ESCUELA SUPERIOR DE INGENIERÍA INFORMÁTICA

INGENIERÍA TÉCNICA EN INFORMATICA DE GESTIÓN

Curso académico 2011/2012

Proyecto de Fin de Carrera

Estudio y Desarrollo de Animación Empática en Agentes Pedagógicos Conversacionales

Autor: Laura Muñoz Panadero

Tutora: Diana Pérez

Agradecimientos

Quiero dar las gracias a todas las personas que han estado a mi lado durante la realización de este proyecto. En primer lugar a Diana Pérez, mi tutora, que es la que me ha guiado y ayudado a conseguirlo, pero también a Beatriz Hernández Pajares, compañera y amiga, con la cual comenzamos juntas a darle una nueva vida a Shamael y su ayuda a sido imprescindible para la realización de este proyecto. También a las personas que han participado en las pruebas, por cederme un poco de su tiempo para poder mejorar este agente.

A mi hermana Ana, por colaborar conmigo a la hora de realizar los vídeos y prestando su voz a Shamael. Y por último, a mis padres y al resto de mi familia y a todos mis amigos que no han dejado de apoyarme en todo momento, aguantándome y evitando que me desmotivara. Gracias.

Resumen

Para la realización de este proyecto fin de carrera se ha querido realizar un Agente Pedagógico conversacional para el aprendizaje de inglés. En esta ocasión hemos utilizado un temario dirigido a unos niños de entre 9 a 11 años.

El objetivo principal era ampliar un agente ya existente, llamado Shamael, con la introducción de vídeos que acompañaran las explicaciones y posteriores preguntas que se han ido realizando al alumno. Estos vídeos sustituyen a las imágenes fijas haciendo que el agente sea más dinámico y dan una mayor expresividad ya que, por ejemplo si se ha acertado una pregunta la animación aparecerá contenta, felicitando al alumno y si la respuesta se ha contestado erróneamente la animación aparecerá más triste pero tampoco enfadada o regañando ya que se quiere animar al alumno para que lo siga intentado y no desmotivarlo.

La aplicación se ha desarrollado en JAVA y consta de una interfaz tipo Messenger donde se realizará la interacción Shamael – Alumno. Las preguntas, divididas en tres niveles, están almacenadas en un fichero XML y a partir de él se genera el diálogo de turnos pregunta-respuesta con el estudiante: mostrará por la pantalla de dicha interfaz las peguntas y el alumno las irá contestando e irá pasando de nivel según las vaya acertando hasta finalizar el curso. Adicionalmente el sistema irá creando un log donde recogerá la información del intercambio de preguntas respuestas. El agente también consta de un área para profesores y para el administrador donde se podrá añadir preguntas nuevas al agente, modificarlas o eliminarlas. El administrador, también podrá dar permisos a otros profesores para que puedan acceder a la aplicación o quitárselos.

Shamael ha sido probado con 10 usuarios de diversas edades y sin unos conocimientos muy avanzados en informática. Se han obtenido valores concretos por medio de su experiencia con el agente y por una encuesta de satisfacción que se realizó después. Con estas pruebas podremos hacernos una idea del funcionamiento que tiene la aplicación y podremos intentar mejorarla. Todos los usuarios que han probado Shamael han quedado muy satisfechos con las condiciones de la aplicación: ser intuitiva y usable.

Índice

Índice de Tablas

Índice de Imágenes

1. Introducción

Cada vez es más común la introducción de nuevas tecnologías en el ámbito de la enseñanza.

Los métodos de e-learning son muy utilizados ya que permiten una gran flexibilidad de organización al alumno, aunque no se recibe apoyo por parte de un tutor y esto, a veces, puede conllevar a una sensación de abandono. En este marco encontramos a los agentes pedagógicos conversacionales, sistemas educativos flexibles, centrados en el alumno y en la construcción conjunta del conocimiento, no solamente en la transmisión de la información declarativa. La presencia de un agente en un entorno interactivo, aunque no sea animado, puede tener un efecto positivo en la percepción de la experiencia educativa por parte del estudiante. Se crea una interacción persona-ordenador en lenguaje natural que permite el acceso a los ordenadores de personas no técnicas. Aunque el agente nunca estará diseñado para reemplazar al profesor, si no como un complemento de aprendizaje [1, HTTP1].

Shamael era un agente ya existente, si bien ha sido rediseñado para una nueva función que permita a los usuarios reforzar unos conocimientos de inglés mediante pequeñas explicaciones y preguntas. Pero la característica principal de este nuevo Shamael es la eliminación del avatar estático y la integración de vídeos para darle un mayor dinamismo y dotar al agente de animación empática. En este caso, se utilizará con niños pero dado que el aprendizaje del inglés puede darse a cualquier edad podría redefinirse con otros vídeos y preguntas adecuadas al nivel. Shamael consta de tres niveles de preguntas, lo que incentiva a los niños a esforzarse en aprender para acertar las preguntas y poder ir superando los niveles.

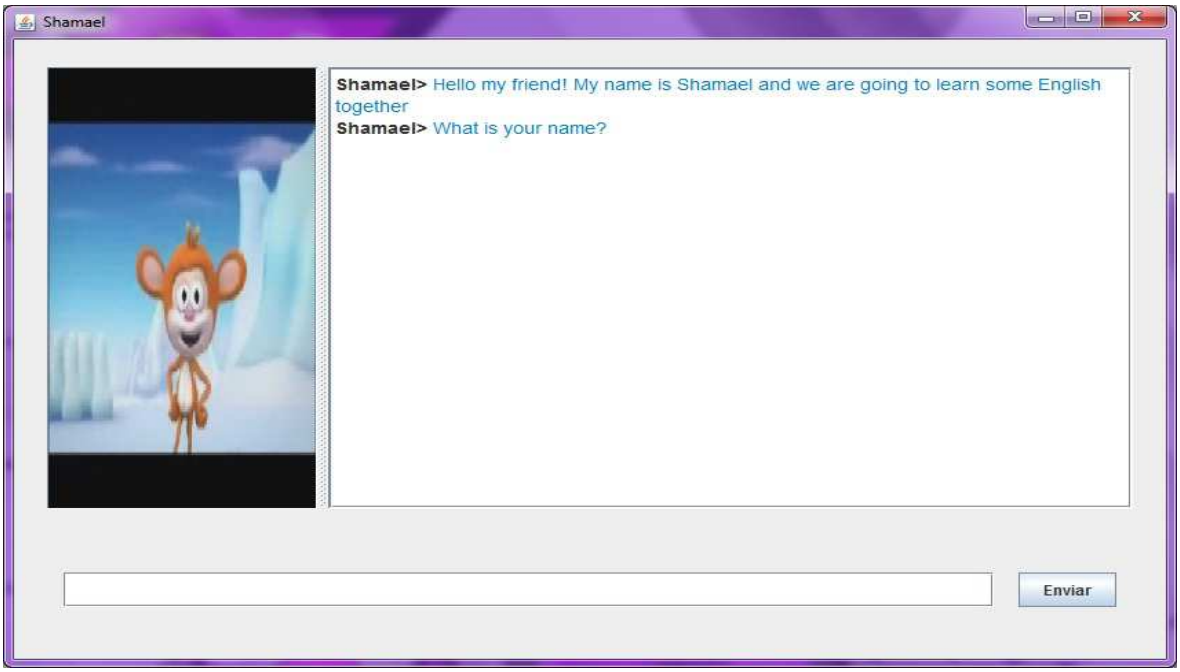


Imagen 1. Interfaz Agente Shamael

Otra característica que se ha añadido a Shamael es una zona de usuario para profesores, dónde cada profesor podrá gestionar sus datos y crear, modificar o eliminar sus propias preguntas para el agente. También estará la figura del administrador. Este usuario tendrá acceso a todas las preguntas del fichero XML, sin distinguir el profesor que la haya realizado, para modificarlas o eliminarlas al igual que podrá añadir nuevas preguntas. También tendrá acceso a los datos del resto de profesores pudiendo modificarlos, eliminarlos o registrar nuevos profesores para que tengan acceso a esta zona. Los profesores y el administrador estarán registrados en un fichero XML que Shamael leerá y accederán a esta zona después de loguearse al inicio de la aplicación.

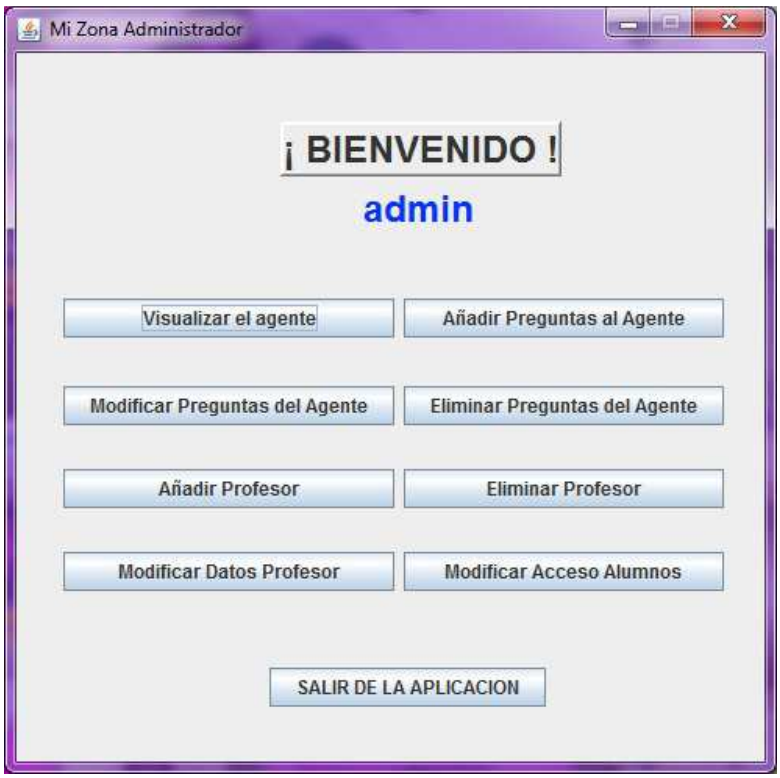


Imagen 2. Interfaz Zona Administrador

2. Objetivos

2.1. Descripción del problema

El objetivo principal del proyecto ha sido la creación de un agente como ayuda al aprendizaje de inglés cuya principal característica es la introducción de vídeos animados como avatar que expresarán los distintos estados por los que pasa el agente durante la conversación. Al ser una aplicación para niños tiene que ser sencilla de usar y atractiva por eso los videos divertirán al niño y le motivarán a seguir aprendiendo, pero sin distraerle. También se incluyen imágenes y sonidos en las distintas preguntas que propone el agente y en las explicaciones y los ejemplos para que sea más visual.

El agente dispone de tres niveles de preguntas. El alumno irá superando los cursos según vaya acertando las preguntas y en el caso que no acierte todas las preguntas repetirá el nivel pero sólo con las preguntas falladas. Esto se hace para no aburrir al alumno y que termine perdiendo el interés por tener que repetir preguntas que ya sabe una y otra vez. Este método, utilizado junto con los elementos multimedia de las preguntas dinamiza el programa y lo puede convertir en un juego para el alumno en el que centrará su atención y a la vez irá aprendiendo.

Para la zona del administrador se ha buscado realizar una aplicación muy intuitiva y manejable que no requiera de conocimientos informáticos para poder trabajar con ella. Esto permitirá al profesor poder modificar fácilmente las preguntas que se realizarán al alumno cambiando o añadiendo imágenes, sonidos, enunciados y poder ir actualizando el repertorio de preguntas.

2.2. Estudio de alternativas

2.2.1. Elección del Agente

La primera decisión que se tomó fue la elección del agente ya que el resto de decisiones vendrían dadas por esto, paradigma, lenguaje de programación, etc. El agente elegido fue Lingu. Lingu es un agente que ayuda al aprendizaje infantil mediante el análisis sintáctico y morfológico de frases. Este agente había sido desarrollado en Java como una aplicación de escritorio y usaba una base de datos para almacenar las palabras y su correspondencia sintáctica y morfológica. El sistema iba cogiendo palabras de la base de datos y formando la estructura de una frase, aunque a veces las frases no tuvieran un sentido coherente estaban bien estructuradas sintácticamente.

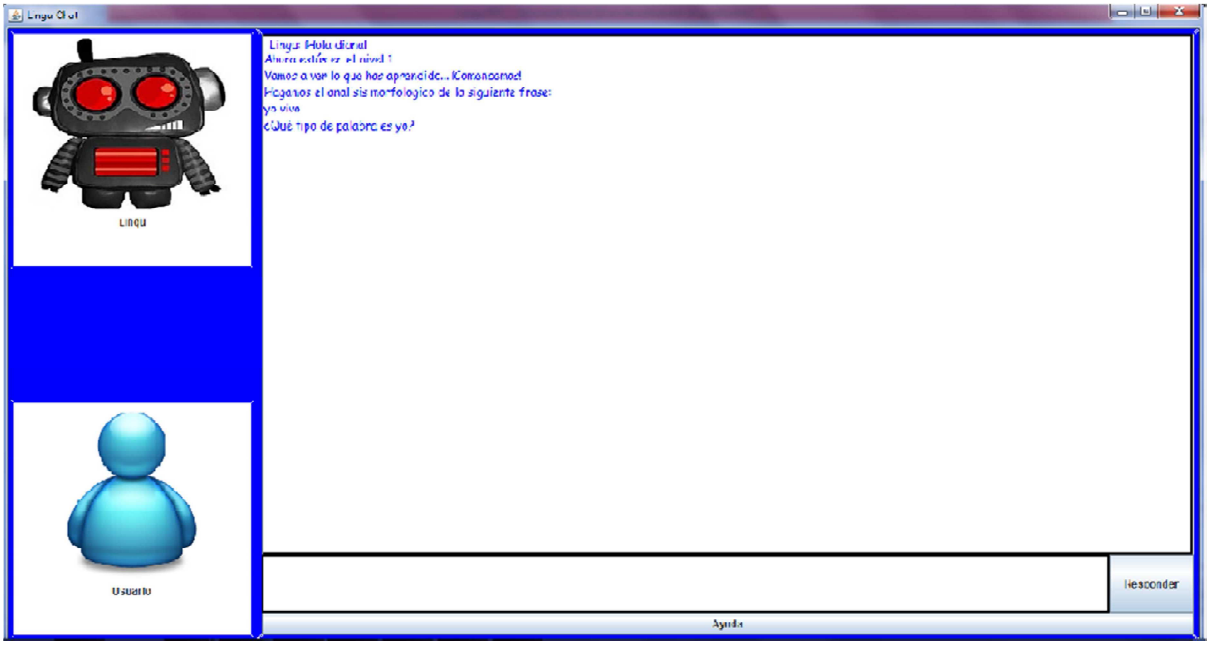


Imagen 3. Interfaz del Agente Lingu

Al instalar la aplicación no se consiguió importar la base de datos. Primero se utilizó la herramienta XAMPP, un paquete formado por un servidor web Apache, una base de datos MySQL y los intérpretes para los lenguajes PHP y Perl, y después se instaló MySQL Server 5.1 y tampoco se pudo realizar la instalación completa.

Se elige un nuevo agente, Shamel, una aplicación de escritorio desarrollada igualmente en Java (JavaSwing). Esta nueva aplicación no utiliza bases de datos para almacenar la información, si no que desde un fichero XML modificable lee las preguntas a realizar al alumno. Con este nuevo método se tendrá un mayor control sobre el diálogo y el

tratamiento de las emociones del agente. Se cambia también la temática del agente, ya que Shamael animaba a los alumnos a estudiar y daba consejos sobre buenos hábitos a la hora de hacerlo. El nuevo ámbito estará acotado al aprendizaje de inglés en niños, lo que aumenta la interacción alumno-agente con la realización de preguntas que el alumno deberá superar prestando atención no sólo al enunciado de la pregunta si no que se incluirán ejemplos, imágenes y archivos de audio para que el alumno utilice varías formas de centrar la atención.

