) con acciones concretas, como por ejemplo ecuaciones escritas. Usando la red resultante, OLAE observa el comportamiento del alumno y calcula las probabilidades de que el alumno conozca y use cada una de las reglas.

En la red bayesiana de OLAE, se consideran cuatro tipos de nodos: nodos de regla, para recoger si el alumno conoce o no una regla del dominio; nodos de aplicación de la regla, para saber si el alumno usó determinada regla durante la resolución del problema propuesto; nodos de hecho, que recogen si el alumno sabe determinado hecho acerca del problema y nodos de acción, que recogen si el alumno ha realizado determinada acción. Estos nodos se conectan mediante arcos dirigidos en la red. Los diferentes caminos que se pueden seguir a través de la red representan la multitud de formas que un alumno puede utilizar para resolver determinado problema. Una vez que el alumno da una respuesta, los algoritmos de propagación actualizan las probabilidades a través de los arcos para determinar la probabilidad a posteriori de que el alumno conozca determinada regla.

El grafo de resolución de problemas es una red dirigida de unos 150 nodos, que se va generando de forma automática de la siguiente forma: siempre que se pueda usar una regla para producir una conclusión a partir de ciertos antecedentes, se introduce un nodo en la red para representar la aplicación de la regla. Asimismo se introduce un arco desde el nodo de aplicación de la regla hasta un nodo de hecho que represente su conclusión (dicho nodo se crea en ese momento si es que no existe). Para cada antecedente (hechos usados para justificar que la regla se dispare) se introduce un arco desde su nodo de hecho hasta el nodo de la aplicación de la regla. También se introduce un arco desde el nodo de la regla hasta el nodo de aplicación de la regla. Si un hecho tiene una acción observable correspondiente, se crea un nodo de acción y se coloca un arco desde el nodo de hecho hasta el nodo de acción. De esta forma OLAE genera automáticamente la red bayesiana a partir del modelo del dominio. Una vez la red bayesiana está generada el alumno resuelve el problema y OLAE propaga esta información a través de la red actualizando las probabilidades de cada uno de los nodos. Otra característica importante de OLAE es que proporciona un segundo tipo de red bayesiana que está diseñada específicamente para el profesor, que consulta el sistema una vez terminado el proceso descrito anteriormente. Esta red para el profesor contiene los siguientes nodos: (a) los nodos de regla de la red bayesiana original que representan el resultado del proceso de inferencias del sistema y (b) nodos dimensionales que almacenan la información de variables más abstractas que representan el dominio que tiene el alumno sobre partes específicas del currículo, como Cinemática o Dinámica. En nuestra opinión, estos nodos podrían incluirse directamente en la red, de forma que sus probabilidades se fuesen actualizando a medida que evolucionan las otras probabilidades de la red5. Esto permitiría además que, si por cualquier circunstancia adquirimos conocimiento acerca de que el alumno domina determinada parte del currículo, este conocimiento afectaría también a la probabilidad de que domine las reglas que lo componen. Cabe resaltar que el sistema OLAE actúa cuando el alumno ha terminado de resolver el problema, puesto que su propósito no era servir de soporte a una enseñanza interactiva, sino simplemente diagnosticar de una forma precisa qué partes del dominio eran conocidas por el alumno.

POLA (Probabilistic On-Line Asessment) es una extensión del sistema OLAE para determinar no sólo las reglas que sabe el alumno sino el camino seguido por el mismo

para la resolución del problema, tratando la incertidumbre en la interpretación de las acciones del alumno de forma consistente utilizando probabilidades. Es decir, mientras que OLAE sólo realiza lo que Anderson y otros llaman traza del conocimiento determinación de qué sabe el alumno, incluyendo conocimiento correcto y errores), POLA realiza también la traza del modelo (seguimiento de la forma de resolver un problema). En particular, cuando existan varios caminos de resolución que sean consistentes con la acción que ha tomado el alumno, POLA tendrá la capacidad de decidir qué camino es más probable que haya sido el seguido por el alumno. A partir de tal información se dota al sistema de nuevas capacidades, como contestar preguntas formuladas por el alumno o generar pistas a un nivel adecuado, y también se pueden tomar decisiones pedagógicas como proporcionar una ayuda, presentar cierto material o elegir el siguiente problema a proponer. Con este objeto, es preciso que el módulo de diagnóstico del sistema conozca las posibles líneas de razonamiento que los alumnos pueden seguir. El conjunto de tales líneas se denomina espacio de soluciones, y a la estructura de datos usada para representarlo grafo solución. El grafo solución se construye automáticamente a partir de una base de conocimientos de reglas de producción y contiene tres tipos de información: a) todos los planes para resolver el problema que se pueden derivar de las reglas de la base de conocimiento; b) todos los caminos algebraicos de resolución que desarrollan dichos planes, y c) el razonamiento que subyace a dichos planes.

2.4.1.4.1. Modelado del alumno con Redes Bayesianas

Los trabajos de Jim Reye [71] en modelado bayesiano del alumno se basan en la hipótesis que el dominio de conocimiento se puede estructurar en una colección abstracta de temas que representan conocimiento conceptual o habilidades que el alumno debe adquirir, y que esos temas admiten una estructuración en forma de relaciones de prerequisitos. Reye propone que la estructura de la red bayesiana apropiada para este problema se base en: a) una parte central, que conecte a todos los nodos de conocimiento (que Reye denomina nodos “student-knows”) ordenándolos en términos de una lista de relaciones de prerrequisito, y b) un grupo de nodos para cada uno de los nodos conocimiento de la parte central, en el que aparezcan el nodo de conocimiento y un conjunto de nodos adicionales relacionados con él, como por ejemplo nodos para medir el interés del alumno en el tema particular (lo que Reye denomina nodos “student-interested-in”). Este tipo de estructura tiene la ventaja de que las actualizaciones sucesivas de la red conforme se va adquiriendo nueva evidencia se llevan a cabo de forma local y sólo afectan a las partes de la red correspondientes a otros temas a través de la parte central, permitiendo aumentar la eficiencia al realizarse los cálculos localmente. Pero en nuestra opinión esta estructura resulta demasiado restrictiva, puesto que como hemos visto en otros sistemas discutidos en esta sección, los nodos en la red pueden utilizarse para representar muchos factores diferentes que no tienen cabida en este tipo de enfoque

Otro trabajo sobre la utilización de redes bayesianas dinámicas para modelado del alumno es la red bayesiana dinámica que utiliza Reye [71]. Este trabajo es muy simple, puesto que el concepto de dinámico en el tiempo se mide en función de interacciones con el sistema en lugar de en función de intervalos de tiempo. De esta forma, para cada i = 1, ..., n se define el nodo Li = "estado del conocimiento del tema que posee el alumno después de la i-ésima interacción con el sistema", y este nodo se hace depender del nodo Li-1 y del nodo Oi-1 (resultado de la interacción n-ésima, que a su vez dependerá también de Li-1). En la figura 3 se muestra la estructura de la Red Bayesiana.

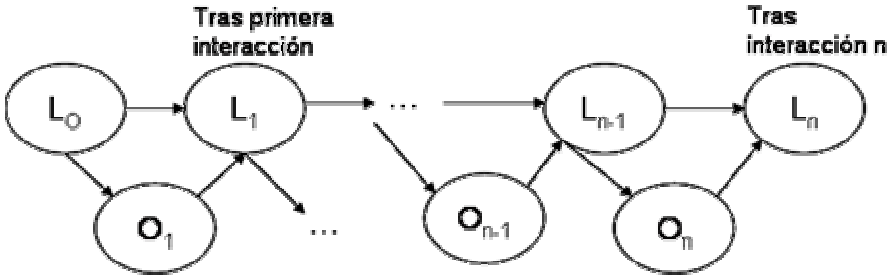


Figura 3. Redes bayesianas dinámicas para modelado del alumno

2.4.1.4.2. Test adaptativos y redes bayesianas

El trabajo del grupo ARIES, investiga el uso de redes bayesianas en test adaptativos [72]. Se basa en la aplicación de redes bayesianas y jerarquías de granularidad para, a partir de un conjunto de preguntas tipo test, evaluar al alumno. En este trabajo se parte de un dominio de conocimiento estructurado en: objetivos a aprender (*learning objectives*) con niveles de logro específicos y un conjunto de preguntas (que no son necesariamente tipo test, sino que pueden ser cualquier tipo de preguntas siempre que aseguremos que podemos comprobar si la respuesta que da el alumno es correcta o incorrecta). Los tipos de relaciones considerados son: relaciones de agregación (que permiten descomponer un objetivo en subobjetivos y que garantizarán tests de contenido equilibrado), relaciones de prerequisitos (que permiten una estructuración del dominio y que ayudan a establecer el orden de las preguntas en el test) y relaciones objetivos-pregunta, entre objetivos de aprendizaje alcanzados y preguntas, que son las que permitirán realizar el diagnóstico. En cuanto a la selección de preguntas, la propuesta de este grupo es elegir la pregunta más informativa (que maximiza cierta medida de utilidad). La medida de utilidad de una pregunta Q para un objetivo O se define en la ecuación mostrada a continuación. Ecuación 1.

$$utilidad(Q) = \left| P(O/Q) - P(\neg O/\neg Q) \right|$$

Ecuación 1. Ecuación de la medida de utilidad

Es decir, la probabilidad que se domine el objetivo O dado que la pregunta se responde correctamente menos la probabilidad de no dominarlo dado que la pregunta se responde incorrectamente. Para calcular dichos valores, cada vez que queremos elegir una pregunta deberemos actualizar la red 2n veces (donde n es el número total de preguntas), construir las diferencias y elegir la máxima. Para nosotros, esta medida de utilidad es muy discutible, puesto que el objetivo debería ser maximizar ambas probabilidades y por tanto no tiene mucho sentido maximizar la diferencia en valor absoluto.

Otro trabajo realizado es el sistema Desktop Associate [73][74] el cual evalúa las habilidades de un usuario que usa un procesador de textos. Los nodos que utiliza Murray son básicamente de dos tipos:

- **Nodos de habilidades**: miden si el alumno es capaz de hacer algo, como dar formato a un párrafo, cambiar el tipo de letra, etc. Dentro de estos nodos, se distinguen las habilidades básicas (habilidades que no admiten descomposición), como cambiar el tipo de letra, y las habilidades más generales (nodos dimensionales), como dar formato a un párrafo. Ambos tipos de nodos aparecen

simultáneamente en la misma red, a diferencia del enfoque adoptado por Conato y Van Lehn en POLA.

- **Nodos evidencia**: son los nodos encargados de recoger la información sobre el alumno que después servirá para determinar su nivel de conocimiento. Dicha información se puede recoger de tres formas distintas (que se corresponden con tres clases distintas de nodos evidencia): realizando preguntas al alumno, pidiéndole que realice cierta tarea o, si el profesor observa de forma directa que el alumno tiene cierta habilidad, introduciendo esta información en la red.

En cuanto a las relaciones de causalidad, Murray considera que tener una habilidad tiene influencia causal en ser capaz de realizar una tarea o contestar una pregunta, y que poseer una habilidad general tiene influencia en poseer las habilidades en las que se descompone. En [73] se propone una simplificación para el problema de la obtención de los parámetros de la red. Partiendo de una red en la que sólo hay nodos de habilidad (que pueden tomar n valores) y nodos pregunta (que son binarios, es decir, se considera que cada pregunta se responde correcta o incorrectamente). En general, si para medir una habilidad tenemos q preguntas, se deben especificar n·q·k probabilidades condicionadas, y n-1 probabilidades a priori. Si quisiéramos modelar k habilidades, necesitaríamos (k **x** n **x** q), que es un número muy grande incluso para valores pequeños de n, q y k. Para reducir el número de datos precisos, Murray propone a) agrupar las preguntas por niveles de dificultad, y utilizar los mismos parámetros para preguntas del mismo nivel, lo cual reduce el número de parámetros necesarios de (k **x** n **x** q) a (k **x** n **x** c), donde c es el número de niveles de dificultad considerados y b) asociar estos niveles de dificultad a los valores de los niveles utilizados para las habilidades, es decir, si por ejemplo para cada habilidad se tienen cinco valores {novel, principiante, intermedio, avanzado, experto}, podemos considerar cuatro categorías de preguntas {*nivel-principiante, nivel-intermedio, nivel-avanzado, nivel-experto*} (no necesitamos nivel novel puesto que el alumno se clasificará como novel cuando no pueda contestar bien ni siquiera a las preguntas de nivel principiante). La última reducción en el número de parámetros es resultado de la naturaleza transitiva de esta clasificación de las habilidades en categorías: si un alumno alcanza cierto nivel, entonces debe ser capaz de responder a todas las cuestiones correspondientes a este nivel y a niveles inferiores, y probablemente no responderá correctamente a las preguntas de niveles más avanzados que el suyo. Para modelar las adivinanzas (*respuestas correctas sin tener conocimiento*) y los descuidos (*errores debidos no a una falta de conocimiento, sino a otros factores difícilmente controlables como despistes, errores al teclear, etc.*), que pueden modificar las hipótesis anteriores, se usan dos probabilidades: "**s**" (probabilidad de error) y "**g**" (probabilidad de adivinanza), y las probabilidades condicionadas se construyen como en el siguiente ejemplo:

• P (respuesta correcta a pregunta nivel intermedio/alumno principiante) = g.
• P (respuesta incorrecta a pregunta nivel principiante/alumno principiante) = 1-s.

De esta forma, las k **x** n **x** c probabilidades condicionadas se pueden obtener de s y g. Esta forma de calcular el número de parámetros tiene otra ventaja adicional: el proceso de propagación de probabilidades cuando se adquiere evidencia tiene lugar en tiempo lineal. Como contrapartida, la principal desventaja de esta aproximación es su limitado alcance: sólo permite diagnosticar una habilidad cada vez, y sólo permite usar nodos evidencia binarios. Y por último, la gran limitación de este enfoque es que su validez se restringe a redes con forma de árbol, es decir, en redes cuyos nodos tienen un único padre, lo cual es, una restricción muy fuerte. Basándonos en esta idea, hemos realizado unas extensiones de las puertas AND y OR clásicas [66] que permiten simplificar el problema de la especificación de los parámetros en redes con

cualquier tipo de estructura. Dichos resultados aparecen publicados en [75]. Como continuación de este trabajo Murray propone en [74] una implementación del algoritmo clásico de propagación en árboles que garantiza la actualización en tiempo lineal.

Luego de revisar las diferentes técnicas de razonamiento aproximado existentes se descarta la utilización de alguna de ellas para el modelado del alumno porque:

a. Como ya hemos discutido en este apartado, no recomendaríamos el uso de factores de certeza en modelado del alumno, sobre todo por su falta de una base teórica sólida. Cuando se usan modelos carentes de fundamentos teóricos las inconsistencias pueden hacer que el comportamiento del modelo del alumno sea impredecible, especialmente en situaciones que no han sido consideradas previamente por sus autores. Sin embargo, consideramos que el modelo de factores de certeza es un procedimiento sencillo de entender e implementar en las primeras versiones de un sistema, permitiendo así hacer una primera evaluación antes de utilizar modelos mejores desde el punto de vista teórico pero que exigen un esfuerzo mucho mayor de implementación como el razonamiento basado en casos.

b. La lógica difusa ha sido considerada seriamente como alternativa para modelado del alumno por su capacidad para procesar datos de entrada expresados verbalmente de forma imprecisa, y no descartamos su uso en nuestro trabajo futuro. La lógica difusa debería ser considerada en aquellas situaciones en que:

    - El razonamiento que hay que realizar se pueda describir de forma natural en términos de conceptos, operadores o reglas imprecisas. Este razonamiento puede ser el relativo al alumno cuyo comportamiento estamos intentando anticipar, o al tutor humano cuyo conocimiento estamos intentando transferir al sistema tutor.

    - Necesitamos procesar datos de entrada imprecisos, como por ejemplo en el caso de un tutor que deba procesar afirmaciones en lenguaje natural. Hay que tener en cuenta que si utilizamos lógica difusa nos veremos obligados a elegir entre diferentes interpretaciones para algunos de sus conceptos, como por ejemplo entre diferentes procesos de paso de difuso a nítido o diferentes significados para los operadores AND, OR y NOR.

c. Para la aplicación de la teoría de Dempster-Shafer encontramos principalmente dos problemas:

    - basar una decisión en los resultados del análisis es más complicado que cuando se utiliza CBR, puesto que CBR cada hipótesis se asocia con un caso (s), mientras que en Dempster-Shafer se requieren medidas de compatibilidad y conjunto de criterios adicionales por cada hipótesis generada.

    - la teoría de Dempster-Shafer realiza inferencia abductiva, pero no predictiva, con lo cual no permite realizar predicciones, que tan útiles son en modelado del alumno. Sin embargo, esta teoría parece especialmente recomendable en aquellas situaciones en las que tengamos informaciones no totalmente fiables sobre el alumno, pero que aún puedan tener cierto interés.

d. En la aplicación de Redes Bayesianas se encuentran los siguientes problemas:

- Especificación de los parámetros (probabilidades condicionadas): Es el problema mas complicado cuando se usan redes bayesianas. Al utilizar este tipo de técnica, se supone que tanto la estructura de dependencias como los parámetros son proporcionados por el experto humano; sin embargo, para un profesor puede resultar imposible especificar el gran número de probabilidades condicionadas que se requieren. Todo ello ha motivado que se haya investigado mucho en técnicas de simplificación de los parámetros, o de obtención de los mismos a partir de bases de datos existentes (también para aprender las estructuras, es decir, las relaciones causales, a partir de datos), sin embargo, la aplicación de estas técnicas está condicionada a la existencia de bases de datos, que probablemente sean escasas en el modelado del alumno.

- el esfuerzo que supone especificar el modelo (variables y relaciones causales), la complejidad computacional de los algoritmos de propagación, y la dificultad que supone la implementación de los mismos.

CBR tiene gran versatilidad en el modelado del alumno, y se constituye en una aproximación muy potente para realizar diferente tipo de inferencias. Adicionalmente, el paradigma de razonamiento basado en casos posee fuerte solidez teórica lo que hace que los modelos generados bajo este principio sean coherentes y funcionen razonablemente.

3. SISTEMAS DE RAZONAMIENTO BASADO EN CASOS

3.1. INTRODUCCIÓN

CBR es una reciente aproximación que se ha convertido en una importante área de investigación en los últimos años. Este nuevo paradigma se diferencia en muchos aspectos de otras aproximaciones usadas en Inteligencia Artificial. El CBR se utiliza como paradigma para la resolución de problemas, implicando un enfoque fundamentalmente diferente al de la mayoría de los sistemas de IA. En lugar de basarse exclusivamente en el conocimiento general del dominio de un problema, o establecer asociaciones a través de un conjunto de relaciones generalizadas entre descriptores de problemas y conclusiones, se utiliza el conocimiento específico de experiencias previas en situaciones concretas. Desde entonces, los sistemas CBR se han aplicado con éxito a un amplio espectro de problemas [10][11][14]. En esta sección, se hace una revisión de los sistemas CBR y el conjunto de tecnologías usadas en la construcción de sistemas de razonamiento basado en casos.

3.2. GENERALIDADES DE CBR

El aprendizaje basado en casos consiste en adquirir conocimiento a partir de experiencias precedentes o casos. También se conoce como razonamiento basado en casos por el hecho de que este tipo de aprendizaje no se concibe sin el proceso de razonamiento que conlleva la obtención de una nueva experiencia. Es un método para resolver problemas recordando situaciones similares, reutilizando la información y el conocimiento sobre dichas situaciones [13]. La idea original es simple:

"Un sistema basado en casos resuelve nuevos problemas adaptando soluciones que fueron usadas para resolver problemas anteriores."

La memoria del sistema CBR almacena un cierto número de problemas junto a sus correspondientes soluciones. La solución de un nuevo problema se obtiene recuperando los casos similares almacenados en la memoria del sistema CBR. Por tanto, un caso engloba un problema y la solución dada a dicho problema.

Un sistema CBR define un modelo dinámico que incorpora una concepción incremental y continua del aprendizaje. Los nuevos problemas se añaden a la memoria del CBR, de forma que los problemas parecidos o similares se eliminan y paulatinamente se crean otros mediante la combinación de varios ya existentes. La idea que impulsó el desarrollo de esta metodología se basa en el hecho de que los humanos utilizan lo aprendido en experiencias previas para resolver problemas presentes. Este hecho se experimenta de forma diaria y ha sido probado empíricamente por varios expertos en el área de psicología [76].

... "El sistema CBR es muy efectivo en situaciones donde la adquisición de casos y la determinación de los hechos es sencilla comparada con la tarea de desarrollar un mecanismo de razonamiento."

Los sistemas CBR están especialmente indicados cuando las reglas que definen un sistema de conocimiento son difíciles de obtener, o el número y complejidad de las mismas es demasiado grande para ser representado con un sistema experto. Esta tecnología ha sido utilizada satisfactoriamente en disciplinas como el derecho, la medicina y sistemas de diagnóstico con grandes bases de datos [10].

3.3. ESTRUCTURA GENERAL DEL CICLO DE VIDA DE UN CBR

Como se ha mencionado anteriormente, los sistemas CBR permiten la resolución de nuevos problemas adaptando soluciones de problemas similares que se obtuvieron con anterioridad. Por medio de algoritmos de indexación, de recuperación de problemas previamente almacenados, de técnicas de comparación y adaptación de problemas a una determinada situación, se obtiene la solución a un nuevo problema. Este tipo de razonamiento está basado en el conocimiento almacenado en su memoria en forma de casos o problemas. Todas las acciones llevadas a cabo están estructuradas y pueden ser representadas por una secuencia cíclica de procesos, que requiere generalmente la intervención humana. Un sistema CBR típico, está compuesto por cuatro etapas secuenciales que se invocan siempre que es necesario resolver un problema [9]. La Figura 4, muestra el ciclo de vida de un sistema de razonamiento basado en casos clásico, donde las cuatro etapas del proceso se definen a continuación:

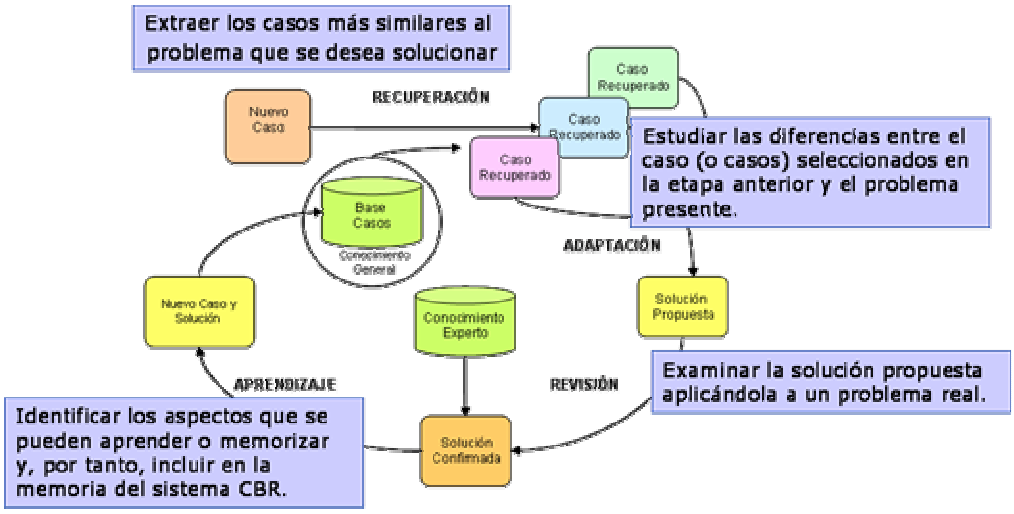


Figura 4. Ciclo de Vida de un sistema CBR [9]

Como ya se indicó previamente, el objetivo de un sistema CBR es encontrar la solución de un problema determinado. La misión del algoritmo de recuperación, consiste en buscar en la memoria del sistema CBR y seleccionar de ella los problemas más similares (junto con sus soluciones) al problema presente. Los casos seleccionados de la memoria son reutilizados para generar una solución inicial, llevándose a cabo normalmente en esta fase la adaptación de casos. Si es posible, la solución dada en la etapa anterior se revisa y finalmente se crea un nuevo caso (constituido por el problema y la solución del mismo) que se almacena en la memoria del sistema CBR. Los casos almacenados en la memoria pueden ser eliminados o modificados, pudiendo crearse nuevos casos a través de la mezcla de características de otros ya existentes. Así, por ejemplo, si la solución dada a un problema es insatisfactoria, una vez encontradas las causas de esta deficiencia, se intentará eliminar de la memoria todos aquellos casos que provoquen dicha irregularidad.

Por tanto, un sistema CBR es capaz de utilizar el conocimiento de situaciones concretas experimentadas con anterioridad, en lugar de realizar asociaciones a lo largo de relaciones generales entre características de problemas, como haría un sistema de razonamiento basado en reglas (*Rule-Based Systems, RBS*) 0, o depender del conocimiento general sobre un determinado problema. Un sistema CBR usa un modelo de aprendizaje incremental, de forma que cada vez que se resuelve un

problema es posible crear un nuevo caso, y almacenarlo en la memoria del sistema CBR para su posterior utilización. De este modo, el aprendizaje tiene lugar de una forma natural como un subproducto del propio método de resolución aplicado, mediante la actualización continua de la base de casos. No obstante, el aprendizaje efectivo en los sistemas CBR requiere un conjunto de métodos que permitan extraer conocimiento relevante a partir de la experiencia, integrar un caso en una estructura de conocimiento ya existente, e indexarlo adecuadamente para que se pueda comparar en el futuro con casos similares.

La intervención humana puede ser necesaria a lo largo del ciclo de vida de un sistema CBR, especialmente en las dos últimas fases del ciclo. La revisión y la retención de casos, se realizan normalmente por profesionales especializados. Este hecho, supone uno de los mayores inconvenientes de esta metodología, y ha sido señalado como un gran inconveniente por sus detractores [77]. En los siguientes sub-apartados se describen las características generales de las etapas que definen el ciclo de vida de un sistema CBR.

3.3.1. Recuperación de Casos

El objetivo de esta fase es extraer los casos almacenados en la memoria del sistema CBR más similares al problema que se desea solucionar. Esta fase se puede dividir en cuatro sub-fases, que normalmente tienen lugar en el orden en que se presentan a continuación:

- **Identificación de figuras**: implica la caracterización de los atributos que definen un problema.
- **Comparación inicial**: implica la selección de los casos que son significativamente similares al problema presente.
- **Búsqueda** y,
- **Selección:** durante estas etapas se identifica el caso (o grupo reducido de casos) más similar al presente problema por el que se empezará a definir la solución final. Dicho caso se extrae de la selección preliminar hecha en la sub-fase anterior.

Aunque un CBR es un sistema bien definido, la naturaleza del problema en el que se centra determina su estructura final. Por ejemplo, antes de implementar los algoritmos de recuperación de casos, es necesario identificar si las técnicas de selección utilizadas deben basarse en la comparación sintáctica o semántica de las características que definen dichos casos. Los sistemas CYRUS [12], ARC [78] y PATDEX-1 [79] utilizan mecanismos de recuperación de casos basados en las similitudes sintácticas existentes entre las características que describen un problema. Esta aproximación, se denomina de conocimiento escaso (*knowledge poor approach*). Por el contrario, PROTOS [80], CASEY [81] CREEK [82][83][84] y MMA [85] recuperan casos basándose en rasgos con fuertes connotaciones semánticas. Esta aproximación, recibe el nombre de conocimiento intensivo (*knowledge intensive approach*) y precisa de un dominio amplio del problema, para analizar la relación existente entre diversos casos.

Otro aspecto importante, a tener en consideración antes de implementar una estrategia de recuperación de casos, es el decidir si los casos tienen que adaptarse al problema a solucionar en la etapa posterior. Esta decisión es fundamental para la identificación de una estrategia adecuada de aproximación, y conocer así el número adecuado de casos que se deben recuperar. La caracterización de un problema puede realizarse simplemente con el empleo de rasgos descriptivos, sin embargo, si se utiliza una aproximación de conocimiento intensivo, se necesita un mecanismo más sofisticado que requiere el pleno entendimiento del problema en su contexto.

Las tres formas más normales de llevar a cabo la recuperación de casos son [9]:

- Utilizando punteros directos indexados que señalan las características de los problemas.
- Realizando búsquedas en una estructura de índices.
- Realizando búsquedas en modelos de conocimiento general.

En algunos sistemas es necesario, para poder recuperar un caso, que todas las características que lo definen sean iguales o muy similares a las del presente problema. En otros, es necesario cuantificar las características que definen un problema y su importancia, de esta forma se puede realizar una comparación más flexible y establecer límites de aceptación. Si el problema requiere un conocimiento intensivo de un dominio en particular y un entendimiento profundo de todas sus características, es necesario realizar unos análisis de condiciones, objetivos, motivaciones, tendencias, etc., para lograr definir una estrategia de recuperación adecuada.

El caso más próximo al problema a resolver, se selecciona tras un estudio profundo de los casos recuperados inicialmente. Este estudio depende de las características del problema. Por ejemplo, la selección del caso más adecuado se puede realizar, después de un estudio que defina por qué el resto de los casos no son adecuados desde un punto de vista semántico. Este proceso, en general, es mucho más complicado que la selección inicial de los casos y en la mayoría de los sistemas actuales requiere la intervención humana.

3.3.2. Reutilización de Casos

El objetivo de esta fase es el estudio de las diferencias entre los casos seleccionados en la etapa anterior y el problema presente. En esta etapa también se tienen en cuenta cuáles son las características de los casos recuperados, que pueden trasladarse al presente problema [9].

En problemas simples de clasificación, la diferencia entre las características de los casos se oculta y sólo se tienen en consideración las similitudes. En estas situaciones el presente problema adopta la solución del caso recuperado. Sin embargo, en la mayoría de los sistemas donde las diferencias son importantes, es necesario realizar una adaptación en las soluciones.

Las dos formas más utilizadas para la adaptación de casos son:

- ***Reutilización transformacional:*** consistente en la reutilización de soluciones dadas a problemas con anterioridad. La solución del caso recuperado no es la solución adoptada por el presente problema, pero existe una función de transformación que genera la solución del presente problema utilizando la solución del caso recuperado.

- ***Reutilización derivacional:*** consistente en la reutilización de los métodos que fueron empleados para generar soluciones en problemas pasados. El caso recuperado tiene que contener información relacionada con el método utilizado para alcanzar la solución del problema representado por dicho caso, incluyendo así una justificación para cada operación utilizada, objetivos, alternativas generadas, problemas encontrados en la búsqueda de la solución, etc. En este tipo de reutilización, se ejecuta el método de recuperación utilizando los detalles del presente problema y siguiendo el antiguo plan en el contexto actual. En esta etapa se pueden utilizar algoritmos generales de resolución de problemas para la planificación del sistema.

3.3.3. Revisión de Casos

En esta fase, la solución propuesta por la etapa anterior se examina minuciosamente. Si la solución es aceptada, será la definitiva y se tendrá en cuenta durante la etapa de aprendizaje, tal como se mostrará en el próximo sub-apartado. Pero si la solución no es satisfactoria, se puede modificar o reparar utilizando conocimiento específico acerca del problema en cuestión [9].

Normalmente la evaluación de una solución se realiza aplicando dicha solución a un problema real, y son sus características las que determinan cuándo se puede llevar a cabo esta evaluación, siendo en ocasiones necesarias el uso de simuladores. Si la solución tiene que modificarse, es necesario identificar y poder explicar los errores cometidos durante su generación, utilizando técnicas basadas en explicaciones. Una vez que los errores han sido identificados, la solución se repara utilizando la explicación del error, lo que garantiza que no volverá a producirse.

3.3.4. Retención o aprendizaje de Casos

Durante esta etapa, se identifican los aspectos que se pueden aprender o memorizar y, por tanto, incluir en la memoria de un sistema CBR. Los algoritmos de aprendizaje deben de tener en cuenta los resultados de la fase anterior [9].

Las tareas más importantes a realizar durante esta etapa son:

- Seleccionar la información contenida en los casos que debe ser memorizada.
- Definir cómo se debe almacenar dicha información.
- Establecer los mecanismos de indexación del nuevo problema, para así poder reutilizarlo en un futuro.
- Definir la forma de integrar el nuevo caso en la estructura de la memoria.

Para establecer los algoritmos de aprendizaje, es necesario definir si las fuentes de dicho aprendizaje son: las características que describen un problema, la solución del problema o el resultado obtenido tras la puesta en práctica de la solución.

Los CBR son sistemas dinámicos en los cuales la memoria se debe modificar continuamente. Para explicar mejor el concepto, considérense las dos situaciones siguientes:

- Si un caso es reutilizado y adaptado para generar una nueva solución, este caso se puede generalizar y absorber así parte del nuevo problema, o incluso se puede crear un nuevo caso, añadiéndolo a la memoria del sistema CBR.

- Si para la creación de la solución es necesaria la intervención de un experto humano, normalmente se crea un nuevo caso que representa el conocimiento del experto.

Para memorizar un caso es muy importante determinar cómo y cuándo se debe indexar. En esta operación, la estructura de indexación es crucial.

La actualización de la memoria es el paso final de esta fase. El aprendizaje se realiza por diversos medios: la integración de varios casos, la eliminación de casos, la indexación de los casos ajustándolos en función del éxito de su reutilización. Además, en aproximaciones de conocimiento intensivo, el aprendizaje tiene lugar durante la creación del modelo conceptual del conocimiento, por medio de la interacción de un experto (humano) u otro método artificial de aprendizaje.

3.4. REPRESENTACIÓN DE LOS CASOS

Un caso es un problema bien estructurado sintáctica y semánticamente, que contiene una experiencia pasada e información acerca del contexto en la que se adquirió dicha experiencia. Un caso puede estar compuesto por:

Un problema, que muestra el estado de una situación en un momento concreto.
1. Una solución a dicho problema.
2. El resultado de aplicar dicha solución a un problema concreto.

Los casos que describen problemas y sus soluciones, pueden utilizarse para proporcionar soluciones a nuevos problemas [81]. Los casos compuestos por problemas y resultados, se emplean para evaluar nuevas soluciones. Los casos formados por problemas, soluciones y resultados se pueden utilizar para evaluar los resultados de posibles soluciones, y por tanto evitar riesgos así como posibles problemas, como por ejemplo MEDIATOR [11]. Los casos se pueden representar de forma diferente dependiendo del problema en cuestión, de la estructura o de los algoritmos utilizados en cada fase del ciclo de vida del sistema CBR, etc. Por tanto, dichos casos se pueden representar utilizando objetos, predicados, estructuras, reglas, etc. La información representada en los casos deber ser funcional, fácil de entender, de interpretar y de recuperar.

3.5. TIPOS DE SISTEMAS DE RAZONAMIENTO BASADO EN CASOS

Esta sección presenta una clasificación de los diferentes modelos de sistemas CBR. Aunque todos comparten características similares, cada uno de ellos se adapta mejor a un tipo de problema en particular. Algunos sistemas CBR no requieren todas las fases de un sistema CBR típico, mientras que otros se diferencian tan solo en la tecnología empleada para la implementación de alguna de estas fases, además, como se ha citado anteriormente, en ocasiones es necesaria la intervención humana. Todas estas características, junto con la dependencia y heterogeneidad de los dominios de aplicación, han conducido a la construcción de un número elevado de sistemas de este tipo. La clasificación que se presenta en esta sección se ha extraído de [9] y en ella se identifican cinco grandes grupos de sistemas.

*3.5.1. Razonamiento Basado en Ejemplares (***Exemplar-Based Reasoning, EBR***)*

En la literatura hay diferentes formas de definir el término concepto, aunque podría considerarse como un conjunto de ejemplos. Los sistemas CBR que se centran en el aprendizaje de definiciones de conceptos, se denominan normalmente sistemas de razonamiento basados en ejemplares. PROTOS es un exponente de este tipo de sistemas. Los casos más parecidos se agrupan en clases, las cuales absorben las características más representativas de los casos asociados a ellas. La solución de un problema requiere encontrar la clase que, de forma general, representa las características fundamentales de un ejemplar (problema) en particular. La solución de la clase a la que pertenece el caso recuperado más similar al problema presente se utiliza como la solución de dicho problema [9].

*3.5.2. Razonamiento Basado en Instancias (***Instance-Based Reasoning, IBR***)*

Este tipo de razonamiento puede ser considerado como un razonamiento basado en ejemplares, centrado en problemas con fuertes connotaciones sintácticas [9]. Este tipo de sistemas CBR se centran en problemas en los que hay un número muy grande de instancias (casos), necesarias para representar un amplio espectro de posibilidades. Se utiliza en situaciones en las cuales existe una falta de conocimientos generales

acerca de las leyes que rigen el comportamiento de un sistema. La representación de un caso puede ser realizada con vectores de características, y las fases del ciclo de vida de este tipo de sistemas CBR pueden llegar a ser automatizadas, eliminando así el factor humano. Corchado y Fdez-Riverola [14] son algunos de los autores que han trabajado con este tipo de modelos.

*3.5.3. Razonamiento Basado en Memoria (***Memory-Based Reasoning, MBR***)*

En este tipo de sistemas de razonamiento, la memoria se representa como una colección de casos, mientras que el razonamiento se corresponde con el proceso de recuperación de esos casos de la memoria [9]. Estos sistemas utilizan técnicas de procesamiento paralelo, y pueden ser utilizados tanto en problemas con fuertes connotaciones sintácticas como semánticas, tal y como se muestra en PARADYME [13].

3.5.4. Razonamiento Basado en Casos (Case-Based Reasoning, CBR)

Bajo este nombre se engloba al conjunto de los diferentes mecanismos de razonamiento presentados en esta clasificación [9]. Las características de un sistema típico de razonamiento basado en casos son las siguientes:

- Un caso típico debe de tener un cierto número de características distintivas y relevantes. Además, la estructura interna es normalmente compleja.

- Un sistema CBR típico, debe ser capaz de adaptar una solución recuperada al contexto en el que está inmerso el nuevo problema. Además, requiere cierto conocimiento general del problema. El tipo de conocimiento y el uso que se hace del mismo varía en cada caso.
- El algoritmo típico de un sistema CBR, se debe inspirar en las teorías desarrolladas en el ámbito de la psicología cognitiva.

3.5.5. Razonamiento Basado en Analogías (Analogy-Based Reasoning, ABR)

De esta forma se nombran a todos aquellos sistemas CBR que intentan resolver un nuevo problema utilizando casos antiguos provenientes de un dominio de conocimiento diferente. Por tanto, no actúa como un sistema CBR típico, que se basa en la resolución de problemas utilizando casos antiguos procedentes del mismo dominio [31]. Este término también se utiliza para nombrar las técnicas que se centran en el estudio de mecanismos para la identificación y utilización de analogías entre diferentes dominios de conocimiento. En este sentido, la investigación se ha centrado en la fase de reutilización y en el desarrollo de técnicas para proyectar problemas, o dicho de otro modo, transferir la solución de unos problemas a otros.

3.6. CBR COMO METODOLOGÍA PARA LA RESOLUCIÓN AUTOMÁTICA DE PROBLEMAS

La IA se describe a menudo desde el punto de vista de las diferentes tecnologías desarrolladas en las tres o cuatro últimas décadas de investigación. Tecnologías como programación lógica, razonamiento basado en reglas, redes neuronales, agentes y sistemas multi-agente, algoritmos genéticos, lógica difusa, programación basada en restricciones, etc. Estas tecnologías vienen caracterizadas por entornos o lenguajes de programación específicos (prolog, shells basados en reglas, etc.) o por técnicas o algoritmos concretos (A*, retropropagación, etc.). Cada una de estas tecnologías proporciona, en mayor o menor medida, diferentes formas o métodos de resolución de problemas (búsqueda primero en profundidad, generación y prueba, etc.) que mejor se adaptan a las características de cada tecnología.

Los sistemas de razonamiento basados en casos, han sido considerados desde su nacimiento como una tecnología equiparable a las comentadas anteriormente. Riesbeck y Schank [86] cuando definen un sistema CBR, indican que la resolución de nuevos problemas se realiza mediante la utilización o adaptación de soluciones dadas a problemas anteriores. Esta definición se centra en lo que el sistema CBR hace, pero no en cómo lo hace. Conceptualmente, un sistema CBR se describe como una secuencia cíclica de cuatro fases como la representada en la Figura 4, lo que de nuevo propone un esquema de funcionamiento para la resolución de problemas, pero sin realizar ninguna aportación acerca de las tecnologías a utilizar en cada fase.

Checkland y Scholes [87] describen el término metodología como un conjunto de principios organizados, que guían el manejo de situaciones problemáticas del mundo real. En este sentido, el ciclo de vida de un sistema CBR encaja perfectamente con la definición dada. En concreto, el principio fundamental que persigue un sistema CBR es el deseo de resolver un problema, utilizando situaciones pasadas. Para ello, el sistema CBR deberá recuperar casos de la memoria y de algún modo, estimar la similitud entre dichos casos y el presente problema. Después, tratará de reutilizar la solución sugerida por el conjunto de casos más similares recuperados y realizar una revisión de la misma. Por último, el sistema CBR debe incrementar su base de conocimiento añadiendo nuevos casos a su memoria.

Watson [88] muestran las implicaciones que supone la visión de sistemas CBR como una metodología, en contraposición con la idea tradicional de su interpretación como tecnología. La conclusión más importante que se extrae de sus trabajos, es que un sistema de razonamiento basado en casos puede utilizar cualquier tecnología existente, siempre y cuando se respeten los principios que define dicha metodología. En este sentido, Medsker [89] presenta una revisión de las diferentes tecnologías que se pueden utilizar en las distintas fases del ciclo de vida de un sistema CBR clásico:

- Los sistemas basados en conocimiento (KBS) se han utilizado en las fases de recuperación, adaptación y aprendizaje.

- Las redes neuronales (*Artificial Neural Networks - ANN*) en la representación y recuperación de casos.

- Los algoritmos genéticos (*Genetic Algorithms - GA*) se consideraron adecuados en la representación de casos.

- Los sistemas basados en reglas (RBS) se han empleado en la recuperación, adaptación y aprendizaje de casos.

- Técnicas de razonamiento cualitativo (*Qualitative Reasoning - QR*) fueron utilizadas en la fase de revisión.

- Sistemas difusos (Fuzzy Systems - FS) en la fase de recuperación de casos.

- Problemas de satisfacción de restricciones (Constraint Satisfaction Problems CSP), utilizados en las fases de adaptación y aprendizaje. Por lo tanto cada sistema CBR, debe construirse a medida del problema a resolver, puesto que en función del conocimiento disponible (en forma de datos o reglas) se seleccionarán unas u otras tecnologías para cada una de las fases.

Resulta inevitable, cuando se trabaja con este tipo de sistemas, plantearse las distintas posibilidades de interconexión de los mecanismos seleccionados para la ejecución de cada fase del ciclo de vida. En la siguiente sección se realiza un estudio más detallado de las distintas tecnologías utilizadas en las diferentes fases del ciclo de vida de un CBR.

3.7. TECNOLOGÍAS USADAS EN LA CONSTRUCCIÓN DE SISTEMAS CBR

A continuación se hace una revisión de las diferentes tecnologías que pueden formar parte de un sistema de razonamiento basado en casos.

3.7.1. Sistemas Expertos y CBR

Los sistemas expertos se han utilizado en colaboración con sistemas CBR en muchos casos y de forma muy variada. La sinergia existente entre ambas formas de razonamiento se debe a que comparten áreas similares de aplicación. Los dos modelos de IA son alternativas naturales, utilizadas de forma independiente para aprender y solucionar problemas, o conjuntamente, de tal forma que se puede utilizar un modelo para aprender acerca de un problema, y así implementar satisfactoriamente el sistema final utilizando el segundo modelo (por ejemplo a través del modelo transformacional). Algunos sistemas que presentan esta combinación son:

- Vo y Macchion [90] utilizan un sistema CBR para aprender más acerca de dominios de conocimiento y mejorar sistemas expertos utilizados en el desarrollo de satélites.
- Rissland et al. [91] utiliza sistemas expertos para mejorar la interacción entre un sistema CBR y el usuario, asegurando que los algoritmos de recuperación de datos reflejan las intenciones de los usuarios.
- Medsker [89] presenta una revisión de este tipo de sistemas.

3.7.2. Lógica Difusa y CBR

Autores como Medsker piensan que con sistemas construidos con estas dos tecnologías se pueden mejorar los resultados obtenidos con híbridos que contienen sistemas CBR y sistemas expertos. Lynn [92] ha investigado en áreas relacionadas con la predicción del SIDA (Síndrome de Inmunodeficiencia Adquirida). Su sistema utiliza reglas difusas para recuperar casos relevantes a la situación bajo estudio. En la actualidad, este campo está atrayendo el interés de muchos investigadores, como lo demuestra el número creciente de publicaciones realizadas en esta área. La utilización de reglas difusas basadas en similitud como una herramienta básica para modelar y formalizar la parte de inferencia de la metodología de razonamiento basado en casos, dentro del marco de razonamiento aproximativo, también está siendo bastante desarrollado. Fdez-Riverola [14] utiliza un sistema de reglas difusas en la fase de revisión de un sistema CBR. Su investigación se centra en modelos neuro-simbólicos para la predicción no supervisada de entornos cambiantes. Este modelo se aplica, dentro de su labor de investigación en la predicción de mareas rojas.

3.7.3. Algoritmos Genéticos y CBR

Louis et al. [93] utilizan un sistema CBR para explicar los resultados obtenidos con algoritmos genéticos. Oppacher y Deugo [94] han demostrado que los algoritmos genéticos pueden mejorar el aprendizaje en medios ruidosos.

3.7.4. Razonamiento Cualitativo y CBR

CADET [95] es uno de los sistemas CBR implementados más potentes y complejos que se conocen. Es un sistema especializado en problemas hidromecánicos. Tiene una memoria de diseños previamente realizados, la cual se usa para realizar diseños de elementos en nuevos artefactos. El módulo de evaluación consta de un sistema de razonamiento cualitativo y otro de control de condiciones.

3.7.5. Sistemas de Satisfacción de Restricciones y CBR

CADSYN [96] es un sistema de diseño estructural de edificios, que utiliza la combinación de un sistema CBR y un modelo de satisfacción de restricciones (CSP). En este contexto, el CSP se utiliza para transformar un diseño recuperado por el sistema CBR y adaptarlo al presente problema. JULIA [97] es un sistema de planificación de comidas, en el cual la adaptación de los casos se efectúa utilizando un CSP que identifica y resuelve incompatibilidades.

3.7.6. Redes Neuronales Artificiales y CBR

Las redes neuronales artificiales no son apropiadas para resolver problemas que requieran el uso de razonamientos escalonados [98]. Por otra parte, es muy difícil explicar su forma de razonamiento y justificar los resultados obtenidos. Sin embargo, sus capacidades de generalización y adaptación son muy interesantes y se pueden utilizar para resolver determinados problemas. Por el contrario, son apropiadas en áreas en las que sea necesaria la generalización de datos. Si los datos son adecuados y la red neuronal se entrena correctamente, el aprendizaje de la misma se realiza fácilmente. Esta propiedad de las redes neuronales puede resultar particularmente útil, ya que en los sistemas CBR el aprendizaje es parte intrínseca de su ciclo de vida. Así pues, las redes neuronales se pueden utilizar para aprender a recuperar los casos más adecuados de la memoria, o dicho de otra forma, para aprender a identificar la distancia más corta entre varios casos. Incluso, una red neuronal puede hacer que a un sistema CBR le resulte relativamente fácil proporcionar nuevas soluciones, por generalización a partir de casos conocidos y adaptándolos a situaciones presentes.
Las redes neuronales artificiales son aconsejables en problemas de tipo numérico, mientras que los sistemas CBR proveen una metodología orientada a tratar con conocimiento simbólico. Aunque el conocimiento simbólico se puede explicitar numéricamente y viceversa, hay que considerar que la transformación supone siempre un riesgo de pérdida de información y precisión. La utilización de redes neuronales como tecnología en el ciclo de vida de un sistema CBR, supone la eliminación de una transformación de este tipo y un aumento de precisión de los resultados. Como se mencionó anteriormente, la capacidad de generalización es una excelente característica de las redes neuronales, pero en algunas ocasiones es conveniente mantener información relativa a casos concretos y ésta es una capacidad natural de los sistemas CBR.

En la literatura se describen varios ejemplos de sistemas CBR, que incorporan redes neuronales en alguna etapa de su ciclo de vida [14]. La forma más común de relacionar redes neuronales y CBR es aquella en la que los casos forman parte de una red neuronal simbólica, o dicho de otra forma, un sistema CBR conexionista. De esta manera, el sistema CBR puede usar satisfactoriamente los casos en la etapa de recuperación y durante la indexación. Por ejemplo, en el sistema creado por Tirri [99] los casos están contenidos en las neuronas del nivel intermedio, creando así una red neuronal simbólica. El sistema CBR hace uso de probabilidades bayesianas y está implementado como si se tratara de una red conexionista, que utiliza dichas

probabilidades para determinar la activación entre las neuronas y proporcionar una explicación teórica de las soluciones dadas.

Fdez-Riverola [14] propone la utilización de una red neuronal tanto en la fase de recuperación, como en la fase de adaptación del ciclo de vida de un sistema CBR. El trabajo de este autor ha sido una gran referencia en el desarrollo de la presente investigación, los resultados obtenidos en su trabajo de investigación con la utilización de la red neuronal GCS en la fase de recuperación, motivaron que se adecuara para utilizarse en la fase de recuperación del modelo de razonamiento de un agente BDI.

Shinmori [100] propone la utilización de redes bayesianas en la fase de recuperación de casos en un sistema CBR, para la implementación de un sistema prototipo interactivo que asista en la resolución de problemas software, a la hora de instalar y trabajar con dichas aplicaciones. El sistema se basa en palabras clave, que un usuario emplea a la hora de intentar encontrar ayuda ante un problema. El objetivo de las redes bayesianas es reformular la petición del usuario, para generar una nueva consulta que recupere casos con información relevante del error soportado.

Los últimos avances en este campo, se centran en la utilización de redes de creencia o redes bayesianas en la fase de indexación y recuperación de casos, integrando conocimiento concreto obtenido en situaciones pasadas, con conocimiento general de un dominio particular. Habitualmente, estos sistemas están parcialmente acoplados, donde el CBR controla el sistema gracias al ciclo de vida que propone la metodología y utiliza la red bayesiana como soporte en alguna de sus fases.

4. SISTEMAS MULTI-AGENTE Y SOCIEDADES DE AGENTES

4.1. INTRODUCCIÓN

La teoría de agencia, los agentes y sistemas multi-agente son una de las áreas de la IA con más auge en la última década. El interés que suscita se debe a que es un punto de convergencia entre distintas áreas de conocimiento como son la inteligencia artificial, la informática distribuida, la psicología, la sociología, la robótica, etc. Esto implica la confluencia de investigadores de distintos campos, lo que provoca un enfoque multidisciplinar [8].

Los sistemas multi-agente son la mejor aproximación para caracterizar o diseñar sistemas de computación distribuidos. En este capítulo se muestran algunos aspectos fundamentales sobre los agentes y se hace una revisión de su aplicación en el contexto de los Sistemas Tutores Inteligentes.

4.2. DEFINICIÓN DE AGENTE

Se pueden encontrar propuestas en la literatura de un gran número de definiciones del concepto de agente, sin que ninguna de ellas haya sido plenamente aceptada por la comunidad científica, siendo quizás la más simple la que considera un agente como una entidad que percibe y actúa sobre un entorno. En la figura 5 se muestra el esquema de un agente inteligente.

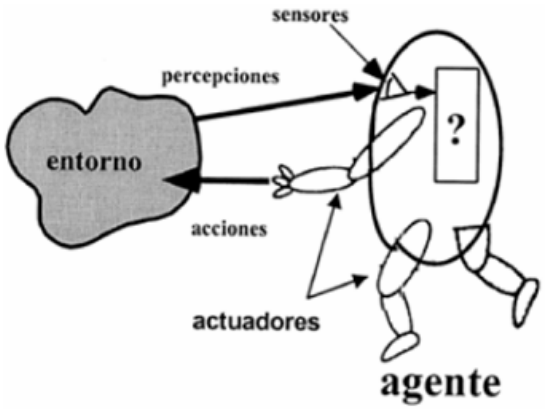


Figura 5. Visión esquemática de un agente inteligente

Basándose en esta definición, se pueden caracterizar distintos agentes de acuerdo a los atributos que posean (y que van a definir su comportamiento) para resolver un determinado problema. Quizás esta definición es muy básica y puede producir un efecto contenedor en el sentido de que es una definición muy amplia y numerosos sistemas pueden ser etiquetados como agentes cuando realmente no lo son o por lo menos no deberían haber sido enfocados desde este punto de vista. En este sentido la Figura 6 muestra las características de un agente, las cuales sirven a su vez de base para definir un modelo genérico de agente. De todas las características mencionadas en la figura, las principales son autonomía, sociabilidad, reactividad e iniciativa (proactividad). Muchos sistemas software ("demonios", servidores de páginas web, etc) podrían satisfacer algunas de ellas, por ello son necesarias otras propiedades más específicas que determinen el concepto de agente: aprendizaje, veracidad, cooperación y racionalidad.

Figura 6. Características de los agentes

Un agente autónomo actúa sin la intervención directa de las personas y tiene algún tipo de control sobre sus acciones y estado interno [8]. La sociabilidad establece que los agentes interactúen con otros agentes mediante algún mecanismo de comunicación. Un agente reactivo percibe su ambiente y responde a los cambios de éste. También son capaces de orientar su comportamiento hacia metas al tener iniciativa.

Un agente para ser inteligente necesita aprender del ambiente que le rodea. No debe comunicar de forma deliberada información falsa (agente veraz). Como consecuencia de la sociabilidad, la cooperación entre agentes es una razón de ser para tener múltiples agentes encargados de resolver un determinado problema. Un agente racional debe realizar todas aquellas acciones que favorezcan la obtención del máximo rendimiento, basándose en las evidencias aportadas por la secuencia de percepciones y en todo su conocimiento incorporado.

Un agente va a venir caracterizado por una serie de calificativos, los cuales vienen a denotar ciertas propiedades a cumplir por el agente. Esto lleva a plantear otra definición bastante aceptada de agente donde se emplean tres calificativos que se consideran básicos [8]. Esta definición ve a un agente como un sistema de computación capaz de actuar de forma autónoma y flexible en un entorno, entendiendo por flexible que sea:

- **Reactivo**: el agente es capaz de responder a cambios en el entorno en que se encuentra situado.
- **Pro-activo**: a su vez el agente debe ser capaz de intentar cumplir sus propios planes u objetivos.
- **Social:** debe de poder comunicarse con otros agentes mediante algún tipo de lenguaje de comunicación de agentes.

Pero, en definitiva, ¿cuáles son las características básicas y de qué mas características disponemos para poder calificar a un agente?. Algunas de las características que en la literatura se suelen atribuir a los agentes en mayor o menor grado para resolver problemas particulares son:

- **Continuidad Temporal:** se considera un agente un proceso sin fin, ejecutándose continuamente y desarrollando su función.
- **Autonomía:** un agente es completamente autónomo si es capaz de actuar basándose en su experiencia. El agente es capaz de adaptarse aunque el entorno

cambie severamente. Por otra parte, una definición menos estricta de autonomía sería cuando el agente percibe el entorno.

- **Sociabilidad:** este atributo permite a un agente comunicar con otros agentes o incluso con otras entidades.
- **Racionalidad:** el agente siempre realiza «lo correcto» a partir de los datos que percibe del entorno.
- **Reactividad:** un agente actúa como resultado de cambios en su entorno. En este caso, un agente percibe el entorno y esos cambios dirigen el comportamiento del agente.
- **Pro-actividad:** un agente es pro-activo cuando es capaz de controlar sus propios objetivos a pesar de cambios en el entorno.
- **Adaptatividad:** está relacionado con el aprendizaje que un agente es capaz de realizar y si puede cambiar su comportamiento basándose en ese aprendizaje.
- **Movilidad:** capacidad de un agente de trasladarse a través de una red telemática.
- **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.

En un sistema multi-agente diversos agentes autónomos trabajan juntos para resolver problemas [15]. En este tipo de sistemas cada agente tiene una información o capacidad incompleta para solucionar un problema, no hay un sistema global de control, los datos están descentralizados y la computación es asíncrona [101].

Se han identificado seis problemas inherentes al diseño e implementación de agentes y sistemas multi-agente:

1. Cómo formular, describir, descomponer, asignar problemas y sintetizar resultados entre grupos de agentes inteligentes.
2. Cómo lograr la comunicación e interacción entre agentes. Qué lenguajes y protocolos de comunicación utilizar para el desarrollo e implementación de sistemas multi-agente.
3. Cómo asegurar que los agentes actúan coherentemente en la toma de decisiones o en la ejecución de acciones, de forma tal que se eviten interacciones dañinas debidas a decisiones particulares que pudiesen afectar decisiones globales.
4. Cómo permitir que agentes individuales representen y razonen sobre los planes, acciones y conocimientos de otros agentes con el propósito de coordinarse con ellos.
5. Cómo reconocer y reconciliar perspectivas opuestas e intenciones en conflicto entre agentes tratando de coordinar sus acciones.
6. Cómo diseñar y desarrollar sistemas distribuidos prácticos. Cómo diseñar plataformas tecnológicas y metodologías de desarrollo para sistemas multi-agente.

Estos problemas han centrado la investigación en el campo de los agentes y sistemas multi-agente en tres áreas diferentes: teoría, arquitecturas y lenguajes de agentes. La teoría de agentes se refiere a cómo conceptualizar agentes, qué propiedades deberían tener, y cómo se puede representar y razonar sobre dichas propiedades. La arquitectura define cómo se pueden construir agentes que satisfagan las propiedades especificadas por la teoría de agentes, y qué estructuras de software o hardware son apropiadas. Los lenguajes de agentes deben desarrollar los principios propuestos por la teoría de agentes, de tal forma que permitan la programación de dichos agentes.

En nuestro trabajo, el uso de agentes inteligentes beneficia los procesos de formación y apoyo de profesores y alumnos. La definición de determinados grupos de agentes que soporten las fases del ciclo CBR son un aporte fundamental, sobre todo en las fases de revisión y adaptación, las cuales en muchas ocasiones se suelen obviar , y son realizadas con intervención humana dada su alta complejidad.

4.3. CLASIFICACIÓN DE AGENTES

Los agentes software existen en un espacio multidimensional, existiendo por lo menos cuatro dimensiones posibles como: cuantitativa (*quantity*), movilidad (*mobility*), sensibilidad (*sensitivity*), y comportamiento (*behavioral*). En la figura 7 se ilustra la clasificación:

Figura 7. Clasificación de agentes software

4.3.1. Comportamiento

Los agentes se pueden clasificar en tres características primarias como son: autonomía, aprendizaje y cooperación.

Autonomía: Se refiere al principio de que los agentes pueden operar por si mismos sin necesitar la guía humana. Un elemento clave de su autonomía es su proactividad.

Aprendizaje: Se refiere a la habilidad de un agente para aprender como reaccionar, y/o interactuar con su ambiente externo y modificar su comportamiento.

Cooperación: La cooperación entre múltiples agentes denota la habilidad para interactuar con otros agentes y posibles humanos usando un lenguaje de comunicación, coordinando para mejorar su habilidad.

4.3.2. Sensibilidad

Teniendo en cuenta la forma como los agentes responden o reaccionan ante una petición, pueden ser clasificados en deliberativos o reactivos.

Deliberativos: Estos agentes, derivan del paradigma de pensamiento deliberativo. Contienen un modelo simbólico del mundo representado internamente de forma explicita y que toma decisiones mediante razonamiento lógico.

Reactivos: Estos agentes no incluyen un modelo lógico del mundo, ni usan razonamiento simbólico complejo.

4.3.3. Cuantitativa

Numerosas tecnologías existentes pueden ser clasificadas como agentes que trabajan individualmente, y sistemas multi-agentes. En sistemas que trabajan con agentes individuales, un agente desempeña una tarea en nombre del usuario o de cierto proceso. Mientras ejecuta su tarea, el agente puede comunicarse con el usuario, y con los recursos del sistema ya sean locales o remotos; pero nunca se comunica con otros agentes.

A diferencia de los agentes individuales, los sistemas multi-agente, además de comunicarse con el usuario y los recursos, también se comunican con otros agentes, resolviendo problemas que están más allá de sus capacidades individuales.

4.3.4. Movilidad

Los agentes se pueden clasificar también por su movilidad (e.j. Su habilidad para moverse alrededor de las redes). Los agentes estáticos no tienen la capacidad de moverse de una máquina a otra, pero puede acceder a los recursos de red para acceder a recursos remotos.
El agente móvil, tiene la capacidad de moverse entre una red heterogénea bajo su propio control, migrando de servidor en servidor e interactuando con otros agentes y recursos, típicamente retornando a su servidor de origen cuando la tarea esta hecha. Los agentes móviles son un buen paradigma para aplicaciones distribuidas y un excelente paradigma cuando están involucradas computadoras móviles

4.4. SISTEMAS MULTI-AGENTE

Los sistemas multi-agente se componen de entidades software independiente (agentes), las cuales colaboran de forma coordinada en dominios complejos del mundo real [102]. El comportamiento óptimo y robusto se consigue a través de las interacciones entre agentes software inteligente y autónomo. El desarrollo de módulos separados, donde cada uno proporciona una solución, les permite cooperar e intercambiar información para resolver problemas mayores, lo que hace que el proceso de resolución de problemas sea sencillo de manejar.

Los agentes MAS poseen una serie de características comunes con la noción de agente inteligente:

- Capacidad de tomar la iniciativa,
- Capacidad de compartir conocimiento,
- Capacidad de cooperar y negociar,
- Capacidad de comprometerse con metas comunes.

Adicionalmente, en los sistemas multi-agente aparecen conceptos de gran interés como:

- Orientación a actividades conjuntas y cooperación;
- Necesidad de establecer mecanismos cooperativos de resolución de conflictos;
- Necesidad de establecer mecanismos de negociación;
- Establecimiento cooperativo de compromisos y planificación de actividades;
- Necesidad de establecer modelo del conocimiento, de su comunicación y de su adquisición que pueda abstraerse del entorno cooperativo que caracteriza a MAS.

4.5. COMUNICACIÓN ENTRE AGENTES

La comunicación es un requisito clave para la construcción de sistemas multi-agente. Los agentes se comunican con el objetivo de alcanzar más eficientemente sus metas o las de la sociedad/sistema en los cuales ellos existen. Este proceso de comunicación, habilita a los agentes a coordinar sus acciones y su comportamiento, obteniendo como resultado sistemas más coherentes. En la figura 8, se presenta una clasificación de algunas formas utilizadas por los agentes para coordinar su comportamiento y actividades.

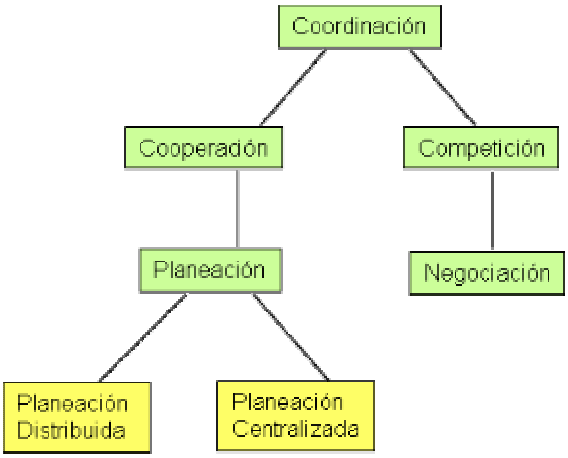


Figura 8. Clasificación de formas de coordinación entre los agentes

4.6. SISTEMAS MULTI-AGENTE EN SISTEMAS TUTORES INTELIGENTES

La utilización de arquitecturas multi-agente en sistemas tutores posee una gran ventaja en relación a las arquitecturas tradicionales de ITS, ya que presentan mayor flexibilidad en el tratamiento de los elementos que componen el sistema. Adicionalmente, el uso de agentes para modelar los componentes del ITS posibilita el agrupamiento de una arquitectura tradicional (un módulo = un agente) en un conjunto de agentes, como se puede ver en el trabajo [103].

Según Claude Frasson [104] una de las aplicaciones más promisorias de los agentes autónomos hace referencia a la educación. Existen diversos ejemplos en la literatura sobre la utilización de agentes en sistemas educativos. Según Shoham [105], una sociedad de agentes para aprender a enseñar, puede ser la solución para construir ambientes de enseñanza-aprendizaje; siempre que esos agentes actúen de forma concurrente o autónoma para alcanzar sus objetivos. Los agentes en un ambiente de

enseñanza aprendizaje son considerados autónomos porque: (a) las actividades de los agentes individuales no requieren constante supervisión humana, (b) no hay una autoridad central proyectada para controlar todas las interacciones desempañadas entre los agentes.

El uso de agentes en la concepción de sistemas educativos proporciona innumerables ventajas, tales como: (a) permitir reaccionar ante las acciones del usuario, (b) proveer credibilidad, (c) permitir el modelamiento de sistemas colaborativos multi-usuario y modularidad. Adicionalmente, otras ventajas pueden ser consideradas:

- El conocimiento puede ser distribuido entre varios "tutores" cada uno con sus creencias, deseos, objetivos, emociones y planos de acción. Esta distribución crea mayores oportunidades para diferentes estrategias pedagógicas.
- El estudiante puede interactuar con el tutor de forma más eficiente.
- El estudiante puede comunicar los conocimientos a su tutor, los cuales pueden ser almacenadas para ser utilizados por otros nuevos estudiantes.

Teniendo en cuenta a [106], las tecnologías basadas en Web en conjunto con las metodologías multi-agente forman una nueva tendencia en el modelamiento y desenvolvimiento de ambientes de aprendizaje. La educación basada en Web provee grandes beneficios. Como por ejemplo: mayor alcance de información, no existe condicionamiento del espacio físico, facilidad de actualización de contenidos, etc. De esta forma, las metodologías multi-agentes han surgido para concebir aplicaciones de aprendizaje distribuido debido a un conjunto de características inherentes al concepto de MAS y a las peculiaridades de una sociedad de agentes.

En [107], se afirma, que en el futuro los ambientes de aprendizaje estarán disponibles en cualquier lugar y a cualquier hora. Los estudiantes de dichos ambientes estarán distribuidos en el espacio y en el tiempo. De esta forma, los trabajos que involucren el uso de arquitecturas multi-agente ofrecen un promisorio abordaje ya que permiten que estos ambientes sean distribuidos. Adicionalmente, la modularidad y uniformidad de los agentes, permite mayor escalabilidad e interoperabilidad lo que se reflejará en mejores resultados. Actualmente, no existe gran cantidad de ambientes distribuidos de aprendizaje, ni arquitecturas multi-agentes que permitan constante crecimiento y heterogeneidad del ambiente software. Esto debido a:

- La complejidad en modelar e implementar ITS en ambientes de enseñanza-aprendizaje inteligentes. Su arquitectura es mas compleja y el modelado de sus componentes e interrelaciones es demorado, necesitando de una gran cantidad de trabajo colaborativo en un equipo interdisciplinario.
- La tecnología de agentes aplicada a estos ambientes, agrega más complejidad al sistema y puede ser considerada de muy reciente aplicación. Los trabajos de investigación de los últimos cinco años pueden ser encontrados en [108].

Los principios de sistemas multi-agente han mostrado un potencial bastante adecuado en el desarrollo de sistemas de enseñanza, debido al hecho de que la naturaleza del problema de enseñanza-aprendizaje es mas fácilmente resuelto de forma cooperativa. Además se puede decir que los ambientes de enseñanza basados en arquitecturas multi-agente posibilitan el desarrollo de sistemas tutores mas robustos, de forma mas rápida y con menores costos.

4.6.1. Sistemas Tutores desarrollados con tecnología de agentes

En esta sección se describen algunos ITS desarrollados usando tecnologías de agentes.

4.6.1.1. White Rabbit

White Rabbit [109] es un sistema que tiene como objetivo aumentar la cooperación entre un grupo de personas, por medio del análisis de sus conversaciones. Cada usuario es asistido por un agente inteligente el cual establece un perfil de sus intereses. Con un comportamiento móvil o autónomo, el agente investiga agentes personales de otros usuarios con el fin de encontrar aquellos que tengan intereses comunes y finalmente ponerlos en contacto. Se utiliza un agente mediador para la comunicación entre los agentes personales y para realizar agrupamientos de perfiles. En este proyecto, se usan agentes inteligentes para descubrir los intereses particulares de un grupo de personas trabajando en un dominio particular con la intención de ponerlos en contacto y aumentar el nivel de cooperación. Los agentes analizan la conversación entre los usuarios a través de un chat, para descubrir los perfiles de su interés.

4.6.1.2. LeCS

LeCS (Learning from Case Studies) es un sistema inteligente de aprendizaje a distancia el cual posee una arquitectura basada en un sistema federativo de agentes. LeCS da soporte al aprendizaje colaborativo a través del web usando un método de estudio de casos. El escenario de aprendizaje para el sistema son un grupo de alumnos geográficamente dispersos cursando una disciplina específica.

4.6.1.3. Lanca

En el proyecto LANCA [104] los autores procuran exponer la importancia del uso de agentes inteligentes en ITS y como pueden ser adaptados para aprendizaje a distancia. Lanca presenta una arquitectura de agentes cognitivos y su funcionalidad en ambientes distribuidos. Dentro del conjunto de agentes se encuentra: un agente que supervisa el aprendizaje local, un agente pedagógico que utiliza un analizador de soluciones para descubrir las respuestas, y un agente de dialogo que ofrece explicaciones adicionales a los estudiantes, además de un conjunto de estrategias de enseñanza como son:

Libro: Conjunto de índices del curso, relacionado con el entendimiento de conceptos específicos.
Tutor: Una estrategia dirigida, que provee respuestas y consejos según los requisitos del estudiante
Compañero: Colega virtual, capaz de discutir, generar preguntas, enfocando la atención en puntos específicos.
Creador de casos: Compañero particular que involucra al estudiante en la proposición de soluciones que ayuden a aumentar su motivación.

4.6.1.4. Baguera

El proyecto Baguera [106] fue creado con el objetivo de desarrollar una fundamentación teórica y metodológica que guíe la concepción y modelamiento de

ambientes de aprendizaje. La plataforma Baguera está fundamentada en el principio de que una función educativa del sistema son las interacciones establecidas entre los componentes: agentes y humanos, y no solamente en la funcionalidad de una de sus partes. El primer resultado de este proyecto incluye un prototipo de una arquitectura multi-agente para aprendizaje en geometría. Cada agente fue extendido por un módulo de interacción que favorece el soporte para gerenciamiento de protocolos entre los agentes. Los agentes poseen habilidad para comunicarse, razonar y tomar decisiones. La arquitectura de la plataforma es concebida por la metodología AEIO (Agent, Environment, Interactions, Organization), una metodología para análisis de proyectos orientados a agentes. Como resultado de estos procesos los estudiantes y profesores interactúan con un grupo de agentes como son: tutor, mediador, profesor, y asistente.

4.6.1.5. I-Help

I-Help [107] ofrece una infraestructura multi-agente en un ambiente de aprendizaje basado en web para estudiantes auxiliares en la solución de problemas. El sistema contiene una variedad de recursos de aprendizaje, foros, materias on-line, chat, etc. El sistema está basado en una arquitectura multi-agente, integrada por agentes personales (usuarios, humanos) y agentes de aplicación. Estos agentes usan una ontología y un lenguaje de comunicación común. Cada agente controla recursos específicos de usuario (o de la aplicación) que representa, incluyendo por ejemplo, los conocimientos del usuario sobre determinados conceptos, y otros materiales instructivos que pertenecen a una aplicación. Los agentes usan sus recursos para conseguir objetivos de sus usuarios, sus propios objetivos, y los objetivos de otros agentes. Todos los agentes son autónomos. Cada agente posee un modelo de usuario de otros agentes y se comunican por medio de combinadotes para encontrar recursos apropiados para sus usuarios, dependiendo del tipo de ayuda requerida.

Teniendo en cuenta que las arquitecturas multi-agente involucran varios niveles de organización, incluyendo la negociación entre los agentes, se logra de esta forma conseguir sistemas distribuidos, multiusuario, multiaplicación, y auto-organizados que soportan la ubicación de recursos de ayuda (otros usuarios, aplicaciones, información)

4.6.1.6. The explanation agent

El agente de explicación (explanation agent) tiene como principal objetivo proveer respuesta y explicaciones sobre contenidos, con mayor calidad, identificando problemas que pasan durante el proceso de explicación de resolución de problemas. Este agente tiene dos objetivos específicos: (1) descubrir la fuente de mal entendimiento del estudiante a través del modelo del estudiante, (2) ayudar al proyectista del curso a adaptar sus explicaciones de acuerdo a estas operaciones. Este agente utiliza teorías de mapas conceptuales para estructurar las explicaciones en una representación formal. Esta representación es usada por los agentes de explicaciones para hacer sus deducciones sobre conceptos mal entendidos por el aprendiz.

4.6.1.7. AME-A

AME-A [110] es un ambiente multi-agente de enseñanza-aprendizaje en el cual se propone un estudio para del desarrollo de un sistema educativo interactivo para enseñanza-aprendizaje. La propuesta de enseñanza genérica se adapta a las características psico-pedagógicas del estudiante. Las características principales de este sistema son el aprendizaje estático y dinámico. El aprendizaje estático

corresponde a la primera interacción del estudiante con el ambiente, donde un agente modela un estudiante según sus características afectivas, motivacionales o de conocimiento. El aprendizaje dinámico ocurre durante la interacción, cuando el modelo del alumno es validado y se establecen un conjunto de estrategias pedagógicas.

Este ambiente utiliza sistemas multi-agente, en donde cada uno de los agentes actúa concurrentemente realizando sus tareas o intercambiando mensajes entre si, con el objetivo de que el estudiante tenga un aprendizaje efectivo. Las características psico-pedagógicas son relevantes para la enseñanza adaptada y son viables para la representación del material de enseñanza de una manera individualizada.

En AME-A [110] cada agente es responsable por sus tareas con la finalidad de cooperar para promover un aprendizaje inteligente y adaptado a las características de aprendizaje.

4.5.1.7. Electrotutor

Las primeras versiones de Electrotutor funcionaban como tutorial Web. Electrotutor III es una nueva versión, que implementa un ambiente distribuido de enseñanza-aprendizaje inteligente (Intelligent Learning Environment - ILE) basado en una arquitectura multi-agente, en la cual los agentes poseen las siguientes características:

- Percibir dinámicamente las condiciones del ambiente
- Tomar decisiones para modificar las condiciones del ambiente
- Interpretar percepciones
- Resolver problemas
- Extraer inferencias y determinar acciones

El ambiente de Electrotutor aborda un contenido constituido por algunos capítulos de electrodinámica, un capítulo de física que estudia algunos fenómenos de electricidad y aborda las relaciones entre algunos tópicos de electricidad.

La sociedad de agentes esta compuesta por siete agentes cada uno de los cuales posee una función específica y su objetivo principal es el aprendizaje del alumno.
En este ambiente es de vital importancia la coordinación del comportamiento de los agentes de forma que compartan sus objetivos, habilidades, para en conjunto tomar acciones y resolver un problema. Con el objetivo de actuar sobre el ambiente, cada uno de los agentes posee una representación interna parcial del mundo que los rodea. Para esto se emplea una metáfora de estados mentales para modelar la base de conocimientos que representa los estados del ambiente donde el agente esta inmerso.
La sociedad de agentes propuesta contempla agentes autónomos que se comunican entre si con otros, cada agente posee funciones y objetivos dentro de su especialidad.
El ambiente de aprendizaje propuesto contempla un agente responsable para la recuperación de conocimiento del dominio sobre cada punto a ser representado por el alumno, agentes responsables en la tarea de proposición de ejercicios y validación de respuesta.

5. ARQUITECTURA DEL SISTEMA

Este capitulo presenta una descripción detallada de la arquitectura propuesta la cual toma como referencia las ideas en las que se fundamentan los Sistemas Tutores Inteligentes, y presenta una nueva aproximación que involucra el uso de Razonamiento Basado en Casos para la implementación del ITS.

5.1. DESCRIPCIÓN DE LA ARQUITECTURA

El Sistema Tutor Inteligente es implementado como un sistema CBR bajo una aproximación multi-agente, como se muestra en la figura 9.

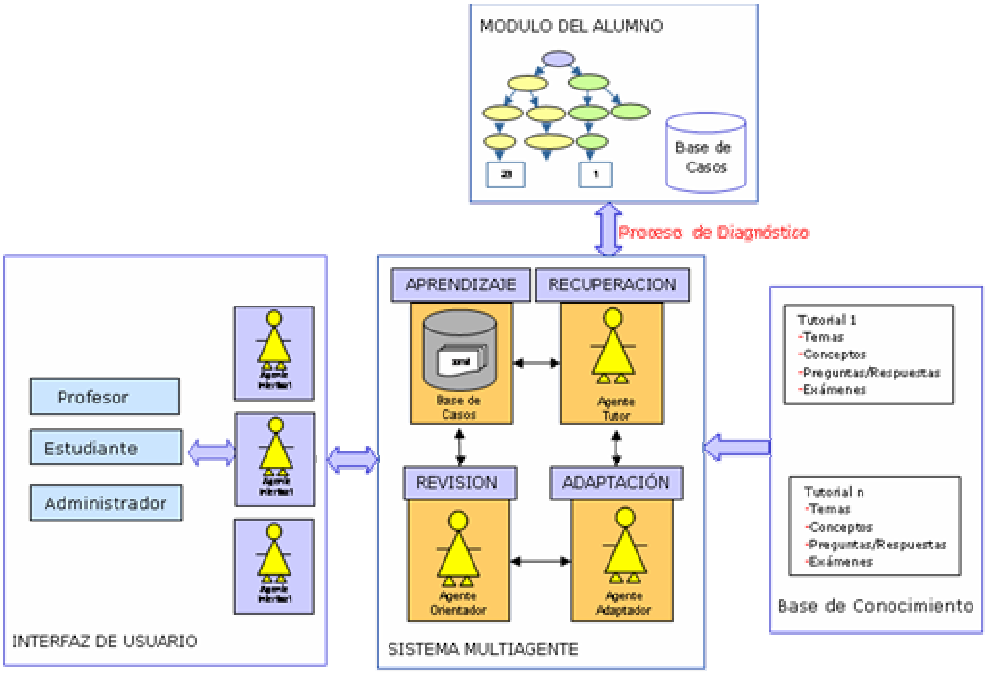


Figura 9. Arquitectura ITS-CBR

La aproximación propuesta está compuesta por un sistema de razonamiento basado en casos, que encapsula un sistema multi-agente. El sistema híbrido así definido constituye un mecanismo autónomo. A continuación se hace una descripción de los componentes de la arquitectura propuesta:

5.1.1. Módulo Tutor

Las tareas del modulo tutor bajo nuestra aproximación son desempeñadas por un Sistema Multi-agente. Las propias características que brindan los agentes, la versatilidad a la hora de combinar e integrar los agentes en sistemas de razonamiento y aprendizaje y sobre todo la posibilidad de utilizar diferentes tipos de agentes para realizar las distintas tareas que desempeña el módulo tutor, son las principales razones que han influenciado en la decisión de elegir como módulo tutor un sistema multiagente.

Las tareas fundamentales del sistema multi-agente son las siguientes:

- Dar soporte al proceso de diagnóstico del alumno
- Monitorizar el comportamiento del alumno durante el proceso de aprendizaje.
- Evaluar y calcular el grado de conocimiento adquirido por el alumno en cada uno de los conceptos del tutorial
- Aconsejar al alumno sobre las decisiones que éste toma a lo largo del proceso de aprendizaje del tutorial.

Estas tareas son llevadas a cabo por tres tipos de agentes existentes en el sistema multi-agente. Estos tipos de agentes son el agente tutor, el agente de adaptación y el agente orientador. A continuación se describen en detalle las funciones que desempeñan estos dos tipos de agentes.

1. Agente Tutor: Este agente inicia la fase de recuperación del sistema CBR. Cada vez que un nuevo estudiante ingresa al sistema este corresponderá a un nuevo caso. El agente tutor, tiene como función recuperar de los casos existentes los que mejor se ajusten al perfil del nuevo alumno que quiere utilizar el sistema de tutorización. El objetivo es seguir los pasos de tutorización que se han utilizado para otros alumnos.

5.1.1.1. Fase de Recuperación de Casos (*Retrieve Cases*)

Dada la descripción de un nuevo problema la primera etapa que un sistema de razonamiento basado en casos debe llevar a cabo es la Recuperación. En nuestro contexto el agente tutor recupera de la base de casos las situaciones pasadas más similares al problema actual (o instancia-problema). Para la recuperación existe un conjunto de técnicas que se circunscriben dentro de los siguientes modelos:

Similitud: Dado un nuevo problema, la relevancia de los casos almacenados en la memoria del sistema CBR, se calcula en base a alguna medida de distancia o similitud predefinida (*k-nearest neighbord*) entre las variables que describen el nuevo problema.

Indexado: Parten de una estructura subyacente que interconecta todas las instancias almacenadas en la memoria del sistema CBR, agrupando aquellas más similares. El cálculo de la relevancia, se determina mediante un procedimiento que recorre la estructura de indexación.

Aproximaciones híbridas: Constituyen una combinación de los dos modelos anteriores.

La regla de los k-vecinos más próximos, conocida comúnmente como modelo de recuperación k-NN (k-Nearest Neighbour) [111] es la escogida en este trabajo por su simplicidad y facilidad de modificaciones y adiciones. Este sistema de recuperación permite determinar los casos más relevantes a uno dado (mediante el cálculo de distancias) que son utilizados posteriormente como base del proceso de razonamiento. El modelo k-NN depende directamente de la ponderación de los pesos usados para calcular la similitud entre casos. La ecuación 2, muestra la formula para el cálculo de la similaridad de los casos.

$$\frac{f(x) = \sum_{i=1}^{n} W_i (Sim(f_i^{\,1}, f_i^{\,r})}{\sum_{i=1}^{n} W_i}$$

Ecuación 2. Cálculo de la similaridad entre casos

Donde:

$W_i =$ Importancia de la dimensión i (Weight)
$Sim =$ Función de similaridad
$f_i^1 and f_i^r =$ Valores para la característica fi en cada entrada.

2. Agente Adaptador: Este agente combina el nuevo caso, con el caso (s) similar recuperado, en orden a obtener la estrategia a seguir. Una vez la estrategia es confirmada, se organizan los recursos para ser presentados al estudiante. La estrategia utilizada se va modificando dependiendo de cómo avance el alumno actual.

5.1.1.2. Fase de Adaptación (*Reuse Cases*)

En general, la adaptación es una de las fases más problemáticas en los distintos tipos de sistemas CBR, especialmente si el problema con el que se trata es complejo, dinámico y heterogéneo, como el que se persigue resolver en este trabajo. Dado un conjunto de casos similares (Caso = Descripción + Solución) a un problema dado, la segunda etapa que un sistema de razonamiento basado en casos debe llevar a cabo, es la de **A**daptar ese subconjunto de casos con el fin de construir una solución inicial al problema. La mayoría de las técnicas de adaptación se basan en heurísticas de generalización y refinamiento [112].

Una vez se recupera el caso, la solución debe adaptarse. La adaptación se fija en las diferencias entre los casos y aplica reglas de adaptación. La adaptación puede ser:

Estructural: La adaptación con reglas es sobre la solución.

Derivacional: Reutiliza los algoritmos, métodos o reglas usados para generar la solución (la solución debe guardarse con la secuencia o plan utilizado), conocida también como reinstanciación.

Los algoritmos mas usados en la etapa de adaptación son los k-NN. Este tipo de mecanismo resulta idóneo en problemas para los que se pueden identificar reglas o patrones de comportamiento estándar. En este trabajo de tesis, el agente adaptador utiliza como técnica alternativa el algoritmo del K- vecino más cercano.

3. Agente Orientador: Este agente revisa cada paso que el estudiante realiza en el tutorial. Para ello utiliza un sistema de evaluación el cual, además de evaluar los conocimientos que va adquiriendo el alumno, también evalúa el sistema para saber si le está aconsejando bien.

5.1.1.3. Fase de Revisión y Aprendizaje (*Revise and Retain Cases*)

Dada una solución inicial a un nuevo problema, la tercera etapa que un sistema de razonamiento basado en casos debe llevar a cabo, es la de revisar dicha solución, en función del conocimiento disponible en el sistema acerca del dominio del problema. La fase de revisión ha sido la etapa, de todo el ciclo de vida de los sistemas CBR, que históricamente más ha costado automatizar [113]. Las características propias de los problemas a resolver, así como el tipo de técnicas utilizadas en esta etapa, han hecho que los modelos utilizados en esta fase fueran dependientes, en mayor o menor medida, del dominio del problema, y que en la mayoría de los casos tuviera que ser un experto humano el que refrendara la solución final propuesta por el sistema.

La fase de revisión se puede automatizar utilizando diferentes técnicas de simulación [77], técnicas de Belief-Revision [114], un sistema difuso, o simplemente con reglas que describen el comportamiento de un experto humano.

La última etapa del ciclo de vida de un sistema de razonamiento basado en casos, es la incorporación a la memoria del sistema CBR de lo aprendido tras resolver un nuevo problema. Cuando se dispone del valor real a un problema planteado al sistema con anterioridad, se construye un nuevo caso (Descripción del problema + Solución + Resultado), que se almacena en la memoria del sistema CBR. Aparte de la actualización global de conocimiento que supone la inserción de un nuevo caso en la memoria del sistema CBR, el sistema propuesto debe realizar una adaptación local de las estructuras de conocimiento que mantiene cada una de las partes que lo componen.

Cada una de las tecnologías seleccionadas como mecanismos de recuperación y adaptación del sistema CBR, modifican su estructura en función del error obtenido tras la aplicación de la solución proporcionada por el sistema.

Las fases de revisión y aprendizaje son las más críticas dentro del sistema CBR. La revisión mediante un agente orientador es uno de nuestros principales aportes.

5.1.2. Modulo del Alumno

Este módulo se estructura como un *sistema multi-agente* y representa el conocimiento que el alumno tiene sobre el dominio que intenta aprender. Está formado por el *modelo del alumno* y el *proceso de diagnóstico*. En la figura 10 se presenta el modelado del alumno.

En el modelado del alumno, hay un *agente interfaz* por cada estudiante. Este agente es responsable en asistir al estudiante y detectar su perfil considerando las diferencias entre los alumnos. Se maneja una base de casos individual por cada agente, con el objetivo de hacer que los agentes cooperen para mejorar:

(a) el desempeño individual,
(b) la calidad de las soluciones,
(c) la eficiencia para encontrar soluciones,
(d) alcanzar tareas que no podrían ser resueltas solos, y
(e) aprender de los errores.

Esta cooperación es relevante cuando los estudiantes se encuentran geográficamente distantes y se requiere colaboración para resolver el perfil. En este sentido, un *agente intradominio* se encarga de gestionar dicha cooperación. En la figura 10 se presenta el sistema multi-agente mencionado anteriormente. Para la cooperación entre los agentes se utiliza el esquema de intercambio de casos que puede ser revisado en [115].