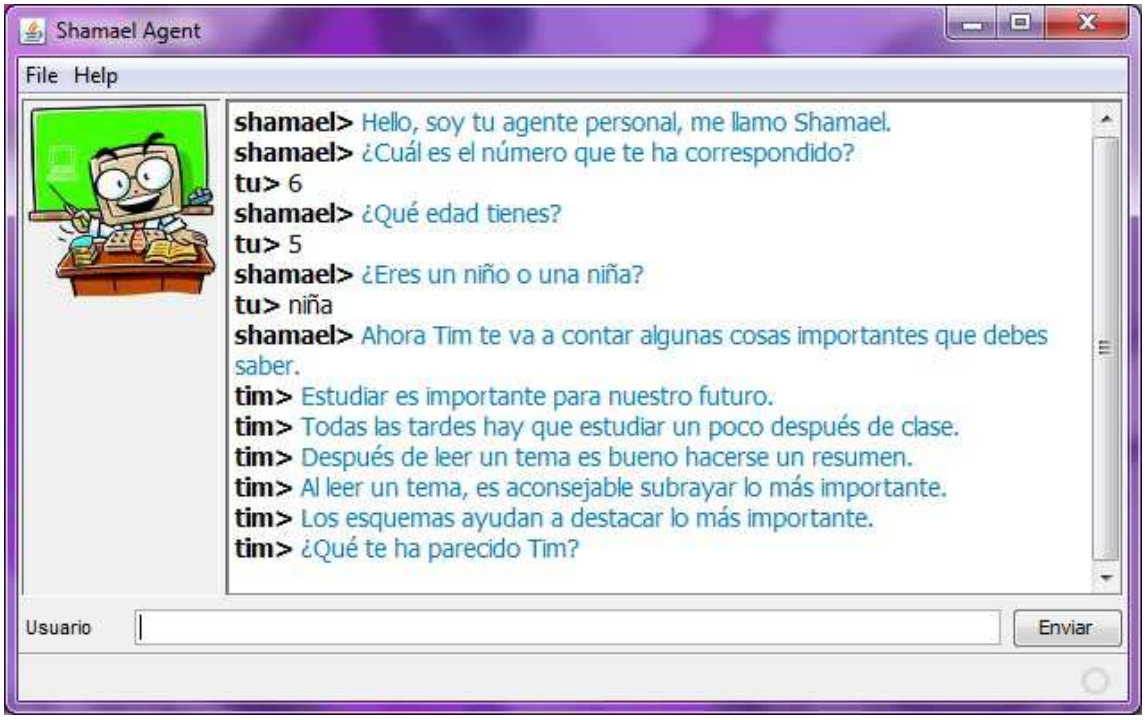


Imagen 4. Interfaz Agente Shamael

Los vídeos utilizados en el agente son recortes de la serie infantil “Monkey see, Monkey do ™”. A pesar de ser “Trade Mark” (Marca Registrada) puesto que el fin de este proyecto es educativo y no va a haber intenciones lucrativas por ninguna de las partes que vayan a utilizar este agente no se considera inadecuado el uso de dichas imágenes.

Para la realización del archivo XML con el temario se ha utilizado imágenes, sonidos y ejercicios del libro de 4º de primaria de inglés Beep de la editorial Richmond cedido por el profesorado de inglés del C.P. Santo Domingo (Alcorcón).

2.2.2. Lenguaje de Programación

Debido a que se ha ampliado un agente ya existente no ha habido posibilidad de elegir el lenguaje de programación. Se ha utilizado JAVA, un lenguaje orientado a objetos, que era el lenguaje en el que estaba ya implementado el agente.

- **Programación Orientada a Objetos (POO) [**HTTP2**]:** La programación orientada a objetos es una de las formas más populares de programar y viene teniendo gran acogida en el desarrollo de proyectos de software desde los últimos años debido a sus grandes capacidades y ventajas frente a las antiguas formas de programar. La idea principal de esta forma de programación es separar las partes complejas del programa en módulos o segmentos que sean ejecutados conforme se requieran. De esta manera tenemos un diseño modular, compuesto por módulos independientes que puedan comunicarse entre sí.

  Para entender este modelo vamos a revisar 4 conceptos básicos:

  - Objetos: Todo el programa está construido en base a diferentes componentes (Objetos), cada uno tiene un rol específico en el programa y todos los componentes pueden comunicarse entre ellos de formas predefinidas. Todo objeto tiene dos componentes: características y comportamiento. Un objeto de software mantiene sus características en una o más "variables", e implementa su comportamiento con "métodos". Un método es una función o subrutina asociada a un objeto. Definición teórica: Un objeto es una unidad de código compuesto de atributos y métodos relacionados.

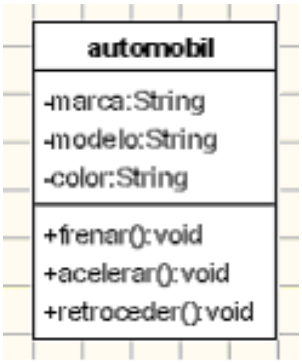


Imagen 5. Ejemplo clase en Java. Atributos y Métodos

  - Clases: La clase es un modelo o prototipo que define las variables y métodos comunes a todos los objetos de cierta clase. También se puede decir que una clase es una

plantilla genérica para un conjunto de objetos de similares características. Por otro lado, una instancia de una clase es otra forma de llamar a un objeto. En realidad no existe diferencia entre un objeto y una instancia. Sólo que el objeto es un término más general, pero los objetos y las instancias son ambas representación de una clase.

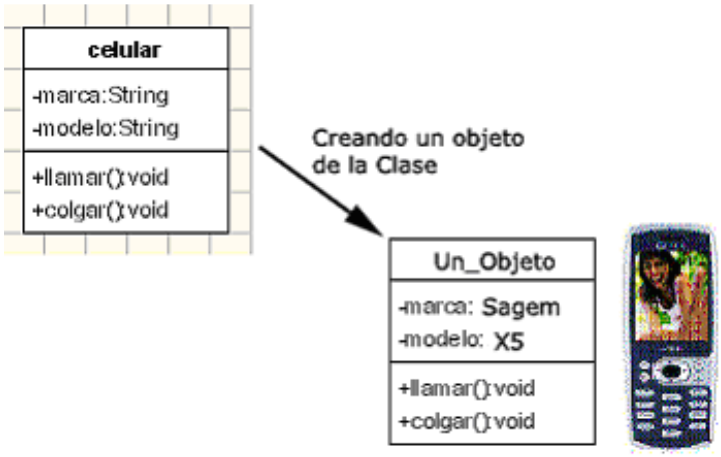


Imagen 6. Ejemplo Instanciación de un Objeto de la clase Celular

- Herencia: La herencia es uno de los conceptos más cruciales en la POO. La herencia básicamente consiste en que varias subclase pueden heredar las variables y métodos de otra clase, llamada superclase o clase padre. Esto significa que una subclase, aparte de los atributos y métodos propios, tiene incorporados los atributos y métodos heredados de la superclase. De esta manera se crea una jerarquía de herencia.

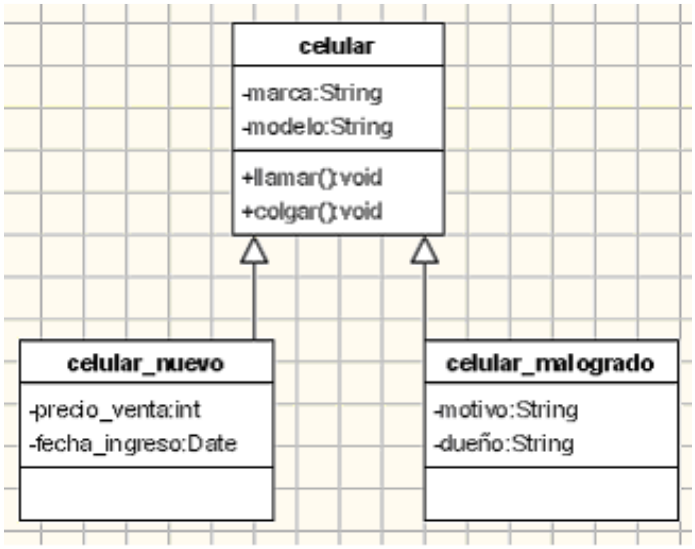


Imagen 7. Ejemplo herencia en clases java.

Características asociadas al POO:

- Abstracción: La abstracción consiste en captar las características esenciales de un objeto, así como su comportamiento. En los lenguajes de programación orientada a objetos, el concepto de Clase es la representación y el mecanismo por el cual se gestionan las abstracciones. Por ejemplo, en Java tenemos:

```
public class Automóvil {
// variables
// métodos
}
```

- Encapsulamiento: El encapsulamiento consiste en unir en la Clase las características y comportamientos, esto es, las variables y métodos. Es tener todo esto es una sola entidad. En los lenguajes estructurados esto era imposible. Es evidente que el encapsulamiento se logra gracias a la abstracción y el ocultamiento que veremos a continuación. La utilidad del encapsulamiento va por la facilidad para manejar la complejidad, ya que tendremos a las Clases como cajas negras donde sólo se conoce el comportamiento pero no los detalles internos, y esto es conveniente porque nos interesará será conocer qué hace la Clase pero no será necesario saber cómo lo hace.
- Ocultamiento: Es la capacidad de ocultar los detalles internos del comportamiento de una Clase y exponer sólo los detalles que sean necesarios para el resto del sistema. El ocultamiento permite dos cosas: restringir y controlar el uso de la Clase. Restringir porque habrá cierto comportamiento privado de la Clase que no podrá ser accedido por otras Clases. Y controlar porque daremos ciertos mecanismos para modificar el estado de nuestra Clase y es en estos mecanismos dónde se validarán que algunas condiciones se cumplan. En Java el ocultamiento se logra usando las palabras reservadas: public, private y protected delante de las variables y métodos.

¿Cuáles son las ventajas de un lenguaje orientado a objetos?

- Fomenta la reutilización y extensión del código.
- Permite crear sistemas más complejos.
- Relacionar el sistema al mundo real.
- Facilita la creación de programas visuales.
- Construcción de prototipos
- Agiliza el desarrollo de software
- Facilita el trabajo en equipo

- Facilita el mantenimiento del software

Lo interesante de la POO es que proporciona conceptos y herramientas con las cuales se modela y representa el mundo real tan fielmente como sea posible.

- **JAVA [**2**]:** El lenguaje de programación JAVA ha evolucionado en respuesta a una necesidad de un lenguaje que sea portable, permitiendo reutilizar sus componentes, ejecutarse en diferentes sistemas operativos, tiene un bajo mantenimiento y es fácilmente actualizable. Otro motivo de su evolución era la necesidad de realizar una programación fácilmente entendible por cualquier programador.

  Anteriormente, el lenguaje de programación C era importante para el desarrollo de todo tipo de proyectos. Ofrece una gran librería de funciones portables, y los compiladores de C han sido desarrollados para la mayoría de sistemas operativos, sin embargo, es un lenguaje difícil de aprender y podía provocar errores de código. Por otra parte, su portabilidad es limitada, es necesario recompilar el código para cada sistema operativo distinto. Tampoco disponía de una interfaz de usuario gráfica estándar, ni de multithreads o seguridad, carecía de ciertas funciones. C++ es un lenguaje de programación orientado a objetos que surgió de la evolución de C, utiliza la misma sintaxis pero su enfoque es más estructurado, aunque el mayor problema es descomponerlo en objetos hasta llegar a un código más robusto y seguro.

  Java fue desarrollado por Sun MicroSystems y se distinguía por ser un lenguaje orientado a objetos puro, altamente estructurado. Java también es muy portable gracias a su Máquina Virtual (VM), que funciona como una capa intermedia de la máquina que aísla su código del que se ejecuta en la CPU. Esta debe ser escrita y portada a sistemas operativos individuales, pero una vez que se ha hecho se funciona como un CPU virtual que puede ejecutar cualquier aplicación Java sin ser recompilado y sin tener en cuenta el sistema operativo. Esta propiedad es usualmente referida como Write One, Run Anywhere –WHORA- (escríbelo una vez, ejecútalo en cualquier sitio) y hace que Java sea ideal para el trabajo en red.

  Java prescinde del uso de punteros, haciéndolos invisibles al programador y, así, eliminando uno de los mayores problemas del área de programación. Además Java incorpora seguridad, el código puede ser cargado y compilado en tiempo de ejecución. También resuelve problemas de herencia mediante la supresión de la herencia múltiple y

mejora el mantenimiento previniendo la sobrecarga de operadores. Otro problema que revuelve es las pérdidas de memoria con una gestión de memoria automática.

Otros Lenguajes de Programación Orientada a Objetos:

- En 1985, E. Stroustrup extendió el lenguaje de programación C a C++, es decir C con conceptos de clases y objetos, también por esas fechas se creó desde sus bases el lenguaje EIFFEL.
- El lenguaje de desarrollo más extendido para aplicaciones Web, el PHP 5, trae todas las características necesarias para desarrollar software orientado a objetos.
- Además de otros lenguajes que fueron evolucionando, como el Pascal a Delphi.

2.2.3. Almacenamiento de Datos

Para almacenar la conversación que ofrecerá Shamael se utiliza un archivo XML, respetando la manera en la que estaba implementado agente en el que se basa este proyecto, llamado Temario.xml.

Desde este archivo se controlará la conversación mediante distintas etiquetas: cuando el agente da una explicación, cuando formula una pregunta, cuando es el turno del usuario de contestar, cuando mostrar imágenes y sonidos relativos a las preguntas y lo más importante controla el video de estado del agente (formulando una pregunta, felicitando, etc.) que se tiene que reproducir en este momento. Y entre estas etiquetas estará la información que se quiere mostrar por la interfaz al usuario.

A la hora de acceder al agente, se distingue entre los usuarios alumno, profesor y administrador. Para ello se creó una base de datos MySQL para almacenar la información de cada usuario dependiendo del tipo al que pertenezcan. Así se recopilaría la información acerca de las conversaciones mantenidas para ir avanzando en el aprendizaje.

MySQL [HTTP3] es un sistema de gestión de bases de datos relacionales Open Source, cualquiera puede usar y modificar el software y bajarlo de internet sin pagar nada. El servidor de base de datos MySQL es muy rápido, fiable y fácil de usar. Ha sido probado con un amplio rango de compiladores diferentes y funciona en diferentes plataformas. Tiene un uso completo de multi-threaded mediante threads del kernel (núcleo), lo que hace que sea

un sistema de reserva de memoria muy rápido. Pueden usarse fácilmente en múltiples CPUs si están disponibles.

El servidor está disponible como un programa separado para usar en un entorno de red cliente/servidor. También está disponible como biblioteca y puede ser incrustado (linkado) en aplicaciones autónomas. En cuanto a la seguridad, es un sistema de privilegios y contraseñas que es muy flexible y seguro, y que permite verificación basada en el host. Las contraseñas son seguras porque todo el tráfico de contraseñas está cifrado cuando se conecta con un servidor.

La siguiente imagen muestra la tabla "Usuarios", de la base de datos, dónde estaban almacenados los datos de cada uno de los usuarios del agente.