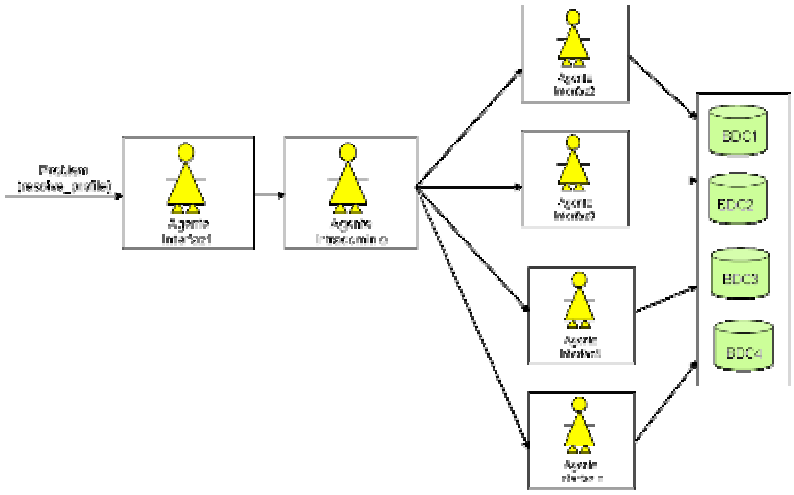


Figura 10. Sistema Multi-agente para modelado del alumno

Para el modelado del alumno se prueban dos aproximaciones:

5.1.2.1 Modelo del Alumno con Redes Bayesianas

Teniendo en cuenta la dependencia del dominio el contenido del modelo del estudiante consiste de dos partes:

- ***Información especifica del dominio:*** Representa el estado del estudiante, nivel de conocimiento y habilidades en términos de un tema particular del dominio.
- ***Información independiente del dominio:*** Incluye metas de aprendizaje, aptitudes cognitivas, medidas de motivación, preferencias de aprendizaje, datos históricos, etc.

En la figura 11, se describe el modelo de red bayesiana que hemos utilizado para representar el modelo del alumno. En la estructura de la red bayesiana se pueden distinguir fácilmente los niveles jerárquicos de agregación que se corresponden con los elementos en los que se divide la estructura del tutorial. Cada nodo de la red va a representar a cada uno de los elementos de la estructura de contenidos del tutorial. A continuación se describe en detalle la representación de los nodos de la red.

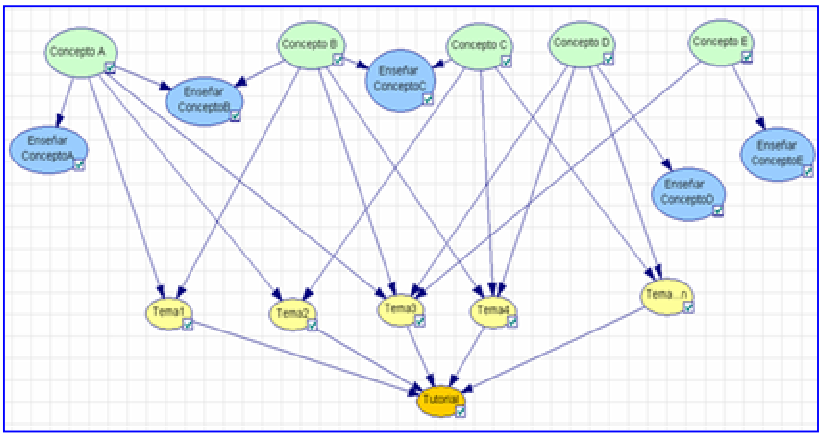


Figura 11. Modelo de Red Bayesiana usada

- **Nodos de Concepto:** Los nodos que representan a los conceptos del tutorial, se encuentran representados en la Figura 11. Estos nodos van a representar el nivel de conocimiento que el alumno ha adquirido en cada uno de los conceptos del tutorial. Cada nodo de tipo concepto va a tener asociados dos estados, el estado Conocido y el estado Desconocido. En el estado Conocido se almacena el porcentaje de conocimiento adquirido por el alumno en el concepto que representa el nodo, que representa al mismo tiempo la probabilidad que tiene el alumno de conocer dicho concepto. Como la suma de los valores de los dos estados debe ser uno, en el estado Desconocido se va a almacenar el valor (1-Conocido), y va a representar la probabilidad de que el alumno no conozca el concepto. Inicialmente, los nodos concepto de la red van a tener un valor cercano a cero en el estado Conocido, valor que irá aumentando a medida que el alumno adquiere conocimientos en los conceptos a los que representan hasta llegar a un máximo de 1 que indica que el alumno conoce la totalidad de los contenidos del concepto.

- **Nodos "Enseñar Concepto":** Los nodos enseñar concepto de la Figura 11, se utilizan para indicar el grado de conveniencia o viabilidad de que el alumno comience a estudiar un concepto determinado del tutorial. El nodo enseñar concepto tendrá dos estados asociados. El estado Viable indica la probabilidad de que sea viable que el alumno comience a estudiar el concepto representado por el nodo concepto al que está asociado. El estado Inviable indica la probabilidad contraria, es decir, la probabilidad de que no sea viable que el alumno estudie el concepto. Inicialmente, cuando el valor del estado Conocido del nodo concepto es cero, el valor del estado Viable del nodo enseñar contendrá un valor cercano a uno ya que como el alumno no conoce nada sobre el concepto, es viable que el alumno comience a estudiarlo. A medida que el alumno adquiere conocimientos en el concepto, el valor del estado Conocido del nodo aumenta, con lo que la viabilidad de estudiar el concepto disminuye paulatinamente.

  Se puede dar el caso de que a un nodo enseñar concepto lleguen más arcos dirigidos provenientes de otros nodos conceptos aparte del que le llega del nodo concepto al que está asociado. Como se puede ver en la figura 4 con los conceptos A y B. En este caso el nodo enseñar concepto del concepto B, sigue representando el grado de conveniencia de que el alumno estudie el concepto B, solo que ahora este grado de conveniencia no solo va depender del nivel de conocimientos adquiridos por el alumno, sino también del nivel de conocimientos adquiridos por el alumno en el nodo A. De esta manera se puede conseguir por ejemplo, que la viabilidad de que el alumno estudie el concepto B dependa del hecho de si el alumno conoce o no previamente el concepto A. Si el alumno conoce el concepto A podrá comenzar a estudiar el concepto B, en caso contrario no podrá hacerlo.

  Mediante las restricciones introducidas en la red bayesiana con los nodos enseñar concepto, no solo se va a conseguir determinar la viabilidad de estudio de determinados conceptos por parte del alumno sino que se va a establecer un itinerario de estudio a seguir por los alumnos que comiencen a estudiar el tutorial. Por ejemplo en el caso de la red bayesiana de la Figura 12, como inicialmente el conocimiento de un alumno en los conceptos del tutorial al que representa es cero, solo es viable que el alumno comience a estudiar el concepto A y el concepto D ya que los nodos enseñar concepto de los conceptos B y C indican que el alumno debe conocer previamente los conceptos Ay D para poder comenzar a estudiar los otros dos conceptos. A la hora de construir la red bayesiana asociada a un tutorial hay que crear convenientemente la tabla de probabilidades condicionadas de los nodos enseñar concepto para obtener este tipo de comportamiento de la red a la

hora de calcular el grado de conveniencia de estudio de cada nodo y definir el itinerario de estudio que seguirán los alumnos en el tutorial.

1. **Nodos de tema:** Los nodos de tema que se representan en la Figura 11, se utilizan para almacenar el grado de conocimiento que adquiere el alumno en un tema determinado del tutorial. El nivel de conocimiento del tema por parte del alumno que representan este tipo de nodos, va a ser un nivel de conocimientos global respecto al conocimiento adquirido por el alumno en cada uno de los conceptos del tema, es decir, que el grado de conocimiento indicado por el nodo tema va a depender directamente del grado de conocimiento indicado por los nodos concepto para cada uno de los conceptos que pertenecen al tema.

   Del mismo modo que ocurre con los nodos de concepto, cada nodo tema va a tener dos estados asociados, el estado Conocido y el estado Desconocido. En el estado Conocido se almacena el porcentaje de conocimiento que el alumno va adquiriendo a medida que aumenta el porcentaje de conocimiento almacenado en los nodos concepto que representan a los conceptos del tema. El estado Conocido representa por lo tanto, la probabilidad de que el alumno conozca el tema representado por el nodo. En el estado Desconocido se va a almacenar el valor (1-Conocido), y va a representar la probabilidad de que el alumno no conozca el tema representado por el nodo. Cuando el estado Conocido alcanza el valor 1 significa que el alumno conoce la totalidad de los conceptos contenidos en el tema. El aumento del valor de conocimiento en los nodos de tema, se va a producir cuando se realice la propagación de probabilidades en la red en el proceso de diagnóstico del alumno como expondremos más adelante. Este aumento en el valor de conocimiento del tema, dependerá del peso que tenga el concepto en el que el alumno ha aumentado su conocimiento dentro del tema. Estos pesos se utilizan para crear las tablas de probabilidades condicionadas de los nodos de tema, con respecto a los nodos concepto en el proceso de creación de la red bayesiana y son los mismos pesos que el experto especifica en el proceso de creación de la estructura del tutorial.

2. **Nodos Tutorial:** El nodo tutorial representado en la Figura 12, se utiliza para almacenar el grado de conocimiento adquirido por el alumno en la totalidad del tutorial. Del mismo modo que ocurre con los nodos de concepto y los nodos de tema, cada nodo tutorial va a tener dos estados asociados, el estado Conocido y el estado Desconocido. En el estado Conocido se almacena el porcentaje de conocimiento que el alumno va adquiriendo a medida que aumenta el porcentaje de conocimiento almacenado en los nodos tema que representan a los temas del tutorial (conocimiento que se aumenta a su vez gracias a los nodos de concepto). El estado Conocido representa por lo tanto, la probabilidad de que el alumno conozca el tutorial representado por el nodo. En el estado Desconocido se va a almacenar el valor (1-Conocido), y va a representar la probabilidad de que el alumno no conozca el tutorial representado por el nodo. Cuando el estado Conocido alcanza el valor 1 significa que el alumno conoce la totalidad de los temas y conceptos contenidos en el tutorial. El nivel de conocimiento del tutorial adquirido por el alumno y representado en este nodo, va a ser por lo tanto, un nivel de conocimiento dependiente de lo que el alumno haya aprendido en los temas del tutorial, y en consecuencia en los conceptos del tutorial. Del mismo modo que ocurre con los nodos tema respecto a los nodos concepto, los nodos tutorial aumentan su valor en su estado conocido cuando se realiza una propagación de las probabilidades en la red y los valores de conocimiento

adquiridos por el alumno en los conceptos del tutorial, se propagan hasta llegar al nodo tutorial.

5.1.2.1.1. Proceso de Diagnostico

El proceso de diagnostico del alumno se realiza mediante la propagación de las probabilidades contenidas en los nodos de la red, a medida que el alumno va adquiriendo conocimientos en cada uno de los conceptos del tutorial.

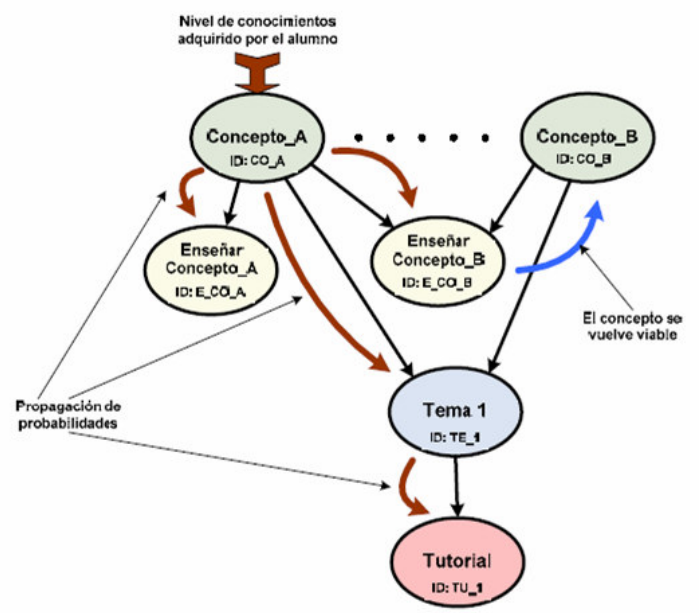


Figura 12. Propagación de probabilidades en la Red Bayesiana

Cuando el alumno aumenta su nivel de conocimiento en un concepto, el valor sobre el nivel de conocimiento adquirido por el alumno en ese concepto se introduce en el estado Conocido del nodo concepto asociado. Una vez actualizado el estado del nodo, se realiza la propagación de las probabilidades de la red, lo que implicará una actualización automática de los nodos directa o indirectamente dependientes del nodo concepto que se ha actualizado.

Esta propagación de probabilidades va a implicar que se actualice el nivel de conocimiento adquirido por el alumno en el nodo tema que depende del nodo concepto actualizado y a su vez se va a actualizar el nivel de conocimiento adquirido por el alumno en el nodo tutorial. La propagación de probabilidades también afecta a los valores de los estados de los nodos enseñar concepto que dependan del nodo concepto actualizado. De esta manera puede suceder que después de realizar una propagación determinados conceptos del tutorial sean accesibles cuando antes no lo eran, debido a que sus nodos enseñar concepto han actualizado sus valores de estado tras la propagación, aumentando el grado de viabilidad de estudio por parte del alumno de los conceptos a los que representan. En la Figura 12 se describe gráficamente el proceso de propagación de probabilidades en una red bayesiana.

El encargado de actualizar los valores de conocimiento en los estados de los nodos concepto así como de realizar la propagación de probabilidades en los nodos de la red, es el agente tutor que está a cargo del alumno en el tutorial.

5.1.2.2. Modelado del Alumno con CBR

Luego de usar las redes bayesianas como aproximación inicial para el modelado del estudiante encontramos un conjunto de problemas entre los cuales podemos mencionar [116]:

(1) la dificultad presentada en el proceso de adquisición del conocimiento, es decir el esfuerzo realizado para especificar el modelo (variables y relaciones causales), y estimar los parámetros (probabilidades condicionales),

(2) la complejidad del modelo incrementaba constantemente,

(3) los resultados obtenidos en algunos eventos fueron incorrectos o inadecuados debido a nuevas asunciones en el modelo,

(4) el número de variables y de operaciones en la red creció exponencialmente, lo cual llegó a ser computacionalmente difícil de manejar.

En este sentido, hemos planteado una nueva aproximación al modelado del estudiante utilizando CBR el cual incluye la representación del conocimiento y razonamiento del estudiante, y la forma como este adquiere el conocimiento para poder llevar a cabo un aprendizaje inteligente. Para la implementación se utiliza la herramienta CBR-Works [116]. La información acerca de los estudiantes es almacenada como casos. Cuando un estudiante comienza una sesión de aprendizaje la información acerca de él es extraída de la base de datos de estudiantes (*student model*) y convertida dentro de un nuevo caso.

Los casos son almacenados en la base de casos y descritos en términos de conceptos, atributos y tipos. En el proceso de recuperación, se crean nuevas búsquedas (querys) donde se asignan valores específicos a los conceptos con el fin de crear un nuevo caso (se pueden usar casos existentes como búsquedas). La mayor similaridad de casos es obtenida por medidas de similaridad basadas en atributos. En este caso la herramienta nos permite seleccionar diferentes funciones entre las que podemos mencionar: Euclidiana, K-nearest, etc. Una revisión mas detallada del modelado del estudiante con CBR se encuentra en [116][117].

La figura 13 indica los pasos a seguir para la ejecución del ciclo CBR en el modelado del alumno.

a. Representación de Casos: Teniendo en cuenta la definición formal de caso descrita en [9], un caso es considerado como un pareja ordenada (P x S) donde **P** corresponde al problema y **S** se refiere a su respondiente solución. Entonces $P(X_1, X_2, X_3, X_4,\ldots X_n)$ es una descripción general de los problemas y $S(Y_1, Y_2, Y_3, Y_4,\ldots Y_n)$ corresponde a una descripción general de las soluciones. Siendo X_i y Y_i las variables del problema y la solución respectivamente. El proceso CBR consiste en asignar un valor a las variables problema y encontrar las instancias correctas para las variables solución.

El primer paso cuando se diseña un sistema CBR es la representación de casos. Un caso deberá contener el contenido y el contexto, típicamente compuesto del problema y su solución; en nuestro trabajo hemos adicionado un componente de **salida** que almacenará información sobre las modificaciones realizadas a los casos, atributos que causaron fallos y los nuevos valores asignados, resultados obtenidos al aplicar las actividades, etc.

Para la representación de casos se pueden utilizar diferentes clases de estructuras: árbol, esquema relacional, pares atributo-valor, objetos, predicados, redes semánticas, reglas, etc, dependiendo de la estructura, contenido de las casos y las preferencias del desarrollador.

Para representar la información de los estudiantes usamos CaseML un lenguaje de marcada estructurado derivado de XML [118]. En la figura 14 la estructura de casos en CaseML es descrita.

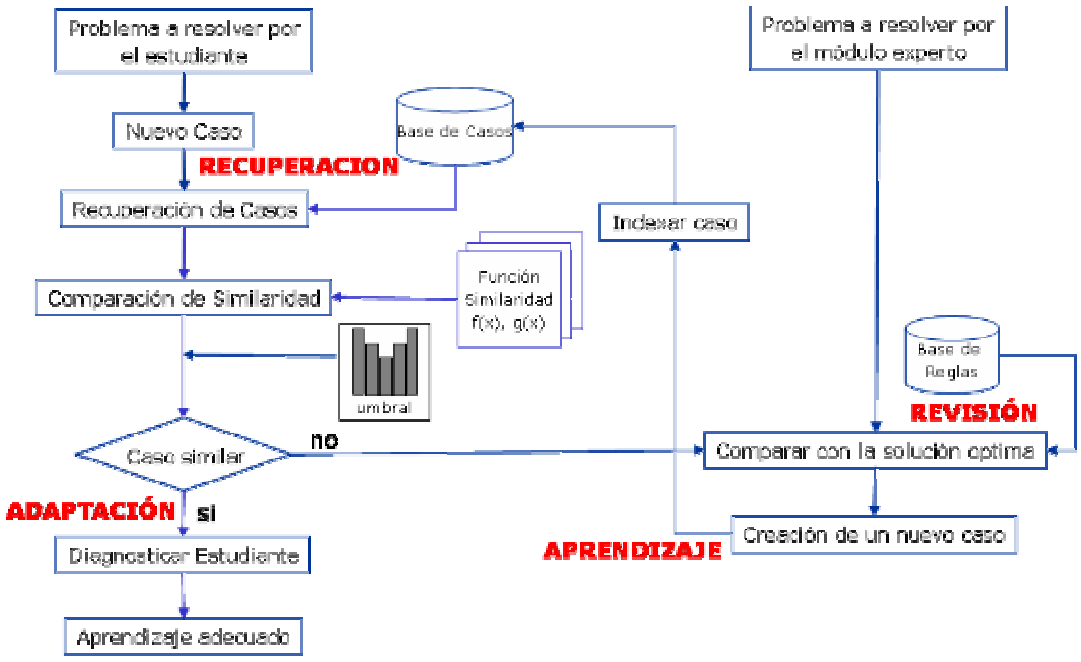


Figura 13. Modelado del Estudiante & CBR

En nuestro trabajo el conocimiento general del estudiante se estructura de la siguiente forma:

- **Información específica del dominio (Domain Specific Knowledge – DSK):** Esta representa una reflexión del estado del estudiante, su nivel de conocimiento y habilidades en terminos de un tema particular del dominio.

- **Información independiente del dominio (Domain Independen Knowledge – DIK):** Esta incluye información que es diferente entre sistemas. Puede incluir metas de aprendizaje, aptitudes cognitivas, medidas de motivación, preferencias acerca de la presentación del método de aprendizaje, datos históricos, etc. La figura 14 muestra un ejemplo de la representación de casos.

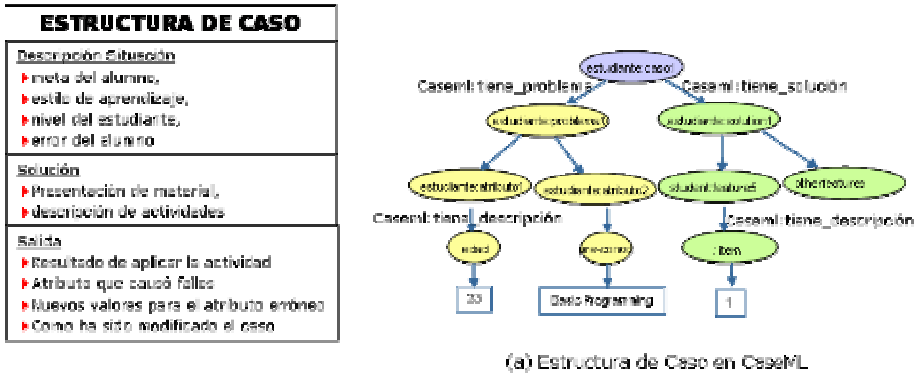


Figura 14. Estructura de casos

b. Indexación de Casos, Almacenamiento y Recuperación: La recuperación de casos consiste de subtareas, referentes a como identificar características, correspondencias iniciales, búsqueda y selección [9]. Una vez los casos son representados e indexados, estos deben ser organizados en una estructura eficiente para su recuperación. En nuestro trabajo, los casos son representados como estructuras y la clase de similaridad que se aplica tiene en cuenta la similaridad estructural entre los casos. En este sentido, un agente de recuperación que use la técnica LID (Lazy Induction of Descriptions) [119] desempeña la tarea de recuperación (Etapa I en Figura 15). LID permite definir la similaridad a partir de una descripción simbólica de las características compartidas entre un nuevo problema y los casos precedentes, evaluando la importancia de las relaciones involucradas con el propósito de clasificar correctamente el problema. Si el grado de similaridad encontrado es muy bajo o no existen casos similares al problema dado, el nuevo caso se almacena como no existente en la base de casos. El sistema en esta fase chequea los resultados de la resolución del problema. Si los resultados son correctos el sistema crea un caso resuelto en la base de casos. Sin embargo si los resultados son incorrectos el sistema asume que el estudiante no conoce el método de resolución e infiere sus errores. Finalmente el sistema provee una solución óptima al estudiante.

c. Inicialización del modelo: En el proceso de inicialización se utilizan test previos en los cuales el sistema analiza la respuesta de los estudiantes y almacena la información para construir el modelo. En algunos casos, se utiliza la técnica de estereotipos en donde se coleccionaron atributos comunes sobre cierto grupo de personas que comparten los mismos intereses, siguiendo una serie de criterios definidos previamente. Esta información fue utilizada como parámetros por defecto para inicializar el modelo del estudiante.

c. Adaptación de Casos: Una vez los mejores casos son recuperados, ellos son reusados o adaptados. La adaptación puede avanzar, si los casos recuperados y el caso objetivo son extremadamente similares en términos de atributos fundamentales, o si los casos son muy similares en algunas componentes. En nuestro caso a diferencia de problemas simples de clasificación donde el resultado de la adaptación corresponde a la resolución el problema usando los casos existentes, el proceso de adaptación requiere de un análisis detallado y en la mayoría de los casos las soluciones existentes tienen que ser modificadas**.**

Una nueva estrategia pedagógica puede ser construida incluyendo componentes del caso que mejor se ajusten al problema que han sido extraídos de diferentes casos recuperados. En nuestra aproximación el uso de un agente de Adaptación permite automatizar esta tarea (Etapa II en Figura 15).

d. Revisión y Aprendizaje: La etapa de revisión ha sido tradicionalmente la tarea más difícil dentro de un CBR. En nuestra aproximación un agente evaluador que hace uso de un sistema de evaluación desempeña esta tarea. Cuando la solución generada por un caso en el proceso de adaptación es equivocada, el agente revisor es responsable de modificar la solución teniendo en cuenta el conocimiento disponible acerca del problema. En esta fase el agente ejecuta las siguientes tareas:

- **Evaluar la solución del caso generada por reuso:** En esta tarea, si el agente detecta condiciones irregulares este reacciona para evaluarlas y tomar las acciones apropiadas. Si esto no ocurre, la solucion inicial es considerada exitosa y el caso es retenido. (Etapa IV en Figura 15)

- **Reparar la solución del caso, usando información específica del dominio:** Esta tarea involucra detectar los errores de la solución inicial y recuperar o generar explicaciones para ellos. El agente usa las explicaciones a fallas con el objetivo de asegurar que estas fallas no ocurran de nuevo. Una vez se asegura la correctitud de la solución el nuevo caso puede ser retenido.

Nuestro modelo considera las diferencias que tienen los estudiantes para procesar la información. Tiene en cuenta que el aprendizaje depende de varios factores personales y cada estudiante posee un estilo propio para aprender el cual no permanece invariable sino que cambia con el tiempo y depende del contexto de las tareas educativas. Por esta razón, nuestro modelo es dinámico ya que el conjunto de agentes involucrados deben ser capaces de aprender de su entorno y de la interacción con otros agentes con el fin de incorporar estos cambios en su base de casos. El agente debe aprender de las interacciones con el estudiante para adaptar el entorno de aprendizaje a sus preferencias las cuales han sido percibidas mediante la actualización de su estilo de aprendizaje.

5.1.3. Modulo Experto

En el ámbito de los ITS, los elementos que representan el conocimiento del dominio, se incluyen en una de las partes en la que se divide ese tipo de sistemas, conocida con el nombre de *módulo del experto*. En la base de conocimiento de nuestro sistema se almacenan los elementos que representan todo el conocimiento a adquirir por un alumno en un tutorial determinado; elementos introducidos en el sistema por un experto del dominio del tutorial. Para representar la estructura del currículo, se ha optado por seguir una división del conocimiento del dominio de cada tutorial basada en conceptos, utilizada eficazmente en sistemas como BITS [120]. De esta manera, el alumno adquiere el conocimiento del dominio a medida que adquiere conocimiento en cada uno de los conceptos en que se divide el tutorial.

Así pues, en la estructura del currículo se distinguen los siguientes elementos:

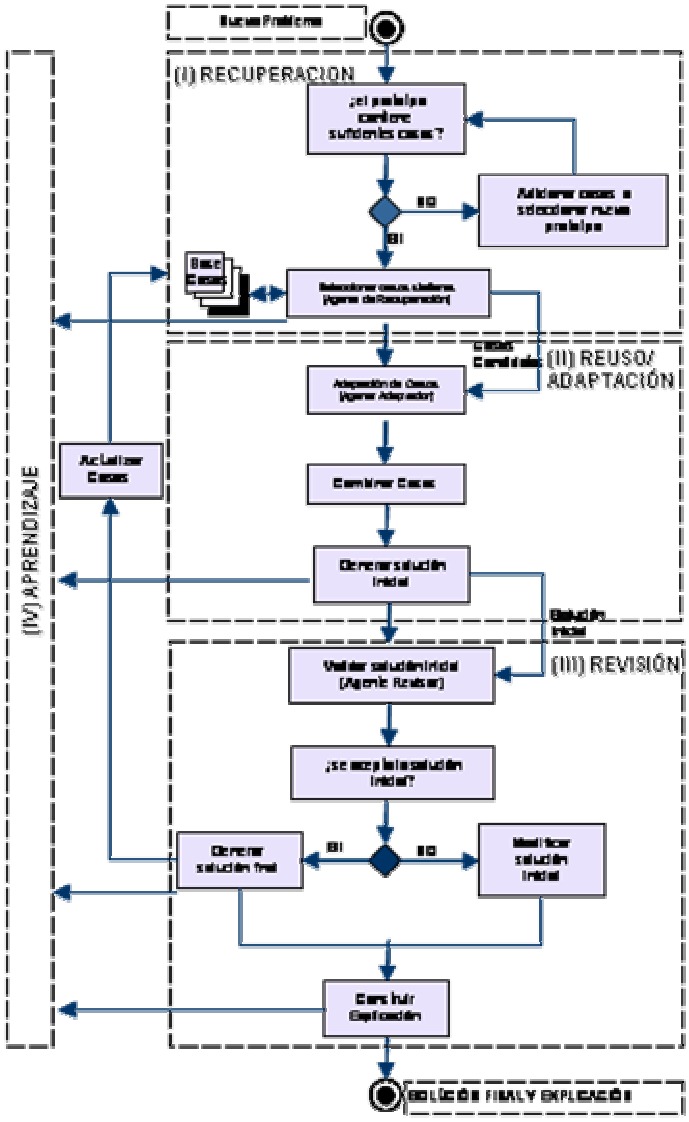


Figura 15. Proceso de modelado del estudiante con CBR

Tutorial: En un nivel superior se encuentra el elemento tutorial que engloba todo el conocimiento sobre un dominio determinado.

Temas: En un primer nivel, cada tutorial se divide en un número determinado de temas. Cada tema contiene un número de conceptos relacionados con el conocimiento que se pretende enseñar en el tema y un examen que debe ser superado por el alumno.

Conceptos: Los conceptos representan las unidades de conocimiento que deben ser aprendidas por el alumno a lo largo del tutorial. Cada concepto está formado por una serie de contenidos textuales que explican el conocimiento que representa el concepto.

Preguntas de selección: Intercalas entre los contenidos textuales de cada concepto se encuentran una serie de preguntas de selección que tienen como objetivo, evaluar

el conocimiento que el alumno adquiere sobre el concepto a medad que examina las aplicaciones.

Preguntas de examen: Son las preguntas contenidas en los exámenes de los temas. Se trata de ejercicios de desarrollo que el alumno debe resolver y que serán evaluados y corregido por un experto.

5.1.3. Modulo del Entorno

La interfaz de usuario del sistema se compone de partes diferentes, teniendo en cuenta el tipo de usuarios que interaccionan con el sistema.

La interfaz de usuario del alumno que aprende le tutorial, y varia su estructura y distribución dependiendo de la tarea que se encuentre realizando el alumno. La interfaz de usuario del profesor o experto es similar a la del alumno, solo que carece de ciertos elementos de interacción con el sistema CBR ya que el profesor no necesita ser evaluado por el alumno.

5.1.4. Comunicación entre los agentes

Para lograr la comunicación entre los agentes, se usa el conjunto de preformativas KQML [121].

Paso 1. El agente interfaz teniendo en cuenta la información contenida en el modelo del estudiante, determina su perfil.

Paso 2. El agente interfaz envía el perfil de cada estudiante al agente tutor.

Paso 3. El agente tutor recupera los casos que mas se ajusten al perfil de alumno de la base de casos.

Paso 4. El agente tutor envía la información de casos recuperados al agente adaptación. El agente adaptación combina los casos recuperados con el nuevo, generando la estrategia a seguir y organizando los recursos.

Paso 5. El agente tutor se comunica con el agente orientador con el caso resuelto. Este agente orientador realiza un proceso cíclico de revisión, obteniendo una nueva estrategia modificada.

Paso 6. La nueva estrategia obtenida por el agente orientador, es enviada a la base de casos donde es almacenada.

5.2. METODOLOGIA PARA LA CONSTRUCCIÓN DEL SISTEMA MULTIAGENTE

La arquitectura considera la metodología INGENIAS [122] la cual concibe el sistema multi-agente como la representación computacional de un conjunto de modelos. Cada uno de estos modelos muestra una visión parcial del sistema: los agentes que lo componen, las interacciones que existen entre ellos, cómo se organizan para proporcionar la funcionalidad del sistema, qué información es relevante en el dominio y cómo es el entorno en el que se ubica el sistema a desarrollar.

Para especificar cómo tienen que ser estos modelos se definen meta-modelos. Un meta-modelo es una representación de los tipos de entidades que pueden existir en un modelo, sus relaciones y restricciones de aplicación. Los meta-modelos que describe INGENIAS son: agente, organización, dominio, tareas-objetivos, e interacciones. La figura 16 presenta los modelos de MAS en INGENIAS.

Un meta-modelo define las primitivas y las propiedades sintácticas y semánticas de un modelo. A diferencia de otros enfoques más formales, 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 el lenguaje de modelado unificado (UML) [123] construido también con meta-modelos, este tipo de notación facilita enormemente el desarrollo de sistemas.

Figura 16. Modelos de Sistemas Multi-agente en INGENIAS

Modelo de la Organización: Define cómo se agrupan los agentes, la funcionalidad del sistema y qué restricciones hay que imponer sobre el comportamiento de los agentes. El modelo de organización es el equivalente a la arquitectura del sistema en un MAS. El valor principal de un modelo de organización, como ocurre en las organizaciones humanas, son los flujos de trabajo que define. Del estudio de estos flujos surgen nuevas interacciones que reflejan con detalle cómo se coordinan los participantes del flujo. El modelo de organización también contribuye al modelo de tareas y objetivos identificando las tareas relevantes para la organización así como los objetivos que se persiguen globalmente. También define restricciones en el comportamiento de los agentes mediante relaciones como la de subordinación. Gracias a estas restricciones, el diseñador asegura que unos agentes obedecerán a otros o que se comprometerán a la ejecución bajo demanda de tareas respetando sus prioridades.

Modelo de Objetivos/Tareas: Se usa para asociar el estado mental del agente con las tareas que ejecuta. El meta-modelo de objetivos y tareas tiene como propósito recoger las motivaciones del SMA, definir las acciones identificadas en los modelos de organización, interacciones o de agentes y cómo afectan estas acciones a sus responsables. Esta información constituye parte de la especificación de cómo se quiere que sea el control del agente a alto nivel.

Modelo de Agente: El meta-modelo de agente describe agentes particulares y los estados mentales en que se encontrarán a lo largo de su vida. 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:

- *Responsabilidades.* Se trata de las tareas que sabe ejecutar y de los objetivos que se compromete a alcanzar. Generalmente se alude al término rol para agrupar la funcionalidad y las propiedades que aparecen con frecuencia en el diseño. Aunque es similar al concepto interfaz de la programación orientada a objetos, el rol se diferencia principalmente en que se le puede asociar estado. En este caso, es de interés el conjunto de estados mentales asociados a un rol que participa en una interacción.
- *Comportamiento*. Existen diversas formas de expresar el comportamiento del agente. En las metodologías existentes, el comportamiento se entiende como un conjunto de llamadas a procedimiento (UML), paso de mensajes entre agentes o transiciones en máquinas de estado. La tónica general es definir el comportamiento de los agentes en las interacciones, pero no es el único modo. En este trabajo se habla del control del agente, esto es, mediante qué mecanismos se va a asegurar la ejecución de tareas dentro de los parámetros acordados. Este control toma como entrada un conjunto de datos que se denominará estado mental. Además, se considerará el estado mental como algo dinámico que evoluciona con el tiempo. Esta idea es necesaria para poder incluir el aprendizaje entre las capacidades del agente.

Modelo de Interacciones: En este modelo se detalla el proceso de coordinación y comunicación entre los agentes. Las interacciones determinan el comportamiento de los agentes mostrando cuál es su reacción cuando actúan sobre ellos. Y cómo el comportamiento va a ser función de las objetivos de los agentes y las tareas a ejecutar, se puede concluir que existe un importante vínculo entre interacciones, objetivos y tareas.

Modelo del Entorno: Define qué existe alrededor del nuevo sistema y cómo lo percibe cada agente.

5.2.1. Integración de INGENIAS con el proceso unificado (*Rational Unified Process – RUP*)

En el RUP [124], el esfuerzo del análisis y diseño se encuentra localizado en tres fases: inicio, elaboración y construcción. Dentro de cada fase se desarrollan las iteraciones (ciclos completos de desarrollo incluyendo análisis, diseño, implementación y pruebas) que construyen gradualmente el sistema.

La generación de modelos a partir de meta-modelos se guía por el conjunto de actividades que no sustituyen las indicadas por el RUP, sino que se integran en el paradigma como complemento de los elementos de especificación ya existentes. De hecho, en las actividades de generación se mencionan *técnicas convencionales* refiriéndose a la utilización de notaciones como UML y modelos de desarrollo como RUP para atacar el diseño de elementos concretos. Para orientar la integración, se plantean una serie de asociaciones entre elementos del RUP y elementos de los meta-modelos (ver Tabla 1).

ENTIDAD MAS (ITS-CBR)	Entidad RUP
Agente	Clase
Organización	Arquitectura
Grupo	Subsistema

Interacción	Escenario
Roles, tareas y flujos de trabajo	Funcionalidad

Tabla 1. Asociaciones entre los elementos del RUP y entidades del ITS

El *agente*, como la *clase*, define tipos. Lo que aquí se ha denominado *agente en ejecución* sería un *objeto* en el RUP. La *organización* equivale a la *arquitectura* en el RUP por su carácter estructurador. La *organización* da una visión global del sistema agrupando agentes, roles, recursos y aplicaciones y estableciendo su participación en los flujos de trabajo del MAS. El *grupo* es la unidad de estructuración utilizada en la organización. Su similitud con un *subsistema* se debe a que como éste, se utiliza para organizar elementos en unidades de abstracción mayores y define un conjunto de interfazs para interaccionar, los roles en este caso. La *interacción*, se ve como una generalización de los diagramas de colaboración y secuencia. En el RUP, los diagramas de secuencia y colaboración se ven como *escenarios* que describen cada *caso de uso*. Por último, *roles*, *tareas* y los *flujos de trabajo* proporcionan el encapsulamiento de acciones que en el RUP dan *métodos* e *interfazs*.

6. PROTOTIPO DE SISTEMA TUTOR INTELIGENTE PARA APRENDIZAJE EN SALUD

SITUA[1] es un sistema de tutorización basado en agentes desarrollado por el Grupo Web de Agentes Inteligentes - GWAI de la Universidad de Vigo y que ha sido integrado en la herramienta de teleformación Ariadna, la cual ha sido desarrollada por el GWAI a partir del ambiente Teleduc conjuntamente con el Núcleo de Informática Aplicada a la Educación (Nied) y el Instituto de Computación (IC) de la Universidad Estatal de Campinas (Unicamp).

El sistema tutor se basa en la creación de un modelo de usuario que representa las preferencias del mismo y los progresos que realiza en el proceso de adquisición de conocimientos. Su principal característica es la adaptabilidad al usuario.

El modelado del estudiante propuesto en SITUA es realizado por medio de Redes Bayesianas, como una de las aproximaciones mas utilizadas en la actualidad. Nuestra aproximación considera un conjunto de deficiencias con las redes bayesianas para modelado del estudiante y propone la utilización de Razonamiento Basado en Casos.

A continuación se describen las diferentes fases de desarrollo del sistema.

6.1. ESPECIFICACIÓN DE REQUISITOS

El objetivo principal de SITUA es desarrollar un sistema con ciertas capacidades inteligentes, que permita a los estudiantes aprender cualquiera de los tópicos incluidos en el mismo, siendo adaptable a las características de cada alumno. La interacción entre el alumno y el tutor le permitirá al estudiante, acceder a los temas y conceptos del tutorial para aprender los contenidos que en ellos se describen; y permitirá al sistema tutor evaluar los conocimientos adquiridos por el alumno, aconsejarlo y guiarlo durante todo el proceso de aprendizaje.

Desde el momento que el alumno comienza el tutorial, es guiado por el sistema a través de los elementos que componen el tutorial. Al mismo tiempo, el sistema realizará una evaluación simultánea de los conocimientos adquiridos y de los resultados obtenidos por el alumno en cada uno de los elementos del tutorial. A partir de esta información y mediante un proceso de inferencia, el sistema debe ser capaz de aconsejar al alumno cual es el mejor camino a seguir para finalizar el tutorial con un nivel de conocimientos adecuado.

El sistema permitirá que un profesor o experto pueda seguir en todo momento el proceso de aprendizaje de uno o varios alumnos que tenga a su cargo. La interacción entre profesor y tutor permitirá al profesor acceder a la información sobre el grado de conocimientos y el progreso de los alumnos en cada uno de los elementos del tutorial. El profesor también realizará la evaluación de algunos elementos del tutorial que el sistema no realizará automáticamente.

6.1.1. Estructura de los contenidos de los tutoriales

Los tutoriales de SITUA deberán seguir una estructura de contenidos de carácter jerárquico. A mismo tiempo debe ser suficientemente flexibles para ser capaz de

[1] Investigación soportada por los proyectos nacionales TIC2002-04516-C03-01 y TIC2001-5108-E

representar cualquier tipo de contenidos. La estructura del tutorial estará compuesta por una serie de temas, secciones o lecciones que formarán la estructura general del tutorial.

- Cada uno de los temas o secciones estará dividido en partes más pequeñas que representarán los conceptos que es necesario conocer para superar esa parte o tema del tutorial. Cada tema tendrá como mínimo un concepto, sin existir límite para el número máximo de conceptos por tema.

- Adicionalmente, temas del tutorial tendrán asociados una serie de exámenes de tema, formados por preguntas de desarrollo relacionadas con los contenidos del tema. Estos exámenes podrán contener un número de preguntas variable, y cada tema podrá tener un examen o más de uno.

- Para establecer un control sobre el itinerario de estudio de cada tutorial, será necesario que determinados conceptos del tutorial sean conocidos por el alumno antes de emprender el aprendizaje de otros conceptos del mismo tema o de diferentes temas. El sistema debe tener en cuenta estas restricciones jerárquicas entre conceptos a la hora de conducir el proceso de aprendizaje del alumno a través del tutorial.

- Los conceptos de cada tema estarán formados por dos tipos de contenidos diferentes: las explicaciones y las preguntas cortas de cada explicación.

    - Las explicaciones de un concepto estarán formadas por los contenidos textuales necesarios para que el alumno logre aprender dicho concepto. Estos contenidos estarán en un formato fácilmente extensible y manejable (XML, HTML…) y seguirán un orden previamente establecido dentro del concepto. Este orden es en el que le serán mostradas al alumno cuando se encuentre estudiando el concepto. Un concepto deberá tener al menos una explicación, sin existir límite específico para el máximo de explicaciones por concepto.

    - Cada explicación de un concepto podrá tener una serie de preguntas cortas para ser respondidas por el alumno cuando se encuentra estudiando la explicación y así demostrar el nivel de conocimiento adquirido. Estas preguntas cortas, serán preguntas de selección donde se proponen una serie de respuestas a la pregunta donde solo una de ellas es correcta. Una explicación puede no contener ninguna pregunta corta, pero no existe límite para el máximo de preguntas cortas por explicación.

6.1.2. Interacción del sistema con los alumnos del tutorial

Cuando el alumno accede al sistema, le son mostrados los tutoriales disponibles. El alumno, si lo desea, podrá registrarse en cualquiera de ellos.

Una vez que el alumno accede a un tutorial, le serán mostrados los elementos de la estructura del tutorial al que ha accedido (temas, conceptos, explicaciones, etc.). Los elementos le son mostrados en forma de enlaces para poder ser seleccionados por el alumno y acceder así a sus contenidos.

Cuando el alumno selecciona un tema del tutorial, el sistema debe ser capaz de determinar si es aconsejable que empiece el estudio de los conceptos de ese tema, y

en caso contrario proponer una alternativa de estudio al alumno en otros conceptos del tutorial.

Cuando el alumno selecciona un concepto del tutorial, el sistema debe determinar si el alumno puede o no empezar a estudiar los contenidos del tutorial, proponiendo alternativas válidas de estudio de otros conceptos del tutorial.

El alumno no podrá acceder a los contenidos de un concepto mientras el sistema considera que el alumno no debe acceder a ese concepto.

Cuando el alumno accede a un concepto, se le mostrarán los contenidos didácticos de sus explicaciones siguiendo la estructura y el orden establecido para ellas al crear el tutorial. A la vez debe realizar las preguntas cortas de cada explicación. El alumno debe ser capaz de responder a esas preguntas seleccionando la respuesta correcta para poder avanzar y estudiar los contenidos didácticos de la siguiente explicación del concepto.

El alumno podrá avanzar a lo largo de las explicaciones del concepto mediante la selección directa en los respectivos enlaces, o bien mediante una barra de botones de navegación con los botones siguiente y anterior. Solo se podrá acceder a una explicación determinada si la explicación anterior es conocida por el alumno.

Cuando el alumno se encuentra examinando los contenidos de un concepto, el sistema debe evaluar el conocimiento adquirido por alumno sobre ese concepto, en base al número de respuestas correctas respondidas por el alumno en las explicaciones del concepto. Al mismo tiempo el sistema tomará en cuenta el tiempo dedicado por el alumno a la hora de estudiar los contenidos del concepto.

Solo cuando el alumno conozca todos los conceptos de un tema, podrá realizar el examen del tema, para que el profesor a cargo del alumno compruebe si realmente ha adquirido los conocimientos necesarios en el tema.

El examen de cada tema, se presentara al alumno en un formulario adecuando con el enunciado de las preguntas y el espacio suficiente para las respuestas. Una vez respondidas, las respuestas le serán enviadas al profesor para su evaluación. Una vez corregidas, se permite al alumno ver las correcciones.

El alumno debe ser capaz de ver en cualquier momento, el grado de conocimiento alcanzado en cada uno de los conceptos o secciones del tutorial de forma numérica o gráfica (barras de progreso).

El alumno debe ser capaz de solicitar un consejo al sistema cuando se encuentra realizando el tutorial. El sistema, basándose en la información sobre los progresos del alumno en el tutorial, debe ser capaz de aconsejar cual será el paso o acción más beneficiosa para el alumno que puede llevar a cabo en ese momento.

El estado en el que se encuentra el alumno dentro del tutorial, será guardado de manera persistente cuando el alumno abandona el mismo, para poder ser recuperado de nuevo y permitir al alumno continuar donde lo dejó cuando acceda de nuevo al tutorial. De la misma manera, se guardarán de manera persistente las respuestas del alumno a las preguntas que se le han realizado en el tutorial.

6.1.3. Interacción del sistema con los profesores de un tutorial.

Al acceder a un tutorial, el profesor debe ser capaz de ver un listado con los alumnos que tiene a su cargo en el tutorial.

El profesor al entrar en el tutorial y seleccionar uno de los alumnos que tiene a su cargo, debe ser capaz de acceder a la información sobre los conocimientos adquiridos en el tutorial por el alumno, lo que le permitirá hacerse a una idea de los avances del alumno en el tutorial. Podrá también obtener información sobre el grado de conocimiento que el alumno ha adquirido en cada una de las secciones y conceptos del tutorial. Esta información le será mostrada o bien numéricamente o gráficamente de la misma manera que le es mostrada al alumno.

El sistema debe avisar al profesor cuando éste entra en el tutorial, si hay exámenes pendientes para corregir pertenecientes a los alumnos que tiene a su cargo.

A la hora de corregir los exámenes, se le debe mostrar al profesor el formulario adecuado con las respuestas del alumno y espacio suficiente para realizar las correcciones de las respuestas incorrectas También se le proporcionará espacio para evaluar la respuesta del alumno. Al terminar, las correcciones, se deben enviar de vuelta al alumno para que las examine si lo desea.

La evaluación realizada por el profesor, será tenida en cuenta por parte del sistema a la hora de considerar como completados y plenamente conocidos, los temas del tutorial.

El sistema debe avisar al alumno cuando el profesor haya terminado de corregir las preguntas de un examen de tema, para que el alumno pueda examinar las correcciones del profesor.

El profesor debe se capaz de visualizar los contenidos del tutorial distribuidos en los elementos que forman la estructura del tutorial. Estos contenidos le serán mostrados de forma similar a la utilizada por el sistema para mostrar los contenidos al alumno, solo que en este caso no es necesaria la evaluación simultánea por parte del sistema.

6.1.4. Gestión de tutoriales

El sistema debe permitir al administrador, introducir nuevos tutoriales. Los contenidos de los tutoriales a introducir en el sistema deben ser incluidos en ficheros XML con una estructura fija y bien definida.

El administrador debe ser capaz de eliminar en cualquier momento un tutorial del sistema. Al eliminar el tutorial, se deben eliminar los contenidos del tutorial y toda la información relativa al estado de los alumnos en el tutorial.

El sistema debe permitir al administrador de SITUA, exportar los contenidos de un tutorial a ficheros XML, los cuales seguirán la misma estructura y formato que los ficheros utilizados en el proceso de importación de contenidos.

6.1.5. Gestión de usuarios y control de acceso

Al tratarse de un sistema que se integrará dentro de la herramienta de tele-formación ARIADNA, los usuarios de SITUA serán los usuarios que ya han sido dados de alta como usuarios de ARIADNA.

Solo aquellos usuarios que son alumnos en alguno de los cursos de ARIADNA, podrán acceder a los tutoriales de SITUA como alumnos. Los alumnos deben ser capaces de registrarse en cualquier tutorial de SITUA si lo desean. Solo aquellos usuarios que son profesores en alguno de los cursos de ARIADNA, podrán acceder a los tutoriales de SITUA como profesores.

El administrador del sistema será el encargado tanto de registrar como eliminar el registro de los profesores en los tutoriales del sistema. También podrá registrar o eliminar el registro de los alumnos en dichos tutoriales.

A cada profesor se le asignarán un número determinado de alumnos del tutorial. El profesor solo tendrá acceso a la información sobre el proceso de aprendizaje de este grupo de alumnos en el tutorial en que ha sido registrado. La asignación de alumnos a cada profesor será realizada por el administrador del sistema pudiendo eliminar estas asignaciones cuando lo considere necesario.

Solo se podrán asignar a un profesor aquellos alumnos registrados en un tutorial y que no tienen profesor asignado. Cada alumno solo tendrá un profesor a su cargo en cada tutorial, mientras que un profesor podrá tener varios alumnos a su cargo en los tutoriales en los que esté registrado. No será necesario que un alumno tenga asignado un profesor para poder acceder y estudiar los contenidos de un tutorial. Los exámenes de los temas del tutorial, que son realizados por el alumno permanecerán sin corregir, sin que esto sea impedimento para que el alumno siga estudiando los contenidos del tutorial.

Para el acceso al sistema será necesario introducir correctamente el nombre de usuario y contraseña en un formulario de acceso. Si se introducen erróneamente se debe notificar el error al usuario y dar la posibilidad de volver a introducir los datos de acceso. Una vez que un usuario acceda al sistema, se le mostrará una interfaz adecuada según el tipo de usuario y que le permita realizar sus tareas dentro de SITUA.

6.2. ANÁLISIS DE REQUISITOS

6.2.1. Identificación de los Procesos de negocio.

A continuación se describen los procesos de negocio y de mantenimiento de los que consta la aplicación y los distintos roles externos que interactúan con ella. En la figura 15 se describe el proceso de análisis de requisitos mediante un diagrama de contexto del sistema. Una descripción detallada de estos procesos junto con sus diagramas de actividades puede ser revisada en el anexo A.

- **Acceso al sistema:** Incluye todas las actividades relacionadas con el control de acceso de los usuarios al sistema y a los tutoriales y la presentación de las interfazs adecuadas para los usuarios autorizados.

- **Contenidos:** Incluyen todas las actividades relacionadas con el aprendizaje por parte del alumno de los conceptos que componen los tutoriales, y la evaluación por parte del sistema, del nivel de conocimiento adquirido por el alumno en cada uno de esos conceptos.

- **Exámenes:** Incluyen todas las actividades relacionadas con la realización de los exámenes de evaluación de cada tema en un tutorial por parte del alumno y de la corrección y evaluación de dichos exámenes por parte del profesor.

- **Informes:** Incluyen todas las actividades relacionadas con la solicitud de informes por parte del alumno o por parte del profesor, para conocer la evolución del alumno en el tutorial.

- **Gestión de tutoriales:** Proceso de mantenimiento donde se incluyen las actividades relacionadas con los procesos de creación de los contenidos del tutorial (contenidos de los temas y conceptos del tutorial, preguntas de selección, exámenes, etc.).

- **Gestión de usuarios:** Proceso de mantenimiento donde se incluyen las actividades relacionadas con los procesos de asignación de alumnos y profesores a los tutoriales del sistema.

6.2.2. Identificación de los Roles externos.

- **Navegante:** El rol navegante lo desempeña cualquier persona que quiere acceder al sistema SITUA. Estas personas pueden ser usuarios de ARIADNA (alumno, profesor o administrador) que al estar ya registrados en esta herramienta son considerados usuarios de SITUA automáticamente o pueden ser cualquier persona no registrada que intenta acceder al sistema. En la figura 17 los diferentes roles son descritos.

- **Usuario:** El rol de usuario, lo desempeña cualquier persona que está registrada en ARIADNA y por lo tanto es considerada como usuario de SITUA.

- **Alumno:** El rol de alumno lo desempeña aquel usuario que accede al sistema para aprender los conceptos de un tutorial, realizar exámenes y solicitar informes de evaluación.

- **Profesor:** El rol de profesor lo desempeña aquel usuario que accede al sistema para corregir exámenes y solicitar informes de evaluación de un alumno.

- **Administrador:** El rol de administrador lo desempeña aquel usuario que tiene los permisos necesarios para eliminar tutoriales del sistema SITUA o de asignar o eliminar alumnos a un profesor en un tutorial determinado.

6.2.3. Diagrama de Contexto

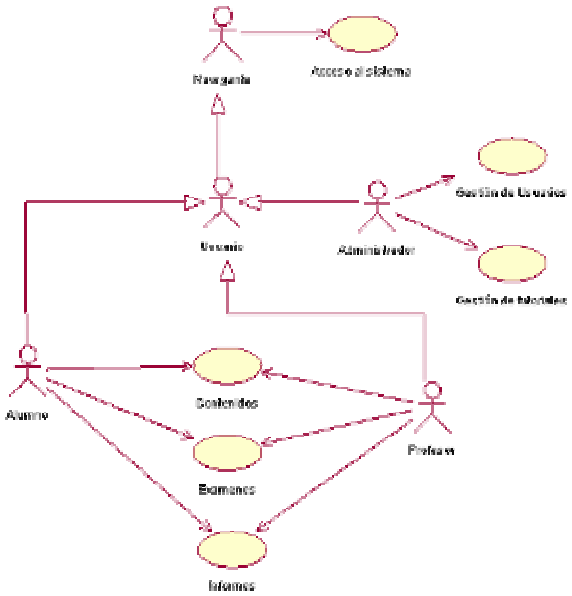


Figura 17. Análisis de requisitos. Diagrama de contexto del sistema

6.3. ANÁLISIS DEL SISTEMA

6.3.1. Casos de Uso

A continuación se identifican cuales son los casos de uso de cada proceso de negocio y quienes son los actores que participan en ellos. Los casos de uso se extraen a partir de las actividades de los diagramas de actividades construidos para cada proceso de negocio en el análisis de requisitos.

Para cada proceso de negocio se incluye uno o varios diagramas de casos de uso y las plantillas de descripción de cada uno de los casos de uso que aparecen en dichos diagramas. Al mismo tiempo, cada plantilla de descripción de los casos de uso se acompaña de un diagrama de actividad donde se muestra la secuencia de actividades que tienen lugar con la realización del caso de uso. La totalidad de los casos de uso y sus correspondientes diagramas de actividades pueden ser revisados en el Anexo B.

6.3.1.1. Caso de Uso del proceso de negocio "Acceso al Sistema"

En la figura 18 se hace una descripción gráfica del caso de uso Acceso al Sistema.

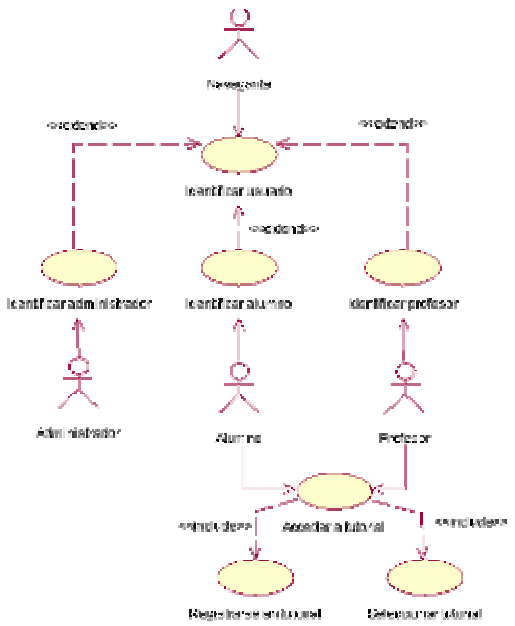


Figura 18. Diagrama de caso de uso Acceso al Sistema

Caso de Uso "Identificar Usuario"

Caso de uso: *Identificar usuario.*
Objetivos: Comprobar que el navegante que solicita el acceso al sistema es un usuario legítimo del sistema para permitirle o denegarle el acceso a éste.
Actores: Navegante.
Puntos de Extensión: Tipo de usuario.
Extends: ---
Includes: ---
Pre-condiciones: - El navegante quiere acceder al sistema.
Post-condiciones: - La identidad del navegante ha sido comprobada y se le ha concedido o denegado el acceso al sistema.
Flujo de Eventos
Principal: 1. ***Controlador Interfaz:*** Muestra un formulario para que el navegante introduzca los datos de acceso (nombre de usuario y contraseña). 2. ***Actor:*** Introduce el nombre de usuario y la contraseña en el formulario. 3. ***Sistema de identificación:*** Comprueba si el nombre de usuario y la contraseña pertenecen a un usuario del sistema. 4. ***Sistema de identificación:*** Si los datos son correctos devuelve la identidad del usuario y concede el acceso del usuario al sistema. 5. ***Controlador de interfaz:*** Construye una interfaz adecuada al tipo de usuario que ha accedido al sistema. 6. Finaliza el caso de uso.
Variaciones: 3.1 ***Sistema de identificación:*** Si los datos son incorrectos deniega el acceso al navegante. 3.2 ***Controlador de interfaz:*** Muestra un mensaje de error de datos de identificación incorrectos al navegante. Ir a **1**.

Tabla 2. Descripción del caso de uso Identificar Usuario

Caso de Uso "Seleccionar Tutorial"

Caso de uso: *Seleccionar Tutorial.*
Objetivos: Permitir a un usuario de SITUA seleccionar el tutorial al que quiere acceder de una lista de tutoriales.
Actores: Usuario (Profesor, Alumno, Administrador).
Puntos de Extensión: ---
Extends: ---
Includes: ---
Pre-condiciones: ---
Post-condiciones: - El usuario ha seleccionado un tutorial de la lista de tutoriales.
Flujo de Eventos
Principal: 1. ***Servicio de tutorial:*** Devuelve los tutoriales disponibles en el sistema. 2. ***Controlador de interfaz:*** Muestra una lista con los tutoriales disponibles en el sistema. 3. ***Usuario:*** Selecciona un tutorial de la lista de tutoriales. 4. Finaliza el caso de uso.
Variaciones: ---

Tabla 3. Descripción del caso de uso Seleccionar Tutorial

Caso de Uso "Registrarse en Tutorial"

Caso de uso: *Registrarse en Tutorial.*
Objetivos: Permitir a un alumno de SITUA registrase en tutorial al que no tiene acceso.
Actores: Alumno.
Puntos de Extensión: ---
Extends: ---
Incluyes: ---
Pre-condiciones: - El alumno no tiene acceso al tutorial seleccionado.
Post-condiciones: - El alumno se ha registrado en el tutorial seleccionado.
Flujo de Eventos
Principal: 1. ***Alumno:*** Selecciona la opción de registrarse en el tutorial seleccionado. 2. ***Sistema de identificación:*** Asigna el alumno al tutorial seleccionado. 3. Finaliza el caso de uso.
Variaciones: ---

Tabla 4. Descripción del caso de uso Registrarse en tutorial

6.3.1.2. Caso de Uso del Proceso de Negocio "Contenidos"

En la figura 19 se presenta una descripción del caso de uso Contenidos.

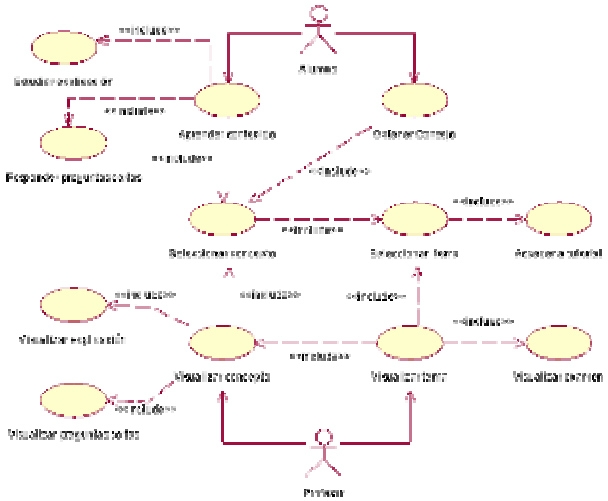


Figura 19. Diagrama de casos de uso de Contenidos

Caso de Uso "Seleccionar Tema"

Caso de uso: *Seleccionar tema.*
Objetivos: Permitir a un alumno seleccionar un tema de entre los que aparecen en la lista de temas que componen los contenidos del tutorial seleccionado previamente.
Actores: Alumno.
Puntos de Extensión: ---
Extendí: ---
Incluyes: Acceder tutorial.
Pre-condiciones: - Se ha mostrado al alumno una lista de temas disponibles en el tutorial.
Post-condiciones: - El alumno ha seleccionado el tema al que quiere acceder. - Se ha comprobado la viabilidad de que el alumno aprenda los conceptos del tema. - Se ha mostrado al alumno la lista de conceptos del tema.
Flujo de Eventos
Principal: 1. Pasos del CDU *Acceder tutorial.* 2. ***Alumno:*** Selecciona un tema de la lista de temas. 3. ***Agente Tutor:*** Comprueba si es viable que el alumno estudie los conceptos del tema. 4. ***Servicio de tutorial:*** Si es viable que el alumno aprenda los conceptos del tema, devuelve la lista de conceptos y el examen que pertenece al tema seleccionado. 5. ***Controlador de interfaz:*** Muestra al alumno un listado de los conceptos y el examen que pertenecen al tema seleccionado. 6. Finaliza el caso de uso.
Variaciones: ***Agente tutor:*** Si no es viable que el alumno aprenda los conceptos del tema se aconseja al alumno que no lo haga. ***Controlador de interfaz:*** Muestra un mensaje de error notificando al alumno que no es viable que aprenda los conceptos del tema. Ir a **1**.

Tabla 5. Descripción del caso de uso Seleccionar Tema

Caso de Uso "Aprender Contenido"

Caso de uso: *Aprender Contenido.*
Objetivos: Permitir a un alumno estudiar los contenidos (ya sean explicaciones o preguntas cortas) de un concepto previamente seleccionado y que pertenece a un tema del tutorial. Al mismo tiempo se realiza una evaluación del proceso de aprendizaje del alumno por parte del sistema.
Actores: Alumno.
Puntos de Extensión: ---
Extendí: ---
Incluyes: Seleccionar concepto, Estudiar explicación, Responder pregunta corta.
Pre-condiciones: - El alumno debe haber seleccionado el concepto que desea aprender.
Post-condiciones: - El alumno ha aprendido los contenidos del concepto. - Se ha realizado una evaluación automática por parte del sistema, del proceso de aprendizaje del alumno.
Flujo de Eventos
Principal: 1. Pasos del CDU *Seleccionar concepto*. 2. ***Agente de adaptación:*** Selecciona el contenido a mostrar al alumno que puede ser una explicación teórica o una pregunta corta de selección. 3. Si el contenido es una explicación realizar los pasos del CDU *Estudiar explicación*. 4. Si el contenido es una pregunta corta realizar los pasos del CDU *Responder pregunta corta*. 5. ***Agente de adaptación:*** Comprueba si hay más contenidos para mostrar al alumno. 6. ***Agente de orientador:*** Si no hay más contenidos que mostrar, realiza una evaluación global del alumno. 7. ***Agente tutor:*** Actualiza el estado del concepto que mantiene el alumno al aprender el concepto actual. 8. ***Agente tutor:*** Actualiza el modelo pedagógico del alumno según los conocimientos adquiridos en el concepto actual. 9. ***Agente tutor:*** Actualiza el estado del tutorial mantenido por el alumno. 10. Finaliza el caso de uso.
Variaciones: Si hay más contenidos para mostrar (ir a **2**).

Tabla 6. Descripción del caso de uso Aprender Contenido

Caso de Uso "Estudiar Explicación"

Caso de uso: *Estudiar Explicación.*
Objetivos: Permitir al alumno estudiar la explicación teórica perteneciente al concepto que está estudiando y que le es proporcionada por el agente de concepto. Al mismo tiempo el agente de concepto monitorizará el tiempo que el alumno emplea en el aprendizaje de la explicación.
Actores: Alumno.
Puntos de Extensión: ---
Extendí: ---
Incluyes: ---
Pre-condiciones: - El contenido seleccionado por el agente de concepto debe ser una explicación textual.
Post-condiciones: - El alumno ha aprendido la explicación. - Se ha actualizado el estado de la explicación con el tiempo empleado por el alumno.

Flujo de Eventos
Principal: 1. ***Controlador de interfaz:*** Muestra el contenido de la explicación al alumno. 2. ***Alumno:*** Visualiza y estudia la explicación. 3. ***Agente orientador:*** Al mismo tiempo que el alumno visualiza la explicación, el agente de concepto monitoriza el tiempo que el alumno emplea en estudiar la explicación hasta que el alumno solicite el siguiente contenido. 4. ***Alumno:*** Solicita el siguiente contenido. 5. ***Agente tutor :*** Actualiza el estado de la explicación que mantiene el alumno, con el tiempo empleado en aprender la explicación. 6. Finaliza el caso de uso.
Variaciones: ---

Tabla 7. Descripción del caso de uso Estudiar explicación

Caso de Uso "Responder Preguntas Cortas"

Caso de uso: *Responder pregunta corta.*
Objetivos: Permitir al alumno responder a las preguntas cortas de selección que le hace el agente de concepto como parte de los contenidos del concepto que esta estudiando.
Actores: Alumno.
Puntos de Extensión: ---
Extends: ---
Includes: ---
Pre-condiciones: - El contenido seleccionado por el agente de concepto deben ser preguntas cortas de selección.
Post-condiciones: - El alumno respondió a las preguntas cortas de selección. - Se ha actualizado el estado de la explicación con las respuestas seleccionadas por el alumno.
Flujo de Eventos
Principal: 1. ***Controlador de interfaz:*** Muestra los enunciados de las preguntas de selección y las posibles respuestas a las preguntas, de las cuales una es verdadera. 2. ***Alumno:*** Visualiza los enunciados de las preguntas. 3. ***Alumno:*** Selecciona las respuestas. 4. ***Alumno:*** Solicita el siguiente contenido. 5. ***Agente de orientador:*** Evalúa las respuestas seleccionadas por el alumno. 6. ***Agente de tutor:*** Actualiza el estado de la explicación que mantiene el alumno, con las respuestas seleccionadas por éste. 7. Finaliza el caso de uso.
Variaciones: ---

Tabla 8. Descripción del caso de uso Responder preguntas cortas

Caso de Uso "Obtener Consejo"

Caso de uso: *Obtener Consejo(Pedagógico).*
Objetivos: Permitir a un alumno poder obtener un consejo sobre cual es el siguiente paso que tiene realizar en el tutorial para finalizarlo de la forma más beneficiosa para él.
Actores: Alumno.
Puntos de Extensión: ---

Extends: ---
Includes: Seleccionar tema.
Pre-condiciones: - El alumno debe haber seleccionado el tema donde desea solicitar el consejo.
Post-condiciones: - Se ha mostrado al alumno el consejo adecuado para que realice el siguiente paso en el tutorial.
Flujo de Eventos
Principal: 1. Pasos del CDU *Seleccionar Tema*. 2. ***Alumno:*** Solicita al agente pedagógico que le proporcione un consejo. 3. ***Agente tutor:*** Analiza el estado del tutorial y el estado del tema que contiene la información sobre el estado de los conocimientos del alumno en el tutorial. 4. ***Agente tutor:*** Analiza el modelo pedagógico del alumno y determina el mejor paso a dar por el alumno en el tutorial. 5. ***Controlador de Interfaz:*** Muestra al alumno el mensaje del consejo proporcionado por el agente pedagógico. 6. Finaliza el caso de uso.
Variaciones: ---

Tabla 9. Descripción del caso de uso Obtener Consejo

Caso de Uso "Visualizar Tema"

Caso de uso: *Visualizar tema.*
Objetivos: Permitir a un profesor visualizar los contenidos (ya sean conceptos o exámenes) de un tema previamente seleccionado.
Actores: Profesor.
Puntos de Extensión: ---
Extends: ---
Includes: Seleccionar tema, Visualizar Concepto, Visualizar Examen.
Pre-condiciones: - El profesor debe haber seleccionado el tema que desea visualizar.
Post-condiciones: - Se han mostrado los contenidos del tema al profesor.
Flujo de Eventos
Principal: 1. Pasos del CDU *Seleccionar Tema*. 2. ***Profesor:*** Selecciona el contenido del tema que quiere visualizar. 3. Pasos del CDU *Visualizar Concepto* si el contenido es un concepto. 4. Pasos del CDU *Visualizar Examen* si el contenido es un examen. 5. ***Profesor:*** Elige si quiere visualizar un nuevo contenido (Ir a **2).** 6. Finaliza el caso de uso.
Variaciones: 5.1 ***Profesor:*** Elige no seleccionar un nuevo contenido (ir a **6**).

Tabla 10. Descripción del caso de uso Visualizar tema

6.3.1.3. Casos de Uso del proceso de negocio "Exámenes"

La figura 20 describe detalladamente el caso de uso Exámenes.

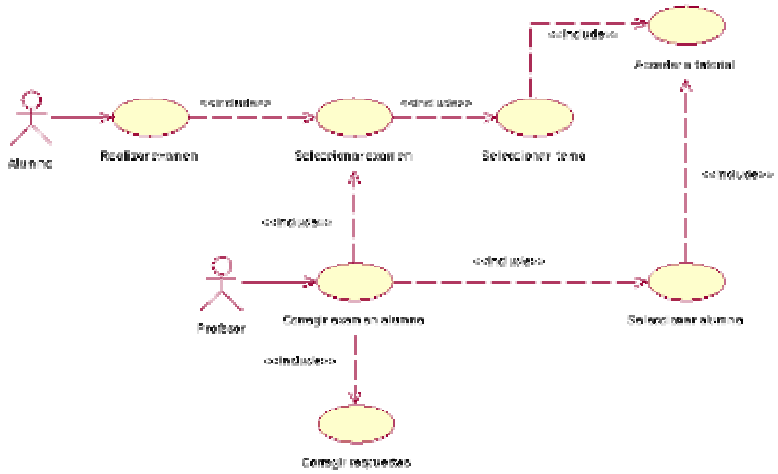


Figura 20. Diagrama de Caso de Uso Exámenes

Caso de Uso "Seleccionar Examen"

Caso de uso: *Seleccionar examen.*
Objetivos: Permitir a un usuario seleccionar un examen de un tema seleccionado previamente.
Actores: Usuario (Alumno, Profesor).
Puntos de Extensión: ---
Extends: ---
Includes: Seleccionar tema.
Pre-condiciones: - Se ha mostrado al usuario una lista con los contenidos del tema seleccionado previamente (conceptos y examen de tema).
Post-condiciones: - El usuario ha seleccionado el examen al que quiere acceder.
Flujo de Eventos
Principal: 1. Pasos del CDU *Seleccionar tema*. 2. ***Usuario:*** Selecciona el examen de tema de la lista de contenidos del tema. 3. ***Servicio de tutorial:*** Obtiene las preguntas del examen seleccionado por el usuario. 4. ***Controlador de Interfaz:*** Muestra al usuario, las preguntas del examen que ha seleccionado. 5. Finaliza el caso de uso
Variaciones: ---

Tabla 11. Descripción del caso de uso Seleccionar Examen

Caso de Uso "Realizar Examen"

Caso de uso: *Realizar examen.*
Objetivos: Permitir a un alumno realizar el examen de evaluación de un tema del tutorial para comprobar que realmente domina el tema.
Actores: Alumno.
Puntos de Extensión: ---

Extends: ---
Includes: Seleccionar examen.
Pre-condiciones: - El alumno debe haber seleccionado el examen del tema que quiere realizar.
Post-condiciones: - El alumno ha realizado el examen del tema. - Se ha actualizado el estado del examen que mantiene el alumno para ese examen. - Se ha notificado al profesor que tiene un examen pendiente de corrección.
Flujo de Eventos
Principal: 1. Pasos del CDU *Seleccionar examen*. 2. ***Alumno:*** Elabora las respuestas a las preguntas del examen seleccionado. 3. ***Controlador de interfaz:*** Muestra las respuestas elaboradas por el alumno. 4. ***Alumno:*** Si el alumno decide finalizar el examen (Ir a **5)**. 5. ***Agente tutor:*** Actualiza el estado del examen con las respuestas del alumno. 6. ***Agente tutor:*** Notifica al profesor a cargo del alumno que tiene un examen con preguntas pendientes por corregir. 7. Finaliza el caso de uso.
Variaciones: ***Alumno:*** El alumno desea responder de nuevo a las preguntas (ir a **2**).

Tabla 12. Descripción del caso de uso Realizar examen

Caso de Uso "Corregir Respuestas"

Caso de uso: *Corregir respuesta.*
Objetivos: Permitir a un profesor corregir y evaluar una respuesta elaborada por un alumno a una pregunta del examen.
Actores: Profesor.
Puntos de Extensión: ---
Extends: ---
Includes: ---
Pre-condiciones: - Se le han mostrado al profesor las preguntas del examen que ha realizado el alumno.
Post-condiciones: - El profesor ha corregido y evaluado la respuesta dada por el alumno a la pregunta seleccionada. - Se ha actualizado el estado de examen que mantiene el alumno.
Flujo de Eventos
Principal: 1. ***Profesor:*** Selecciona la pregunta del examen cuya respuesta del alumno quiere corregir. 2. ***Controlador de interfaz:*** Muestra la respuesta dada por el alumno a la pregunta. 3. ***Profesor:*** Comprueba si la respuesta dada por el alumno es correcta o no. 4. ***Profesor:*** Si la respuesta es correcta, evalúa la respuesta con un valor numérico que indica su grado de corrección. 5. ***Profesor:*** Elige si desea corregir otra respuesta del examen (Ir a **1)**. 6. Finaliza el caso de uso.
Variaciones: 4.1 ***Controlador de interfaz:*** Si la respuesta es incorrecta, muestra un formulario adecuado para que el profesor elabore una corrección si hace falta. 4.2 ***Profesor:*** Si la respuesta dada por el alumno es incorrecta elabora una corrección de la misma en el formulario. 4.3 ***Profesor:*** Evalúa la respuesta incorrecta elaborada por el alumno. (ir a **5**). ***Profesor:*** No elige corregir una nueva respuestas del examen (Ir a **6).**

Tabla 13. Descripción del caso de uso Corregir respuestas

6.3.2 Diagramas de Secuencia

Con los diagramas de secuencia del sistema se intenta determinar, a partir de los casos de uso, las operaciones demandadas por los actores del sistema. De esta manera se definen para cada caso de uso, los posibles eventos que los actores envían al sistema y que son los que desencadenan las secuencias de operaciones necesarias para llevar a cabo las tareas del caso de uso. En el Anexo B la totalidad de los diagramas de secuencia pueden ser revisados.

6.3.2.1. Diagrama de secuencia del sistema de "Identificar Usuario"

El proceso de identificar un usuario comienza en el momento en que un navegante cualquiera accede al formulario de identificación del sistema e introduce un nombre de usuario y una contraseña. Como el sistema admite tres tipos de usuarios diferentes (alumno, profesor y administrador), surgen tres escenarios donde el evento generado, da lugar a una secuencia de operaciones diferente según el tipo de usuario que lo genera. En la figura 21 el diagrama de secuencia es mostrado.

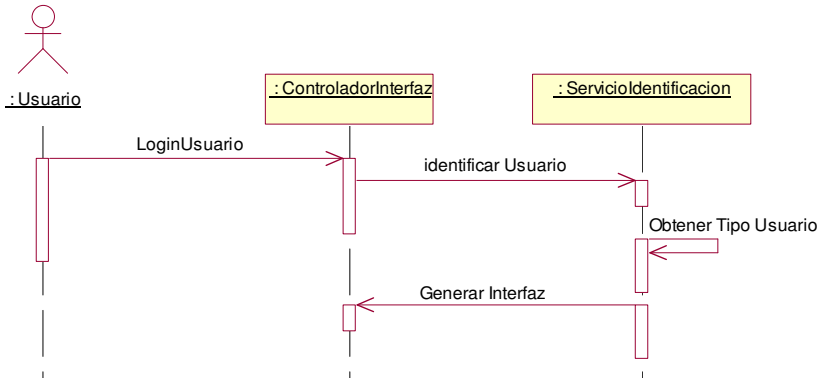


Figura 21. Identificar Usuario

6.3.2.2. Diagrama de secuencia del sistema de "Seleccionar Tutorial"

Este evento se genera cuando un usuario de SITUA, solicita al sistema un listado de los tutoriales disponibles en éste y selecciona uno de ellos. En la figura 22 el diagrama de secuencia Seleccionar Tutorial es presentado.

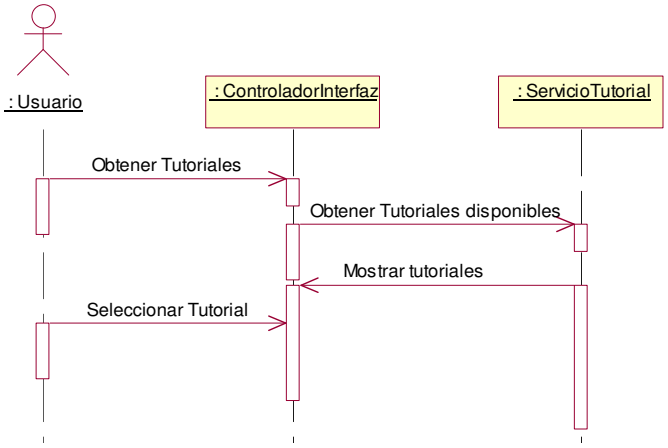


Figura 22. Seleccionar Tutorial

6.3.2.3. Diagrama de secuencia del sistema de "Registrarse en Tutorial"

Este evento se genera cuando un alumno, solicita al sistema el ser registrado en uno de los tutoriales que ha seleccionado previamente. En la figura 23 se muestra el diagrama de secuencia Registrarse en Tutorial.

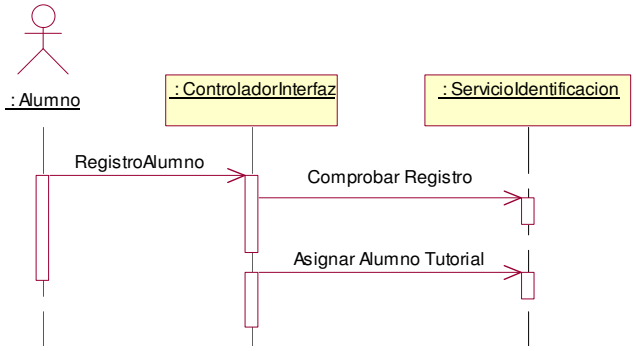


Figura 23. Registrarse en Tutorial

6.3.2.4. Diagrama de secuencia del sistema de "Seleccionar Tema"

El proceso de seleccionar un tema, comienza cuando un usuario (alumno o profesor) selecciona uno de los temas del tutorial en el que se encuentran. Dependiendo de si el usuario que selecciona el tema es un alumno o un profesor se generan eventos diferentes. A continuación se muestra el evento generado para el alumno, quien selecciona un tema para estudiar alguno de sus contenidos. La figura 24 presenta el diagrama Seleccionar Tema.

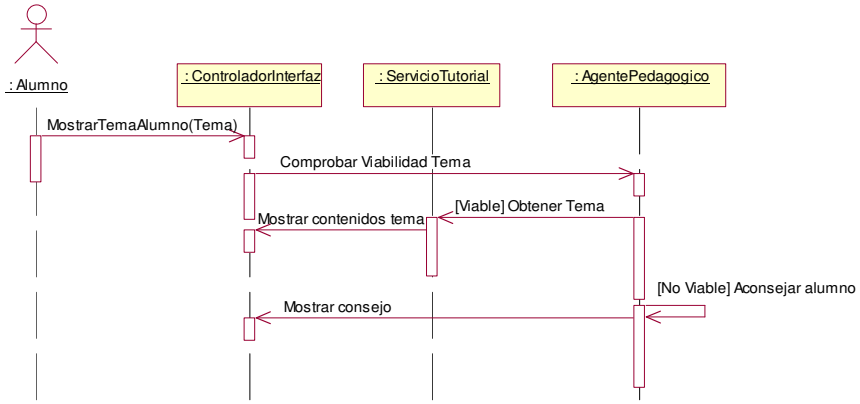


Figura 24. Seleccionar Tema

6.3.2.5. Diagrama de secuencia del sistema "Aprender Contenido"

Este evento se genera cuando un alumno aprende uno de los contenidos de un concepto que ha seleccionado previamente, bien sea una explicación o responder las preguntas cortas de una explicación. En la figura 25 se presenta el diagrama Aprender Contenido.

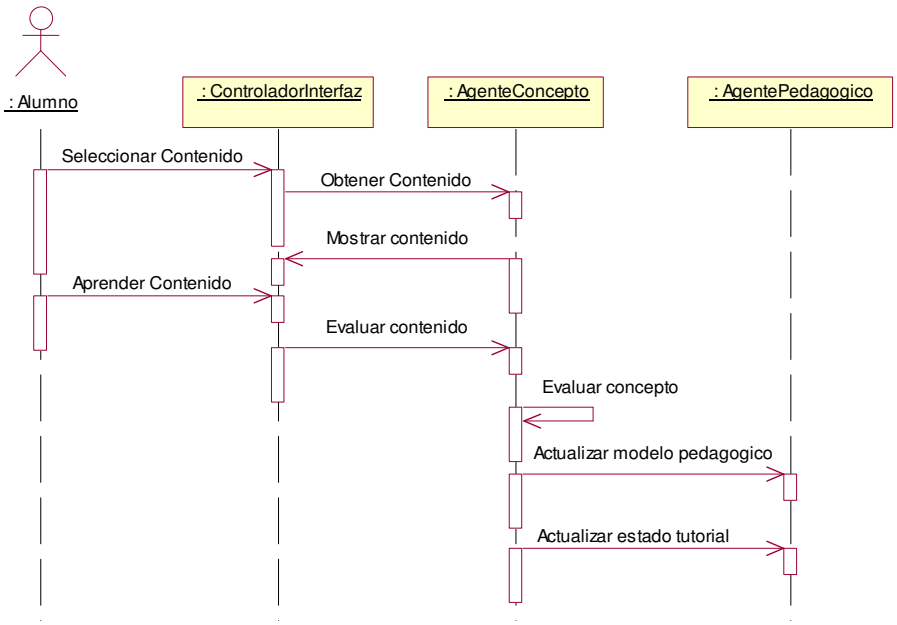


Figura 25. Aprender Contenido

6.3.2.6. Diagrama de secuencia del sistema "Estudiar Explicación"

Este evento se genera cuando un alumno aprende una explicación perteneciente a un concepto que ha seleccionado previamente. En la figura 26 se muestra el diagrama estudiar explicación.

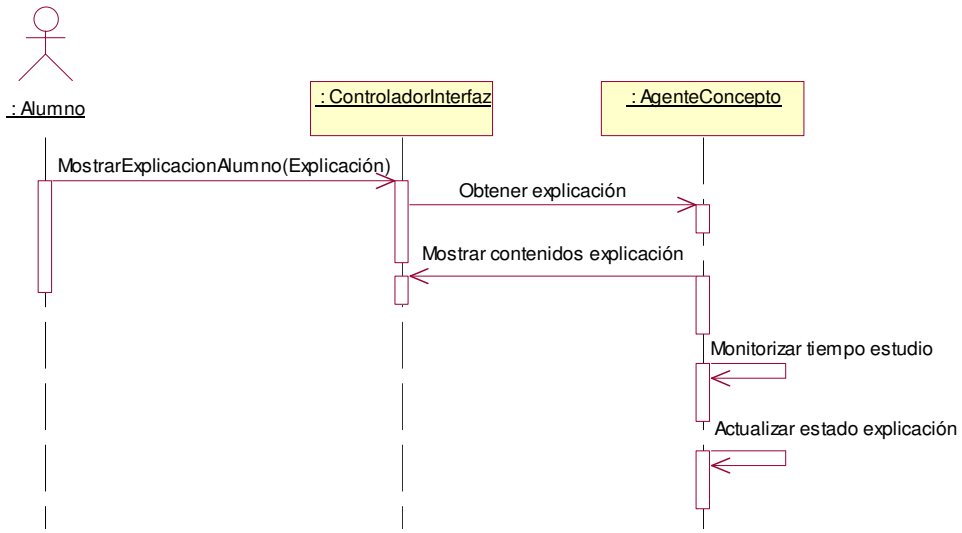


Figura 26. Estudiar explicación

6.3.2.7. Diagrama de secuencia del sistema "Responder Preguntas Cortas"

Este evento se genera cuando un alumno elige responder a las preguntas cortas de una explicación que ha estudiado previamente. En la figura 27 se presenta el diagrama de secuencia correspondiente.

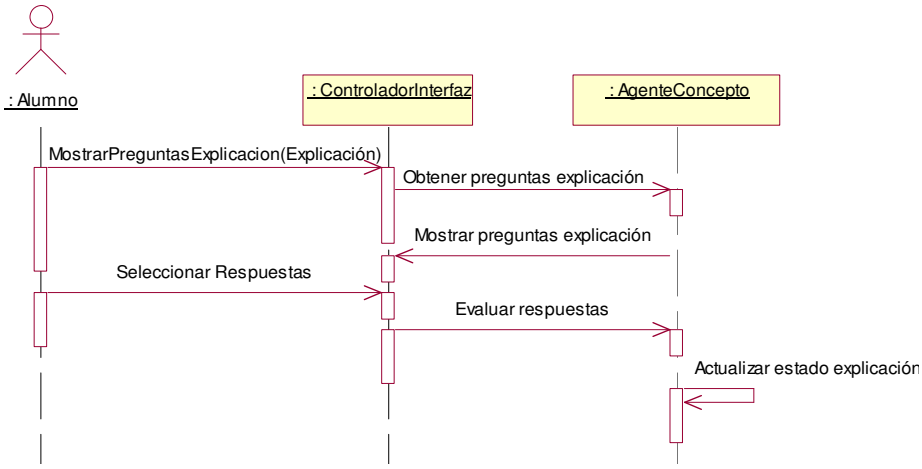


Figura 27. Responder preguntas cortas

6.3.2.8. Diagrama de secuencia del sistema de "Obtener consejo"

A continuación se muestran los diagramas de secuencia de los eventos generados por un alumno cuando solicita un consejo al sistema. El consejo puede ser solicitado cuando el alumno se encuentra estudiando un concepto determinado o cuando se encuentra en cualquier otro lugar del tutorial. La figura 28 muestra el diagrama de secuencia de obtener consejo y la figura 29 muestra el diagrama de secuencia de visualizar tema.