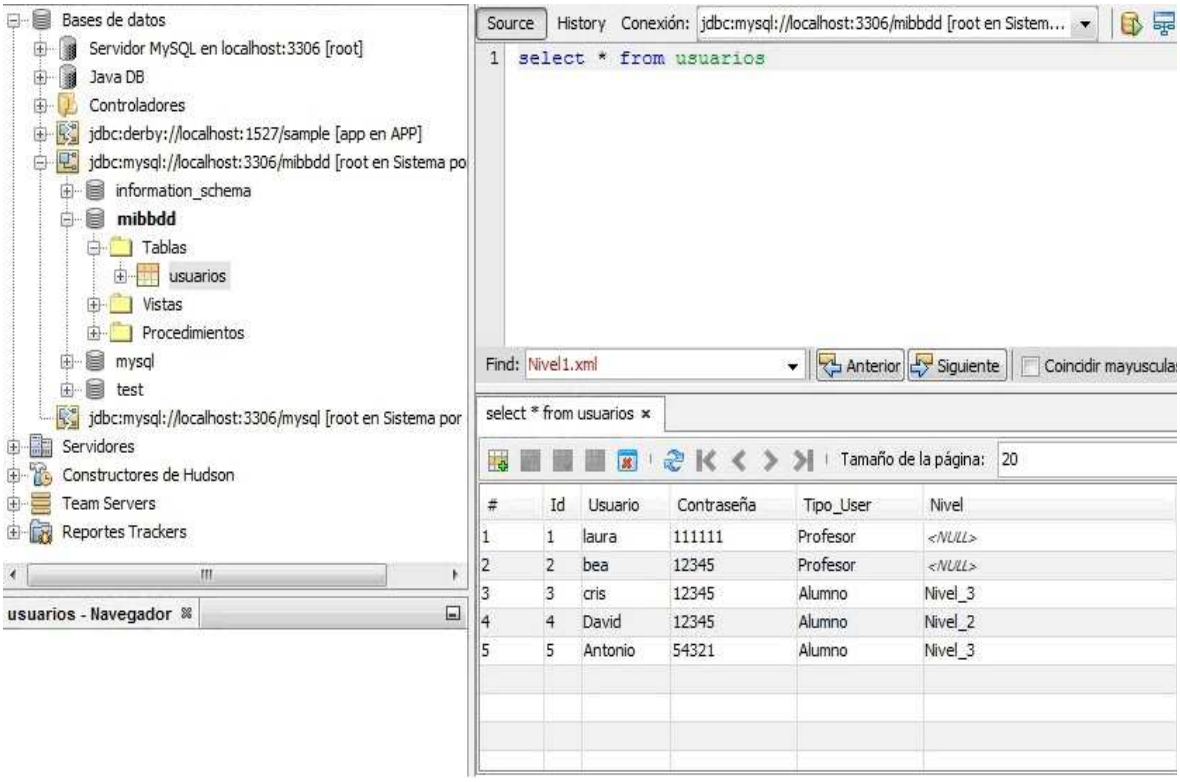


Tabla1. Tabla Usuarios Base de Datos

- ID: Es la clave primaria de la tabla. Será el identificador de cada uno de los usuarios del agente.
- Nombre: No podrá tener valor NULL y será el nombre de usuario con el que accederá a la aplicación.
- Contraseña: No podrá tener valor NULL y será la password utilizada por el usuario para acceder a la aplicación.

- Tipo_User: No podrá tener valor NULL y sólo podrá tener el valor Alumno o Profesor.
- Nivel: Solamente podrá tener valor NULL en el caso que Tipo_User valga Profesor. Para el usuario Alumno, Nivel deberá tener una de estos tres niveles: Nivel_1, Nivel_2 o Nivel_3.

Con esta base de datos se podrán añadir nuevos usuarios o modificar y eliminar los datos de los usuarios ya existentes. También se podrá tomar el nivel en el que está un alumno para utilizarlo al cargar las preguntas del nivel correspondiente del XML.

Posteriormente se eliminó la distinción entre alumnos, se implementó el sistema para que simplemente hubiera un usuario y una contraseña genérica para todos los alumnos. Se decidió entonces pasar toda la información de los profesores a otro XML y eliminar la base de datos. Así, toda la información que manejase el agente lo haría con los mismos procedimientos y no tendría dos fuentes de recogida de información tan diferentes.

Otra de las ventajas de eliminar la base de datos es que no en todas partes está instalado MySQL, por lo tanto habría que hacer un instalador que contenga MySQL y la bases de datos para poder instalar la aplicación. Sin embargo XML es un lenguaje estático que no necesita de instalador, con lo cual se puede instalar la aplicación en cualquier máquina e incluso este XML podría ser leído por otro agente.

Finalmente, toda la información que se ha ido generando en la conversación, preguntas acertadas por el alumno, las falladas, las veces que repite un nivel, etc se recoge en un archivo log. Este archivo también tiene estructura XML y se va generando según se produce el intercambio pregunta-respuesta. Este fichero no se eliminará hasta que el alumno haya superado el curso completo y se irá mandando progresivamente vía e-mail a una cuenta de correo introducida en el código del agente. El envío del log será útil para su posterior estudio, no sólo para ver el avance del alumno, si no porque entre las preguntas se incluyen preguntas a cerca de la aceptación del agente y así podremos ver lo que los alumnos opinan sobre el diseño, temario, usabilidad, etc. de Shamael.

2.3. Metodología empleada

El ciclo de vida software es la descripción de las distintas formas de desarrollo de un proyecto o aplicación informática. Un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso [3].

Los modelos orientados a objetos caracterizan el desarrollo orientado al objeto por:

- La eliminación de fronteras entre fases, ya que debido a la naturaleza iterativa del desarrollo orientado al objeto, estas fronteras se difuminan cada vez más.
- Una nueva forma de concebir los lenguajes de programación y su uso, ya que se incorporan bibliotecas de clases y otros componentes reutilizables.
- Un alto grado de iteración y solapamiento, lo que lleva a una forma de trabajo muy dinámica.

Los expertos en tecnología de objetos proponen seguir un **desarrollo iterativo e incremental**.

- Iterativo: Existe un ciclo de desarrollo análisis-diseño-implementación-análisis que permite hacer evolucionar al sistema.
- Incremental: el sistema se divide en un conjunto de particiones.
- Las actividades de validación, verificación y aseguramiento de la calidad se pueden realizar de forma continuada.

Se pueden combinar los modelos tradicionales de ciclo de vida con los más modernos:

- Idea del “microproceso” y del “macroproceso”. Seguir cualquiera de los ciclos de vida más modernos y seguir un modelo en cascada [HTTP4].

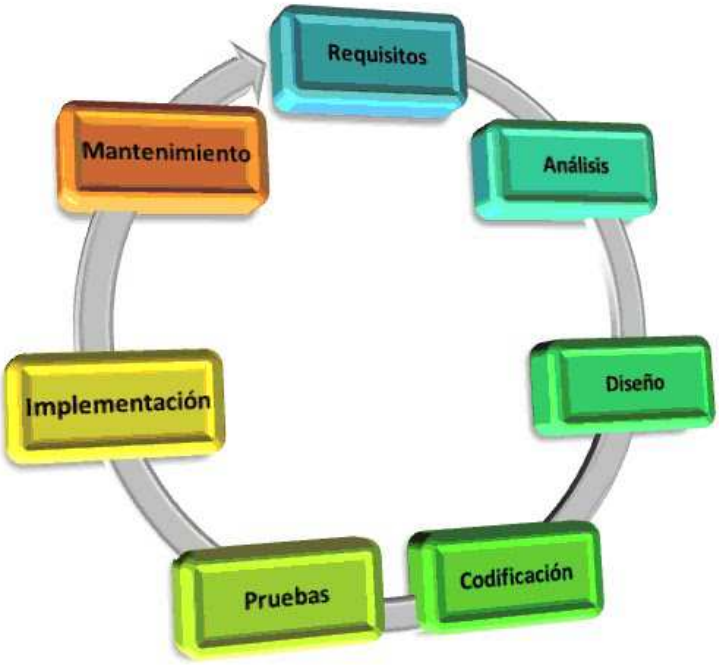


Imagen 8. Modelo de desarrollo Iterativo e Incremental

Para el desarrollo de software orientado a objetos no basta usar un lenguaje orientado a objetos. También se necesitará realizar un análisis y diseño orientado a objetos.

El modelado visual es la clave para realizar el análisis OO. Desde los inicios del desarrollo de software OO han existido diferentes metodologías para hacer esto del modelado, pero sin lugar a duda, el Lenguaje de Modelado Unificado -Unified Modeling Language- (UML) es el más usado actualmente.

Según los mismos diseñadores del lenguaje UML, éste tiene como fin modelar cualquier tipo de sistemas (no solamente de software) usando los conceptos de la orientación a objetos. Y además, este lenguaje debe ser entendible para los humanos y máquinas.

Actualmente en la industria del desarrollo de software tenemos al UML como un estándar para el modelado de sistemas OO. Fue la empresa Racional que creó estas definiciones y especificaciones del estándar UML, y lo abrió al mercado. El UML consta de todos los elementos y diagramas que permiten modelar los sistemas en base al paradigma orientado a objetos. Los modelos orientados a objetos cuando se construyen en forma correcta, son fáciles de comunicar, cambiar, expandir, validar y verificar. Este modelado en UML es flexible al cambio y permite crear componentes plenamente reutilizables [HTTP5].

3. Descripción Informática

3.1. Especificación

3.1.1. Captura de Requisitos

El diseño de esta aplicación cumple con los siguientes requisitos funcionales y no funcionales. Esto permite modelar los requisitos del sistema desde la perspectiva del usuario:

Requisitos Funcionales:

- RF1: El usuario podrá autentificarse en el sistema introduciendo su usuario y contraseña.
- RF2: El programa leerá las preguntas de un archivo XML y las mostrará al usuario alumno por la pantalla de la interfaz tipo messenger.
- RF3: El sistema controlará los turnos de pregunta-respuesta con el alumno.
- RF4: El sistema mostrará los vídeos del agente según los estados de la conversación.
- RF5: El sistema irá creando un log compuesto por la conversación agente-alumno.
- RF6: El sistema enviará a un correo electrónico el log con la información de la sesión.
- RF7: El usuario administrador podrá añadir preguntas nuevas al agente que el sistema almacenará en el archivo XML.
- RF8: El administrador podrá cargar las preguntas del fichero XML para visualizarlas y las podrá eliminar o modificar. El sistema registrará los cambios en el fichero XML.
- RF9: El administrador podrá añadir nuevos profesores para que puedan acceder a la aplicación. El sistema almacenará los nuevos datos en un fichero XML de usuarios.
- RF10: El administrador podrá eliminar a un profesor del fichero XML de usuarios.
- RF11: El administrador podrá modificar los datos de cualquier profesor. El sistema realizará los cambios en fichero de usuarios.
- RF12: El administrador podrá modificar los datos de acceso del usuario alumno. El sistema lo modificará en el fichero XML de usuarios.

Requisitos No Funcionales:

- RNF1: La aplicación deberá cumplir los principios de interacción usuario-ordenador, contará con una interfaz amena, visual, usable e intuitiva para favorecer la comunicación.

- RNF2: El tiempo de respuesta de la aplicación debe ser óptimo.
- RNF3: La aplicación podrá usarse desde cualquier ordenador independientemente del sistema operativo.
- RNF4: Soporte afectivo, el agente puede intentar animar al estudiante y mantener su atención.

3.1.2. Obtención de casos y uso y diagramas

Los diferentes casos de uso son los siguientes:

- **"Autentificar Usuario"**

Actor: Administrador

Descripción: El usuario accede a la aplicación.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
1. El administrador ejecuta la aplicación.	2. La aplicación se abre y aparece una ventana de autentificación.
3. El usuario introduce su usuario y contraseña para acceder al agente.	4. Aparecerá la zona de gestión del administrador.

Tabla2. Caso de Uso "Autentificar Usuario-Administrador"

Imagen 9. Diagrama Caso de Uso "Autentificar Usuario-Administrador"

- **“Añadir Profesor”**

Actor: Administrador

Descripción: el administrador quiere añadir un profesor nuevo para que pueda trabajar con el agente.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
	1. Aparecerá la zona de gestión del administrador.
2. El administrador pulsa sobre el botón “Añadir Profesor”	3. El sistema muestra una ventana solicitando los datos de registro del nuevo profesor (identificador, código de profesor y contraseña)
4. El administrador introduce los datos solicitados y pulsa “Guardar”	5. El sistema comprueba que el nuevo usuario no esté ya registrado. En tal caso archiva sus datos.

Tabla3. Caso de Uso “Añadir Profesor”

Caminos alternativos:
El administrador decide no registrar ningún nuevo profesor y pulsa el botón “Cancelar”.
El nuevo profesor ya está registrado, el sistema pide introducir nuevos datos al administrado

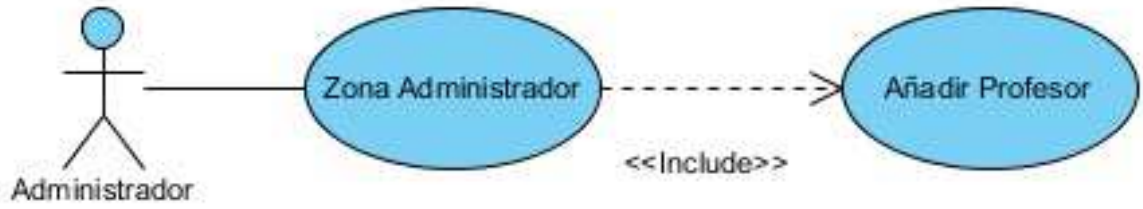


Imagen 10. Caso de Uso “Añadir Profesor”

- **“Eliminar Profesor”**

Actor: Administrador

Descripción: el administrador quiere eliminar un profesor del XML.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
	1. Aparecerá la zona de gestión del administrador.
2. El administrador pulsa sobre el botón “Eliminar Profesor”	3. El sistema muestra una ventana solicitando el código del profesor a eliminar
4. El administrador introduce el dato solicitado y pulsa “Borrar”	5. El sistema comprueba que el profesor exista. En tal caso elimina sus datos.

Tabla 4. Caso de Uso “Eliminar Profesor”

Caminos alternativos:
El administrador decide no borrar ningún profesor y pulsa el botón “Cancelar”.
El profesor no está registrado, el sistema pide introducir de nuevo los datos al administrador

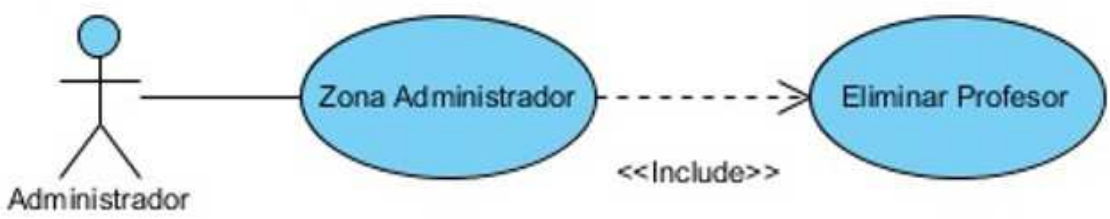


Imagen 11. Caso de Uso “Eliminar Profesor”

- **“Modificar Profesor”**

Actor: Administrador

Descripción: el administrador quiere modificar los datos de acceso de un profesor.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
	1. Aparecerá la zona de gestión del administrador.
2. El administrador pulsa sobre el botón “Modificar Profesor”	3. El sistema muestra una ventana solicitando el nuevo identificador y la nueva contraseña
4. El administrador introduce los datos solicitados y pulsa “Modificar”	5. El sistema comprueba que el profesor exista. En tal caso modifica sus datos.