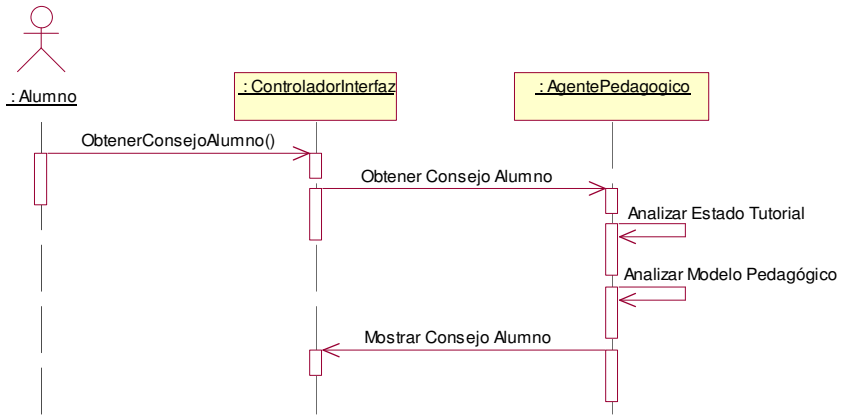


Figura 28. Obtener Consejo

6.3.2.9. Diagrama de secuencia del sistema de "Visualizar Tema"

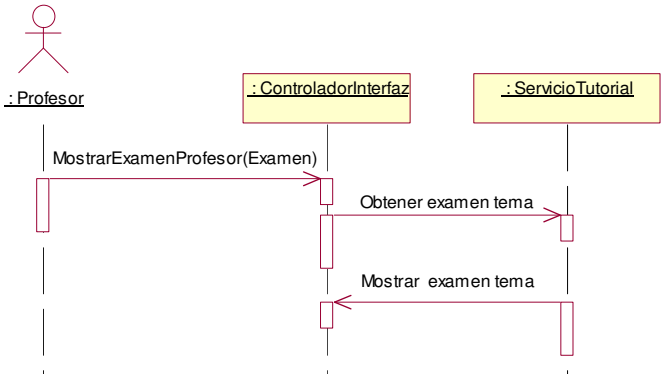


Figura 29. Visualizar Examen

6.3.2.10. Diagrama de secuencia del sistema de "Seleccionar Examen"

Este evento se genera cuando un usuario desea visualizar las preguntas de un examen perteneciente a un tema seleccionado previamente. En la figura 28 se muestra el diagrama correspondiente.

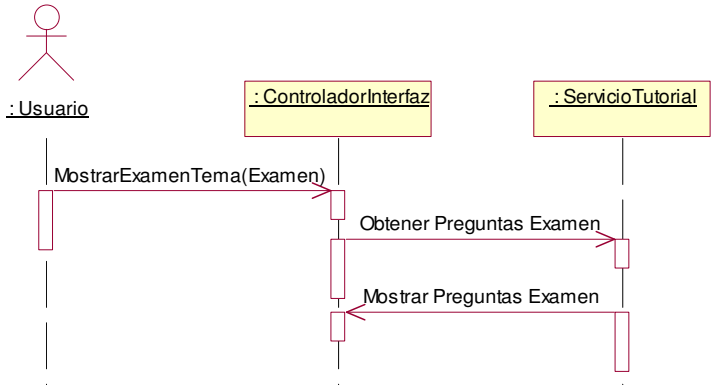


Figura 30. Seleccionar Examen

6.3.2.11. Diagrama de secuencia del sistema de "Corregir Examen"

Este evento se genera cuando un profesor corrige el examen del alumno que tiene a su cargo en un tutorial determinado. Los eventos generados con este caso de uso incluyen los eventos generados con el caso de uno Corregir Respuestas. La figura 31 presenta el diagrama de secuencia correspondiente.

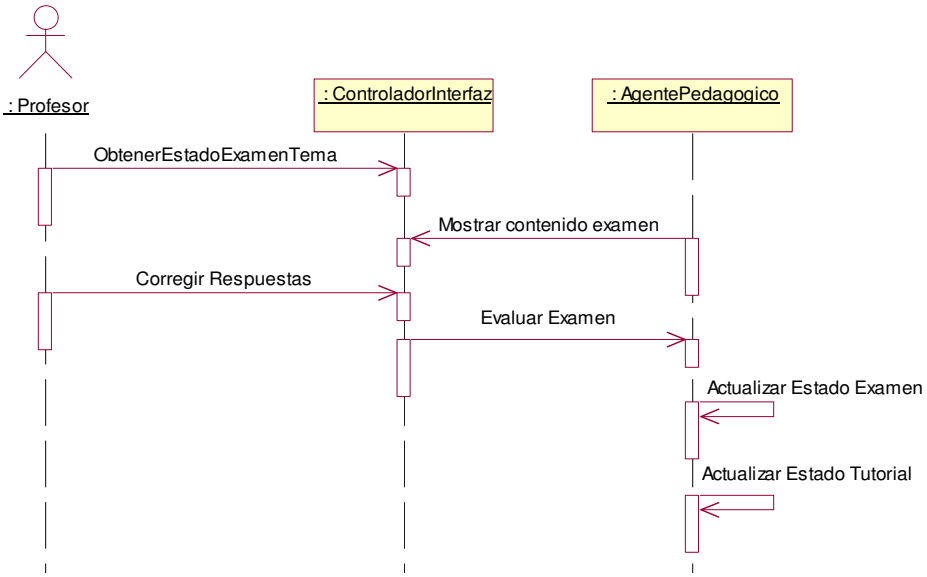


Figura 31. Corregir Examen

6.3.2.12. Diagrama de Secuencia de "Corregir Respuestas "

Este evento se genera cuando un profesor corrige las respuestas elaboradas por el alumno en contestación a las preguntas del examen seleccionado por el profesor. En la figura 32 se muestra el diagrama de secuencia corregir respuestas.

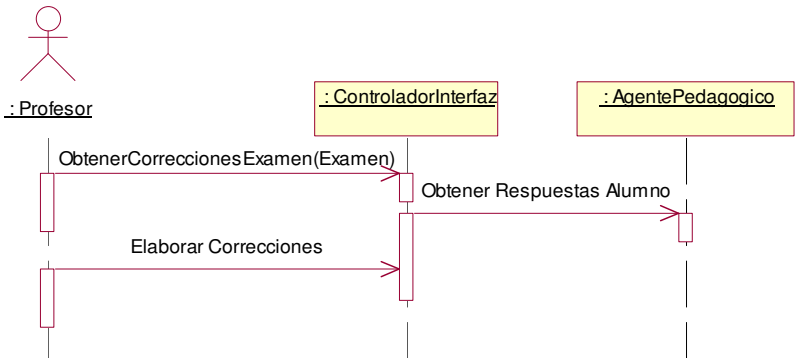


Figura 32. Corregir respuestas

6.3.3. Diagrama de Clases Inicial del sistema

En el diagrama de clases inicial del sistema, se representan las entidades o conceptos del dominio del problema obtenidos a partir de los casos de uso, y como se relacionan entre ellas. La mayoría de las entidades representadas en el diagrama darán lugar a algunas de las clases finales del sistema, que se representarán en el diagrama de clases de la fase de diseño. El diagrama de clases inicial del sistema es descrito en la figura 33.

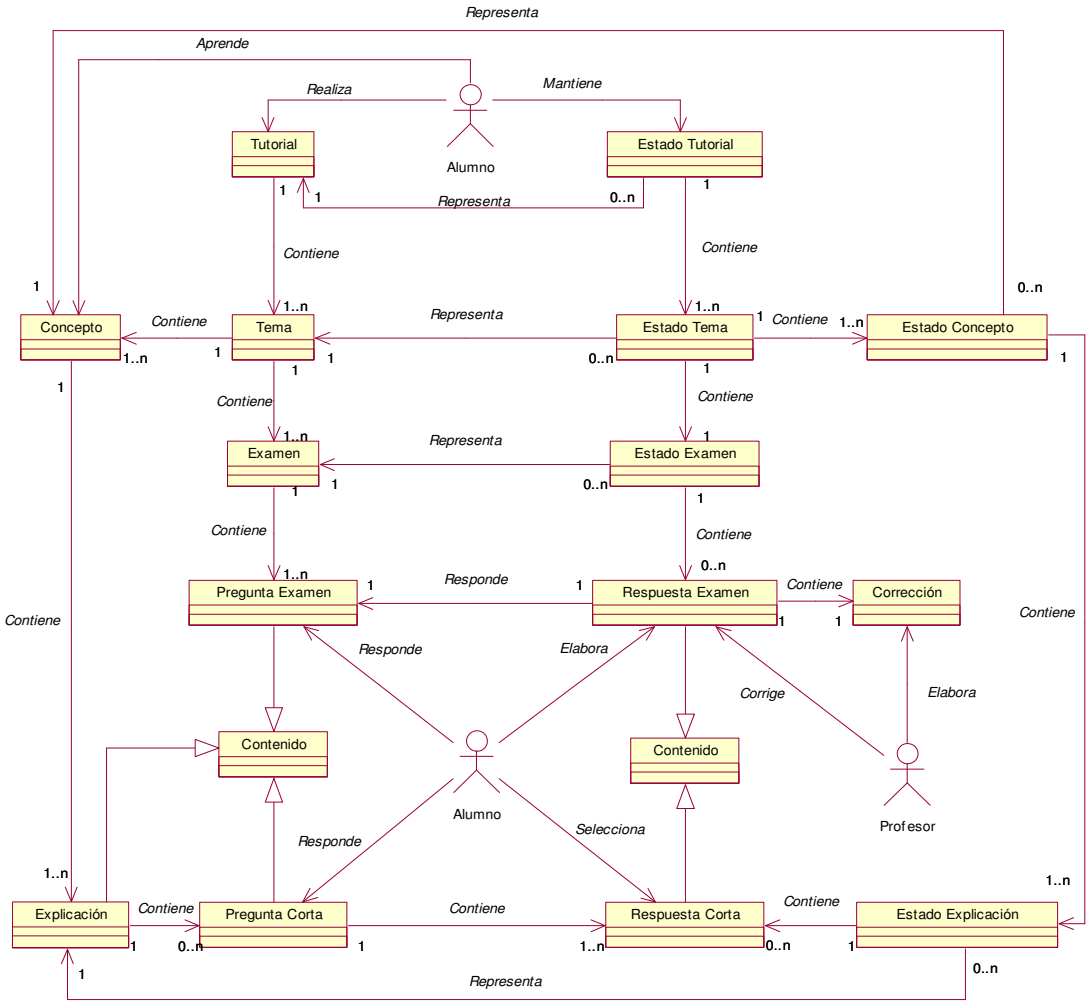


Figura 33. Diagrama de clases inicial

6.4 DISEÑO DEL SISTEMA

6.4.1. Diagrama General de Clases del Sistema

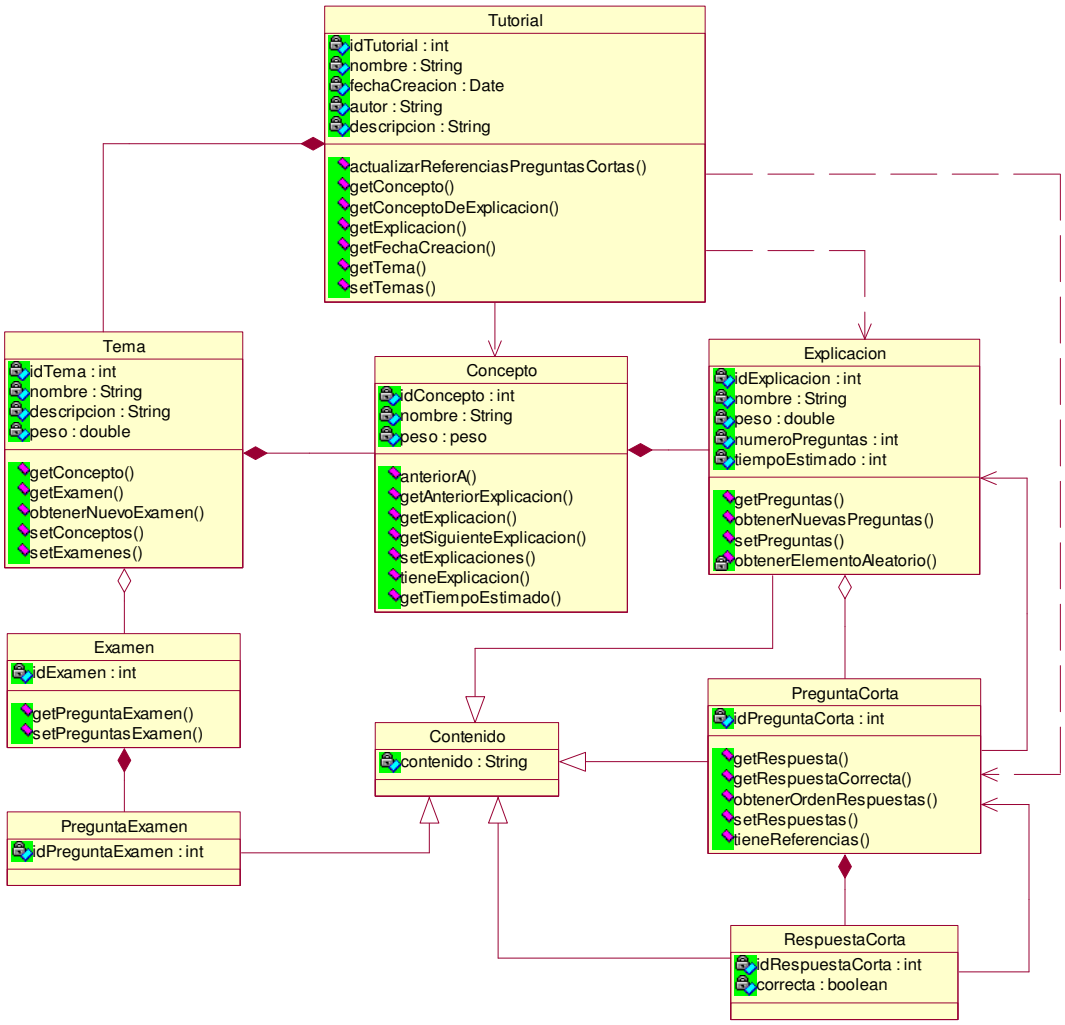


Figura 34. Diagrama general de clases

En la figura 34 se muestra el diagrama de clases completo de la estructura básica de un tutorial en SITUA. En el se muestran los distintos elementos que componen el tutorial y las relaciones estructurales entre ellos.

6.4.2. Diseño de la Base de Datos

En el sistema se utiliza una base de datos relacional para almacenar los contenidos de cada tutorial y la información sobre los estados de tutorial que mantienen cada uno de los alumnos que estudian los tutoriales. Se van a almacenar los contenidos de los temas, conceptos, explicaciones, preguntas cortas y exámenes pertenecientes a cada uno de los tutoriales de SITUA, al mismo tiempo que se va a almacenar la información sobre los estados de tutorial, estados de tema, estados de concepto, estados de explicación y estados de examen pertenecientes a cada uno de los alumnos para cada uno de los tutoriales que se encuentren estudiando en SITUA.

En la figura 35 se muestra el diagrama de entidad – relación que describe la base de datos utilizada en el sistema.

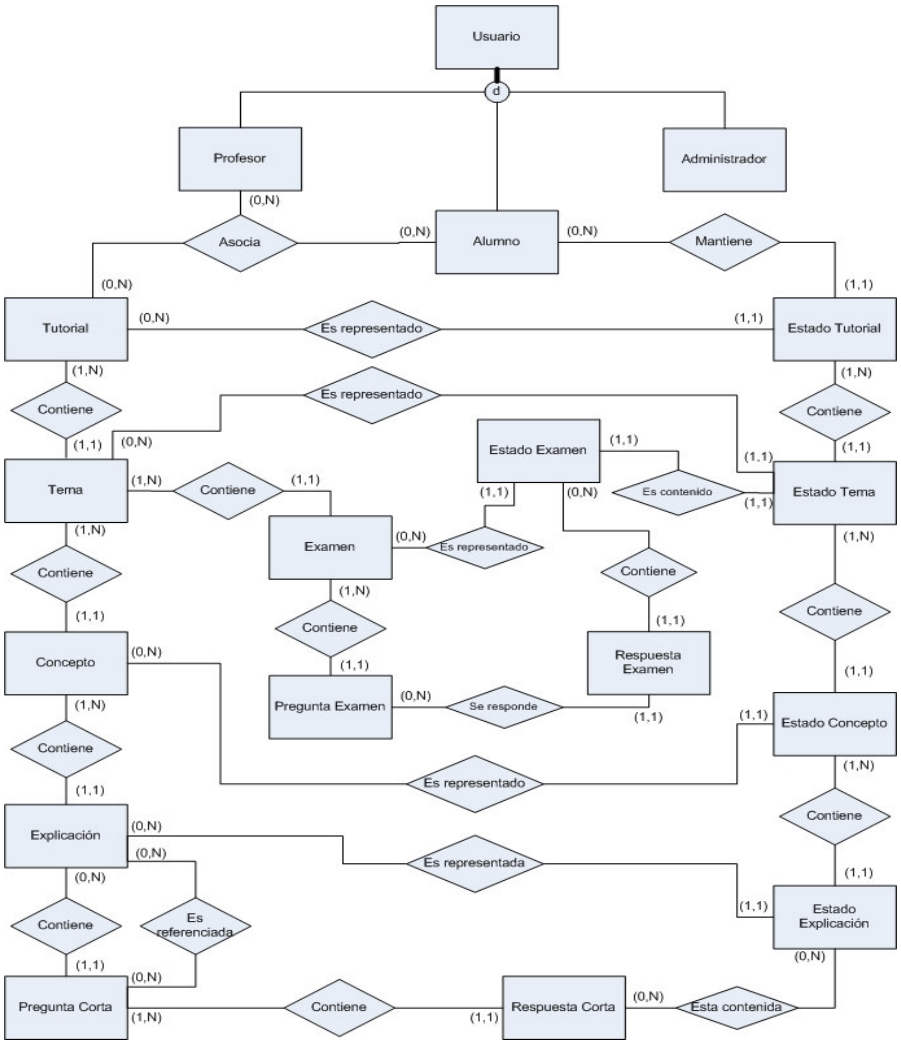


Figura 35. Diseño. Diagrama Entidad-Relación

6.4.2.1. Modelo Relacional

En la figura 36 se describe el modelo relacional realizado en el proceso de diseño del sistema.

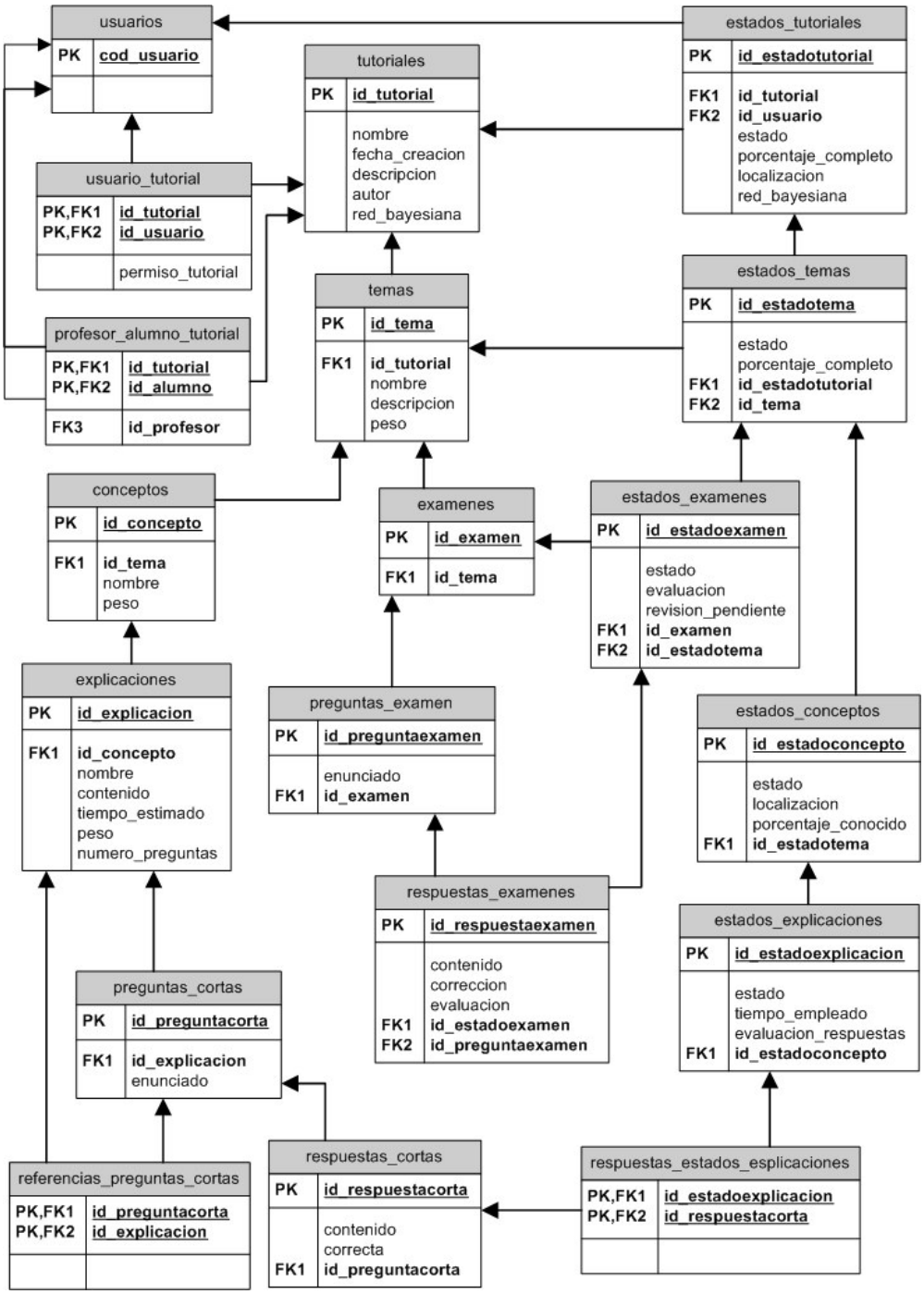


Figura 36. Diseño. Diagrama del modelo relacional

6.4.3. Creación de la Red Bayesiana

El proceso de creación de redes bayesianas se describe brevemente en la figura 37.

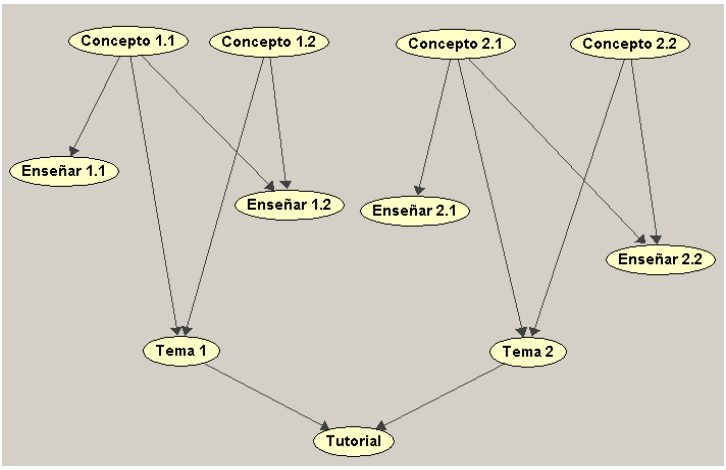


Figura 37. Estructura típica de una red bayesiana de un tutorial en SITUA.

En la figura, se muestra la estructura típica de una red bayesiana utilizada por los tutoriales de SITUA. Las redes que utilizan los tutoriales pueden diferir en mayor o menor medida de ésta pero siguen la misma estructura básica.

Para crear la red hay que tener en cuenta varias aspectos importantes que se comentan a continuación:

- El elemento **Tutorial** tiene un **nodo tutorial** asociado en la red (en la Figura es el nodo *Tutorial*).

- Cada uno de los elementos **Tema** definidos tiene asociado un **nodo de tema** en la red (en la Figura son los nodos *Tema1 y Tema2*).

- Cada uno de los elementos **Concepto** tiene asociado un **nodo de concepto** en la red (en la Figura son los nodos *Concepto1.1…Concepto2.2*).

- Cada uno de los elementos **Concepto** tiene asociado un **nodo enseñar concepto** (en la Figura son los nodos *Enseñar1.1…Enseñar2.2*).

Nodos Concepto.

Los nodos concepto son los que representan a los conceptos en la red. Son los que reciben los datos sobre el nivel de conocimientos que adquieren los alumnos en cada concepto del tutorial, por este motivo, el valor de conocimiento que representan estos nodos inicialmente es 0.

Como se indica en la tabla 14, estos nodos se deben crear con dos estados diferentes que albergan la distribución de probabilidades a priori para cada nodo concepto. El estado *Conocido* y el estado *Desconocido*. Como inicialmente, los conceptos van a ser desconocidos por los alumnos, al estado *Conocido* se le debe asignar un valor

cercano a 0, concretamente el valor 0.001, y al estado *Desconocido* [2] se le asigna la diferencia entre la unidad y el estado *Conocido*.

Conocido	0.0010
Desconocido	0.999

Tabla 14. Distribución de probabilidades a priori de los nodos concepto

Nodos Enseñar Concepto.

Los nodos enseñar concepto son los que indican la viabilidad de que el alumno estudie el concepto al que están asociados y que indican las dependencias que existen entre concepto que necesitan que el alumno conozca otros conceptos para ser estudiados.

Un nodo enseñar concepto tiene como mínimo un nodo padre que representa el concepto al que se controla la viabilidad de estudiarlo por parte del alumno. En la Figura 38 se representa la asociación más simple entre un nodo enseñar concepto y su nodo padre. En este caso el nodo *Enseñar 1.1* controla la viabilidad de que el alumno estudie el concepto representado por el nodo *Concepto 1.1.*

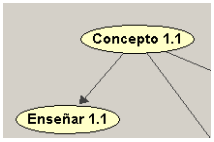


Figura 38. Nodo enseñar concepto simple.

Inicialmente, como los conceptos son desconocidos por el alumno, los nodos enseñar concepto como el de la Figura 3 deben indicar que es viable que el alumno comience a estudiar el concepto ya que dicho concepto no depende de que el alumno conozca previamente ningún otro concepto. Para ello la distribución de probabilidades a priori del nodo *Enseñar 1.1* se debe realizar de la manera que se indica en la Tabla 15. Distribución de probabilidades a priori de un nodo enseñar
.

Concepto 1.1	Conocido	Desconocido
Viable	0.0010	0.999
Inviable	0.999	0.0010

Tabla 15. Distribución de probabilidades a priori de un nodo enseñar

Mientras el *Concepto 1.1* sea desconocido, el nodo *Enseñar 1.1* será viable en un **0.999**, sin embargo cuando el alumno conoce el concepto, el nodo *Enseñar 1.1* indicara que no es viable que el alumno lo estudien en un **0.001**.

En el caso de que un nodo enseñar concepto tenga más de un padre, indica que el concepto al que representa el nodo enseñar concepto, depende de que el alumno conozca otros concepto para poder ser estudiado por el alumno. En la Figura 39 se muestra un ejemplo de este tipo de situación. El nodo *Enseñar 1.2* tiene dos padres.

[2] El estado *Desconocido* es calculado automáticamente por la herramienta al fijar el valor del estado *Conocido*.

Uno de los padres es el propio nodo *Concepto 1.2* al que representa y el otro padre es el nodo *Concepto 1.1*. Esto indica la viabilidad de que el alumno estudie el concepto 1.2 depende de si el alumno conoce previamente el concepto 1.1. Un nodo enseñar concepto podrá tener por lo tanto, un número de padres variable dependiendo de los conceptos que sean necesarios conocer, para poder estudiar el concepto al que representa.

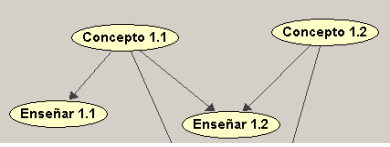


Figura 39. Nodo enseñar concepto múltiple.

En la se muestra la distribución de probabilidades a priori del nodo *Enseñar 1.2*, donde se puede apreciar como dicho nodo tendrá el estado viable sólo cuando el concepto 1.1 sea conocido por el alumno y el concepto 1.2 sea desconocido. En los demás casos, su estado será inviable.

Concepto 1.2	Conocido	Conocido	Desconocido	Desconocido
Concepto 1.1	Conocido	Desconocido	Conocido	Desconocido
Viable	0.0010	0.0010	0.999	0.0010
Inviable	0.999	0.999	0.0010	0.999

Tabla 16. Distribución de probabilidades de un nodo enseñar concepto múltiple

Nodos Tema.

Los nodos tema son los que representan a los temas en la red. Cada nodo tema tendrá un número de padres igual al número de conceptos que tenga el tema en el tutorial. Todos los padres de un nodo tema van a ser nodos concepto. De esta manera, a medida que el alumno adquiere conocimientos en cada concepto, aumentará el nivel de conocimientos en el tema.

Para crear la distribución de probabilidades de un nodo tema, hay que utilizar los valores de los **pesos de cada concepto en el tema**. Las tablas 17 y 18 representan la distribución de probabilidades a priori del nodo *Tema1*, nodo que tiene como padres los nodos *Concepto 1.1* y *Concepto 1.2*. El valor de conocimiento del nodo *Tema1* será de un 0.7 cuando el alumno conoce el *Concepto 1.1* y desconoce el *Concepto 1.2*. El valor 0.7 es el peso del *Concepto 1.1* en el *Tema 1*. Lo mismo ocurre con el *Concepto 1.2* que tiene un peso de 0.3 en el *Tema 1*. Cuando el alumno conoce el *Concepto 1.2* y desconoce el *Concepto 1.1*, el valor de conocimiento del *Tema 1* será de 0.3.

Concepto 1.1	Conocido	Conocido	Desconocido	Desconocido
Concepto 1.2	Conocido	Desconocido	Conocido	Desconocido
Conocido	0.999	0.7	0.3	0.0010
Desconocido	0.0010	0.3	0.7	0.999

Tabla 17. Distribución de probabilidades a priori de los nodos tema

Nodo Tutorial

El nodo tutorial representa al tutorial en la red. El nodo tutorial tendrá un número de padres igual al número de temas que tenga el tutorial. Todos los padres de un nodo tutorial van a ser nodos tema. De esta manera, a medida que el alumno adquiere conocimientos en cada tema, aumentará el nivel de conocimientos en el tutorial.

Para crear la distribución de probabilidades de un nodo tutorial, hay que utilizar los valores de los **pesos de cada tema en el tutorial**. La Tabla 18. Distribución de probabilidades a priori del nodo tutorial representa la distribución de probabilidades a priori del nodo *Tutorial*, nodo que tiene como padres los nodos *Tema1* y *Tema2*. El valor de conocimiento del nodo *Tutorial* será de un 0.45 cuando el alumno conoce el *Tema1* y desconoce el *Tema2*. El valor 0.45 es el peso del *Tema1* en el *Tutorial*. Lo mismo ocurre con el *Tema2* que tiene un peso de 0.55 en el *Tutorial*. Cuando el alumno conoce el *Tema2* y desconoce el *Tema1*, el valor de conocimiento del *Tutorial* será de 0.55.

Tema 1	Conocido	Conocido	Desconocido	Desconocido
Tema 2	Conocido	Desconocido	Conocido	Desconocido
Conocido	0.999	0.45	0.55	0.0010
Desconocido	0.0010	0.55	0.45	0.999

Tabla 18. Distribución de probabilidades a priori del nodo tutorial

6.4.4. Plataforma de Agentes

La plataforma de agentes se encarga de facilitar el acceso al sistema de agentes de SITUA. Los servicios proporcionados por la plataforma de agentes son los siguientes:

- Facilitar métodos para iniciar y detener la plataforma de agentes cuando se inicia o detiene el servidor donde se ejecuta SITUA.
- Permitir la creación e inicialización de los agentes necesarios.
- Proporcionar al servicio de agentes, mecanismos que permitan la comunicación del resto del sistema con los agentes de la plataforma.

En la figura 40 se muestra el diagrama con las clases que forman parte de la plataforma de agentes de SITUA. Está formada por una factoría de plataforma principal representada por la clase abstracta *PlataformaFactory,* la cual define los métodos que proporcionan los servicios de la plataforma. Estos métodos son implementados por la factoría concreta representada por la clase Plataforma jade que es una subclase de la anterior.

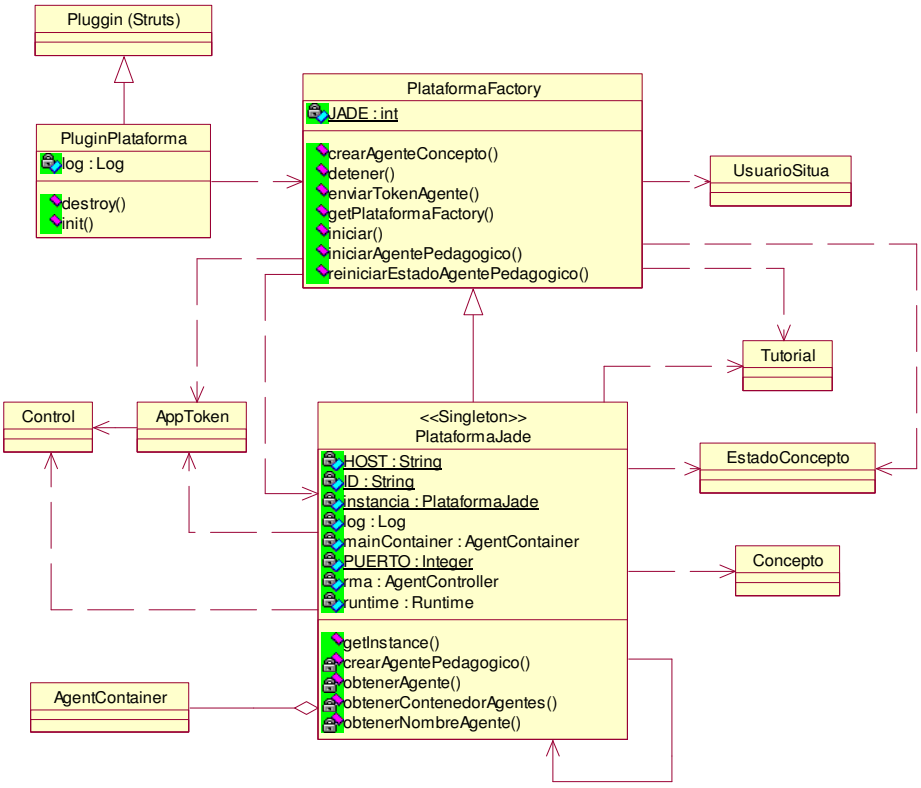


Figura 40. Diseño. Diagrama de clases de la plataforma de agentes.

La factoría concreta utiliza el **Framework JADE** *(Java Agent Development Framework)* [125] para la implementación de los métodos que proporcionan los servicios de la plataforma. Este framework es el mismo que se utiliza para la implementación de los agentes. De esta manera se aprovechan las características y servicios que proporciona la plataforma de agentes implementada con JADE, para crear, controlar y establecer comunicación entre los agentes de SITUA.

6.4.4.1. Mecanismos de Comunicación entre los agentes

El framework JADE proporciona una serie de clases contenidas en el paquete **jade.wrapper** que permiten la interacción entre una aplicación externa y los agentes de una plataforma JADE. Una de estas clases es la clase *AgentController*, que entre los métodos que proporciona, se encuentra el método ***putO2AObject(Object object);*** Este método permite el **envío de un objeto** al agente que está controlado por la instancia de la clase *AgentController* a la que pertenece el método.

Los usuarios de SITUA interaccionan con los servicios proporcionados por el sistema a través de una interfaz Web. Entre los usuarios del sistema, los alumnos y los profesores son los únicos que necesitan establecer algún tipo de comunicación con los agentes de la plataforma a través del servicio de agentes. Este servicio de agentes es considerado una aplicación externa a la plataforma de agentes y por lo tanto utilizará la funcionalidad que brinda el método ***putO2AObject(Object object);*** para comunicarse con ellos. Por otro lado, las funcionalidades que brindan los agentes de SITUA, requieren ciertos objetos de la aplicación (instancias de las clases *Tutorial,*

EstadoTutorial, Tema, EstadoTema, etc.) por lo que la necesidad de utilizar este método es evidente.

En la figura 41 se muestran las clases que se han implementado para facilitar la comunicación entre los distintos subsistemas de SITUA y los agentes de la plataforma, o incluso para la comunicación entre los propios agentes de la plataforma cuando esta comunicación implica el envío de algún objeto al agente destinatario.

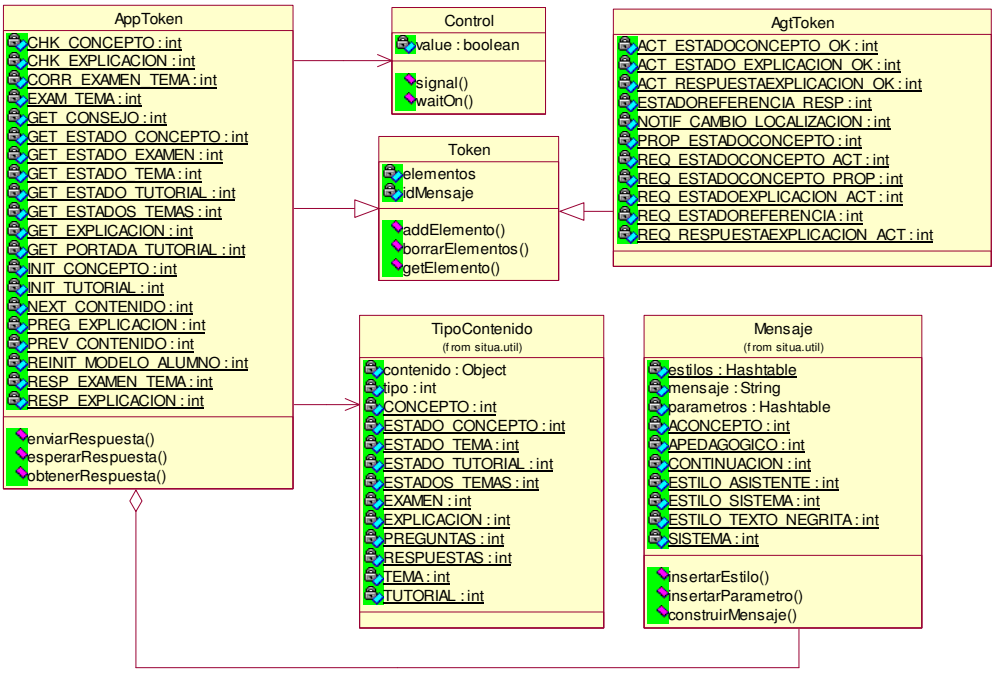


Figura 41 . Diseño. Diagrama de clases de los elementos de comunicación con los agentes.

6.4.5. Diagrama de Paquetes

6.4.5.1. Paquete situa.tutorial

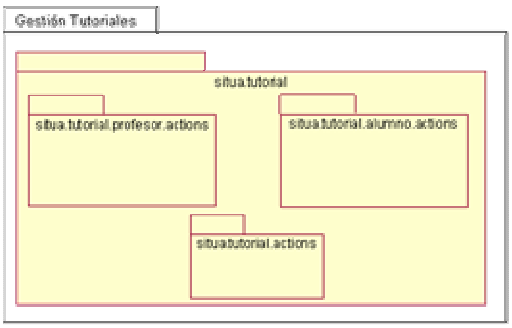


Figura 42. Implementación. Diagrama de componentes del paquete situa.tutorial

En la figura 42 se describe el paquete *situa.tutorial* el cual contiene las clases que forman parte de la estructura de elementos de un tutorial y de la estructura de elementos de los estados de tutorial. Estas clases son necesarias para trabajar con los contenidos de los tutoriales y con la información pedagógica de los alumnos en los demás paquetes del sistema. También incluye la clase que representa a la factoría de tutoriales, cuya función es proporcionar a los demás paquetes operaciones básicas para acceder a los tutoriales y los estados de tutorial que maneja el sistema.

Las clases contenidas en el paquete *situa.tutorial* son:

- *Concepto*
- *Contenido*
- *Corrección*
- *Examen*
- *Explicacion*
- *PreguntaCorta*
- *PreguntaExamen*
- *RespuestaCorta*
- *RespuestaExamen*
- *Tema*
- *Tutorial*
- *TutorialFactory*
- *EstadoConcepto*
- *EstadoExamen*
- *EstadoExplicacion*
- *EstadoTema*
- *EstadoTutorial*

El paquete *situa.tutorial* incluye tres paquetes adicionales:

- *situa.tutorial.actions*
- *situa.tutorial.alumno.actions*
- *situa.tutorial.profesor.actions*

6.4.5.2. Paquete situa.tutorial.actions

El paquete *situa.tutorial.actions* contiene un conjunto de clases que forman parte del controlador del sistema. Estas clases son las encargadas de atender las peticiones procedentes del administrador del sistema cuando se encuentra realizando operaciones de importación / exportación de los contenidos de los tutoriales del sistema.

Las clases contenidas en el paquete *situa.tutorial.actions* son:

- *ObtenerTutoriales*
- *ImportarTutorialXml*
- *ImportarRedBayesiana*
- *ExportarTutorialXml*
- *ImportarPreguntasCortasXml*

- *ExportarPreguntasCortasXml*
- *ImportarExamenesXml*
- *ExportarExamenesXml*
- *EliminarTutorial*
- *ObtenerTutorial*

6.4.5.3. Paquete situa.tutorial.alumno.actions

El paquete *situa.tutorial.alumno.actions* contiene las clases que forman parte del controlador del sistema y que son las encargadas de capturar y atender las peticiones que realizan los alumnos al sistema cuando se encuentran estudiando cualquiera de los tutoriales del sistema.

Las clases contenidas en el paquete *situa.tutorial.alumno.actions* son:

- *IniciarTutorialAlumno*
- *MostrarTutorialAlumno*
- *MostrarTemaAlumno*
- *MostrarConceptoAlumno*
- *SiguienteContenidoAlumno*
- *AnteriorContenidoAlumno*
- *ObtenerRespuestasExplicacion*
- *MostrarPreguntasExplicacion*
- *ObtenerConsejoAlumno*
- *MostrarExamenTema*
- *ObtenerRespuestasExamen*
- *ObtenerEstadoTutorial*
- *ObtenerEstadoTema*
- *ObtenerEstadoConcepto*
- *MostrarExplicacionAlumno*

6.4.5.4. Paquete situa.tutorial.profesor.actions

El paquete *situa.tutorial.profesor.actions* contiene las clases que forman parte del controlador del sistema. Estas clases son las encargadas de capturar y atender las peticiones que realizan los profesores al sistema cuando se encuentran corrigiendo los exámenes de los alumnos que tienen a su cargo en un tutorial determinado o cuando están visualizando los contenidos de cualquier tutorial al que tienen accedo.

Las clases contenidas en el paquete *situa.tutorial.profesor.actions* son:

- *ObtenerAlumnosTutorial*
- *ObtenerEstadosTemasAlumno*
- *AccederEstadoTutorialAlumno*
- *ObtenerEstadoExamenTema*
- *ObtenerCorreccionesExamen*
- *ObtenerInformacionAlumno*
- *ObtenerEstadoTutorialAlumno*
- *ObtenerEstadoTemaAlumno*

- *ObtenerEstadoConceptoAlumno*
- *MostrarTutorialProfesor*
- *MostrarTemaProfesor*
- *MostrarConceptoProfesor*
- *MostrarExplicacionProfesor*
- *MostrarExamenProfesor*
- *MostrarPreguntasCortasProfesor*

6.4.5.5. Paquete situa.dao

El paquete *situa.dao* se compone de las clases que implementan las operaciones que tienen que ver con el acceso a la base de datos del sistema y con la importación y exportación de datos en ficheros XML. La figura 43 describe el paquete de acceso a bases de datos.

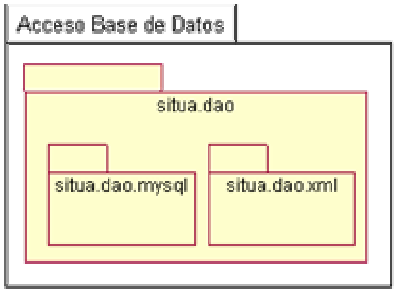


Figura 43. Implementación. Diagrama de componentes del paquete situa.dao.

Las clases incluidas en el paquete *situa.dao* son los interfazs que definen los métodos que van a implementar las clases incluidas en los dos paquetes contenidos en *situa.dao*. De esta manera se consigue definir una interfaz común para acceder a diferentes implementaciones de los servicios del paquete.

Las clases contenidas en el paquete *situa.dao* son:

- *DAOFactory*
- *MySqlDAOFactory*
- *XmlDAOFactory*
- *ExplicacionDAO*
- *EstadoTutorialDAO*
- *EstadoTemaDAO*
- *PreguntaCortaDAO*
- *EstadoExplicacionDAO*
- *ConceptoDAO*
- *EstadoConceptoDAO*
- *RespuestaCortaDAO*
- *TemaDAO*
- *TutorialDAO*
- *UsuarioDAO*
- *ExamenDAO*
- *EstadoExamenDAO*

- *PreguntaExamenDAO*
- *RespuestaExamenDAO*

El paquete *situa.dao* incluye tres paquetes adicionales:

- *situa.dao.mysql*
- *situa.dao.xml*

6.4.5.6. Paquete situa.dao.mysql

Las clases contenidas en este paquete están relacionadas con el acceso a la base de datos del sistema e implementan los métodos definidos en los interfazs del paquete *situa.dao*. Esta clases permiten realizar operaciones de inserción borrado y modificación de los contenidos del los tutoriales del sistema y de la información de los conocimientos y estados que los alumnos mantienen en los tutoriales que se encuentran estudiando. Al mismo tiempo permiten acceder a los datos de los usuarios del sistema que se encuentran almacenados en la base de datos.

Las clases contenidas en el paquete *situa.dao.mysql* son:

- *MySqlConceptoDAO*
- *MySqlEstadoConceptoDAO*
- *MySqlEstadoExplicacionDAO*
- *MySqlEstadoTemaDAO*
- *MySqlEstadoTutorialDAO*
- *MySqlExplicacionDAO*
- *MySqlPreguntaCortaDAO*
- *MySqlRespuestaCortaDAO*
- *MySqlTemaDAO*
- *MySqlTutorialDAO*
- *MySqlUsuarioDAO*
- *MySqlExamenDAO*
- *MySqlEstadoExamenDAO*
- *MySqlPreguntaExamenDAO*
- *MySqlRespuestaExamenDAO.*

6.4.5.7. Paquete situa.dao.xml

Las clases contenidas en este paquete están relacionadas con la importación / exportación de ficheros XML y que permiten introducir nuevos tutoriales en el sistema. Permiten importar los contenidos de los tutoriales, preguntas cortas de selección, exámenes y redes bayesianas almacenadas en ficheros externos y almacenar dichos contenidos en la base de datos. Al mismo tiempo permite exportar los contenidos de cualquier tutorial del sistema a ficheros XML.

Las clases contenidas en el paquete *situa.dao.xml* son:

- *ContenidoExplicacion*
- *DescripcionTema*
- *DescripcionTutorial*
- *XmlConcepto*

- *XmlExplicacion*
- *XmlTema*
- *XmlTutorial*
- *EnunciadoPreguntaCorta*
- *TextoRespuestaCorta*
- *XmlPreguntaCorta*
- *XmlPreguntasCortas*
- *XmlRespuestaCorta*
- *EnunciadoPreguntaExamen*
- *XmlExamen*
- *XmlExamenes*
- *XmlPreguntaExamen*

6.4.5.8. Paquete situa.agentes

El paquete *situa.agentes* incluye todas aquellas clases que forman parte de la plataforma de agentes del sistema y las clases empleadas para la comunicación entre los agentes de la plataforma y los demás subsistemas de SITUA o para la comunicación entre los propios agentes. En la figura 44 se describe el paquete correspondiente a la plataforma de agentes.

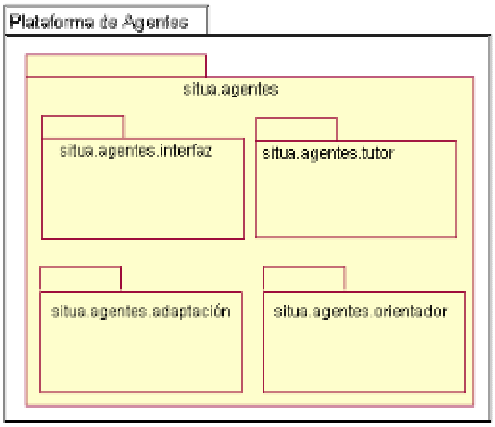


Figura 44. Implementación. Diagrama de componentes del paquete situa.agentes.

Las clases contenidas en el paquete *situa.agentes* son:

- *PlataformaFactory*
- *PlataformaJade*
- *Control*
- *PluginPlataforma*
- *AppToken*
- *Token*
- *AgtToken*

Adicionalmente, el paquete *situa.agentes* contiene dos paquetes con las clases relacionadas con los dos tipos de agentes del sistema. Los agentes pedagógicos y los agentes de concepto. Estos paquetes son:

- *situa.agentes.pedagogico*

- *situa.agentes.concepto*

6.4.5.9. Paquete situa.agentes.pedagogico

El paquete *situa.agentes.pedagogico* contiene las clases que definen el propio agente pedagógico y las clases que implementan las tareas que debe llevar a cabo el agente a lo largo de su existencia en la plataforma de agentes del sistema. Estas tareas son requeridas en los procesos desencadenados cuando los alumnos se encuentran estudiando los tutoriales del sistema.

Las clases contenidas en el paquete *situa.agentes.pedagogico* son:

- *IniciarTutorialBehaviour*
- *MostrarTutorialBehaviour*
- *MostrarTemaBehaviour*
- *GestorTareasPedagogico*
- *GestorMensajesPedagogico*
- *ActualizarEstadoConceptoBehaviour*
- *SolicitarConceptoBehaviour*
- *ActualizarEstadoExplicacionBehaviour*
- *ObtenerConsejoBehaviour*
- *EvaluarReferenciaBehaviour*
- *MostrarExamenBehaviour*
- *RecibirRespuestasExamenBehaviour*
- *ObtenerEstadosTemasBehaviour*
- *ObtenerEstadoExamenBehaviour*
- *RecibirCorreccionesExamenBehaviour*
- *MensajesConsejos*
- *ObtenerEstadoTutorialBehaviour*
- *ObtenerEstadoTemaBehaviour*
- *ObtenerEstadoConceptoBehaviour*
- *SolicitarExplicacionBehaviour*

. 6.4.5.10. Paquete situa.agentes.concepto

El paquete *situa.agentes.concepto* contiene las clases que definen el propio agente de concepto y las clases que implementan las tareas que ejecuta el agente cuando se encuentra enseñando conceptos del tutorial a un alumno.

Las clases contenidas en el paquete *situa.agentes.concepto* son:

- *AgenteConcepto*
- *GestorTareasConcepto*
- *GestorMensajesConcepto*
- *IniciarConceptoBehaviour*
- *SiguienteContenidoBehaviour*
- *AnteriorContenidoBehaviour*
- *EvaluarRespuestasBehaviour*
- *PreguntasExplicacionBehaviour*

- *CalcularTiempoBehaviour*
- *ObtenerConsejoBehaviour*
- *ContenidoExplicacionBehaviour*

6.4.5.11. Paquete situa.bn

El paquete *situa.bn* incluye todas aquellas clases permiten realizar las operaciones básicas con redes bayesianas que utilizan los agentes de concepto. Las clases de este paquete proporcionan métodos para trabajar con la estructura de las redes bayesianas utilizadas en SITUA. Al mismo tiempo incluye la clase que representa el modelo pedagógico de un alumno para un tutorial del sistema. Esta clase que representa el modelo, permite realizar operaciones de inferencia y propagación de probabilidades sobre las redes bayesianas del sistema, y hace de puente entre las estructuras que almacenan los estados de los tutoriales para cada alumno y las redes bayesianas que trabajan con la información pedagógica de dichos alumnos. La figura 45 describe el paquete situa.bn.

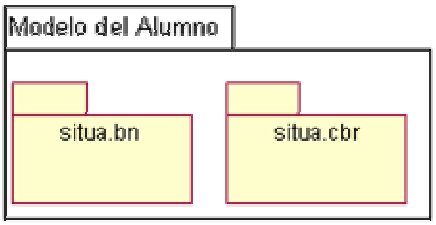


Figura 45. Implementación. Diagrama de componentes de situa.bn.

Las clases contenidas en el paquete *situa.bn* son:

- *BNFactory*
- *ElviraBNFactory*
- *PluginBNFactory*
- *ModeloAlumno*

6.4.5.12. Paquete situa.identificacion

El paquete *situa.identificacion* de la figura 46 contiene las clases que almacenan los datos de los usuarios del sistema. Al mismo tiempo contiene el paquete *situa.identificacion.actions* donde se incluyen las clases del controlador encargadas de gestionar las peticiones procedentes de los usuarios del sistema. Las peticiones que gestionan estas clases son las que están relacionadas con la identificación y control de acceso de los usuarios al sistema, con las operaciones de registro de usuarios en los tutoriales y con las operaciones de asignación de alumnos a los profesores de un tutorial.

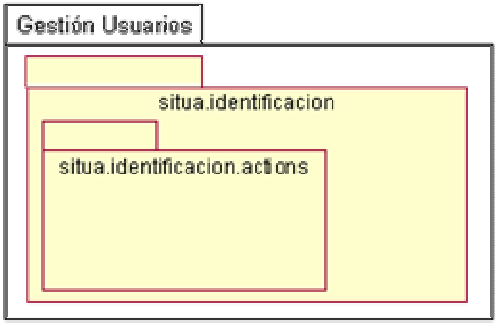


Figura 46. Implementación. Diagrama de componentes de situa.identificacion.

Las clases contenidas en el paquete *situa.identificacion* son:

- *Usuario*
- *UsuarioSitua*

Las clases contenidas en el paquete *situa.identificacion.actions* son:

- *GenerarInterfaz*
- *LoginUsuario*
- *ObtenerTipoUsuario*
- *ComprobarPermisosUsuario*
- *LogoutUsuario*
- *ObtenerProfesoresTutorial*
- *ObtenerProfesoresLibres*
- *AsignarProfesoresLibres*
- *DesasignarProfesorTutorial*
- *ObtenerAlumnosProfesor*
- *DesasignarAlumnoProfesor*
- *ObtenerAlumnosSinProfesor*
- *AsignarAlumnosProfesor*
- *ObtenerProfesor*
- *ObtenerAlumnosTutorial*
- *DesasignarAlumnoTutorial*
- *ObtenerAlumnosLibres*
- *AsignarAlumnosLibres*
- *ComprobarRegistroAlumno*
- *RegistroAlumno*
- *CancelarRegistroAlumno*

6.4.5.13. Paquete situa.servicios

El paquete *situa.servicios* contiene las clases que implementan el sistema de servicios de SITUA. Estas clases proporcionan al controlador acceso a los demás subsistemas para poder atender las peticiones de los usuarios del sistema, como se muestra en la figura 47.

Figura 47. Implementación. Diagrama de componentes de situa.servicios.

Las clases contenidas en el paquete *situa.servicios* son:

- *ServiciosFactory*
- *ServicioIdentificacionImpl*
- *ServicioIdentificacion*
- *ServicioTutorialImpl*
- *ServicioTutorial*
- *ServicioAgentes*
- *ServicioAgentesImpl*

6.4.5.14. Paquete situa.util

El paquete *situa.util* contiene clases auxiliares y clases de utilidad utilizadas las clases de los demás paquetes. Entre las clases que incluye el paquete se encuentran las que definen la clase Action personalizada de la que heredan las demás clases del controlador, así como una clase que define los formularios para subir ficheros al servidor. También se incluyen en este paquete las clases que definen los mensajes y los contenidos que se incluyen en los tokens de comunicación que utilizan los agentes de la plataforma para comunicarse con el resto de la aplicación. En la figura 48 se presnta el diagrama de componentes correspondiente.

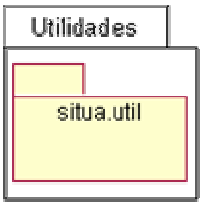


Figura 48. Implementación. Diagrama de componentes de situa.util.

Las clases contenidas en el paquete *situa.util* son:

- *BaseAction*
- *BaseForm*
- *Constantes*
- *ContenedorUsuario*
- *UploadForm*
- *ContenedorUsuario*
- *Mensaje*
- *Mensaje*
- *TipoContenido*

- *TipoContenido*

7. CONCLUSIONES Y TRABAJO FUTURO

En esta tesis se ha profundizado en el diseño de Sistemas de Tutoría Inteligentes, prestando especial atención al problema del modelado del alumno. A continuación vamos a resumir las principales aportaciones realizadas en este trabajo. Así mismo enunciaremos y comentaremos futuras líneas de investigación.

Principales Aportaciones

Las principales contribuciones originales de este trabajo son las siguientes:

- Se ha propuesto una nueva aproximación para el diseño de Sistemas de Tutoría Inteligentes. Mediante la combinación de diferentes técnicas de Inteligencia Artificial como Razonamiento Basado en Casos y Sistemas Multi-agente fue posible la definición de un modelo híbrido para la implementación de ITS complejos y dinámicos.

- Se han hecho dos propuestas para el modelamiento del alumno: Redes Bayesianas y Razonamiento Basado en Casos.

- Se ha realizado un estudio comparativo entre las dos técnicas usadas para modelamiento del estudiante. Este estudio puede ser revisado en: http://www-gist.det.uvigo.es/~cgonzals/ (ver disco compacto anexo a la monografía). La implementación se lleva a cabo dentro del modelo del estudiante en el sistema SITUA.

- Se concluye que las redes bayesianas son útiles para el modelado del estudiante por su gran versatilidad en modelado del alumno y porque constituyen una herramienta muy potente para realizar inferencias abductivas y predictivas. Sin embargo, durante su uso se encontraron algunas desventajas como: a) el esfuerzo realizado al especificar el modelo (variables y relaciones causales) y estimar los parámetros (probabilidades condicionadas), b) la complejidad computacional de los algoritmos de propagación, y c) la dificultad que supone la implementación de los mismos.

- Se elige el paradigma de Razonamiento Basado en Casos como la aproximación mas adecuada para el modelamiento del estudiante en ITSs debido a que permite: (a) facilidad en la actualización y mantenimiento del modelo del estudiante, beneficiando a tutores y estudiantes, (b) reflexión por parte de los estudiantes al recibir por parte del sistema un reporte de sus errores y las razones por las cuales ocurrieron, (c) facilidad de supervisión de los estudiantes habilitando al tutor a tener una base sólida y continua del desempeño del estudiante, incluyendo cuantitativa y cualitativa información. Lo anterior, teniendo en cuenta que CBR cuenta con una memoria permanente y numerosos métodos de búsqueda, haciendo disponible al diseñador gran volumen de experiencia. En este sentido a través de casos resueltos en el pasado el diseñador puede construir nuevas soluciones de calidad. Adicionalmente CBR permite gestionar gran cantidad de casos sin incrementar la complejidad del modelo.

- La utilización de sistemas de agentes como un mecanismo flexible que al trabajar de forma integrada con CBR permite una distribución del ciclo de vida y de la base de casos entre varios agentes, logrando un mecanismo de colaboración inicial que

permite mejorar el desempeño individual de los agentes y la calidad de sus soluciones.

- Se mejoran los resultados proveídos por el sistema SITUA, al implementarlo mediante la utilización de la arquitectura propuesta, y validando los resultados con estudiantes simulados en el aprendizaje de cursos en el dominio médico.

Como trabajo futuro se espera automatizar completamente las etapas del ciclo de vida CBR de forma que el sistema sea totalmente autónomo y no requiera intervención humana. Adicionalmente se plantea el ¿Cómo habilitar a un grupo de agentes con metas compartidas para comportarse cooperativa y coherentemente en dominios altamente complejos, dinámicos e inciertos?. Se pretende estudiar y plantear estrategias de coordinación y colaboración entre agentes que mejoren su desempeño individual y por ende la calidad de sus soluciones.

REFERENCIAS

[1]. McCarthy, J. Artificial Intelligence, Logic and Formalizing Common Sense. In Richmond Thomason. *Philosophical Logic and Artificial Intelligence*. Klüver Academic, 1989.

[2]. Brooks, R. A. Intelligence Without Representation. Artificial Intelligence 47 (1-3): 139-159. 1991.

[3]. Lagos, P. Inteligencia Artificial Distribuida y Razonamiento Basado en Casos en la arquitectura de un sistema basado en el conocimiento para la educación a distancia. Revista Ingeniería Informática. Vol. 9. 2003.

[4]. Moore, J, and Johnson, L. Artificial Intelligence in Education: AI-ED in the Wired and Wireless Future (Proc. of AI-ED2001 (Eds.), Frontiers in Artificial Intelligence and Applications, Volume 68, IOS Perss, 2001.

[5]. Baker, M. The roles of models in Artificial Intelligence and Education research: a prospective view. International Journal of Artificial Intelligence in Education. Vol 11, num 2. pp. 122 -143. 2000.

[6]. Nwana, S. Intelligent Tutoring Systems: An Overview. Artificial Intelligence Review. Num 4, pp. 251 – 277. 1990.

[7]. Ignizio, J. Introduction to Expert Systems: The development and implementation of rule-based expert systems. MacGraw-Hill, 402 pages. 1991.

[8]. Wooldridge, M. Introduction to MultiAgent Systems. John Wiley and Sons. 2002.

[9]. Aamodt, A and Plaza, E. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications. IOS Press, Vol. 7: 1, pp. 39-59. 1994.

[10]. Watson, I and Marir, F. Case-Based Reasoning: A Categorised Bibliography. The Knowledge Engineering Review. Volumen 9. No. 4. pp. 355-381. 1994.

[11]. Kolodner, J. L. Maintaining Organization in a Dynamic Long-Term Memory. Cognitive Science, 7 (4). pp. 243 -280. 1983.

[12]. Kolodner, J. L. Reconstructive Memory: A Computer Model. Cognitive Science, 7 (4). pp. 281. 1983.

[13]. Kolodner, J.L. Case-Based Reasoning. Moorgan Kaufmann. 1993.

[14]. Riverola, F, Corchado, J.M and Torres, J.M. An automated Hybrid CBR System for Forecasting. 6th European Conference on Case Based Reasoning: ECCBR 2002. Aberdeen, Scotland. 2002.

[15]. Wooldridge, M and Jennings, N. Intelligent Agents. Theory and Practice. Knowledge Engineering Review. 1995.

[16]. Sycara, K. Multi-agent systems. American Association for Artificial Intelligence. AI Magazine. 1998.

[17]. Johanssen, D.H and Reeves, T.C. Learning with technology: Using as cognitive tools. In D.H. Johanssen, Handbook of research for communications and technology. pp. 693 -719. 2004.

[18]. Anderson, J., Boyle, C., Farrell, R. and Reiser, B. "Cognitive principles in the design of computer tutors", in P. Morris (ed.), Modeling Cognition. NY: John Wiley. 1987.

[19]. Anderson, J.R. Cognitive Psychology and Intelligent Tutoring. P*roceedings of the Cognitive Science Society Conference*, Boulder, Colorado, pp. 37-43. 1984.

[20]. Bloom, B. The 2-sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher June, 13, 4-16. 1984.

[21]. Skinner, B. Are theories of learning necessary?. Psychological Review, 57. pp. 193 – 216. 1950

[22]. Bracey, G. W. "Computer-Assisted Instruction: What the Research Shows." Electronic Learning 7/3. 22-23. 1987.

[23]. Wenger, E. Artificial Intelligence and Tutoring Systems. Computational and Cognitive approaches to the communication of knowledge. Morgan Kaufman Ed. 1987.

[24]. Urretavizcaya, M. Sistemas Inteligentes en el ámbito de la educación. Revista Iberoamericana de Inteligencia Artificial. No. 12. pp. 5 -12. 2001.

[25]. Gros, S.B. Evaluación de los sistemas de automatización del diseño instructivo. Congreso iberoamericano de telemática. RIBIE, 2006.

[26]. Hartley, J and Sleeman, D. Towards more intelligent teaching systems. International Journal of Man-Machine Studies 2, 215-236. 1973.

[27]. VanLehn K. Student models. In Polson M.C. & Richardson J.J (eds). Foundation of Intelligent Tutoring Systems. Lawrence Erlbaum. Hillsdale. 1988.

[28]. Kearsley, G.P. Artificial Intelligence and Instruction. Addison Wesley. Massachusetts. 1987.

[29]. Anderson, J.R., Boyle, C.F and Yost, G. The Geometry Tutor. Proceedings of the International Joint Conference on Artificial Intelligence. Los Angeles. USA. 1985.

[30]. Brown, J. S. and VanLehn, K. Repair theory: A generative theory of bugs in procedural skill. *Cognitive Science, 4*, 379-426. 1980.

[31]. Carbonell, J. AI in CAI: An artificial intelligence approach to computer aided instruction. *Science*, (167):190-202. 1970.

[32]. Morgan, B. An Introduction to Bayesian Statistical Decision Processes. Prentice-Hall Inc., Englewood Cliffs, N.J. p. 15. 1968.

[33]. Mislevy, R.J. and Gitomer, D.H. The role of probability-based inference in an intelligent tutoring system. User-Modeling and User-Adapted Interaction, 5, 253 – 282. 1996.

[34]. Anderson, J.R. The architecture of cognition, Harvard University Press: Cambridge, Massachusetts -EE.UU. 1983.

[35]. Carmona, C, Millán, E, Pérez de la Cruz, J, Trella, M and Conejo R. Introducing Prerequisite Relations in a Multi-layered Bayesian Student Model. User Modeling 347-356. 2005.

[36]. Clancey, J. Knowledge-Based Tutoring – The GUIDON Program. MIT Press. Cambridge, 1987.

[37]. ShortLife, E.N. Computer-Based Medical Consultation: MYCIN. Elsevier, New York. 1976.

[38]. Self, J. "The defining characteristics of intelligent tutoring systems research: ITSs care, precisely". International Journal of Artificial Intelligence in Education, Vol. 10, num. 3-4, pp. 350-364. 1999.

[39]. VanLehn, K. Student modelling. En Polson, C., Richardson, J.J., (Editors). Intelligent Tutoring Systems. Pp. 55 -77. 1988.

[40]. Shute, V. Smart evaluation: Cognitive diagnosis, mastery learning and remediaton. In Proceedings of Artificial Intelligence in Education. pp. 123 – 130. 1995.

[41]. Bloom, B.S. The search for methods of group instruction as effective as one-to one tutoring educational leadership. 41 (8). Pp. 4 – 17. 1984.

[42]. Cook M. Student modelling in intelligent tutoring systems. Artificial Intelligence Review. vol. 7, pp. 227-240. 1993.

[43]. Ohlsson, S. Constraint-based Student Modeling. Student Modeling: the key to individualized knowledge-based instruction. Berlin. Springer-Verlag. 167 – 189. 1994.

[44]. Mitrovic, A., and Ohlsson, S. Evaluation of a Constraint-Based Tutor for a Database Language. International Journal of Artificial Intelligence in Education, 10. 1999.

[45]. Corbett, A.T, Anderson, J.R., Carver, V.H. and Brancolini, S.A. Individual differences and predictive validity in student modelling. In A. Ram & K. Eiselt (Eds.). The proceedings of the Sixteenth Annual Conference of the Cognitive Science Society. Hillsdale, NJ: Lawrence Erlbaum. 2000.

[46]. Baloian, N, Galdames, P, Collazos, C and Guerrero, L. A model for a collaborative recommeder system for multimedia learning material. LNCS, pp. 281-288. 2004.

[47]. González, M. A., Suthers, D. D. and Icaza, J. Designing and Evaluating a Collaboration Coach: Knowledge and Reasoning. In Artificial Intelligence in Education (Proc. 10th AI-ED, May 19-23, San Antonio Texas), J. D. Moore, C. L. Redfield, & W. L. Johnson (Eds.), Amsterdam: IOS Press, pp. 176-187. 2001.

[48]. Burton, R. The Environment Module of Intelligent Tutoring Systems (Capítulo V). En: Martha C. Polson and J. Jeffrey Richardson. Lea (eds.) Foundations of Intelligent Tutoring Systems, Hove & London. 1988.

[49]. Brusilovsky, P. and Cooper, D. W. ADAPTS: Adaptive hypermedia for a Web-based performance support system. In: P. Brusilovsky and P. De Bra (eds.) Proceedings of Second Workshop on Adaptive Systems and User Modeling on WWW at 8th International Word Wide Web Conference and 7-th International Conference on User Modeling, Toronto and Banff, Canada, May 11 and June 23-24, Computer Science Report # 99-07, Eindhoven University of Technology. 1999.

[50]. Weber, G. and Specht, M. User modeling and adaptive navigation support in WWW-based tutoring systems. Proceedings of User Modeling '97 (pp. 289-300). 1997.

[51]. Lopez, J.M, Millán, E, Pérez de la Cruz, J.L and Triguero F. "ILESA: Design and implementation of web-base tutoring tool for Linear programming problems". 2001.

[52]. Millán, E, Pérez de la Cruz, J.L, Triguero Ruiz, F and Vázquez, L. TUDER- An Intelligent Tutoring System for Symbolic derivation. LNCS. Vol. 1108. pp. 469. 1996.

[53]. Clancey, W.J and Letsinger, R. NEOMYCIN: Reconfiguring a rule-based expert system for application to teaching. In A. Drinan, Ed., Seventh International Joint Conference on Artificial Intelligence, University of British Columbia, Vancouver, B.C., 829-836. 1981.

[54]. Clancey, W. GUIDON Overview. Journal of Computer-based Instruction. 1 (10) , pp. 8-15. 1983.

[55]. Heckerman, D and Horvitz, E. The myth of modularity in rule-based systems form reasoning with uncertainty. UAI. pp. 23-24. 1986.

[56]. Zadeh, L.A. Fuzzy Sets and Systems. Proc. Symposium Syst. Theory, Polytechnic Institute of Brooklyn. pp. 29 – 37. 1965.

[57]. Mendel, J.M. "Fuzzy Logic Systems for Engineering: A Tutorial", Proceedings of the IEEE, vol.83, pp.345-377, Mar.1995.

[58]. Dubois, D and Prade, H. Fuzzy Sets and Systems: Theory and Applications. New York: Academic, 1980.

[59]. Chin D. KNOME: Modeling What the User Knows in UC. In A. Kobsa and W. Wahlster (eds.): User Models in Dialog Systems, 74-107,Berlin-Heidelberg:Springer Verlag. 1989.

[60]. Gonschorek, M, Herzog, C and Kluge, E. Eine Didaktikkomponente für SYPROS - Studentenmodellierung, Lernzielstrukturierung und Lernerführung. GI Jahrestagung ,293-303. 1995.

[61]. Stern, M and Beck, J. Adaptation of Problem Presentation and Feedback in an Intelligent Mathematics Tutor. Beverly Park Woolf Center for Knowledge Intelligent Tutoring Systems. 1996.

[62]. desJardins, M, Denise W, and Schlager M. Representing a Student's Learning States and Transitions. In AAAI &ring Svmnosium on Representing Mental States and Mechanisms held in Stanford, CA , Menlo Park: AAAI, l-9. 1995.

[63]. Shafer, G. A Mathematical Theory of Evidence. Princeton, NJ: Princeton Univ. Press. 1976.

[64]. Carberry, S. Plan Recognition in Natural Language Dialogue. MIT Press. 1990.

[65]. Bauer, M, Biundo, S, Dengler, D, Koehler, J and Gabriele P. PHI - A Logic-Based Tool for Intelligent Help Systems. IJCAI 1993: 460-466. 1993.

[66]. Pearl, J. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers. 1988.

[67]. Villano, M. (1992). Probabilistic student's models: Bayesian belief networks and knowledge space theory. In Second International Conference on Intelligent Tutoring System, p.p. 491-498, Montreal, Canada, 1992.

[68]. Martin, J. and VanLehn, K. Student assessment using Bayesian nets. Int. J. HumanComputer Studies 42, (1995) 575-591. Also available on-line: http://www.pitt.edu/~vanlehn/distrib/journal/HCS95.pdf. 1995.

[69]. Conati, C and K. VanLehn. POLA a student modelling framework for probabilistic on-line assessment of problem solving performance. In Proceedings of the Fifth International Conference on User Modeling, pages 75--82. Kailua--Kona, Hawaii. http://citeseer.ist.psu.edu/conati96pola.html. 1997.

[70]. Schulze, K.G, Shelby, R.N, Treacy, D, Wintersgill, M.C, VanLehn, K and Gertner, A. Andes: A coached learning environment for classical newtonian physics. The Journal of Electronic Publishing. 2000.

[71]. Reye J. Two-Phase Updating of Student Models Based on Dynamic Belief Networks. In Goettle B., Halff H., Redfield C., and Shute V. (Eds.) Proc. of the 4th International Conference on Intelligent Tutoring Systems, Springer-Verlag, pp. 274-283. 1998.

[72]. Greer, J, McCalla, G, Collins, J, Kumar, V, Meagher, P and Vassileva, J. Aries Laboratory, Department of Computer Science, University of Saskatchewan, Saskatoon, SK, S7N 5A9 Canada. International Journal of Artificial Intelligence in Education. 9, pp. 159-177. 1998.

[73]. Murray W. A Practical Approach to Bayesian Student Modeling. In Goettle B., Halff H., Redfield C., and Shute V. (Eds.) Proc. of the 5th International Conference on Intelligent Tutoring Systems, Springer-Verlag, pp. 424-433. 1998.

[74]. Murray W. An easily implemented, linear time algorithm for Bayesian student modelling in multi-level trees. In Lajoie S. and Vivet M., Proc. of the 9th International Conference on Artificial Intelligence and Education (AI-ED 99), IOS Press, pp.413-420. 1999.

[75]. Millán E., Pérez-de-la-Cruz J., and Suárez E. Adaptive Bayesian Networks for Multilevel Student Modelling. In Gauthier G., Frasson C., and VanLehn K. (Eds.), Proc. Of 5th International Conference on Intelligent Tutoring Systems, Springer-Verlag, pp. 534- 543. 2000.

[76]. Klein G, and Calderwood R: Using Analogues to Predict and Plan. Proceedings of a Workshop on Case-Based Reasoning. Pp. 224-232. 1988.

[77]. Corchado, J. M., Pavón, R, Laza, R and Gómez, A. Improving the revision stage of a CBR system with belief revision techniques. Computing and Information Systems Journal. 8 (2). pp. 40-45. 2001.

[78]. Plaza, E and López de Mántaras. Case-Based Apprentice that Learns from Fuzzy Examples. (Z. Ras et al., Eds.) Methodologies for Intelligent Systems-5 North-Holland, pp. 420-427. 1990.

[79]. Richter, M and Wess, S. Similarity, uncertainty and case-based reasoning in PATDEX. In R S Boyer, editor, Automated Reasoning -- Essays in Honour of Woody Bledsoe, pages 249--265. Kluwer, 1991.

[80]. Bareiss, R. Protos: a unified approach to concept representation, classification and learning. Technical report ai88-83, University of Texas at Austin, Dep. of Computer Science, 1989.

[81]. Koton, P. Using experience in learning and problem solving. Massachusetts Institute of Technology, Laboratory of Computer Science (Ph.D. diss, October 1988). MIT/LCS/TR-441. 1989.

[82]. Aamodt, A. Explanation-driven case-based reasoning. In Topics in case-based reasoning. Springer Verlag, 274-288. 1994.

[83]. Skalle, P and Aamodt, A. Knowledge-b ased decision support in oil well drilling, combining general and case-specific knowledge for problem solving. To appear in Proceedings of ICIIP-2004. International Conference on Intelligent Information Processing. October, 2004.

[84]. Aamodt, A. Knowledge intensive case-based reasoning in Creek. Advances in case-based reasoning. 7th European Conference ECCBR 2004. Lecture Notes in Artificial Intelligence. 2004.

[85]. Plaza, E. and Arcos J. L. Reflection and Analogy in Memory-based Learning, Proceedings Multistrategy Learning Workshop, George Mason University. p. 42-49. 1993.

[86]. Reisbeck, C and Schank, R. Inside Case-Based Reasoning. Lawrence Erlbaum Ass. Hillsdale, New Jersey, 1989.

[87]. Checkland, P and Scholes, J. Soft systems methodology in action. Wiley, London, 1990.

[88]. Watson, I.. Case-Based Reasoning is a Methodology not a Technology. Research & Development in Expert Systems XV, Mile, R., Moulton, M. & Bramer, M. (Eds.), pp.213-223. Springer-Verlag, London. 1998.

[89]. Medsker L. R. Hybrid Intelligent Systems. Kluwer Academic Publishers. 1995.

[90]. Macchion, D. Use of case-based reasoning technique in building expert systems. In, Future Generation Computer Systems, 9(4), pp.311-319. 1993.

[91]. Rissland, E, and Daniels, J. Using CBR to Drive IR. In the Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal, Canada, 1995. http://citeseer.ist.psu.edu/rissland95using.html. 1995.

[92]. Lynn L.. A hybrid knowledge-based system applied to epidemic screening. Expert Systems: The International Journal of Knowledge Engineering and Neural NetworksVolume 16 Issue 4 Page 248 - November 1999.

[93]. Louis, S. J., McGraw, G., and Wyckoff, R. Case-based reasoning assisted explanation of genetic algorithm results. *Journal of Experimental and Theoretical Artificial Intelligence*, 5:21-37. 1993.

[94]. Dwight, D and Oppacher, F. Improving the quality of case memory using genetic algorithms. In Proceedings of the Eighth Biennial Conference of the Canadian Society for Computational Studies of Intelligence, p.p. 161—168. 1990.

[95]. Navinchandra, D., Sycara, K. P., and Narasimhan, S. Behavioral synthesis in CADET, a case-based design tool. In Proc. of the 7 th IEEE Computer Society Conference on Artificial Intelligence Applications, pages 217--221, Miami, CA. 1991.

[96]. Maher, M.L and Zhang, D.M. "CADSYN: A Case-Based Design Process Model," Artificial Intdligence for Eng. Design, Analysis, and Manufacturing, vol. 7, no. 2, pp. 97-110, 1993.

[97]. Hinrichs, T. Problem Solving in Open Worlds: A Case Study in Design. Lawrence Erlbaum Associates, 1992.

[98]. Lau, C. Neural Networks, Theoretical Foundations and Analysis, IEEE Press. 1991.

[99]. Tirri, V. Using Neural Networks for Descriptive Statistical Analysis of Educational Data. Annual Meeting of the American Educational. 1997.

[100]. Shinmori, A. A Proposal to Combine Probabilistic Reasoning with Case-Based Retrieval for Software Troubleshooting. Case- Base Reasoning Integrations. Papers from the 1998 Workshop Technical Report WS-98-15. 1998.

[101]. Chavez, A, Dreilinger, D, Guttman, R and Maes, P. “A Real-Life Experiment in Creating an Agent Marketplace” In Proceedings of the Second Conference on Practical Applications of Agents and Multi-Agent Systems, p. 159-178. 1997.

[102]. Ferber, "Multi-Agent Systems - An Introduction to Distributed Artificial Intelligence", Addison-Wesley, 1999.

[103]. Costa, E. B., Lopez, M. A. and Ferneda, E. Mathema: A Learning Enviroment based on a Multi-Agent Architecture. In Lectures Notes in Artificial Intelligence. Advances in Artificial Intelligence. Proceedings of 12th Brazilian Symposium on Artificial Intelligence. SBIA '95, Campinas, Brasil, October. P.141-150. 1995

[104]. Frasson, C, Martin, L, Gouarderes, G, and Aimeur, E. LANCA: A Distance Learning Architecture Based on Networked cognitive Agents. In Lectures Notes in Computer Science. Intelligent Tutoring Systems. Proceedings of 4th International Conference, San Antonio, Texas, August 1998. P.594- 603. 1998.

[105]. Shoham, Y. Agent-oriented Programming. Artificial Intelligence, 60, 51-92, 1993.

[106]. Webber, C, Bergia, L, Pesty, S, and Balacheff, N. The Baghera project: a multi-agent architecture for human learning. Workshop – Multi-Agent Architectures for Distributed Learning Environments. Proceedings International Conference on AI and Education, San Antonio, Texas, P.12-17. 2001.

[107]. Vassileva, J, Deters, R, Geer, J, Mccalla, G, Bull, S, and Kettel, L. Lessons from Deploying I-Help. Workshop – Multi-Agent Architectures for Distributed Learning Environments. Proceedings International Conference on AI and Education, San Antonio, Texas, P.3-11, 2001.

[108]. Giraffa, L.M. STI modelados a través de una sociedad de agentes. Capturado en Novembro de 2001. Online. Disponível na Internet em: http://www.edukbr.com.br/portal.asp. 2001.

[109]. Thiry, M. Uma Arquitectura Baseda en Agentes para Suporte ao Ensino à Distância. CPGCC/UFRGS, Porto Alegre, 2001. DEPS/PPGEP, 1999. Tese de Doutorado. Capturado em Novembro de 2001.

[110]. D'Amico, C, Viccari, R. M and Alvares, L.O. A Framework for Teaching and Learning Environments. In: SIMPÓSIO DE INFORMÁTICA NA EDUCAÇÃO, VIII, São Paulo, SP. 1997.

[111]. Dasarathy, B. V. (ed.) Nearest Neighbor(NN) Norms: NN Pattern Classificatiion Techniques. IEEE Computer Society Press, Los Alamitos, California. 1991.

[112]. Medsker, L. R. Hybrid Intelligent Systems. Kluwer Academic Publishers. 2000.

[113]. Riverola and Corchado, J.M. Employing TSK Fuzzy models to automate the revision stage of a CBR system. Lecture Notes in Artificial Intelligence. 2004.

[114]. Gómez R, Hidalgo, R, Pavón, R and Corchado J.M.. Knowledge Management. (Automating the Revision phase of a Case-Based Reasoning system using Belief Revision), volumen 1, páginas 59-67. L. Joyanes et. Al. (eds), Primera edición, 2002.

[115]. Plaza, E, Arcos, Joseph L, and Martín, F. Cooperative Case-Based Reasoning. Distributed Artificial Intelligence meest Machine Learning. Lecture Notes in Artificial Intelligence, Springer Verlag, num 1221,. pp. 180-201. 1997.

[116]. González, C, Burguillo, J.C, and Llamas, M. A comparison of case-based reasoning and Bayesian networks for student modelling in intelligent learning environments: Source Files. Available on http://www-gist.det.uvigo.es/~cgonzals/. 2005.

[117]. González, C, Burguillo, J.C and Llamas, Martín. Case-based student modelling in multi-agent learning environments. 4th International Central and Eastern European Conference on Multi-Agent Systems. Lecture Notes in Computer Science. Springer Verlag. Septiembre 15 – 17. Budapest. 2005.

[118]. Chen, H and Wu, Z. On case-based knowledge sharing in the semantic web. 15th International Conference on Tools with Artificial Intelligence. 3. 2003.

[119]. Armengol, E and Plaza, E. Lazy Induction of description for relational case-based learning. 12th European Conference on Machine Learning. pp. 13 -24. 2001.

[120]. Devedzic, V, Jerinic, L and Radovic, D. The GET-BITS Model of Intelligent Tutoring Systems. International Journal of Continuing Engineering Education and Life-Long Learning. 1999.

[121]. Munch, A. UMBC. AgentWeb. Specification of the KQML Agent-Communication Language -- plus example agent policies and architectures. Disponible en: http://www.cs.umbc.edu/kqml/papers/. 2004.

[122]. Mestras, J.P. GRASIA. Research group. INGENIAS IDE. http://ingenias.sourceforge.net/. 2003.

[123]. Booch, G, Rumbaugh, J and Jacobson, I. Unified Modeling Language User Guide, 2nd Edition. 2005**.**

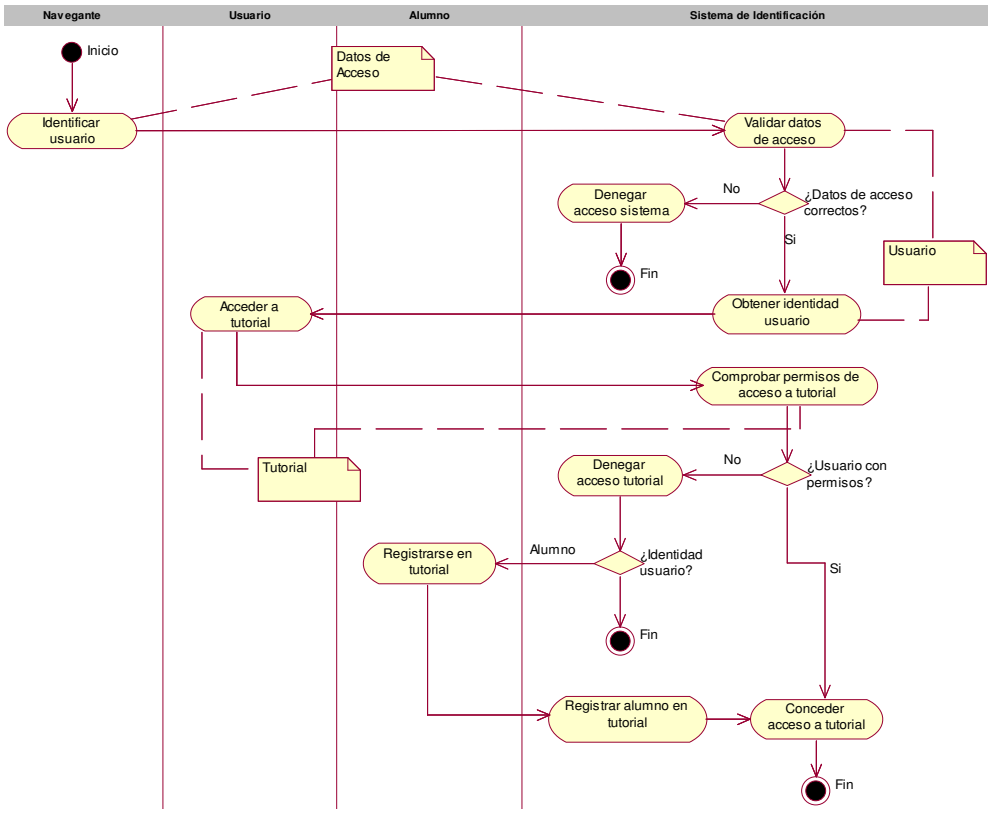
[124]. Krutcehn, P. The Rational Unified Process: An Introduction Second Edition.

[125]. Moraitis, P. and Spanoudakis, N. Combinig Gaia and JADE for Multi-agent Systems Development. 4th International Symposium "From Agent Theory to Agent Implementation" (AT2AI4), in: Proceedings of the 17th European Meeting on Cybernetics and Systems Research (EMCSR 2004), Vienna, Austria, April 13 - 16, 2004.

ANEXOS

ANEXO A (Análisis de Requisitos)

Diagrama de actividades de "Acceso al sistema".



Actividades del proceso "Acceso al sistema".

Las actividades necesarias a llevar a cabo en el proceso de negocio "Acceso al sistema" clasificadas según el agente o rol que las desempeña son las siguientes:

Navegante.
- Identificar usuario.

Usuario.
- Acceder a tutorial.

Alumno.
- Registrar en tutorial.

Sistema de Identificación.
- Validar datos de acceso.
- Denegar acceso.

- Obtener identidad usuario.
- Comprobar permisos de acceso a tutorial.
- Registrar alumno en tutorial.
- Denegar acceso a tutorial.
- Conceder acceso a tutorial.