Tabla 5. Caso de Uso “Modificar Profesor”

Caminos alternativos:
El administrador decide no modificar ningún profesor y pulsa el botón “Cancelar”.
El profesor no está registrado, el sistema pide introducir de nuevo los datos al administrador.

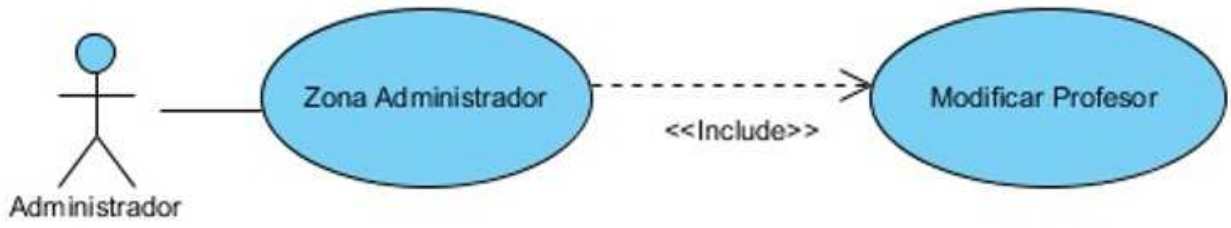


Imagen 12. Caso de Uso “Modificar Profesor”

- **“Modificar Acceso Alumnos”**

Actor: Administrador

Descripción: el administrador quiere modificar los datos de acceso de los alumnos.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
	1. Aparecerá la zona de gestión del administrador
2. El administrador pulsa sobre el botón “Modificar Acceso Alumnos”	3. El sistema muestra una ventana solicitando el nuevo identificador y la nueva contraseña de acceso para los alumnos
4. El administrador introduce los datos solicitados y pulsa “Modificar”	5. El sistema modifica los datos

Tabla 6. Caso de Uso “Modificar Acceso Alumno”

Caminos alternativos:
El administrador decide no modificar los datos de acceso de los alumnos y pulsa el botón “Cancelar”.

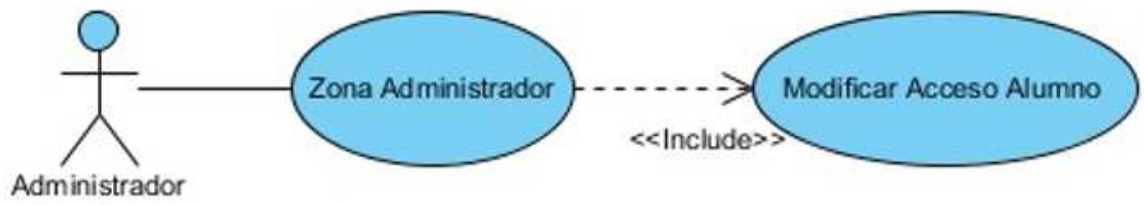


Imagen 13. Caso de Uso “Modificar Acceso Alumnos”

- **“Eliminar Pregunta”**

Actor: Administrador

Descripción: el administrador quiere eliminar una pregunta del XML.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
	1. Aparecerá la zona de gestión del administrador.
2. El administrador pulsa sobre el botón “Eliminar Pregunta”	3. El sistema muestra una ventana solicitando el nivel de la pregunta. Muestras las preguntas de dicho nivel.
4. El administrador selecciona la pregunta que desea eliminar y pulsa “Eliminar”	5. El sistema elimina la pregunta.

Tabla 7. Caso de Uso “Eliminar Pregunta”

Caminos alternativos:
El administrador decide no eliminar ninguna pregunta y pulsa “Volver”.
Al administrador decide no eliminar la pregunta seleccionada y pulsa “Volver” para poder elegir otra pregunta.

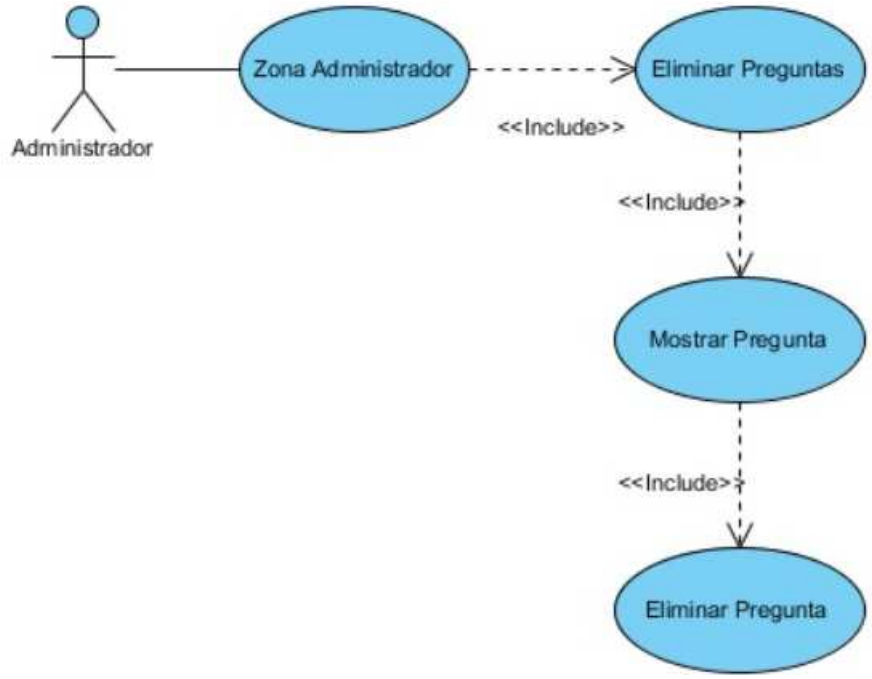


Imagen 14. Caso de Uso “Eliminar Pregunta”

- **“Añadir Pregunta”**

Actor: Administrador

Descripción: el administrador quiere añadir una pregunta nueva al XML.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
	1. Aparecerá la zona de gestión del administrador.
2. El administrador pulsa sobre el botón “Añadir Pregunta”	3. El sistema solicita la nueva pregunta, su respuesta, el nivel, se puede escoger entre un vídeo del profesor u otro mudo por defecto, y la opción de añadirle una imagen o un audio.
4. El administrador introduce los datos solicitados, ve la muestra y pulsa “Guardar”	5. El sistema añade la pregunta nueva.

Tabla 8. Caso de Uso “Añadir Pregunta”

Caminos alternativos:
El administrador decide no añadir ninguna pregunta nueva y pulsa “Volver”.
Al administrador no le gusta como queda la nueva pregunta y en la muestra y pulsa en volver para regresar a la ventana de “Añadir Pregunta”.

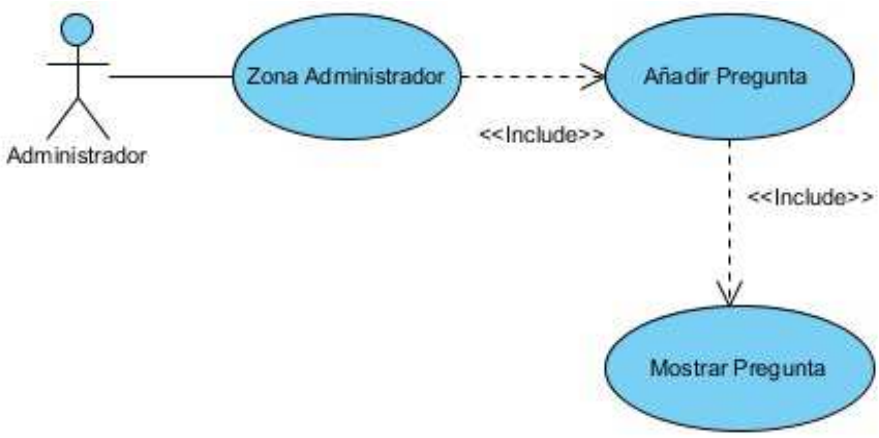


Imagen 15. Caso de Uso “Añadir Pregunta”

- **“Visualizar Agente”**

Actor: Administrador

Descripción: El usuario ver una demo del agente.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
	1. Aparecerá la zona de gestión del administrador.
2. El administrador pulsa el botón “Visualizar Agente”	3. El sistema muestra una demo del Agente

Tabla 9. Caso de Uso “Visualizar Agente”

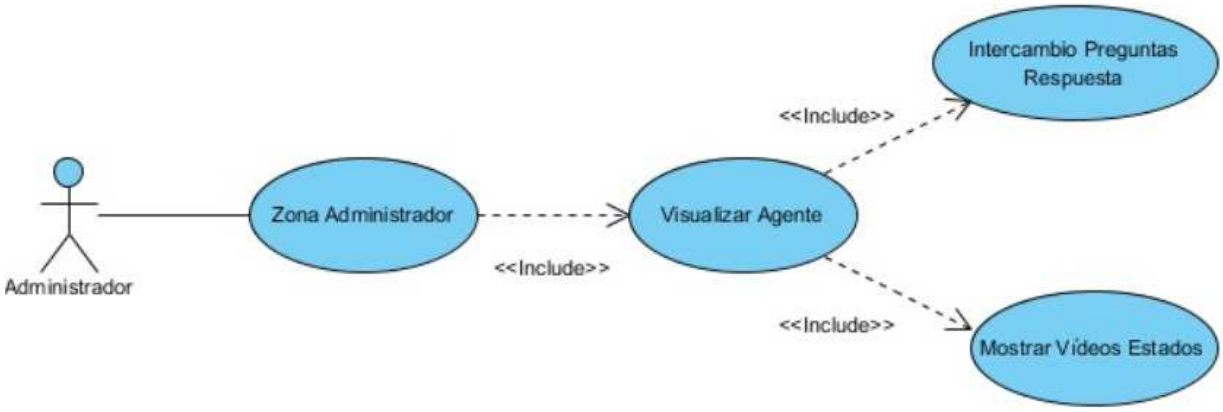


Imagen 16. Caso de Uso “Visualizar Agente”

- **“Modificar Pregunta”**

Actor: Administrador

Descripción: el administrador quiere modificar una pregunta del XML.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
	1. Aparecerá la zona de gestión del administrador.
2. El administrador pulsa sobre el botón “Modificar Pregunta”	3. El sistema muestra una ventana solicitando el nivel de la pregunta. Muestras las preguntas de dicho nivel.
4. El administrador selecciona la pregunta que desea editar y pulsa “Modificar”	5. El sistema elimina muestra una nueva ventana donde introducir los nuevos datos de la pregunta (enunciado, respuesta, imagen, sonido, video)
6. El usuario completa los datos de los campos que desea cambiar y pulsa en “Modificar”	7. El sistema sustituye la pregunta.

Tabla 10. Caso de Uso “Modificar Pregunta”

Caminos alternativos:
El administrador decide no modificar ninguna pregunta y pulsa “Volver”.
Al administrador decide no modificar la pregunta seleccionada y pulsa “Volver” para poder elegir otra pregunta.

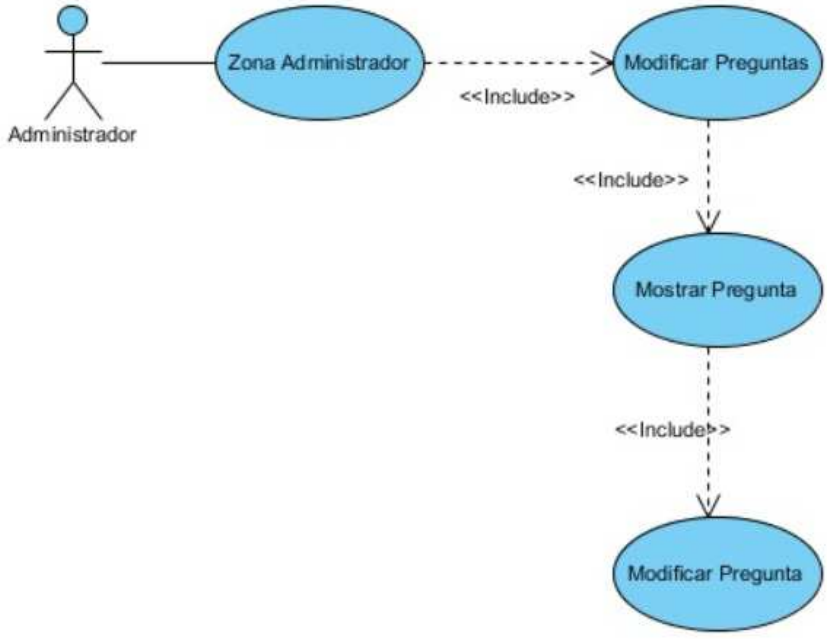


Imagen 17. Caso de Uso “Modificar Pregunta”

- **“Autentificar Usuario”**

Actor: Alumno

Descripción: El usuario accede al agente.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
1. El alumno ejecuta la aplicación.	2. La aplicación se abre y aparece una ventana de autentificación.
3. El usuario introduce su usuario y contraseña para acceder al agente.	4. Aparecerá el agente.

Tabla 11. Caso de Uso “Autentificar Usuario-Alumno”

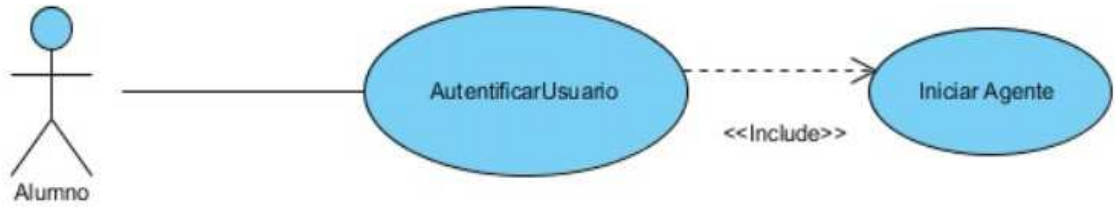


Imagen 18. Caso de Uso “Autentificar Usuario-Alumno”

- **“Interacción Agente”**

Actor: Alumno

Descripción: El usuario interactúa con el agente.

ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
1. El alumno ejecuta la aplicación.	2. La aplicación se abre y aparece una ventana de autentificación.
3. El usuario introduce su usuario y contraseña para acceder al agente.	4. Aparecerá el agente.
5. El usuario irá contestando las distintas preguntas que le formula el agente hasta completar el curso.	