Diagrama de actividades de "Contenidos" (Alumno)

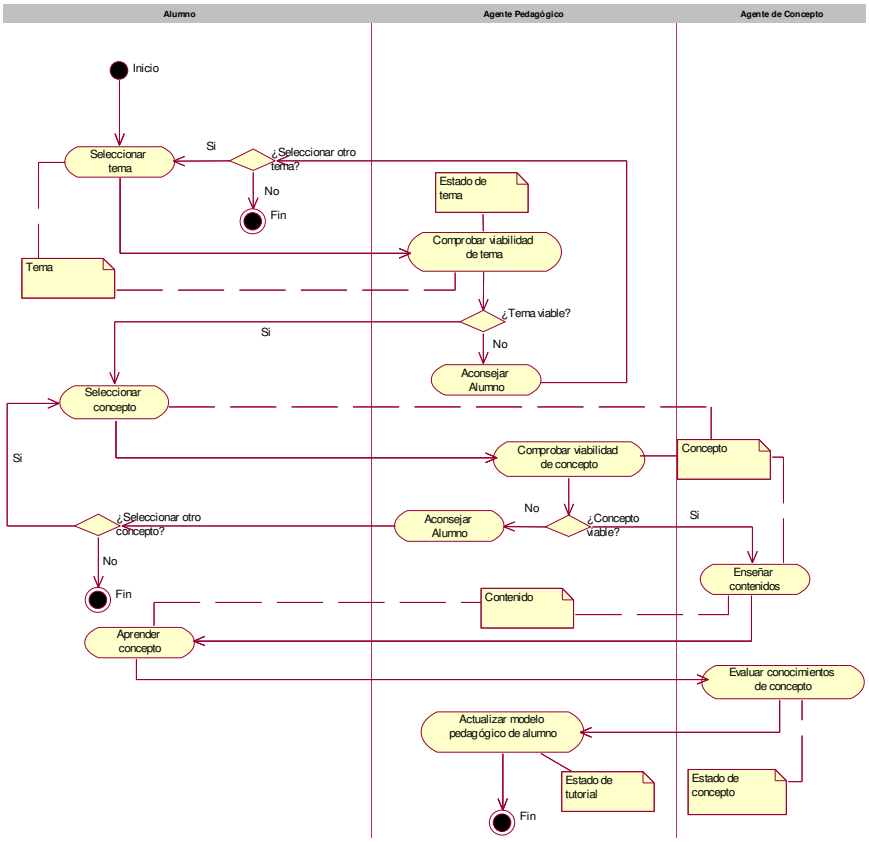
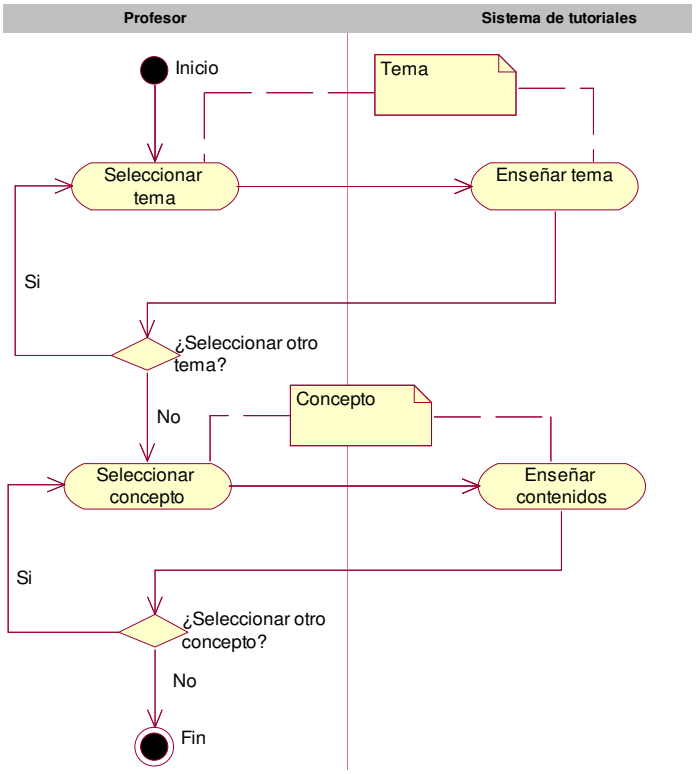


Diagrama de actividades de "Contenidos" (Profesor)



Actividades del proceso "Contenidos".

Las actividades necesarias a llevar a cabo en el proceso de negocio "Contenidos" clasificadas según el agente o rol que las desempeña son las siguientes:

Alumno.
- Seleccionar tema.
- Seleccionar concepto.
- Aprender Contenido.

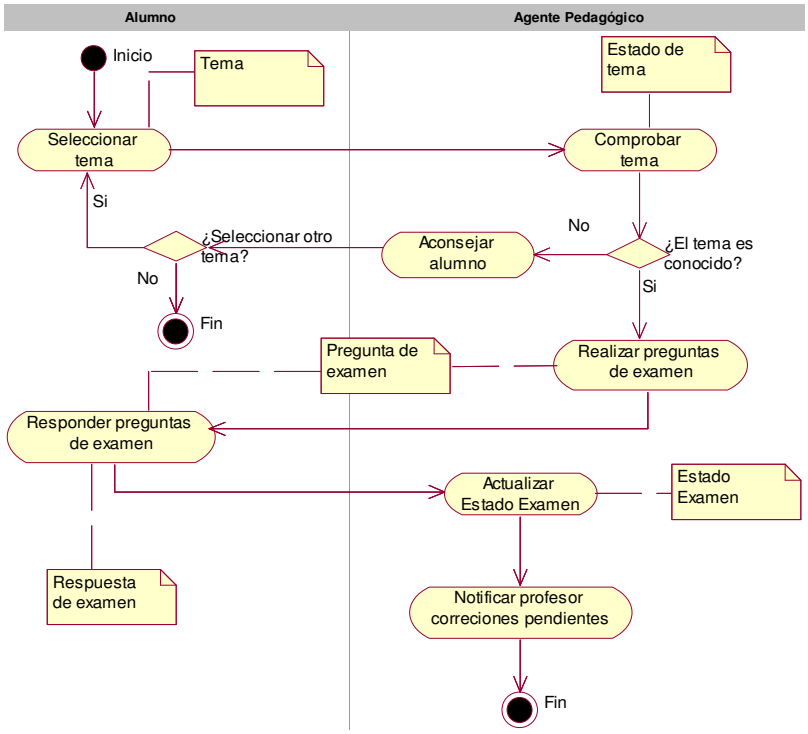
Profesor.
- Seleccionar tema.
- Seleccionar concepto.

Sistema de tutoriales.
- Enseñar tema.
- Enseñar contenidos.

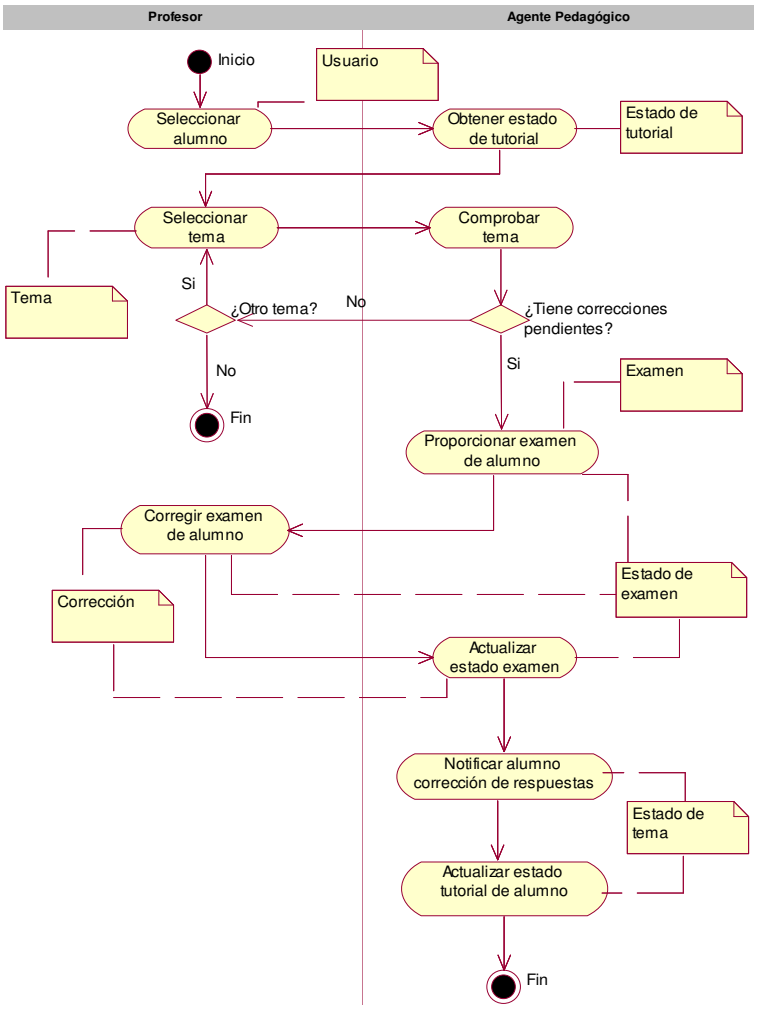
Agente Pedagógico.
- Comprobar viabilidad de tema.
- Aconsejar alumno.
- Comprobar viabilidad de concepto.
- Actualizar modelo pedagógico de alumno.

Agente de Concepto.
- Enseñar contenidos.
- Evaluar conocimientos de concepto.

Diagramas de actividades de "Exámenes".



Corregir respuestas.



Actividades del proceso "Exámenes".

Las actividades necesarias a llevar a cabo en el proceso de negocio "Exámenes" clasificadas según el agente o rol que las desempeña son las siguientes:

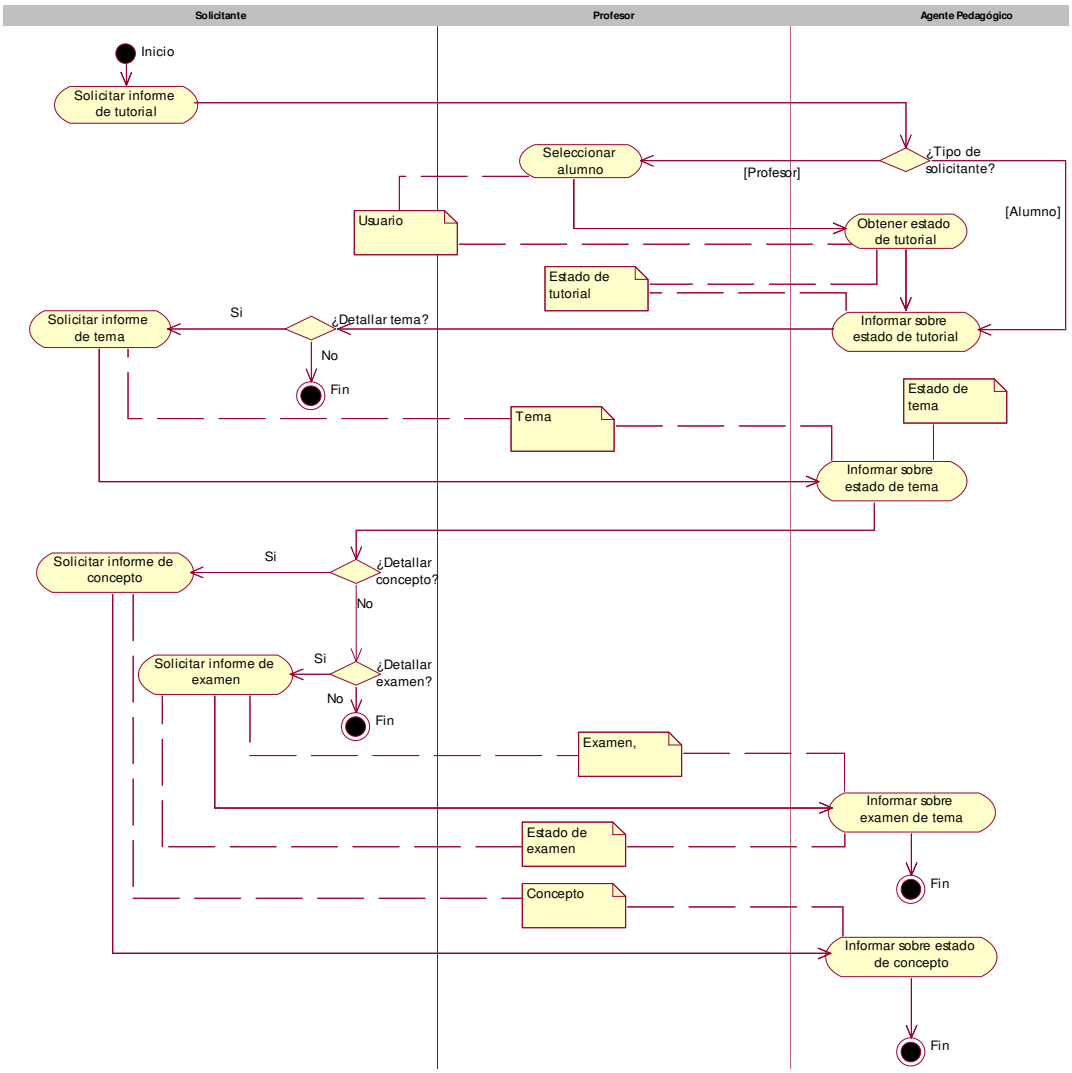
Alumno.
- Seleccionar tema.
- Responder preguntas de examen.

Profesor.
- Seleccionar alumno.
- Seleccionar tema.
- Corregir examen de alumno.

Agente Tutor.

- Comprobar tema.
- Aconsejar alumno.
- Obtener estado de tutorial.
- Proporcionar examen de alumno.
- Realizar preguntas de examen.
- Notificar profesor correcciones pendientes.
- Notificar alumno corrección de respuestas.
- Actualizar estado examen.
- Actualizar estado tutorial de alumno.

Diagrama de actividades de "Solicitar informes".



ANEXO B (Análisis del Sistema)

Diagrama de actividades de Identificar Usuario.

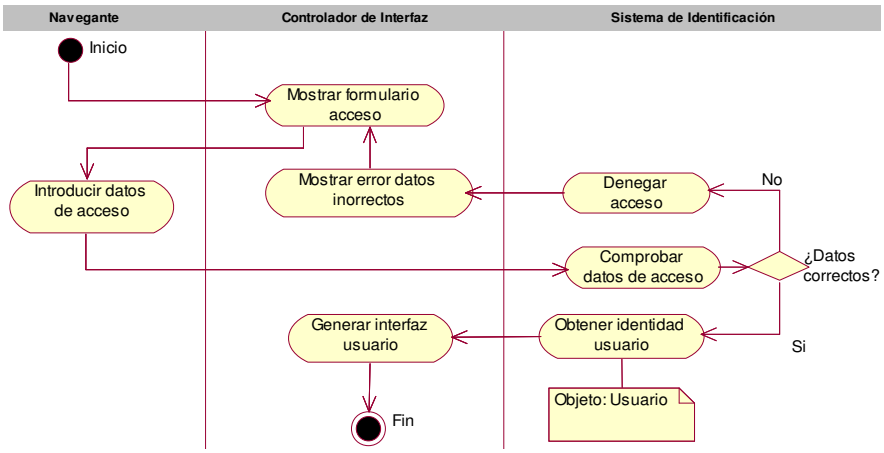


Diagrama de actividades de Identificar Alumno.

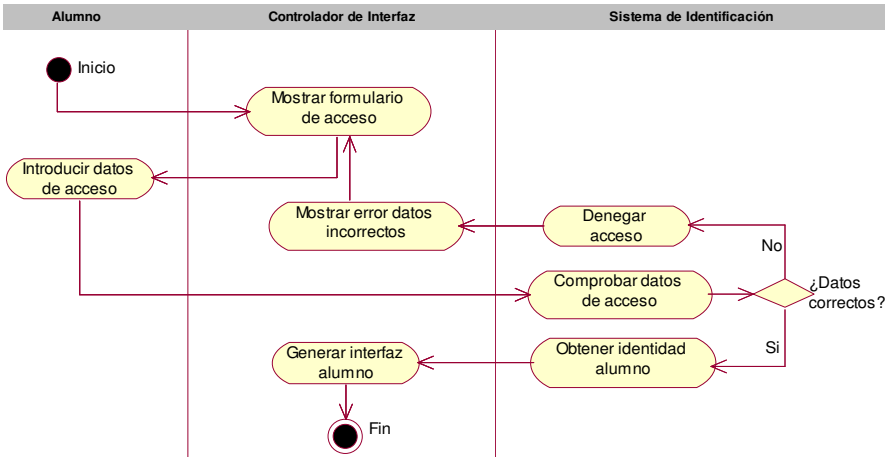


Diagrama de actividades de Identificar Profesor.

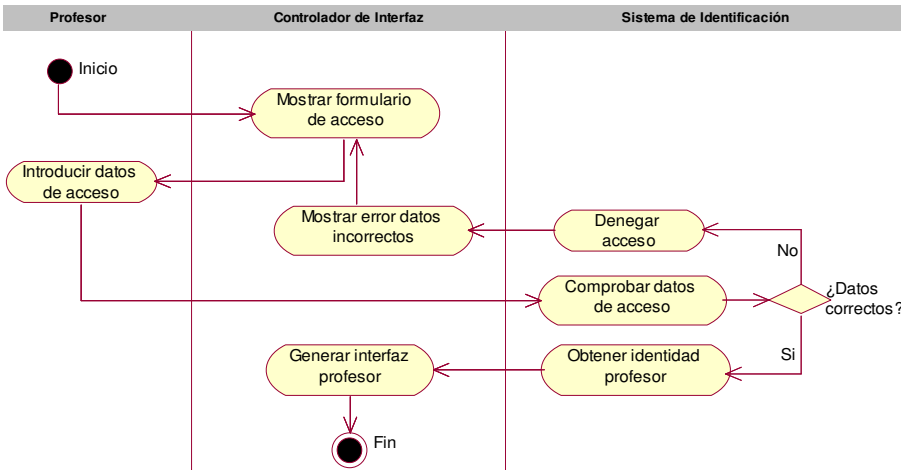


Diagrama de actividades de Identificar Administrador.

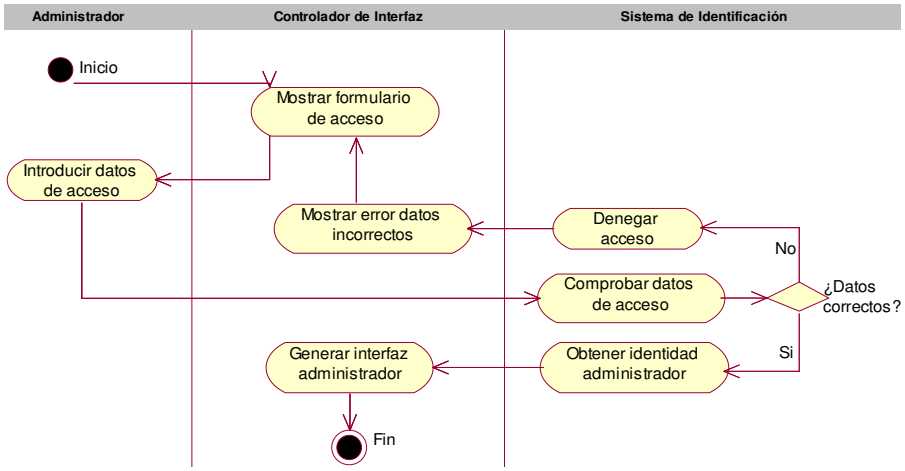


Diagrama de actividades de Seleccionar Tutorial.

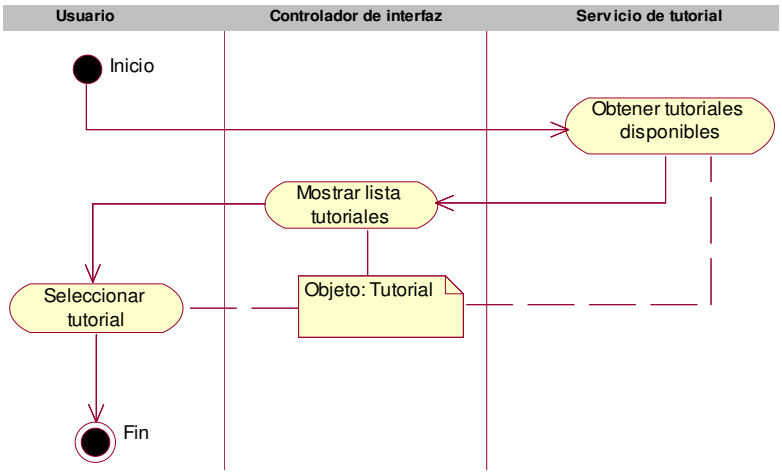


Diagrama de actividades de Registrarse en Tutorial.

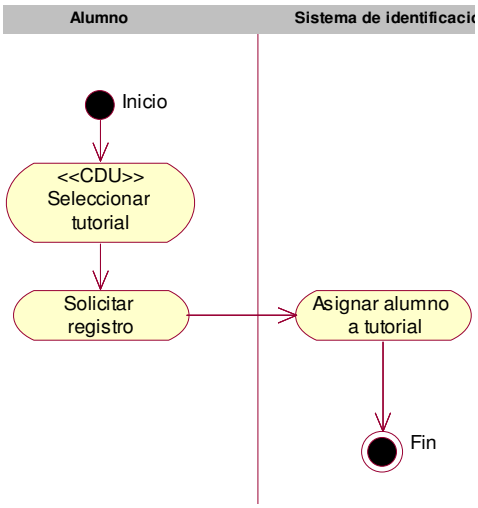


Diagrama de actividades de Acceder a Tutorial (Alumno).

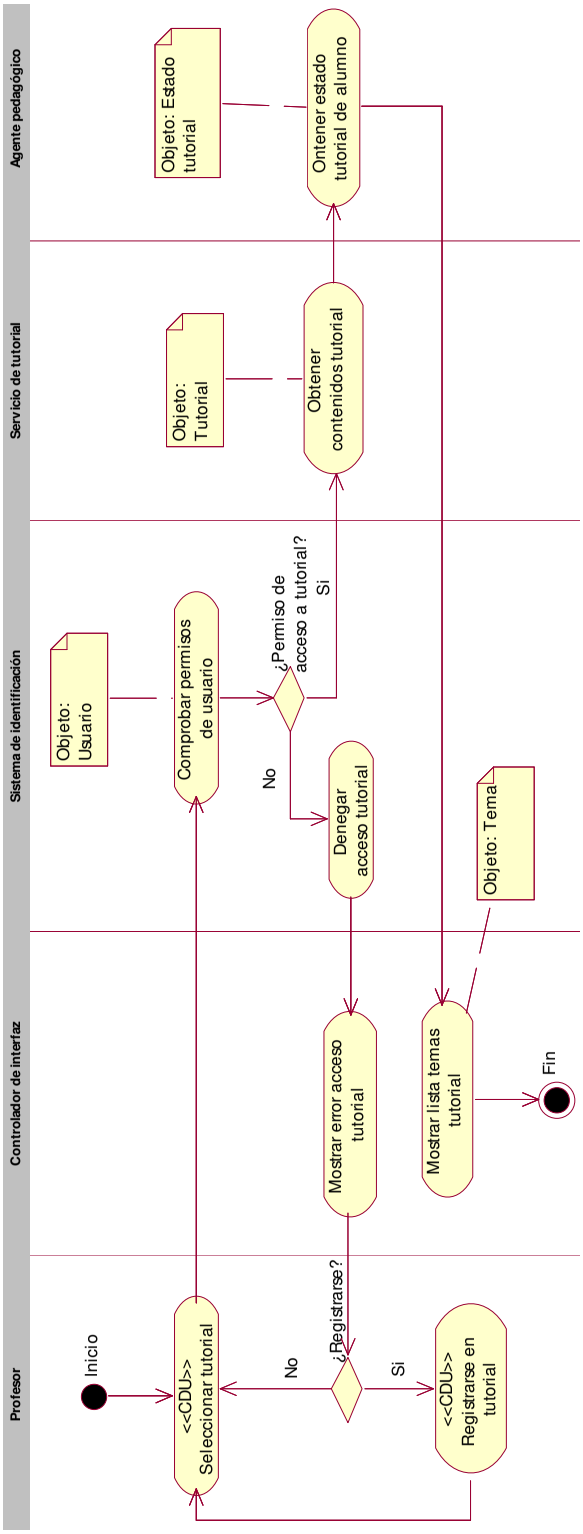


Diagrama de actividades de Acceder a Tutorial (Profesor).

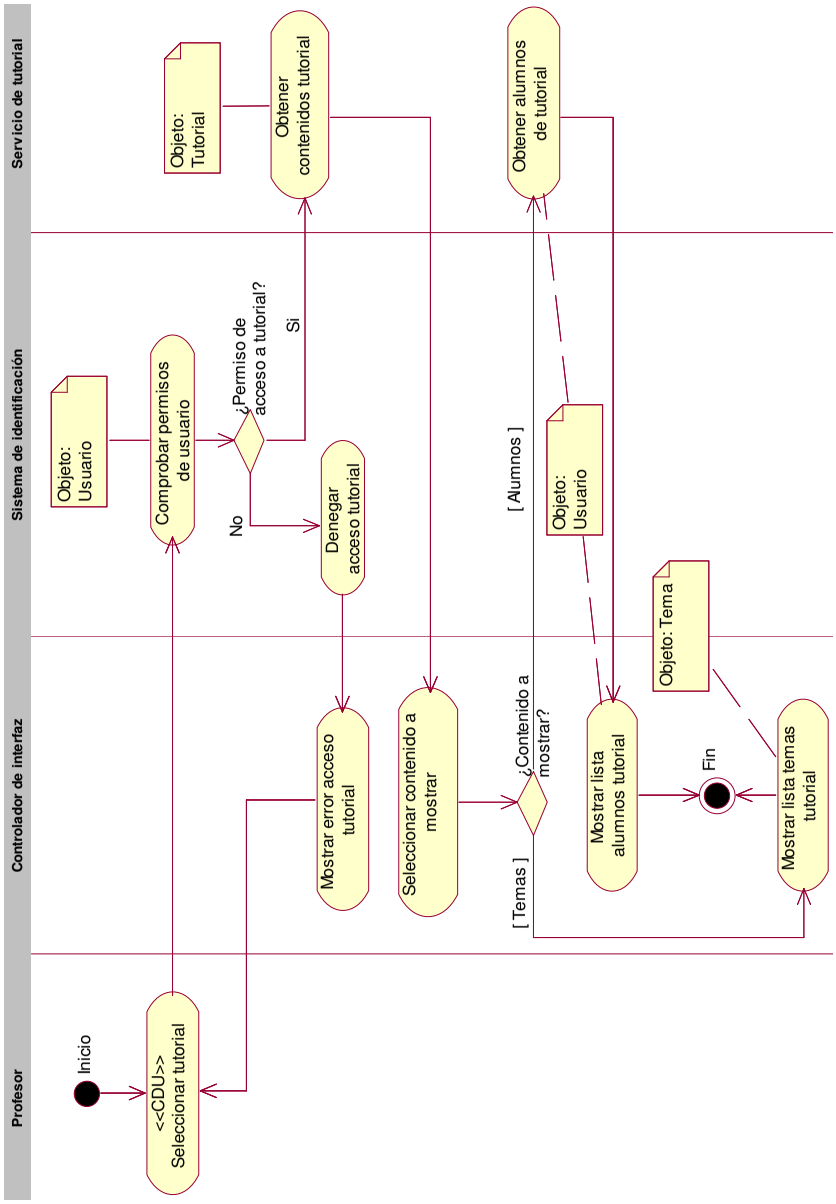


Diagrama de actividades de Seleccionar Tema (Alumno).

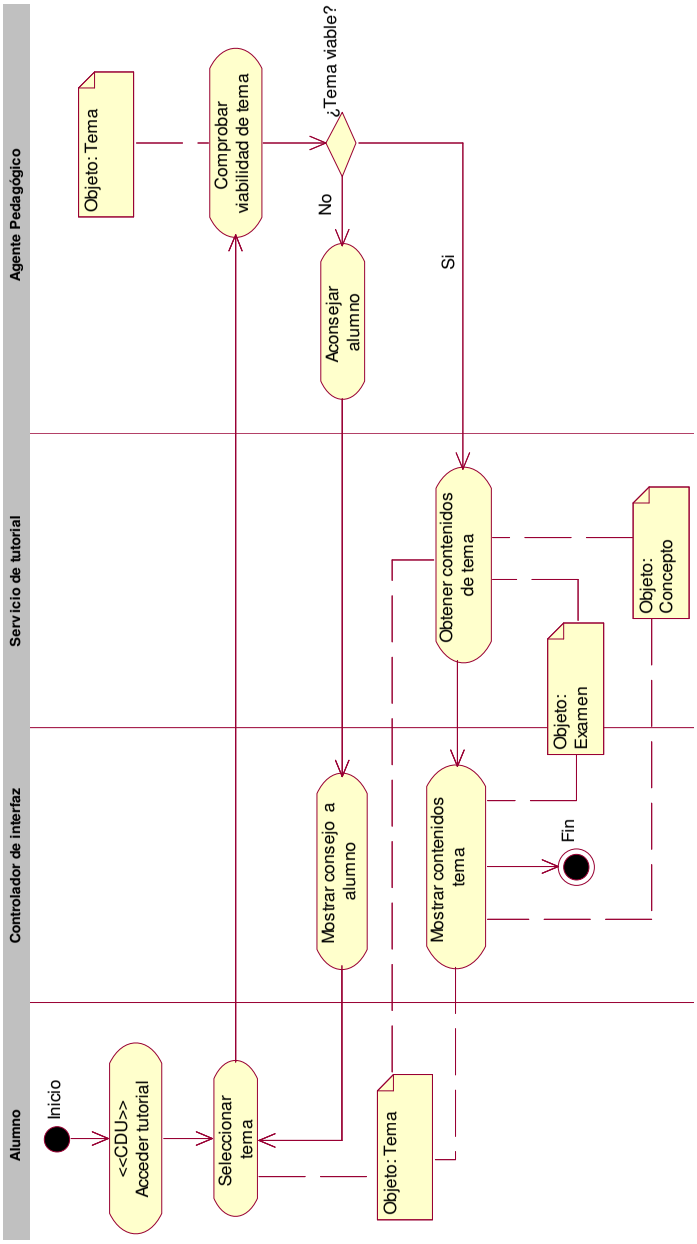


Diagrama de actividades de Seleccionar Tema (Profesor).

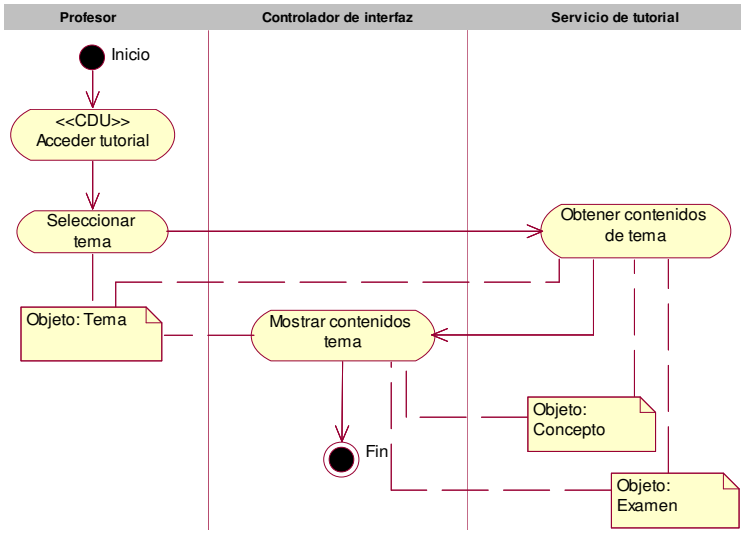


Diagrama de actividades de Seleccionar Concepto (Alumno).

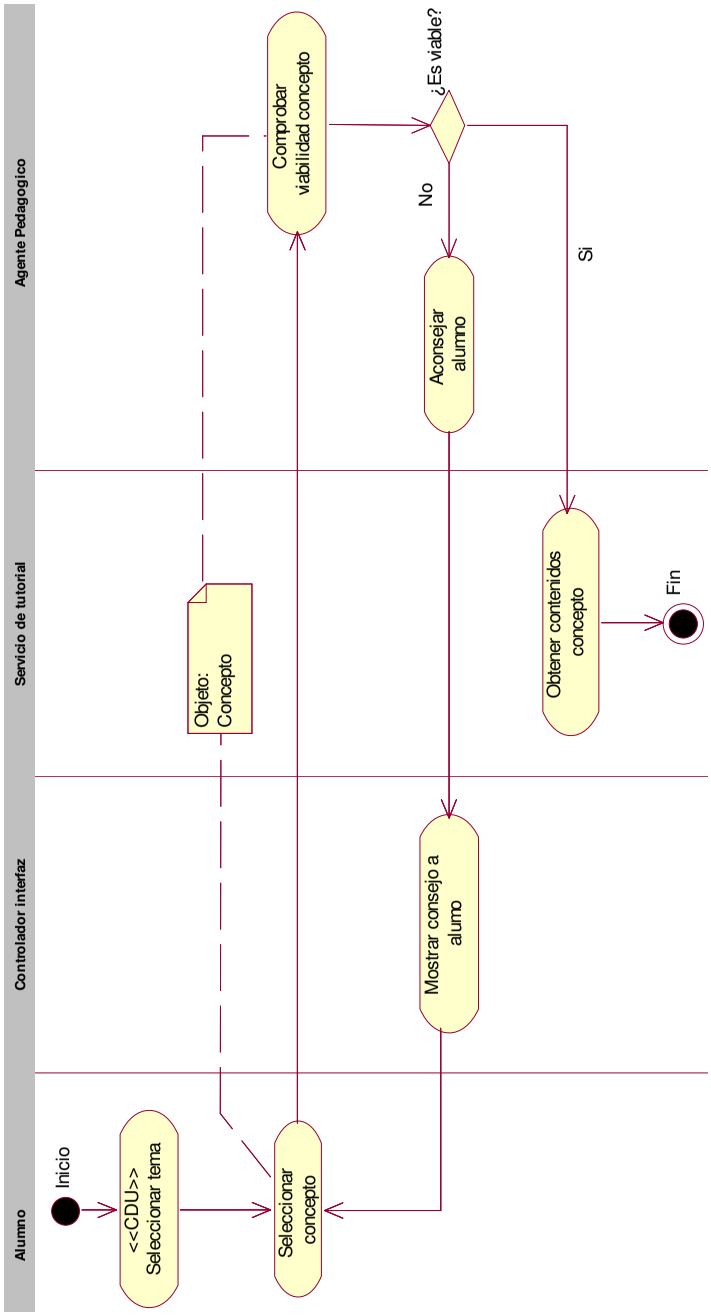


Diagrama de actividades de Seleccionar Concepto (Profesor).

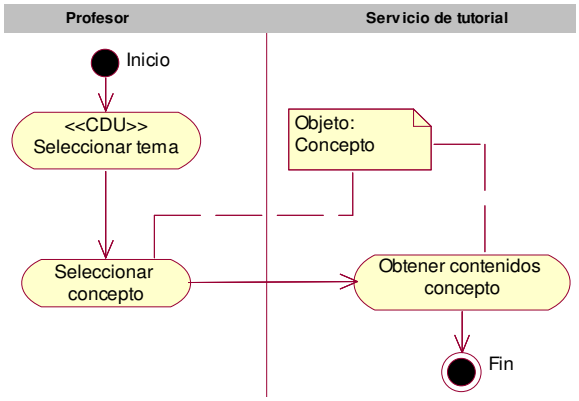


Diagrama de actividades de Aprender Contenido.

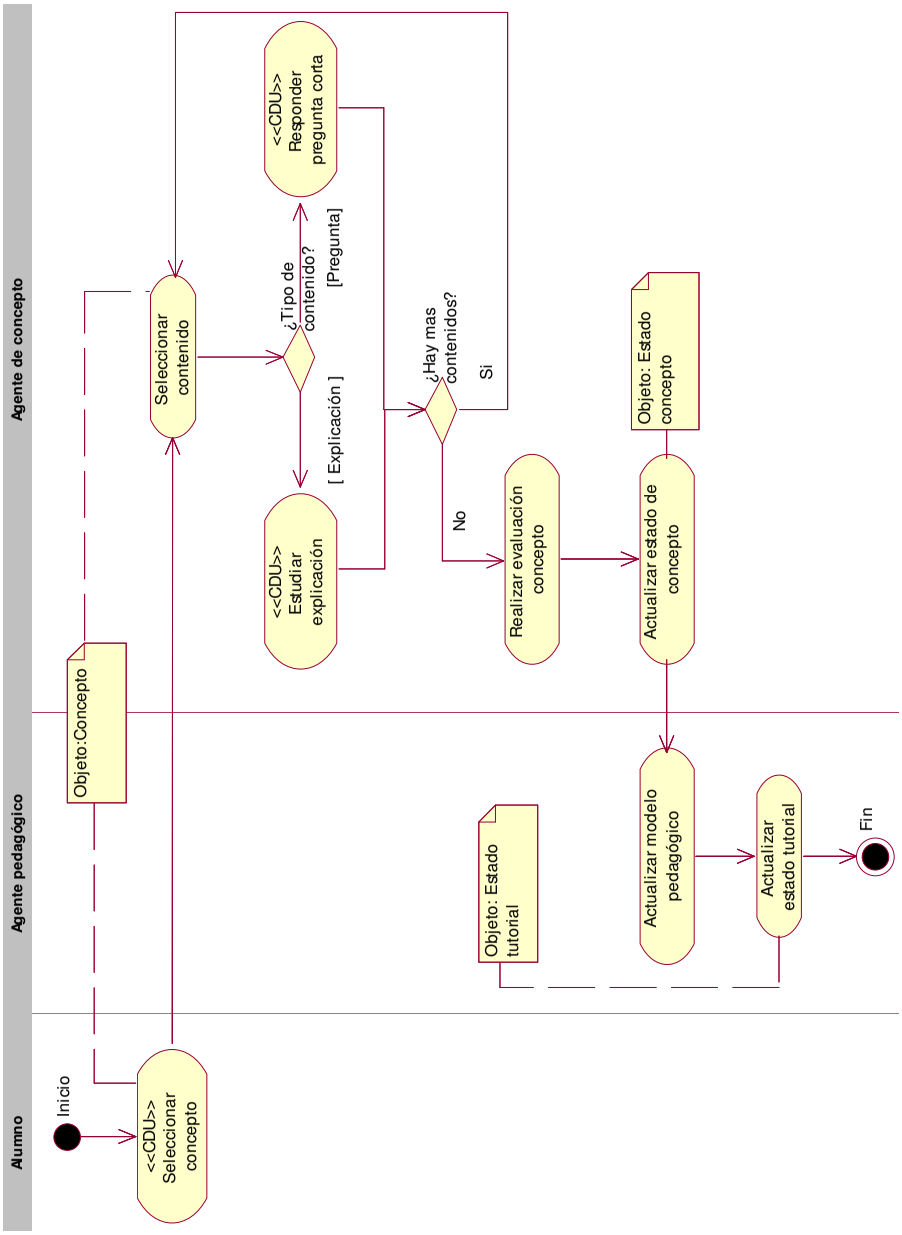


Diagrama de actividades de Estudiar Explicación.

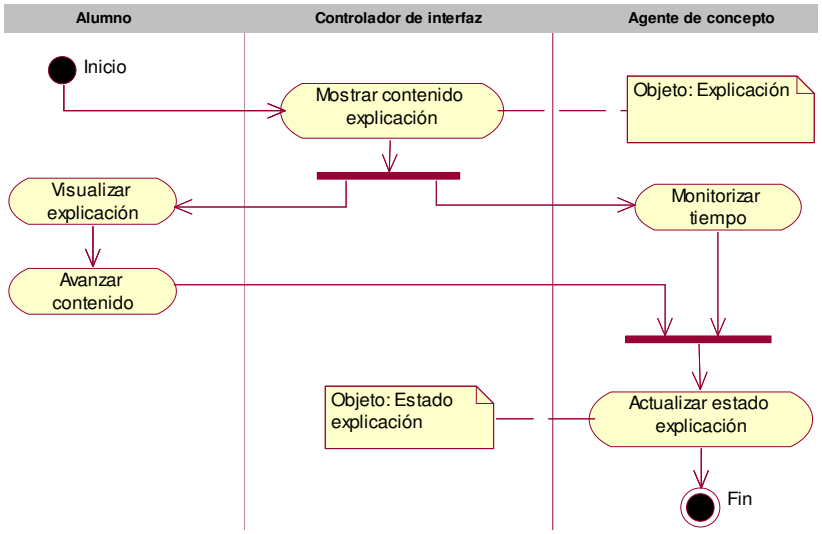


Diagrama de actividades de Responder Preguntas Cortas.

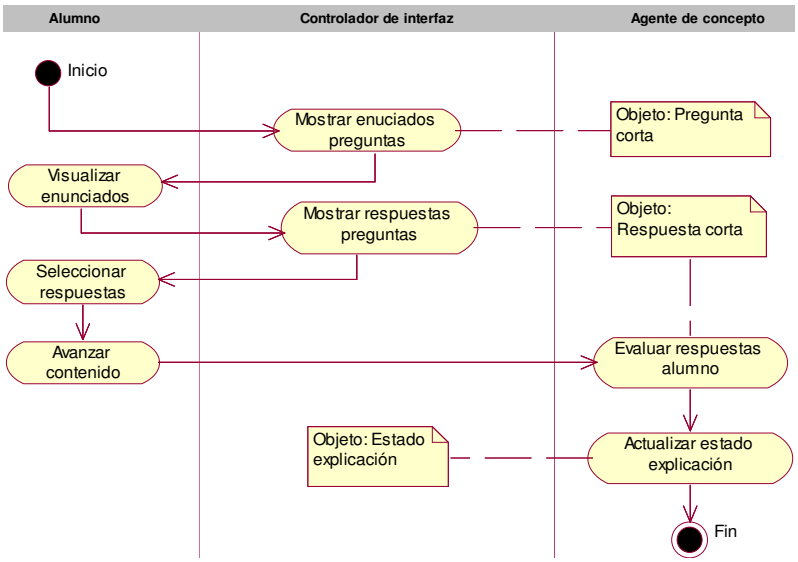


Diagrama de actividades de Obtener Consejo (Pedagógico).

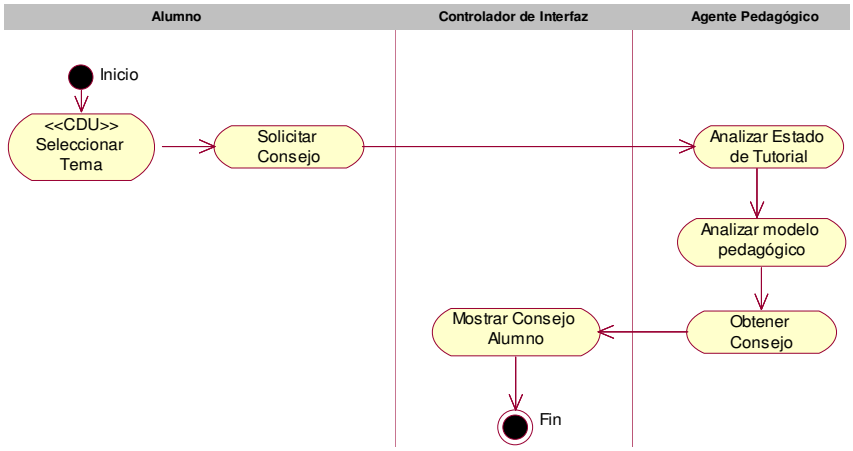


Diagrama de actividades de Obtener Consejo (Concepto).

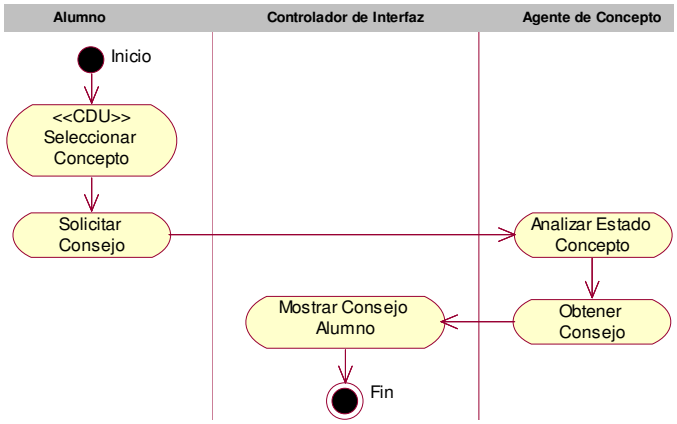


Diagrama de actividades de Visualizar Tema.