Tabla 12. Caso de uso “Interacción Agente”

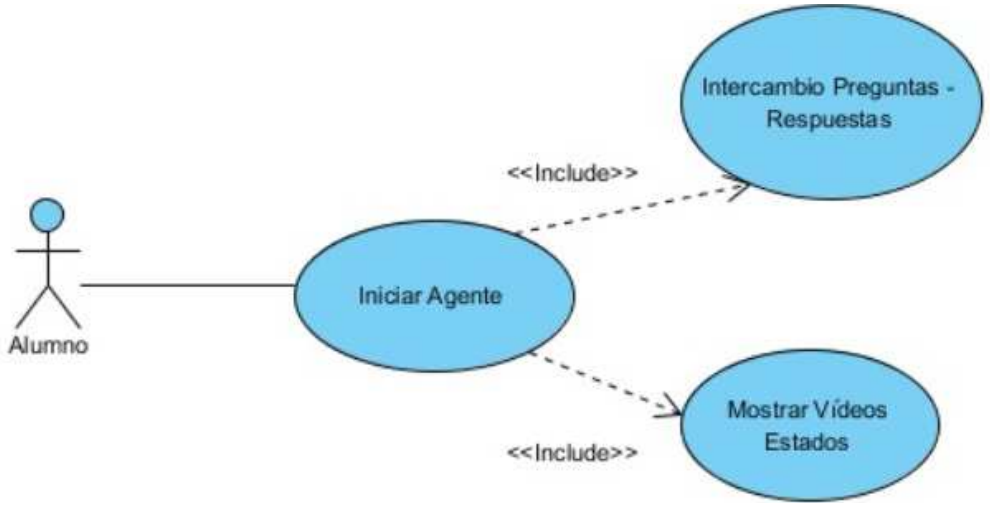


Imagen 19. Caso de Uso "Interacción Agente"

- **Casos de Uso Actor – Administrador:**

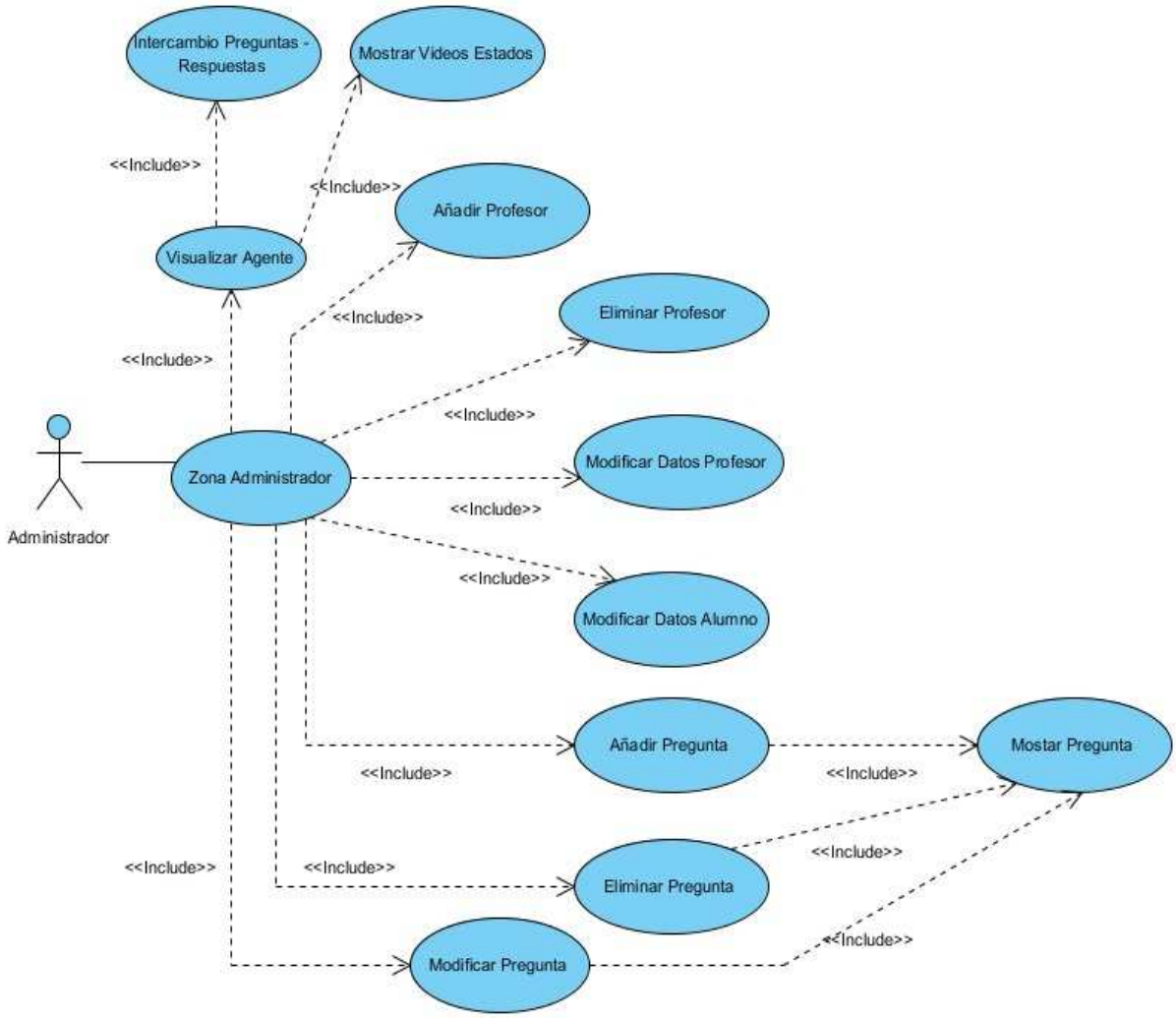


Imagen 20. Casos de Uso Administrador

- **Casos de Uso Actor – Alumno:**

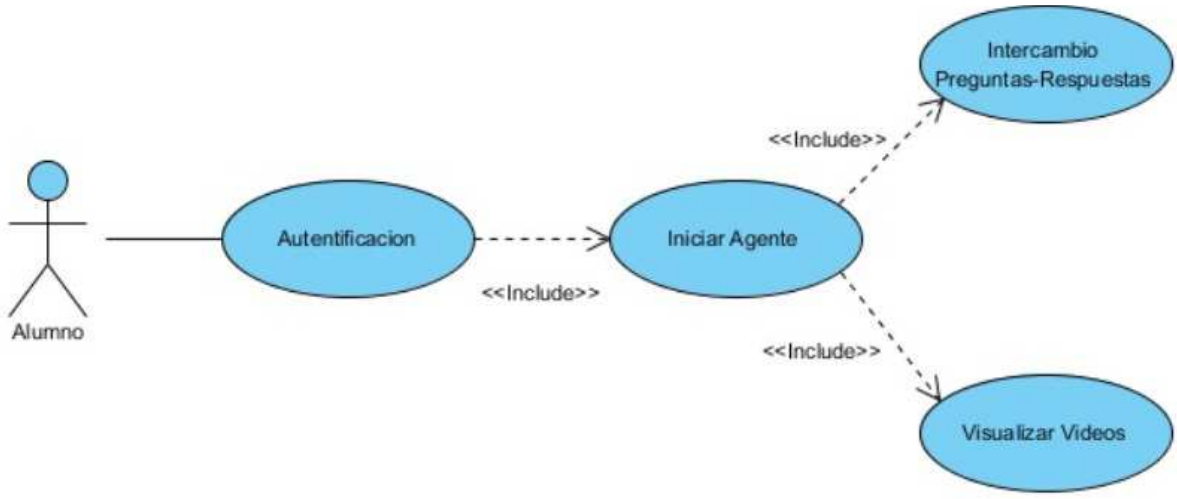


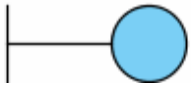
Imagen 21. Casos de Uso Alumno

3.2. Análisis

3.2.1. Descripción de las clases de Análisis

En esta parte se busca representar los requisitos funcionales y casos de uso mediante diagramas de interacción.

Clase Límite o interfaz: Modelan la interacción entre el sistema y los actores, implica recibir (y presentar) información y peticiones de (y hacia) los usuarios y los sistemas externos. También modelan las partes del sistema que dependen de sus actores, por lo que clarifican y reúnen los requisitos en los límites del sistema. Representan la interfaz del sistema pero a nivel conceptual.



Clase Control: Representan coordinación, secuencia, transacciones y control de otros objetos. Se usan para encapsular el control de un caso de uso concreto. No representan ni interacciones con el usuario ni problemas de almacenamiento de información. Manejan y coordinan las acciones y los flujos de control principales, delegando trabajo a otros objetos (de interfaz, de entidad).

Clase Entidad: Modelan la información y el comportamiento asociado de algún fenómeno o concepto. Persisten durante la aplicación y pueden provenir de las entidades del dominio o de las del negocio, pero no tienen por qué corresponderse completamente. Pueden ser pasivas o activas (comportamiento complejo), encapsulan información y operaciones asociadas y suelen mostrar una estructura de datos lógica y contribuyen a comprender de qué información depende el sistema.

3.2.2. Diagramas de interacción

- Diagrama de interacción para el caso de uso: "Autentificar Usuario - Alumno":
  En este diagrama se muestra como es la interacción usuario-agente a la hora de que un Alumno desee acceder a la aplicación.

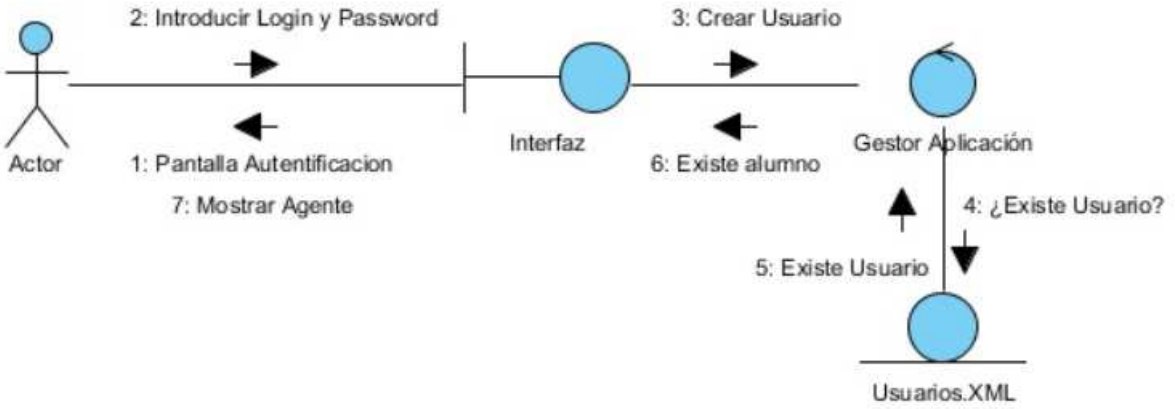


Imagen 22. Diagrama de Interacción "Autentificar Usuario – Alumno"

- Diagrama de interacción para el caso de uso: "Autentificar Usuario - Administrador":
  En este diagrama se muestra como es la interacción usuario-agente a la hora de que un Administrador desee acceder a la zona del Administrador.

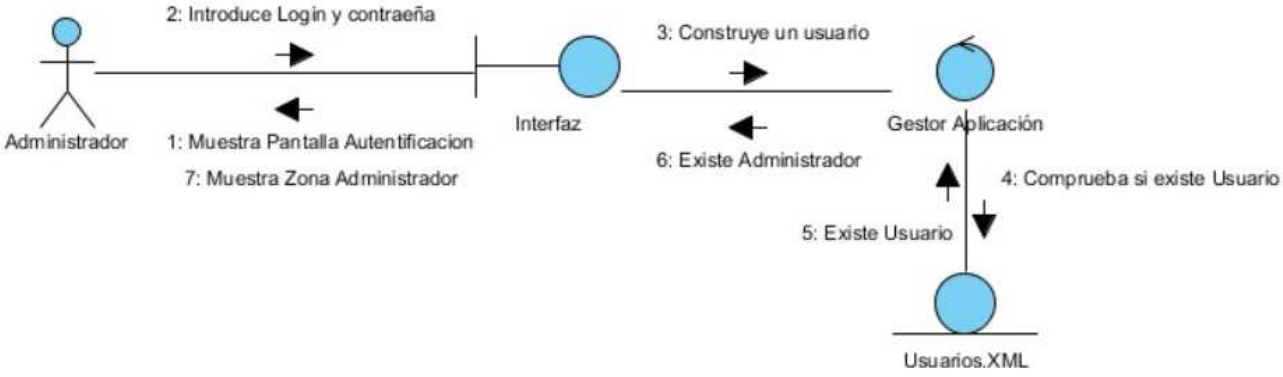


Imagen 23. Diagrama de Interacción "Autentificar Usuario–Administrador"

- Diagrama de interacción para el caso de uso: "Interacción Agente-Alumno":
  En este diagrama se puede observar cómo se produce la interacción entre el alumno y el agente, desde que el agente carga la teoría para mostrárselas al alumno, después le hace la pregunta a la vez que reproduce el vídeo de estado_pregunta. En este caso, el alumno responde correctamente a la pregunta propuesta, el agente lo comprueba y le muestra un vídeo de felicitación a la vez que registra la pregunta como contestada.

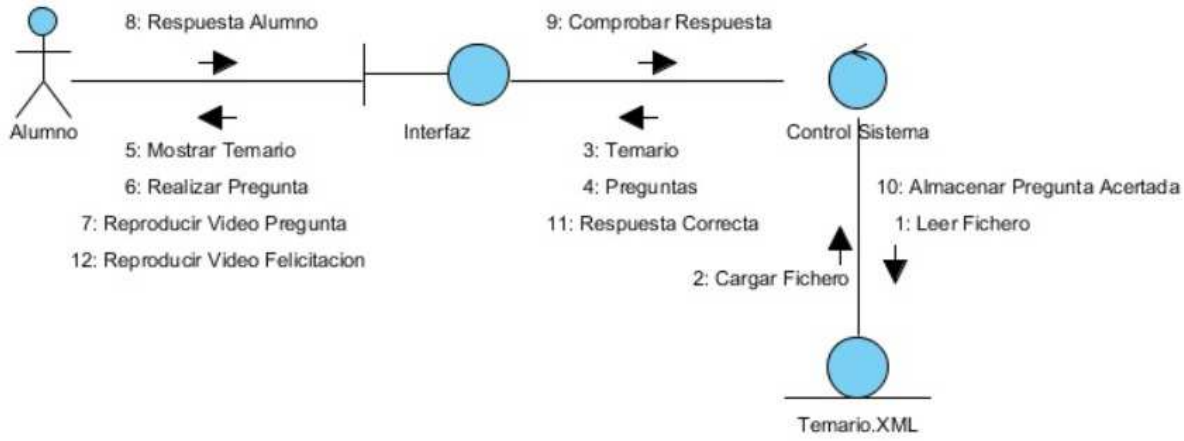


Imagen 24. Diagrama de Interacción "Interacción Agente-Alumno"

- Diagrama de interacción para el caso de uso: "Eliminar Profesor-Administrador":
  El administrador decide Eliminar los accesos a un determinado profesor. Para ello escoge la opción de la Zona de Administrador e introduce, en la pantalla que le muestra el sistema, el código del profesor que desea eliminar, el sistema comprueba si existe el profesor y en tal caso lo elimina.

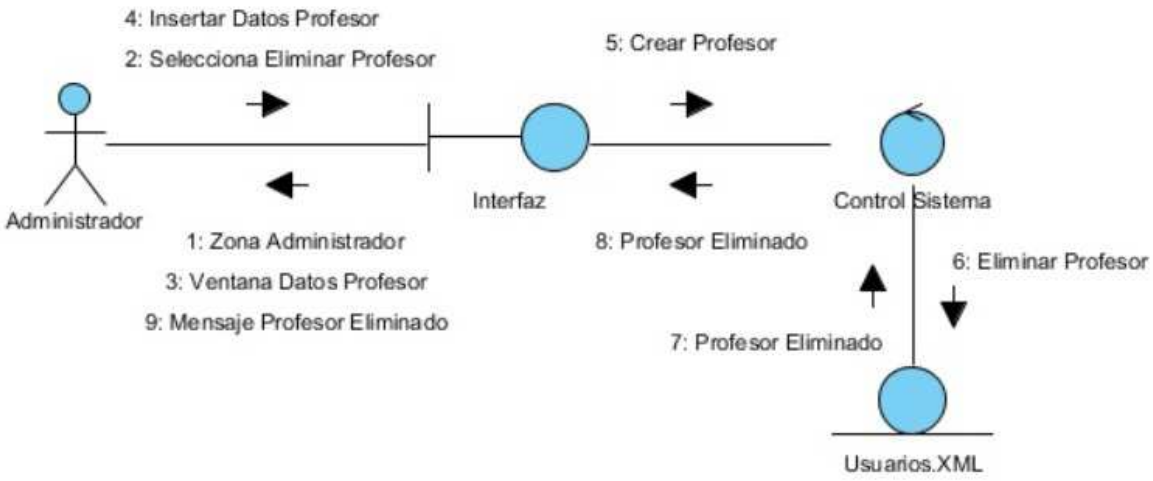


Imagen 25. Diagrama de Interacción "Eliminar Profesor"

3.3. Diseño

En esta fase se busca definir de forma mucho más precisa la aplicación centrándose en la arquitectura del sistema. Para ello se realiza un diagrama con los distintos módulos que componen la aplicación y el diagrama de clases.

3.3.1. Arquitectura de Alto Nivel

A continuación se muestran los diagramas de módulos que forman la aplicación:

- Módulos Administrador y Alumno integrados en la aplicación

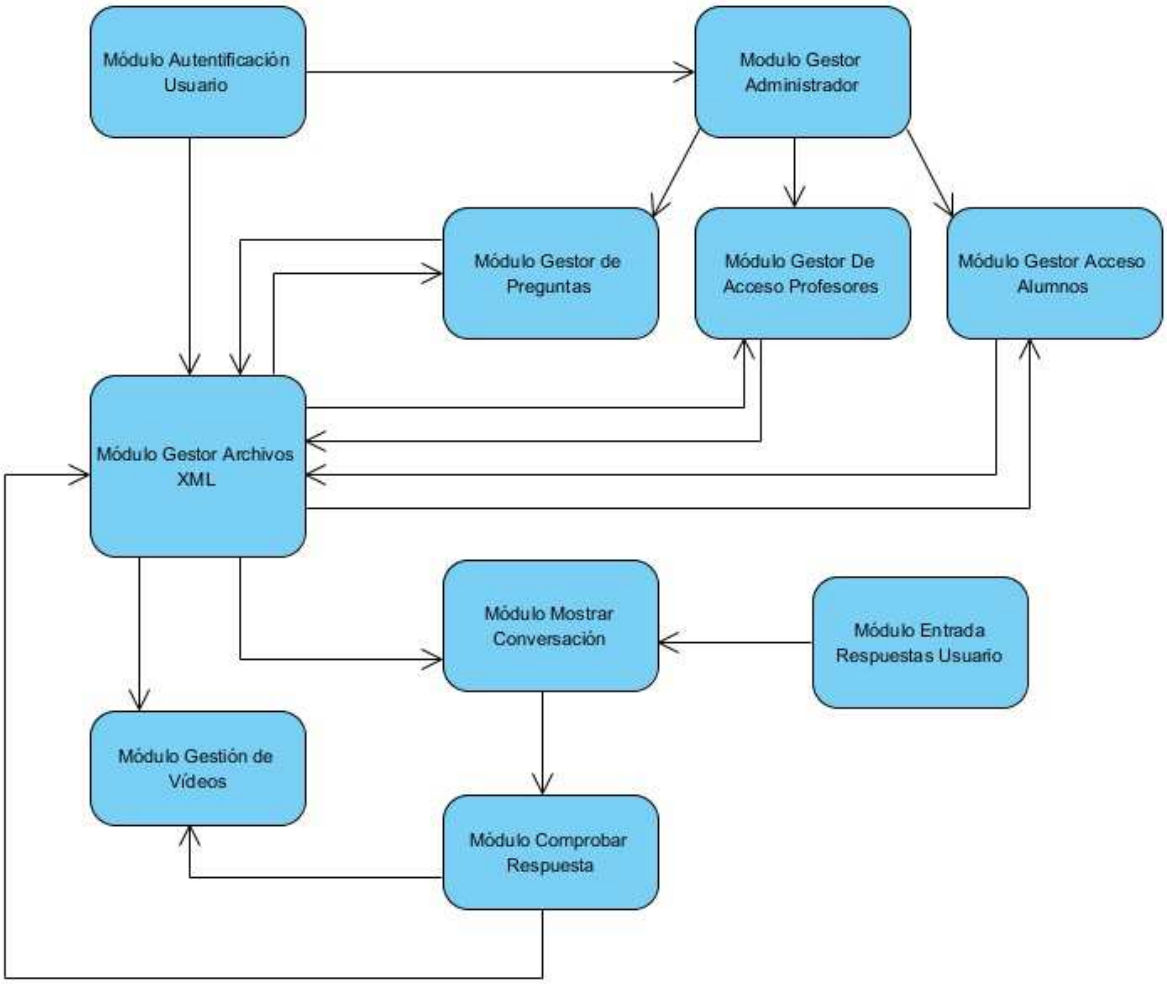


Imagen 26. Diagrama de Módulos de la Aplicación

- Módulos Aplicación Administrador

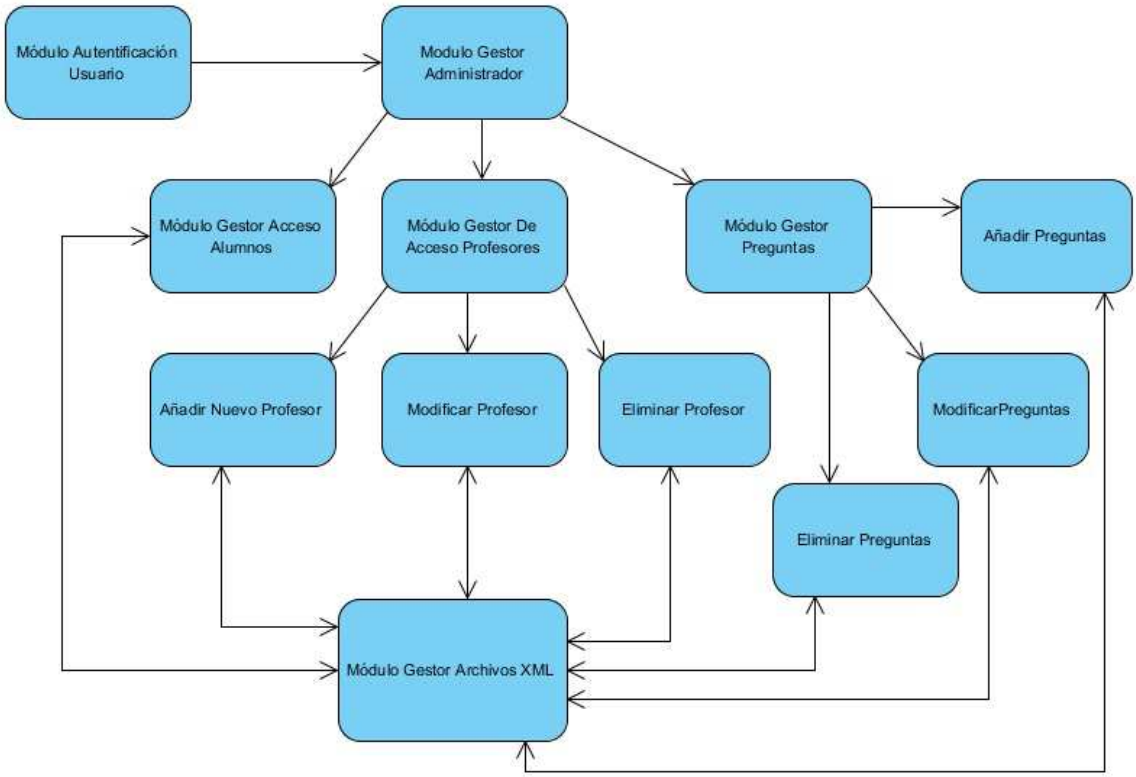


Imagen 27. Diagrama de Módulos Administrador

- Módulos Aplicación Administrador

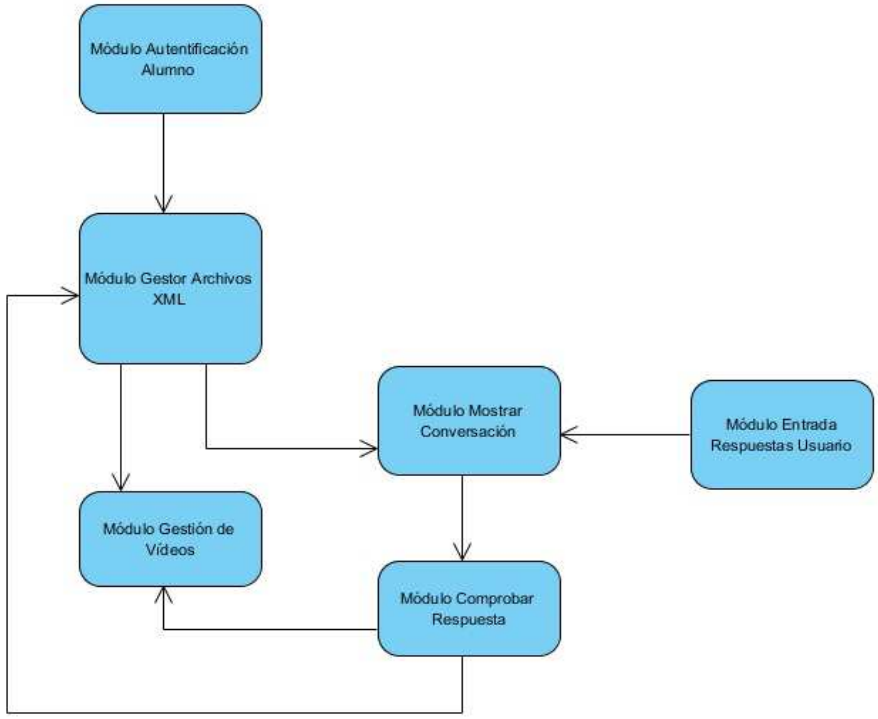


Imagen 28. Diagrama de Módulos Alumno

Como se puede observar en las figuras anteriores los dos roles principales son el administrador y el alumno.

El administrador **gestiona** las preguntas: entra en la aplicación para añadir, modificar o eliminar preguntas. Estas serán las preguntas que se les mostrará a los alumnos en el agente. También gestionarán los accesos de los profesores para que puedan entrar en la aplicación y el de los alumnos.

Los estudiantes **reciben** las preguntas: entran en la aplicación donde contestan a las preguntas que les muestra el agente y se les muestran los distintos tipos de vídeos correspondientes al estado de la conversación. Toda la información extraída de la conversación se almacenará en el log.

El Módulo Gestor de Archivos XML será el módulo principal de la aplicación, pues será el que lea y modifique los XML y el que mandará ordenes a los otros módulos.

3.4. Archivos XML

Como para la realización de este sistema no se han utilizado bases de datos se hará una descripción de cómo se han utilizado los ficheros XML para almacenar datos.

Para manejar los ficheros se ha utilizado el API XML DOM (Document Object Model). DOM genera un árbol de objetos (Nodos) que representan al documento XML. Cada uno de estos nodos puede ser de muchos tipos: documento, elemento, atributo, comentario,… y todos se relacionan entre sí mediante relaciones padre-hijo.

DOM no es sólo un parseador, sino que también permite modificar los documentos parseados, así como crear documentos XML totalmente nuevos. Al haber un árbol de objetos relacionados entre sí, la programación en DOM es más intuitiva para el desarrollador.

Realmente DOM es una especificación de W3C, por lo que su objetivo principal es proporcionar una interfaz estándar que pueda ser utilizada en una amplia variedad de entornos y lenguajes de programación (Java, JavaScript, C++, PPHP, Python, Ruby,…). Esto tiene su parte positiva, ya que es independiente del lenguaje de programación, pero también su parte negativa, ya que muchas clases son interfaces y esto implica que su uso sea un poco más laborioso.

A continuación se revisan los tipos de nodos son más utilizados en DOM. Todas estas interfaces se encuentran en el paquete org.w3c.dom y todas heredan de la interfaz Node:

- **Document**: representa al documento XML en sí. Es el objeto principal del árbol y sólo puede haber uno por documento. El resto de objetos del árbol deben de pertenecer al ámbito de este objeto para poder estar en el árbol. Esto quiere decir que todos los objetos que vayan a ser insertados en el árbol deben ser creados o importados por el Document antes de ser insertados como hijo de cualquier elemento. Almacena información como el encoding (getEncoding), la versión xml (getXmlVersion), si el documento es Standalone (getStandalone), y por supuesto tiene como único nodo hijo a la etiqueta raíz del documento (getDocumentElement).
- **Element**: representa una etiqueta XML. Almacena entre otras la lista de atributos que posee (getAttributes), la lista de nodos hijos (getChildNodes), contenido de tipo texto (getTextContext). Tiene métodos de acceso directo a su primer hijo (getFristChild) y acceso directo a su próximo nodo hermano (getNextSibling). Ambos devuelven un Node por lo que el primer hijo o el próximo hermano pueden ser de cualquier tipo (Element, Text, CDATASection, …)
- **Attr**: Atributo dentro de un Element
- **CharacterData**: texto dentro del XML. Puede ser del tipo CDATASection, Text o Comment. Estas tres interfaces heredan de CharacterData.

DOM te permite interactuar con estos elementos del árbol permitiendo añadir, modificar, eliminar, copiar y mover nodos. Se puede partir de un documento parseado, proveniente de un fichero o crear un documento XML completamente nuevo. En cualquier momento el árbol puede ser convertido a cadena de caracteres o bytes (serializar). Los pasos serían:

1.- Parsear un documento XML en formato String o crear un documento XML completamente bueno, desde cero. Esto nos proporciona el objeto Element.

2.- Modificar, añadir o eliminar elementos al Document y sus hijos.

3.- Serializar el árbol XML en formato String o en un fichero para su almacenaje.

DOM guarda ese árbol en memoria (RAM) y por tanto está ocupando un espacio que puede ser considerable dependiendo del tamaño del fichero XML. En principio no hay por qué alarmarse ya que hoy en día la RAM no es un problema para los ordenadores. Sin embargo, tampoco hay que olvidar del todo esto ya que si se descuida pueden producirse errores de OutOfMemoryException [HTTP6].

Antes de la realización de este proyecto no conocía la librería DOM por lo que se ha tenido que estudiar su funcionamiento y utilidad para poder trabajar con los ficheros XML que se usan en la aplicación.

A continuación se muestran los tres ficheros XML que utilizamos para guardar el temario (temario.xml), la información generada en la conversación (log.xml) y el que utilizamos para almacenar los usuarios que tienes acceso a la aplicación.