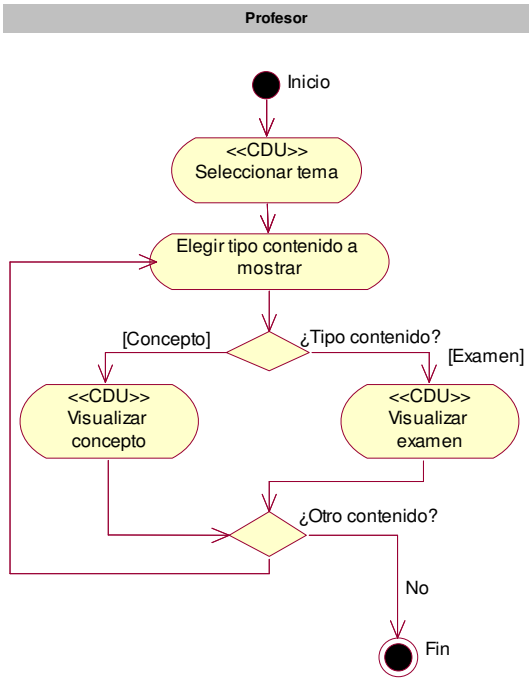


Diagrama de actividades de Visualizar Examen.

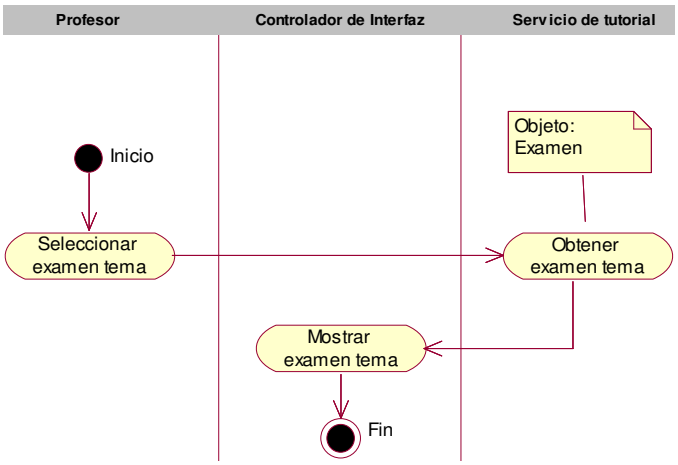


Diagrama de actividades de Visualizar Concepto.

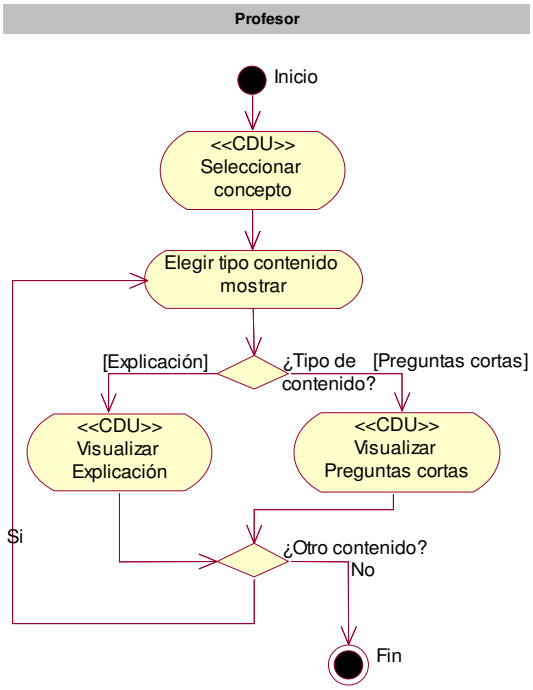


Diagrama de actividades de Visualizar Explicación.

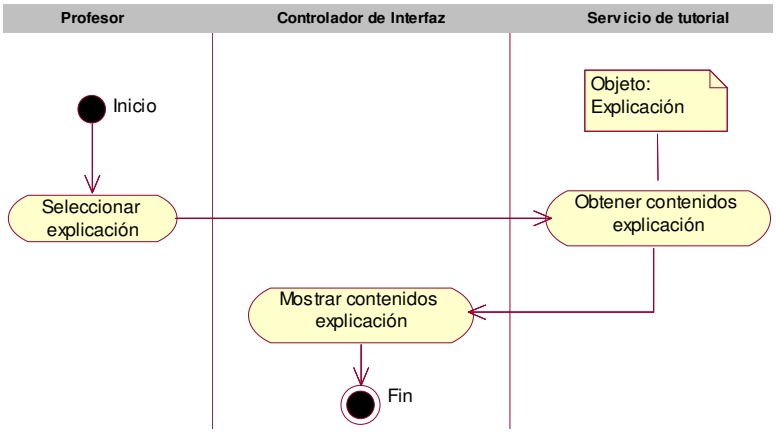


Diagrama de actividades de Visualizar Preguntas Cortas.

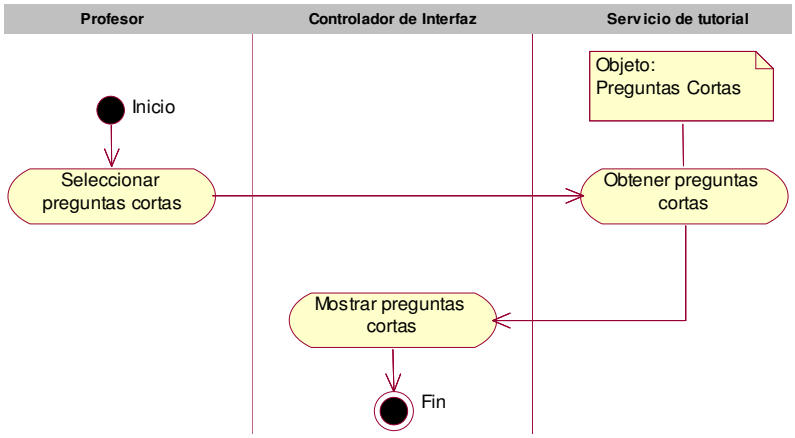


Diagrama de actividades de Seleccionar Examen.

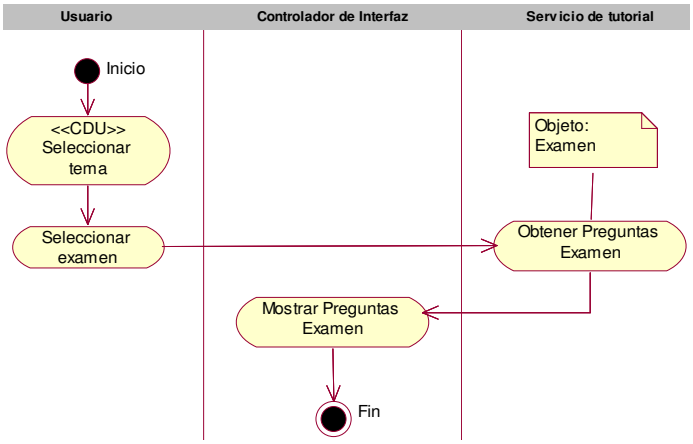


Diagrama de actividades de Realizar Examen.

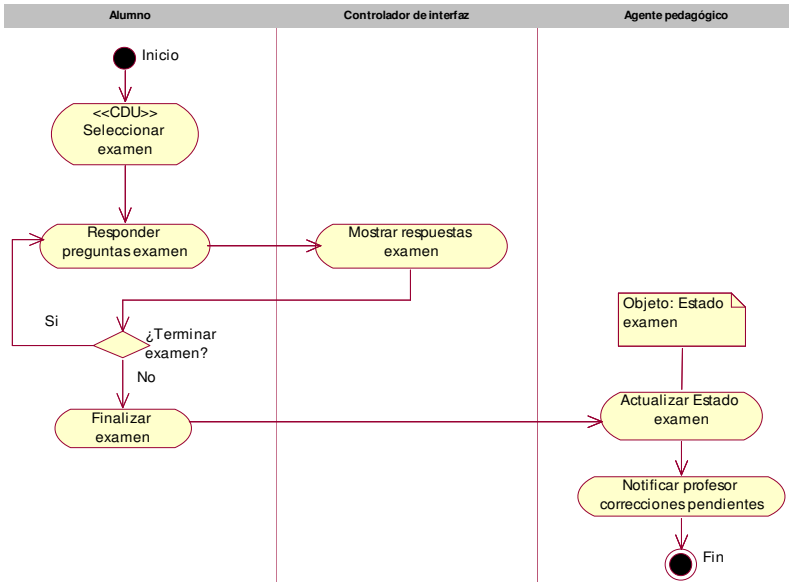


Diagrama de actividades de Seleccionar Alumno.

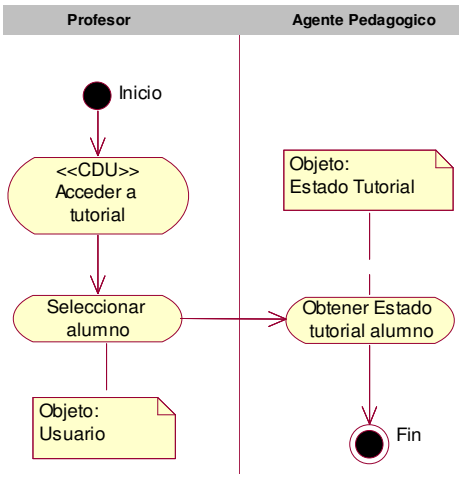


Diagrama de actividades de Corregir Respuestas.

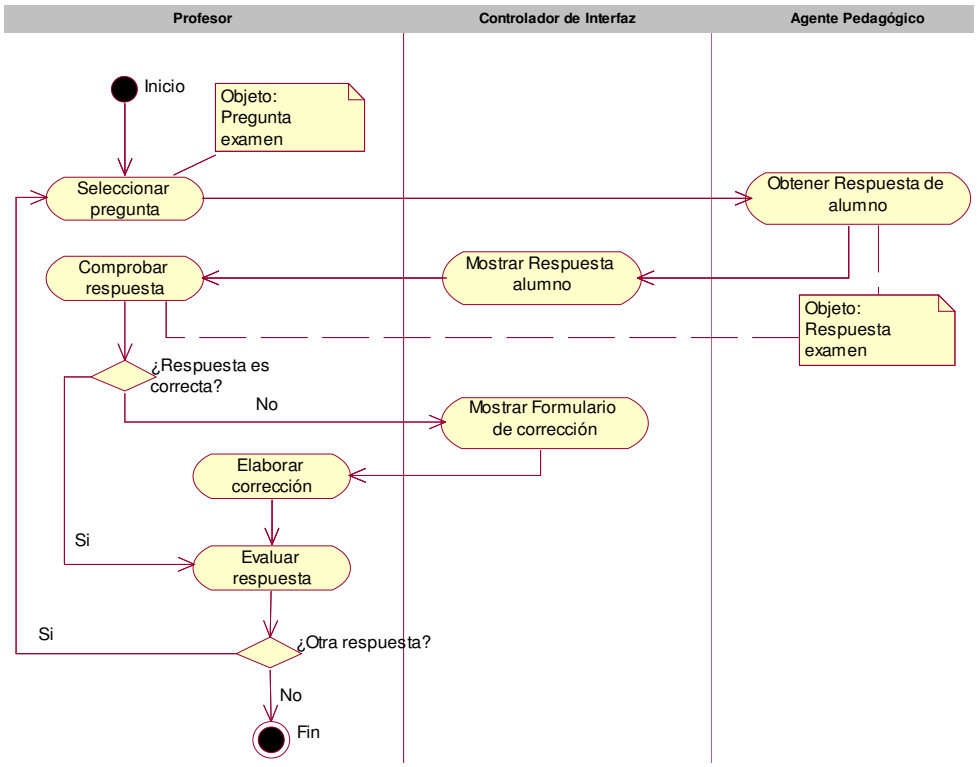


Diagrama de actividades de Corregir examen alumno.

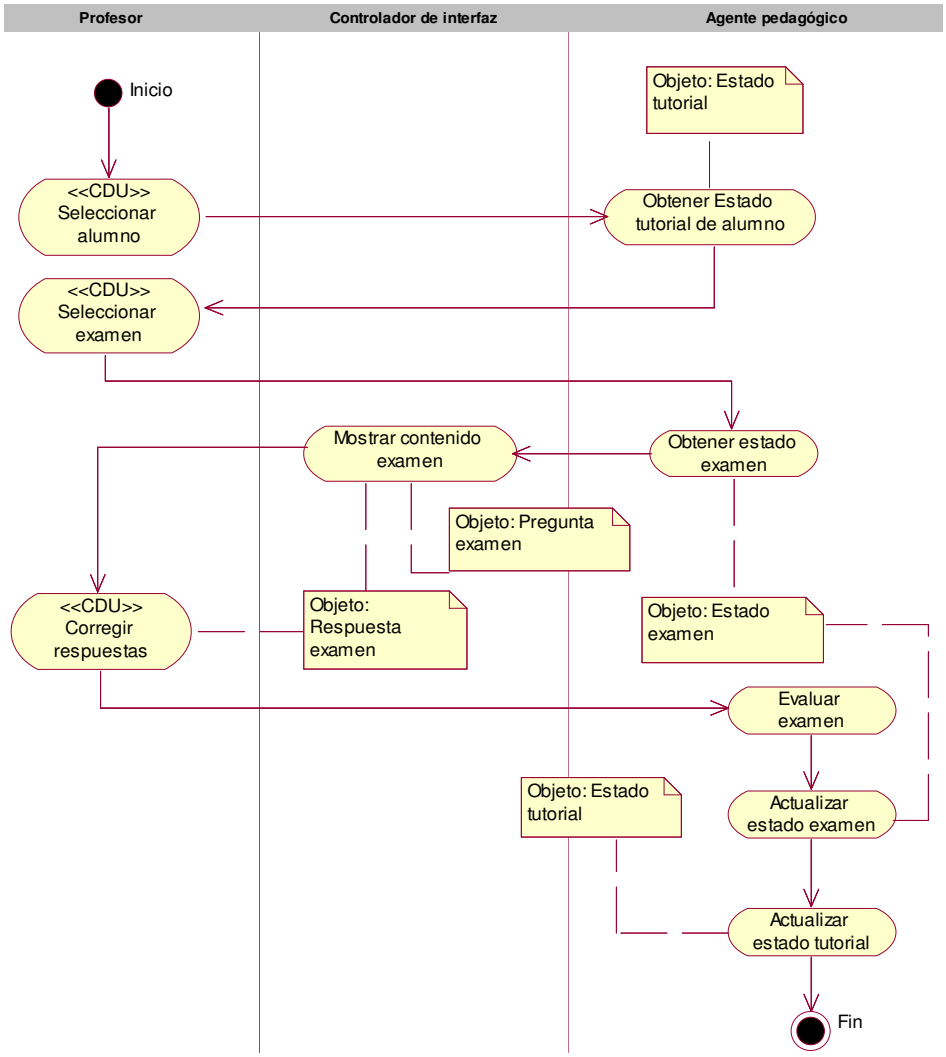


Diagrama de actividades de Solicitar Informe.

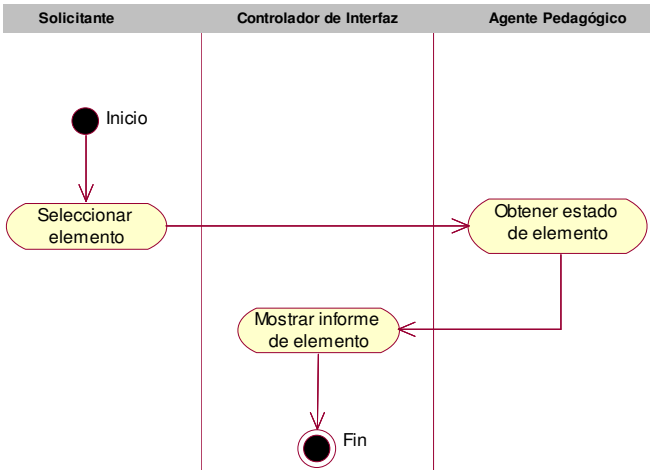


Diagrama de actividades de Solicitar Informe Tutorial (Alumno).

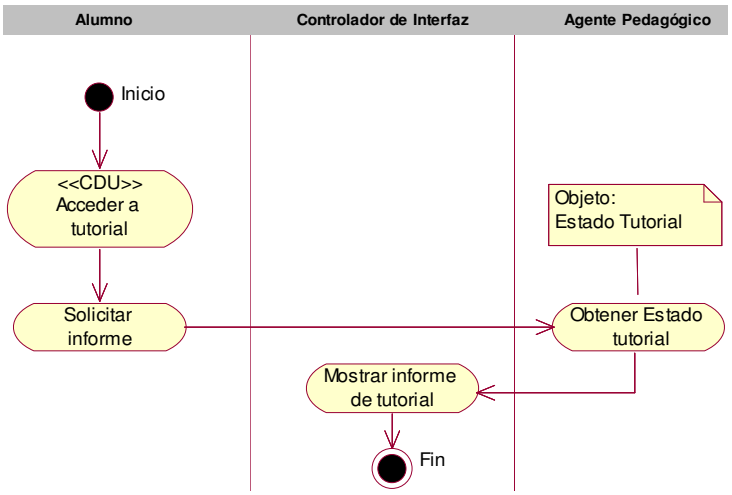


Diagrama de actividades de Solicitar Informe Tutorial (Profesor).

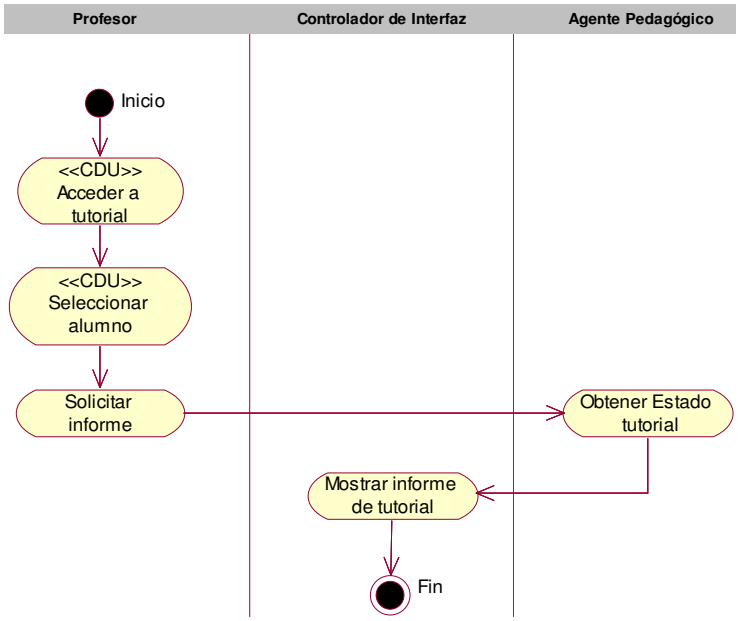


Diagrama de actividades de Solicitar Informe Tema.

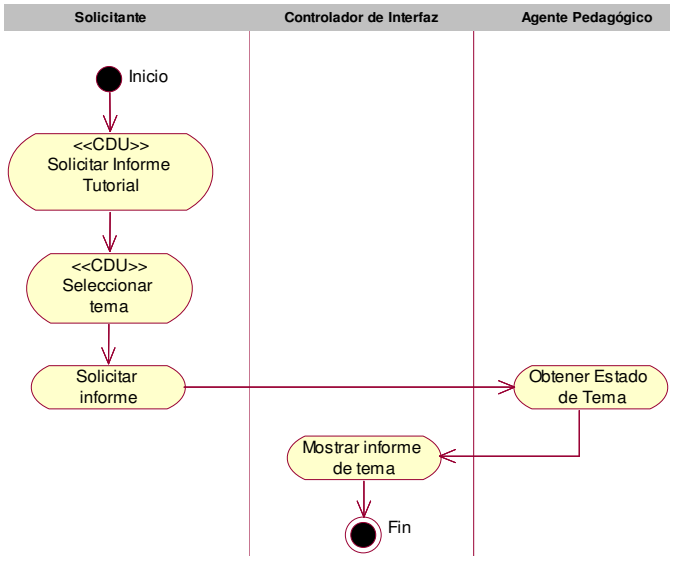


Diagrama de actividades de Solicitar Informe Concepto.

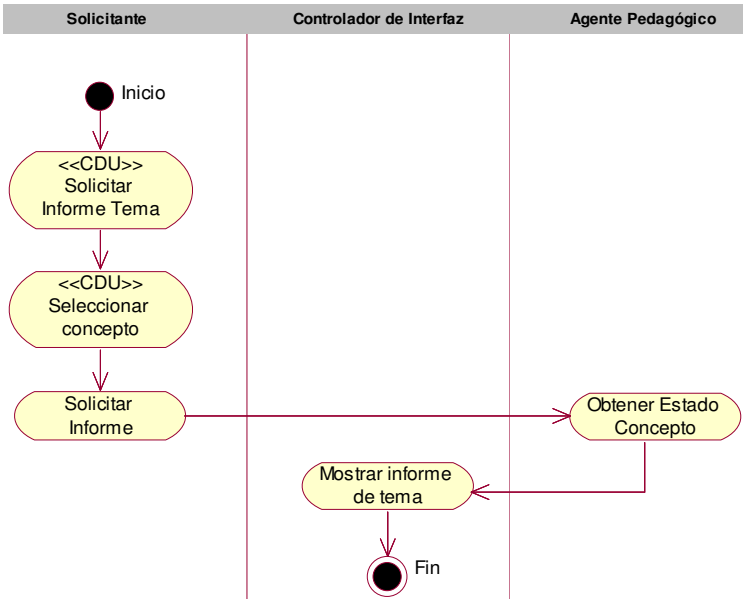


Diagrama de actividades de Registrar Usuario.

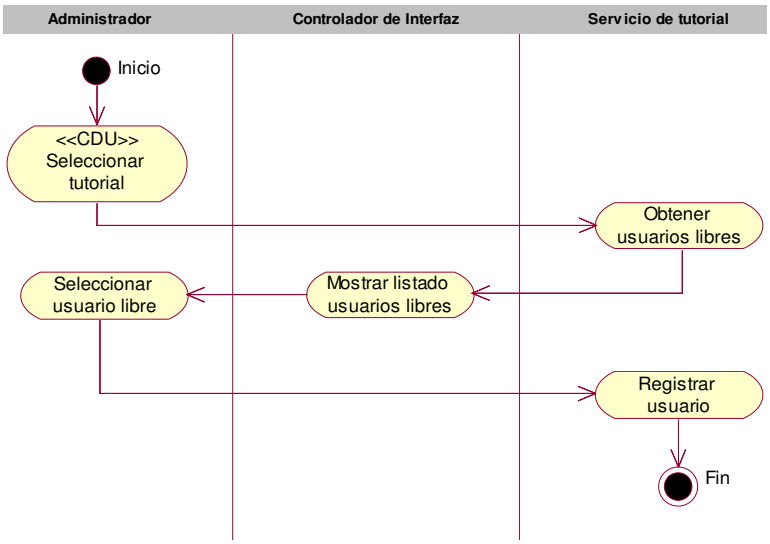


Diagrama de actividades de Registrar Alumno.

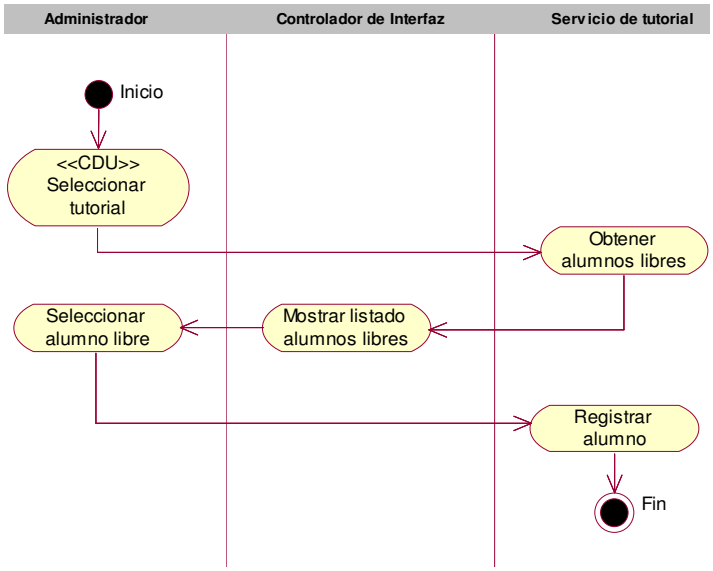


Diagrama de actividades de Registrar Profesor.

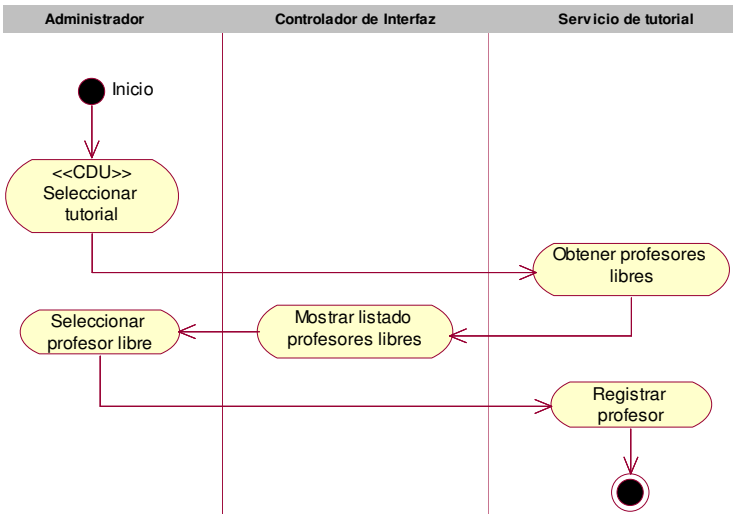


Diagrama de actividades de Eliminar Registro Usuario.

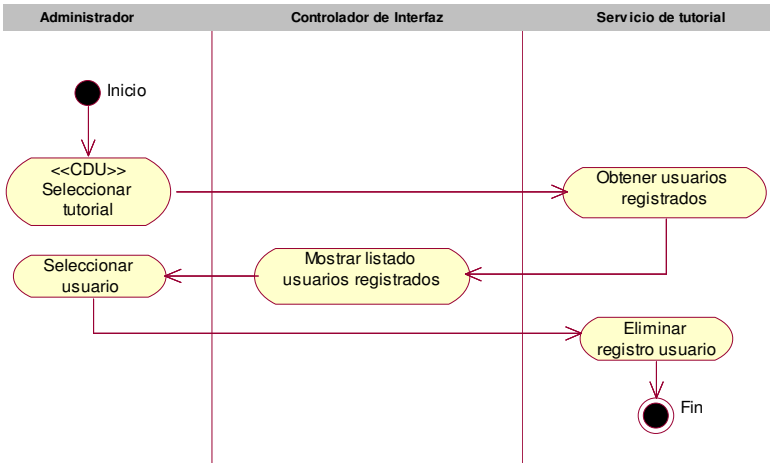


Diagrama de actividades de Eliminar Registro Alumno.

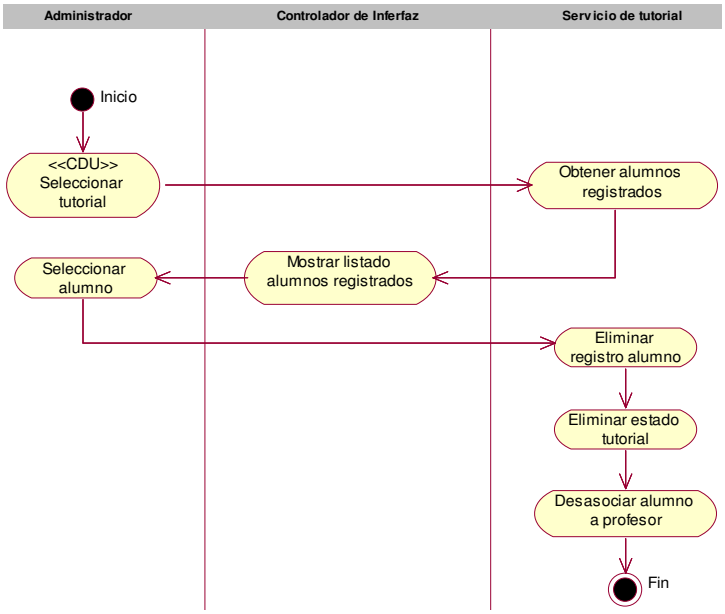


Diagrama de actividades de Eliminar Registro Profesor.

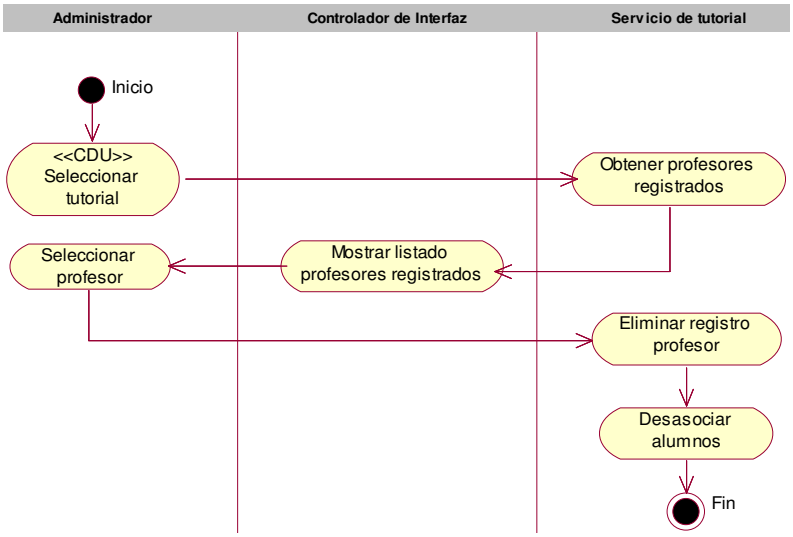


Diagrama de actividades de Asociar Alumno.

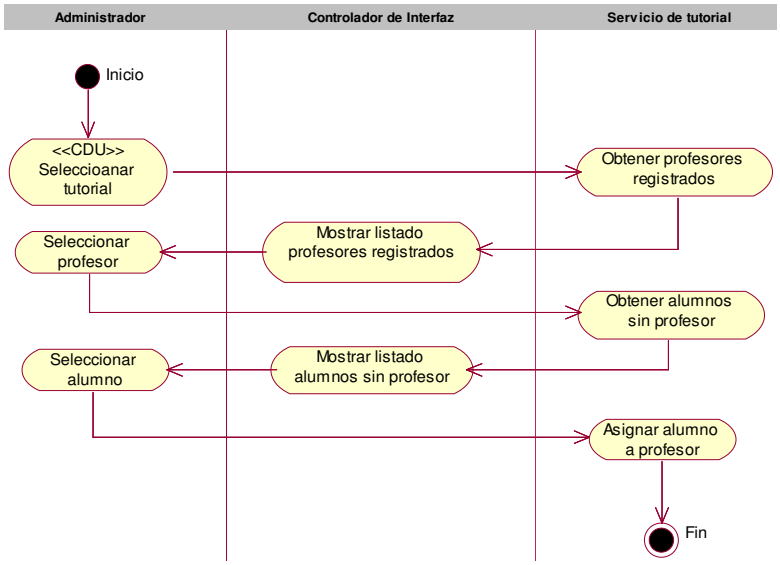


Diagrama de actividades de Desasociar Alumno.

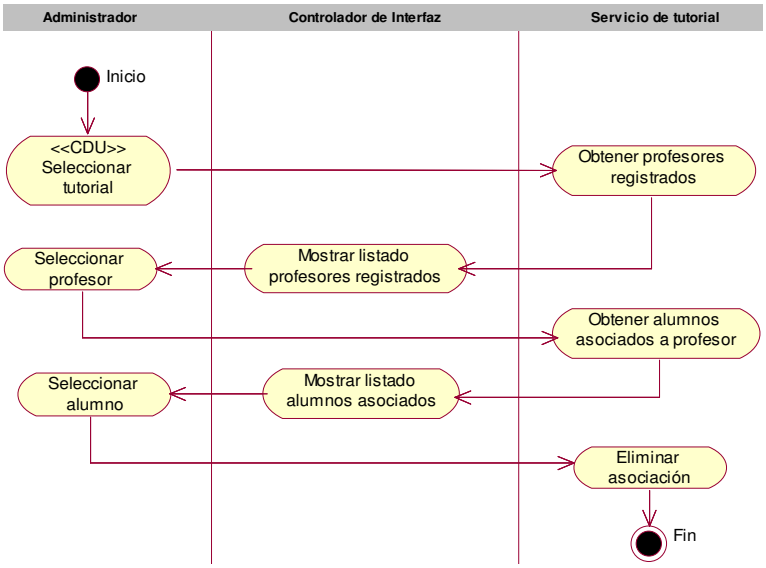


Diagrama de actividades de Importar Contenidos Tutorial.

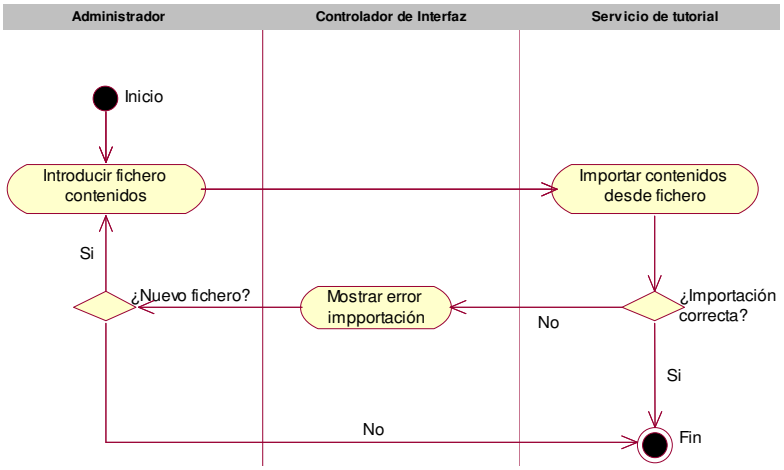


Diagrama de actividades de Importar Preguntas Cortas.

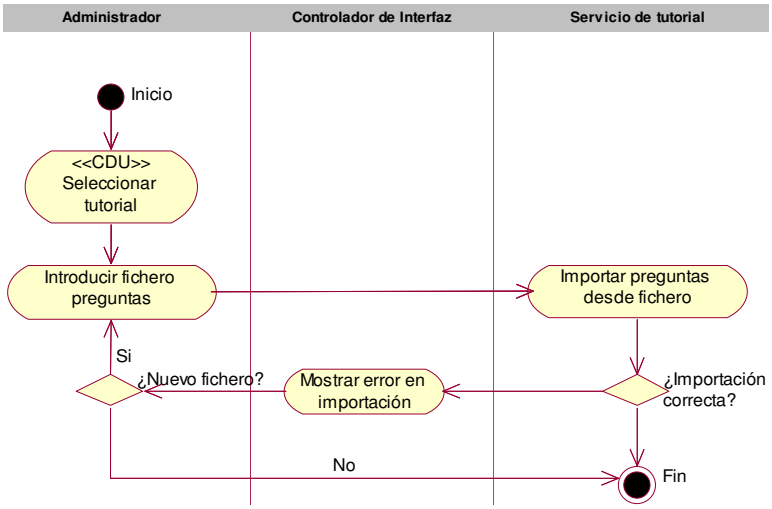


Diagrama de actividades de Importar Exámenes.

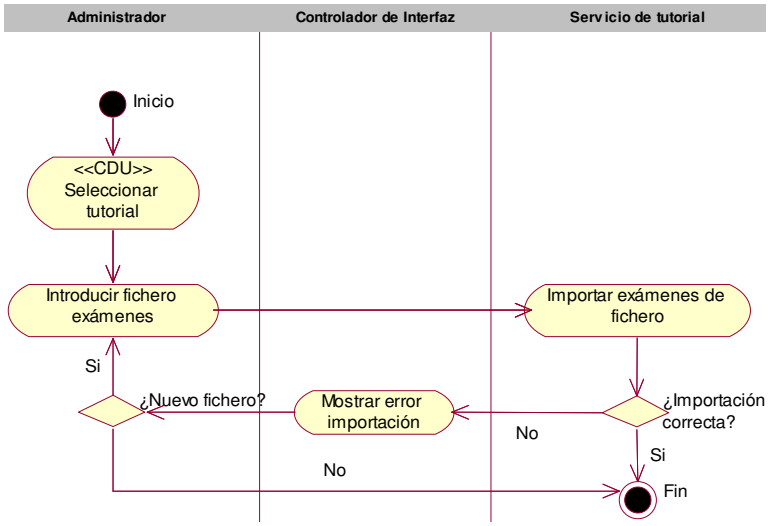


Diagrama de actividades de Importar Red Bayesiana.

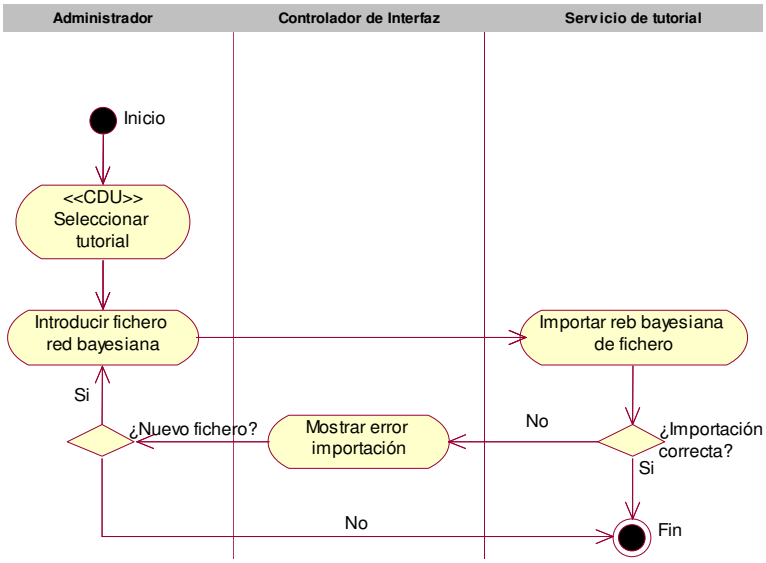


Diagrama de actividades de Exportar Contenidos Tutorial.

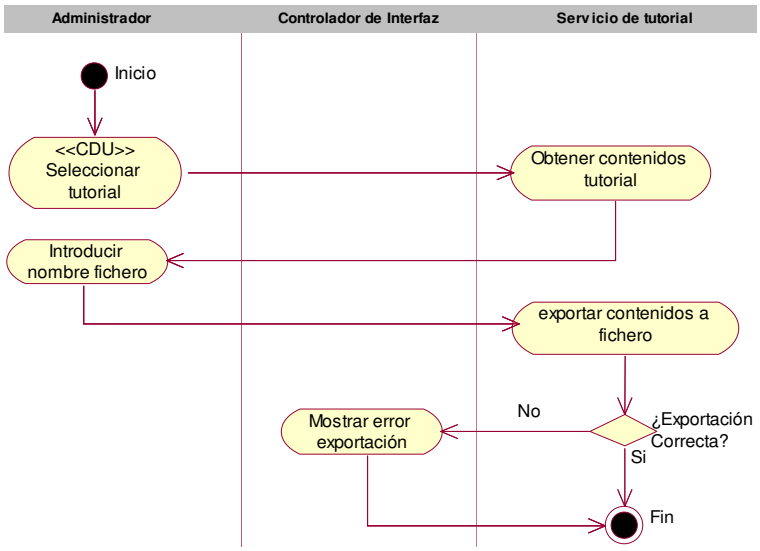


Diagrama de actividades de Exportar Preguntas Cortas.

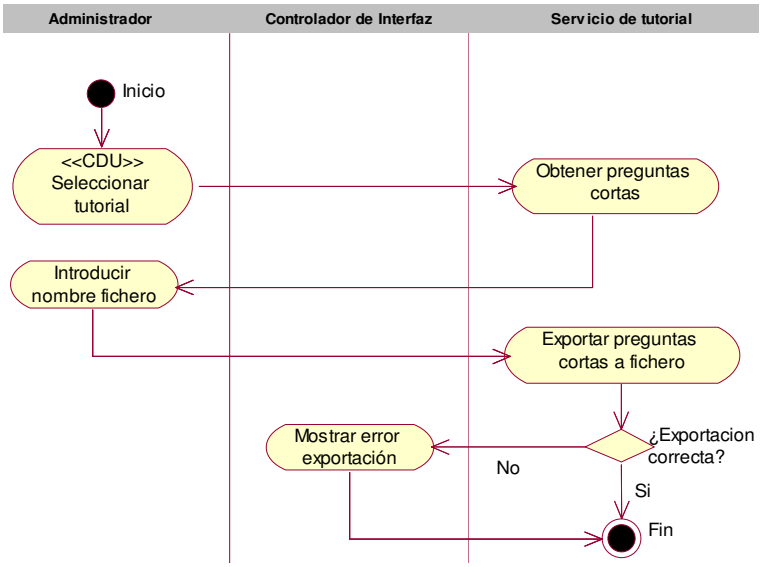


Diagrama de actividades de Exportar Exámenes.

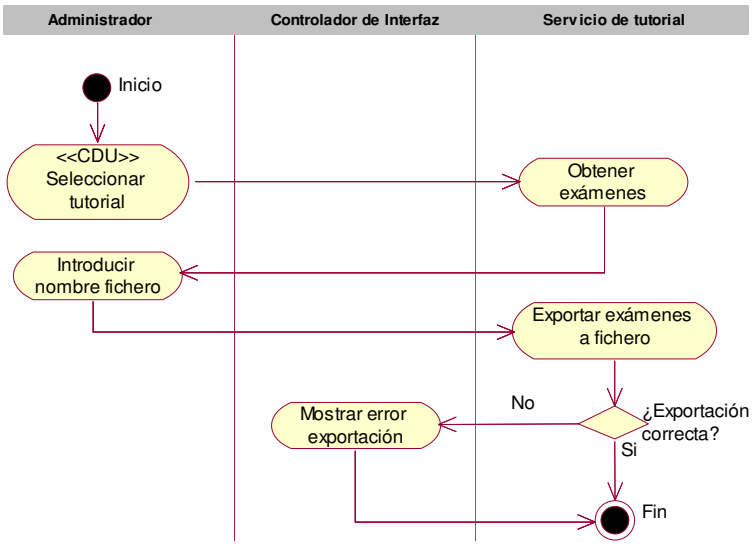
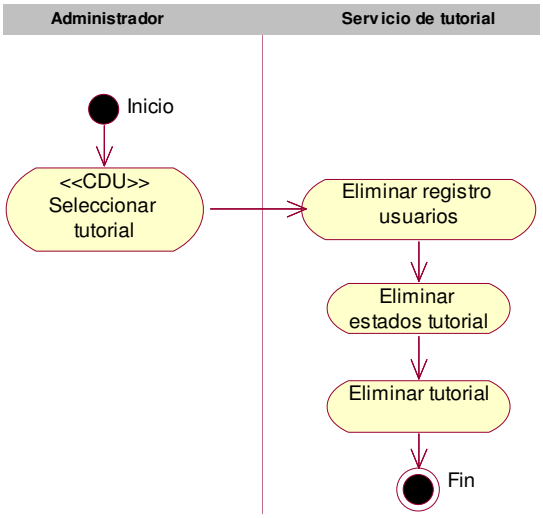


Diagrama de actividades de Eliminar Tutorial.


Administrador
Servicio de tutorial
Inicio
<<CDU>>
Seleccionar
tutorial
Eliminar registro
usuarios
Eliminar
estados tutorial
Eliminar tutorial
Fin