- **Temario.xml:**

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
- <temario>
    - <presentacion>
        <frase id="0">Hello my friend! My name is Shamael and
        we are going to learn some English together</frase>
        - <interaccion id="1">
            <text>What is your name?</text>
        </interaccion>
    </presentacion>
    - <nivel_1>
        <frase id="0">Ok. We are in level 1. Let's start!</frase>
        <frase id="1">We are studying the time</frase>
        <grafica id="2">.\imagenes\horas.jpg</grafica>
        - <interaccion id="3">
            <text>¿Ya has aprendido las horas?Pulsa una tecla para
            seguir...</text>
        </interaccion>
        <frase id="4">And now we are studying subjects at
        school</frase>
        <grafica id="5">.\imagenes\asignaturas.jpg</grafica>
        - <interaccion id="6">
            <text>¿Comenzamos con unas preguntas?</text>
        </interaccion>
        - <pregunta id="7">
            <cod_profesor>0</cod_profesor>
            <enunciado>What time is it?</enunciado>
            <imagen>.\imagenes\reloj2.jpg</imagen>
            <respuesta>It's twenty past eight</respuesta>
            <acertada>false</acertada>
            <video>.\videos\Nivel_1Pregunta7.mpg</video>
        </pregunta>
        - <pregunta id="13">
            <cod_profesor>0</cod_profesor>
            <enunciado>Listen and say the schoolbag</enunciado>
            <imagen>.\imagenes\mochilas.jpg</imagen>
            <sonido>.\sonidos\mochilas4.mp3</sonido>
            <respuesta>1</respuesta>
            <acertada>false</acertada>
```

```
            <video>.\videos\Nivel_1Pregunta13.mpg</video>
        </pregunta>
    - <interaccion id="14">
            <text>Lo estas haciendo muy bien. Continuamos.</text>
    </interaccion>
    - <informacion id="15">
            <text>¿Te gusta que las preguntas tengan imagenes y
            sonido?</text>
    </informacion>
</nivel_1>
- <nivel_2>
    <frase id="0">Well, We are now in level two. Very good!Let's
    continue</frase>
    <frase id="1">Now we are going to learn the countries</frase>
    <grafica id="2">.\imagenes\paises.jpg</grafica>
    - <interaccion id="3">
            <text>Let's continue</text>
    </interaccion>
    <grafica id="4">.\imagenes\from.jpg</grafica>
    - <pregunta id="6">
            <cod_profesor>0</cod_profesor>
            <enunciado>Where is he from?</enunciado>
            <imagen>.\imagenes\usa.jpg</imagen>
            <respuesta>He is from USA</respuesta>
            <acertada>false</acertada>
            <video>.\videos\Nivel_2Pregunta6.mpg</video>
    </pregunta>
    - <informacion id="10">
            <text>¿Te parecen muy dificiles las preguntas?</text>
    </informacion>
    <frase id="16">Look and read</frase>
    <grafica id="17">.\imagenes\ing.jpg</grafica>
    - <interaccion id="18">
            <text>¿Lo has aprendido?Vamos a contestar unas
            preguntas</text>
    </interaccion>
    - <pregunta id="19">
            <cod_profesor>0</cod_profesor>
            <enunciado>What does she like?</enunciado>
            <imagen>.\imagenes\tenis.jpg</imagen>
            <respuesta>She likes playing tennis</respuesta>
            <acertada>false</acertada>
            <video>.\videos\Nivel_2Pregunta17.mpg</video>
    </pregunta>
</nivel_2>
- <nivel_3>
    <frase id="0">You are doing very well!We are now in level
    three.This is the last level.</frase>
    <frase id="1">We continue learning some places in a city</frase>
```

```
            <grafica id="2">.\imagenes\ciudad.jpg</grafica>
            - <interaccion id="3">
                  <text>Press enter to continue</text>
            </interaccion>
            - <pregunta id="4">
                  <cod_profesor>0</cod_profesor>
                  <enunciado>Look at the picture. Is there a cinema?
                  </enunciado>
                  <imagen>.\imagenes\there.jpg</imagen>
                  <respuesta>Yes there is</respuesta>
                  <acertada>false</acertada>
                  <video>.\videos\Nivel_3Pregunta4.mpg</video>
            </pregunta>
                  <text>¿Estas aprendiendo con Shamael?</text>
            </informacion>
            - <pregunta id="15">
                  <cod_profesor>0</cod_profesor>
                  <enunciado>Say true or false. Shamael is in the
                  cinema</enunciado>
                  <sonido>.\sonidos\parque.mp3</sonido>
                  <respuesta>false</respuesta>
                  <acertada>false</acertada>
                  <video>.\videos\Nivel_3Pregunta10.mpg</video>
            </pregunta>
            - <pregunta id="21">
                  <cod_profesor>0</cod_profesor>
                  <enunciado>Listen and say the town</enunciado>
                  <imagen>.\imagenes\towns.jpg</imagen>
                  <sonido>.\sonidos\greenTown1.mp3</sonido>
                  <respuesta>Green Town</respuesta>
                  <acertada>false</acertada>
                  <video>.\videos\Nivel_3Pregunta14.mpg</video>
            </pregunta>
      </nivel_3>
      - <alumno>
            <usuario>Lau</usuario>
            <nivel>Nivel_2</enunciado>
      </alumno>
</temario>
```

<?xml version="1.0" encoding="UTF-8" standalone="no" ?> : Nodo Document que representa al documento XML en sí. Es el objeto principal del árbol.

- <temario>: Es el nodo raíz del documento, sólo puede existir este. Dentro de él estará toda la información del XML.

- <presentacion>: Primer nodo hijo de <temario>. Dentro de él aparece el saludo y la presentación que hará Shamael la primera vez que se acceda a la aplicación.

- <nivel_1>: Segundo nodo hijo de <temario>. Dentro de esta etiqueta estarán guardadas las explicaciones y las preguntas correspondientes al primer nivel.
- <nivel_2>: Tercer nodo hijo de la raíz <temario>. Dentro de esta etiqueta estarán almacenadas las explicaciones y las preguntas correspondientes al nivel dos.
- <nivel_3>: Es el cuarto nodo hijo de <temario>. Dentro de esta etiqueta estarán guardadas las explicaciones y las preguntas pertenecientes al nivel tres.
- <alumno>: Este nodo no forma parte del XML desde el principio, se añadirá una vez el alumno haya introducido su nombre. Una vez suceda esto, esta etiqueta será el quinto hijo de la raíz <temario>. Dentro de esta etiqueta se almacenará el nombre del usuario y el nivel del curso en el que se encuentra. Este nodo se irá modificando según el alumno vaya superando los niveles, actualizando este campo. Esta información será útil a la hora de que un alumno vuelva a acceder al agente y pueda continuar por el nivel en el que lo dejó y no tener que volver a repetir preguntas de niveles ya superados. La estructura de este nodo es:

```
- <alumno>
        <usuario>Nombre introducido por el alumno</usuario>
        <nivel>Nivel en el que se encuentra el alumno</enunciado>
</alumno>
```

Dentro de las etiquetas <nivel_1>, <nivel_2> y <nivel_3> se podrán encontrar las siguientes etiquetas indistintamente, todas ellas tienen un atributo común que es el "id". Este atributo identifica individualmente a cada etiqueta y no se podrá repetir dentro de cada uno de los nodos hijos de la raíz:

<frase id="**0**"> Dentro de esta etiqueta estará guardado un texto que se mostrará por la pantalla de la interfaz sin esperar ninguna interacción por parte del usuario.

<interaccion id="**1**"> Dentro de esta etiqueta habrá otra etiqueta <text> que incluirá un texto, puede ser una pregunta o una frase, a imprimir por la pantalla de la interfaz. Con esta etiqueta se indica que si se espera una respuesta a introducir por el usuario pero no importará el contenido de ésta. Esta es su estructura:

```
-<interaccion id="1">
        <text>Texto a imprimir en la interfaz</text>
</interaccion>
```

<grafica id="**2**"> Dentro de esta etiqueta se encuentra el directorio donde se aloja una imagen que se quiere mostrar en la pantalla de la interfaz.

- <informacion id="**14**"> Esta etiqueta incluye otra etiqueta <text> que incluirá una pregunta que aparecerá en la interfaz pero que no estará relacionada con el proceso de aprendizaje. En esta etiqueta formularemos preguntas que serán útiles a la hora de realizar mejoras y estudios acerca de la aceptación del sistema. Se podrán preguntar al usuario si le gustan los vídeos que se muestran, el nivel de dificultad de las preguntas, si la aplicación es útil para que aprendas, etc. El usuario introducirá sus respuestas personales y luego éstas podrán ser evaluadas al visualizar el log. Esta es su estructura:

```
- <informacion id="14">
        <text>Pregunta a realizar acerca del diseño de la aplicación</text>
</informacion>
```

- <pregunta id="**12**"> Esta es la etiqueta donde se formula la pregunta al usuario y se espera su respuesta, que en este caso si tiene valor puesto que se compara con la respuesta de la pregunta para saber si ha sido acertada o no. Este campo tiene otras etiquetas en su interior, son las siguientes:

<cod_profesor> Entre estas etiquetas se guardará el código del profesor que haya introducido en el agente la pregunta, puesto que solamente él mismo y el administrador podrán modificar y eliminar dicha pregunta.

<enunciado> En esta etiqueta se añadirá el enunciado de la pregunta que se formula al usuario.

<imagen> Esta etiqueta indica que se quiere mostrar una imagen dentro de la pregunta que se está formulando. El texto que almacena es el directorio donde está la imagen. Esta etiqueta no estará siempre, únicamente cuando se quiera mostrar una imagen en una pregunta.

<sonido> Esta etiqueta indica que se reproduce un archivo de audio en la pregunta. Al igual que en la etiqueta <imagen>, esta etiqueta no estará siempre y su contenido será el directorio del sonido a reproducir.

<respuesta> El texto que contiene esta etiqueta es la respuesta a la pregunta que se formuló al alumno, que se comparará con la que introduzco el usuario para saber si la ha contestado correctamente o no.

<acertada> Esta etiqueta contiene el texto False predeterminado. Una vez el alumno haya contestado correctamente a la pregunta se cambiará por True. La función de este campo es que una vez que el alumno haya acertado una pregunta no la tenga que volver a contestar en el caso de que tenga que repetir el nivel en el que se encuentra.

<video> El contenido de esta etiqueta es el directorio del vídeo que se utilizará en esta pregunta para el agente.

La estructura completa de esta etiqueta es la siguiente:

```
- <pregunta id="12">
        <cod_profesor>Código del profesor</cod_profesor>
        <enunciado>Enunciado de la pregunta</enunciado>
        <imagen>Directorio de la imagen a mostrar (Puede existir o
        no)imagen>
        <sonido>. Directorio del sonido a reproducir (Puede existir o
        no)</sonido>
        <respuesta>Respuesta de la pregunta</respuesta>
        <acertada>Pregunta ya acertada o no</acertada>
        <video>Directorio del video a reproducir en el agente</video>
</pregunta>
```

- **Log.xml:** Este archivo recoge toda la información que se ha ido generando en la conversación, tanto la que ya estaba almacenada en el fichero temario.xml como las contestaciones del alumno, a parte de algunos datos más informativos. Cada vez que el alumno termina una sesión se enviará por correo electrónico a una dirección predeterminada.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
- <log>
   - <conversacion>
      - <Nivel_1>
          <frase id="0">Hello my friend! My name is Shamael and we are going
          to learn some English together</frase>
          <interaccion id="1">What is your name?</interaccion>
          <usuario>patry</usuario>
          <frase id="0">Ok. We are in level 1. Let's start!</frase>
          <frase id="1">We are studying the time</frase>
          <grafica id="2">.\imagenes\horas.jpg</grafica>
          <interaccion id="3">¿Ya has aprendido las horas?Pulsa una tecla
          para seguir...</interaccion>
          <usuario />
          <frase id="4">And now we are studying subjects at school</frase>
          <grafica id="5">.\imagenes\asignaturas.jpg</grafica>
          - <pregunta id="7">
                <cod_profesor>0</cod_profesor>
                <enunciado>What time is it?</enunciado>
                <imagen>.\imagenes\reloj2.jpg</imagen>
          </pregunta>
          <usuario>it's twenty past eight</usuario>
          <Shamael>Let's Continue</Shamael>
          <respuesta>It's twenty past eight</respuesta>
          <usuario />
          - <pregunta id="13">
```

```
                <cod_profesor>0</cod_profesor>
                <enunciado>Listen and say the correct schoolbag</enunciado>
                <imagen>.\imagenes\mochilas.jpg</imagen>
                <sonido>.\sonidos\mochilas1.mp3</sonido>
            </pregunta>
            <usuario>4</usuario>
            <Shamael>Let's Continue</Shamael>
            <respuesta>4</respuesta>
            <grafica id="26">.\imagenes\have_got.jpg</grafica>
            - <pregunta id="40">
                <cod_profesor>0</cod_profesor>
                <enunciado>Read and anwser true or false. His favourite
                subject is Music</enunciado>
                <imagen>.\imagenes\resumen.jpg</imagen>
            </pregunta>
            <usuario>false</usuario>
            <Shamael>Let's Continue</Shamael>
            <respuesta>false</respuesta>
        </Nivel_1>
        <Num_Preguntas>14</Num_Preguntas>
        <Num_Respuestas>14</Num_Respuestas>
        <Num_Errores>0</Num_Errores>
    </conversacion>
- <conversacion>
    - <Nivel_2>
            <frase id="0">Well, We are now in level two. Very good!Let's
                continue</frase>
            <frase id="1">Now we are going to learn the countries</frase>
            <grafica id="2">.\imagenes\paises.jpg</grafica>
            <interaccion id="3">Let's continue</interaccion>
            <usuario />
            <grafica id="4">.\imagenes\from.jpg</grafica>
            <interaccion id="5">Vamos a contestar unas preguntas</interaccion>
            <usuario />
            - <pregunta id="6">
                <cod_profesor>0</cod_profesor>
                <enunciado>Where is he from?</enunciado>
                <imagen>.\imagenes\usa.jpg</imagen>
            </pregunta>
            <usuario>he is from united states</usuario>
            <Shamael>Upss do it better the next time</Shamael>
            <respuesta>He is from USA</respuesta>
            <interaccion id="7">Vamos a por la siguiente pregunta</interaccion>
            <usuario />
            <informacion id="19">¿Te gustan los vídeos de
            Shamael?</informacion>
            <usuario>si, son divertidos y el muñeco es muy bonito</usuario>
            <frase id="20">Look and read</frase>
            <grafica id="21">.\imagenes\ing.jpg</grafica>
            <interaccion id="22">¿Lo has aprendido?Vamos a contestar unas
            preguntas</interaccion>
            <usuario />
            - <pregunta id="23">
```

```
            <cod_profesor>0</cod_profesor>
            <enunciado>What does she like?</enunciado>
            <imagen>.\imagenes\tenis.jpg</imagen>
            </pregunta>
            <usuario>she like tennis</usuario>
            <Shamael>Try it again</Shamael>
            <respuesta>She likes playing tennis</respuesta>
            <informacion id="29">¿Que es lo que más te gusta de
            Shamael?</informacion>
            <usuario>me gusta que parece un animal divertido</usuario>
  </Nivel_2>
  <Num_Preguntas>13</Num_Preguntas>
  <Num_Respuestas>10</Num_Respuestas>
  <Num_Errores>3</Num_Errores>
  </conversacion>
- <conversacion>
- <Nivel_2>
      - <pregunta id="6">
            <cod_profesor>0</cod_profesor>
            <enunciado>Where is he from?</enunciado>
            <imagen>.\imagenes\usa.jpg</imagen>
      </pregunta>
      <usuario>he is from usa</usuario>
      <Shamael>Well Done!!</Shamael>
      <respuesta>He is from USA</respuesta>
  </Nivel_2>
- <alumno>
      <nombre>patry</nombre>
      <nivel>Nivel_2</nivel>
  </alumno>
  <Num_Preguntas>3</Num_Preguntas>
  <Num_Respuestas>3</Num_Respuestas>
  <Num_Errores>0</Num_Errores>
</conversacion>
- <conversacion>
      - <Nivel_3>
            <frase id="0">You are doing very well!We are now in level
            three.This is the last level.</frase>
            <frase id="1">We continue learning some places in a city</frase>
            <grafica id="2">.\imagenes\ciudad.jpg</grafica>
            - <pregunta id="4">
                  <cod_profesor>0</cod_profesor>
                  <enunciado>Look at the picture. Is there a
                  cinema?</enunciado>
                  <imagen>.\imagenes\there.jpg</imagen>
            </pregunta>
            <usuario>yes</usuario>
            <Shamael>Try it again</Shamael>
            <respuesta>Yes there is</respuesta>
            <informacion id="12">¿Estas aprendiendo con
            Shamael?</informacion>
            <usuario>si, estoy recordando cosas que habia olvidado</usuario>
            - <pregunta id="14">
```

```
                <cod_profesor>0</cod_profesor>
                <enunciado>Say true or false. Shamael is in the
                cinema</enunciado>
                <sonido>.\sonidos\parque.mp3</sonido>
            </pregunta>
            <usuario>false</usuario>
            <Shamael>Well Done!!</Shamael>
            <respuesta>false</respuesta>
            - <pregunta id="22">
                <cod_profesor>0</cod_profesor>
                <enunciado>Listen and say the town</enunciado>
                <imagen>.\imagenes\towns.jpg</imagen>
                <sonido>.\sonidos\greenTown1.mp3</sonido>
            </pregunta>
            <usuario>green town</usuario>
            <Shamael>Let's Continue</Shamael>
            <respuesta>Green Town</respuesta>
        </Nivel_3>
        <Num_Preguntas>11</Num_Preguntas>
        <Num_Respuestas>9</Num_Respuestas>
        <Num_Errores>2</Num_Errores>
    </conversacion>
- <conversacion>
</log>
```

Como se puede observar el log recoge toda la información que a parece en el fichero temario, y algunas etiquetas nuevas que se detallan a continuación:

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>

- <log>: Es la raíz del documento.

- <conversacion>: Esta etiqueta representa los hijos que pueda tener la raíz. Cada vez que el alumno comience una sesión en el agente se creará una nueva etiqueta de este tipo y dentro se almacenará la información generada en el intercambio de preguntas respuestas. Al pasar de un nivel a otro se creará otra nueva etiqueta <conversacion>.

<Shamael>: En esta etiqueta se almacena el texto que muestra Shamael tras una pregunta acertada o fallada. Son frases de felicitación o ánimo que se han generado aleatoriamente y que llevan asociadas un vídeo.

<Num_Preguntas>: Esta etiqueta almacena el número de preguntas que ha realizado Shamael en un determinado nivel. Este número no será igual si se repite el nivel ya que cuando esto sucede solamente se formulan las preguntas que el alumno había fallado previamente.

<Num_Respuestas>: Esta etiqueta indica el número de preguntas que ha acertado el alumno en un determinado nivel.

<Num_Errores>: En esta etiqueta se registra el número de preguntas que ha fallado un alumno en el nivel que se encuentra.

Estas tres últimas etiquetas servirán para poder observar el aprendizaje del alumno, si es capaz de pasar un nivel a la primera o en caso contrario observar su progreso, si aprende de las respuestas falladas y las acierta después.

<usuario>: Esta etiqueta contiene el texto introducido por el usuario. La información que contiene puede tener valor según la etiqueta de la que vaya precedida.

Cuando la etiqueta que la precede es <interaccion id="**1**"> el texto introducido no tiene ningún valor ni para el desarrollo de la aplicación ni para el estudio del log. El único texto que tiene valor es cuando al inicio de la ejecución se pregunta al alumno por su nombre, que ese dato se guardará en el fichero temario.xml. Aparece de la siguiente manera:

<interaccion id="**1**">**Frase o pregunta que propone el agente**</interaccion>
<usuario>: **Contestación del usuario**</usuario>

Cuando la etiqueta que va delante es <informacion id="**?**"> la contestación introducida por el alumno no tendrá valor para el agente pero si será de gran valor a la hora de hacer el estudio del log ya que estás respuestas corresponden a preguntas sobre la aceptación de la aplicación, si le está gustando al usuario, si le gustan los vídeos, etc. Su estructura sería:

<informacion id="**21**">**Pregunta acerca del Agente**</informacion>
<usuario>**Opinión dada por el usuario**</usuario>

En el último caso de que la etiqueta que vaya delante sea -<pregunta id="**24**"> el valor de la etiqueta <usuario> sí que tendrá un valor importante para el desarrollo de la aplicación, ya que será la respuesta introducida por el usuario a una pregunta de teoría. Esta contestación se comparará con la respuesta correcta a la pregunta y en el caso que sea igual se generará una frase de felicitación o en caso contrario una frase de ánimo (texto guardado en <Shamael>) y un vídeo asociado a dicha frase. En el caso de que la respuesta correcta se cambiará el estado de la pregunta para que quede reflejada como pregunta ya contestada.

- **Usuarios.xml:**

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
- <usuarios>
    - <administrador>
        <cod_profesor>admin</cod_profesor>
        <usuario>admin</usuario>
        <contraseña>admin</contraseña>
    </administrador>
```

```
- <profesor>
      <cod_profesor>0</cod_profesor>
      <usuario>bea</usuario>
      <contraseña>12345</contraseña>
  </profesor>
  - <profesor>
      <cod_profesor>1</cod_profesor>
      <usuario>lau</usuario>
      <contraseña>lau1987</contraseña>
  </profesor>
  - <alumno>
      <usuario>alumno</usuario>
      <contraseña>alumno</contraseña>
  </alumno>
</usuarios>
```

<?xml version="1.0" encoding="UTF-8" standalone="no" ?> : Nodo Document que representa al documento XML. Es el objeto principal del árbol.

- <usuarios>: Es el nodo raíz del documento, sólo puede existir este. Dentro de él estará toda la información del XML.

- <administrador> Primer hijo de la raíz <usuarios>. Esta etiqueta aparecerá una sola vez. Contiene los datos de acceso del administrador al agente (usuario y contraseña). El administrador es el único que puede cambiar estos datos en la aplicación. Esta es su estructura:

```
- <administrador>
      <cod_profesor>admin</cod_profesor>
      <usuario>admin</usuario>
      <contraseña>admin</contraseña>
  </administrador>
```

- <profesor> Segundo hijo de la raíz <usuarios>. Esta etiqueta aparecerá tantas veces como profesores estén autorizados a utilizar el agente. El administrador es el único que puede añadir nuevos miembros a este archivo y para eliminarlos deberá ser el administrador o el propio usuario que quiera eliminar su acceso al agente. La etiqueta <cod_profesor> contendrá el código que tiene asignado cada profesor y que le distingue del resto puesto que no pueden existir dos usuarios profesores con el mismo código. Este código será útil a la hora de modificar y eliminar preguntas ya que cada pregunta nueva añadida se introduce con el código del profesor que la creó para que sólo él y el administrador puedan actuar sobre ella.

La estructura del nodo profesor es la siguiente:

```
- <profesor>
        <cod_profesor>Codigo Profesor</cod_profesor>
        <usuario>Usuario de acceso al sistema</usuario>
        <contraseña>Contraseña de acceso al sistema</contraseña>
 </profesor>
```

- <alumno> Último hijo de la raíz <usuarios>. Esta etiqueta aparecerá una sola vez. Contiene los datos de acceso del alumno al agente (usuario y contraseña). El administrador es el único que puede cambiar estos datos en la aplicación. Esta es su estructura:

```
- <alumno>
        <usuario>Usuario de acceso del alumno</usuario>
        <contraseña>Contraseña de acceso del alumno</contraseña>
 </alumno>
```

3.5. Descripción de clases

En este apartado se describen las clases principales de la aplicación, con sus atributos y métodos fundamentales.

ChatBox.java:

Esta clase pertenece al paquete Shamael_Imagenes y es la que maneja toda la aplicación del agente, los turnos de preguntas-respuestas, verifica las respuestas del usuario, dirige la reproducción de los vídeos, etc.

- **Atributos:**

  private Question: Tipo de etiqueta del fichero Temario.xml

  private Interaction: Tipo de etiqueta del fichero Temario.xml

  private Information: Tipo de etiqueta del fichero Temario.xml

  private Statement: Tipo de etiqueta del fichero Temario.xml

  private Student: Tipo de etiqueta del fichero Temario.xml

  private Graphic: Tipo de etiqueta del fichero Temario.xml

  private FileInputStream: fichero Log

  private boolean goon: variable que controla el turno de intervención

- **Constructor:**

  public ChatBox(Conversation conversacion, String nivel) : Carga la información del fichero correspondiente al nivel en el que está el alumno. Comprueba si existe el fichero log, en caso contrario lo crea.

- **Métodos:**
  - *public void run(boolean activar):* Este método se recorre el array y va identificando el tipo de etiqueta y que debe hacer con cada una. Imprime simultáneamente la información en el log y la manda imprimir por la interfaz. Controla también la reproducción de los videos correspondientes a las preguntas.
  - *private String aleatoriaAcierto():* Este método elige aleatoriamente entre cuatro frases para mostrar cuando el alumno haya acertado la pregunta. También regula el vídeo que se reproducirá junto con la frase.
  - *private String aleatoriaFallo():* Este método elige aleatoriamente entre tres frases para mostrar cuando el alumno haya fallado la pregunta. También elegirá el vídeo que se reproducirá para este caso.
  - *private void reproducir(String directorio):* A este método se le pasa la ruta del vídeo del agente a reproducir y lo reproduce.
  - *protected void logprint(Node nodo):* Este método va añadiendo los nodos con la información perteneciente a cada etiqueta dentro de esta conversación.
  - *private void close():* Este método añade la conversación generada al nodo log.
  - *public void print(final String str):* Este método imprime las preguntas y la información perteneciente al fichero temario.xml por la pantalla de la interfaz.
  - *public void printUser(String str):* Este método muestra en la pantalla de la interfaz lo que el usuario ha introducido en la zona de texto (respuestas a las preguntas).
  - *private void userEntryActionPerformed(java.awt.event.ActionEvent evt):* En este método se manejan las respuestas del usuario dependiendo del tipo de etiqueta que haya generado esa respuesta (interacción, pregunta o información) y se manda almacenar en el log e imprimir por pantalla. En el caso de ser una respuesta a una pregunta la comparará con la respuesta correcta y llamará a aleatoriaAcierto o aleatoriaFallo dependiendo del resultado de la comparación. Irá guardando en un contador el número de respuestas acertadas y cuando se han contestado todas las preguntas, si el contador es igual al número de preguntas (el alumno ha acertado todas las preguntas) actualizará el nivel del alumno para que pueda pasar al siguiente nivel. También llamará al método ModificarAcertadas del paquete ModificarXMLs para cambiar el valor de la pregunta acertada y que no se vuelva a repetir en caso de que el alumno no supere el nivel y lo tenga que repetir. Al acabar las preguntas, tanto si ha superado el nivel como si no, se le da al alumno la opción de continuar con la sesión. En el caso de que no quiera se enviará el log generado por e-mail.

ModificarXMLs.java:

Esta clase pertenece al paquete Shamael_Imagenes y todos sus métodos utilizan el archivo temario.xml para modificarlo o reestructurarlo.

- **Atributos:**

  Esta clase no dispone de atributos.

- **Constructor:**

  public ModificarXMLs(): El constructor de este método es un constructor vacío. No tiene parámetros.

- **Métodos:**

  - *private void nuevoAlumno(InputStream is, String UserEntry, String nivelUser):* En este método se crea un usuario con el nombre del alumno y el nivel en el que se encuentra y se inserta en el fichero temario.xml
  - *private void modificarQuestion (InputStream is, String level, String id):* Cuando el alumno contesta correctamente una pregunta este método se encarga de cambiar el valor de su etiqueta <acertada> para que en el caso que repita el curso esta pregunta ya no la tenga que volver a contestar. También elimina la interacción que hay detrás de cada pregunta ya que si no la siguiente vez que se ejecute ese nivel no aparecería la respuesta acertada pero si la interacción para continuar después.
  - *private void actualizarAlumno(String temario, String nivel, String nuevoNivel):* Cuando el alumno ha contestado correctamente todas las preguntas, pasa de nivel y se modifica en el fichero temario.xml poniendo el nivel nuevo.
  - *private void reestructurarAcertadas(String temario):* Una vez superado el curso se modifica la etiqueta <acertada> de todas las preguntas para que queden como no contestadas. También llama al método obtenerNodoInteracción para que añada una interacción detrás de cada pregunta.
  - *public void reEscribirIds(Document Doc):* Este método rescribe los ids de cada etiqueta después de haber tenido que borrar las interacciones de las preguntas en el método ModificarQuestion o al añadirlas en reestructurarAcertadas.
  - *public Node obtenerNodoInteraccion(Document Doc, String Id):* Este método genera el nodo del tipo Interaction con una frase como valor generado aleatoriamente. Esta interacción se colocará detrás de cada pregunta y servirá para dar paso a la siguiente.

AniadirPreguntas.java:

Esta clase formulario pertenece al paquete Usuarios y ofrece la posibilidad de añadir una pregunta nueva a un nivel al archivo Temario.xml. Se mostrará una ventana al usuario donde elegirá el nivel y podrá insertar el texto de la pregunta, el de la respuesta y también habrá la opción de elegir entre un vídeo para el avatar por defecto (mudo) o que el profesor pueda elegir el video que él desee. Estos tres campos serán necesarios para poder formar una pregunta, sería un mínimo obligatorio. El usuario también tendrá la opción de añadir a la pregunta un archivo de audio o también una imagen. Una vez tenga configurada la pregunta la clase ofrece la opción de visualizar como quedaría la pregunta en el agente antes de ser guardada.

- **Atributos:**

  private String pregunta: Este atributo toma como valor el texto que ha introducido el usuario como Enunciado de la pregunta.

  private String respuesta: La respuesta a la pregunta que el usuario introduce se asigna a este atributo.

  private String id: Identificador que se le asigna a la pregunta.

  private String imagen: Directorio donde se encuentra la imagen seleccionada para añadir a la pregunta.

  private String sonido: Directorio donde se encuentra el archivo de audio seleccionado para añadir a la pregunta.

  private String video: Directorio del video asociado a la pregunta.

  private String code: Este atributo almacena el código de profesor del usuario que crea la nueva pregunta.

- **Constructor:**

  public AniadirPreguntas(String código, String nombre): Este constructor contiene dos parámetros el nombre y el código de profesor del usuario que va a crear una nueva pregunta.

- **Métodos:**

  - *private void loadFromFile(InputStream is):* Este método parsea el archivo temario.xml y se sitúa en el nivel en el que el usuario desea añadir una nueva pregunta. Aquí crea un nuevo nodo pregunta con los campos pregunta, respuesta y vídeo como mínimo y con los elementos imagen y sonido en el caso de que el usuario haya decidido introducirlos. El nuevo nodo se colocará al final del nivel seleccionado

4. Pruebas

Las pruebas se han realizado en dos fases. En primer lugar se ha validado que la aplicación funciona correctamente y en segundo lugar se ha verificado que el sistema cumple con los requisitos que anteriormente se definieron. La estructura de las pruebas realizadas sigue un enfoque ascendente desde las pruebas unitarias de cada módulo del sistema hasta valorar si la aplicación cumple con los requisitos no funcionales y la comprobación del nivel de satisfacción alcanzado por los usuarios con o sin conocimientos informáticos.

4.1. Pruebas Unitarias

Con este tipo de pruebas se comprueba el correcto funcionamiento del algoritmo y de los distintos módulos de forma independiente. Para ello, se verifica que a partir de la pantalla de inicio todas las bifurcaciones y posibilidades en un módulo son correctas, esto son las pruebas de caja blanca. Y con las pruebas de caja negra se comprueba que los resultados de cada módulo coinciden con los resultados esperados.

4.1.1. Pruebas Caja Blanca

Para la realización de las pruebas de caja blanca se ha utilizado el algoritmo para acceder a la aplicación.

```
1.  Introducir Usuario y Contraseña
2.  SI existe_Usuario{
3.      SI Usuario = Admin{
4.          Mostrar Zona_Admin
5.      }SI NO{
6.          Nivel=ObtenerNivelAlumno
7.          SI nivel=Nivel_1{
8.              ParseXML(Nivel_1)
9.          }SI NO{
10.             SI nivel=Nivel_2{
11.                 ParseXML(Nivel_2)
12.             }SI NO {
13.                 ParseXML(Nivel_3)
14.             }
15.         }
16.         Mostrar Agente
17.     }
18. }SI NO{
19. No_Existe_Usuario
20. }
```

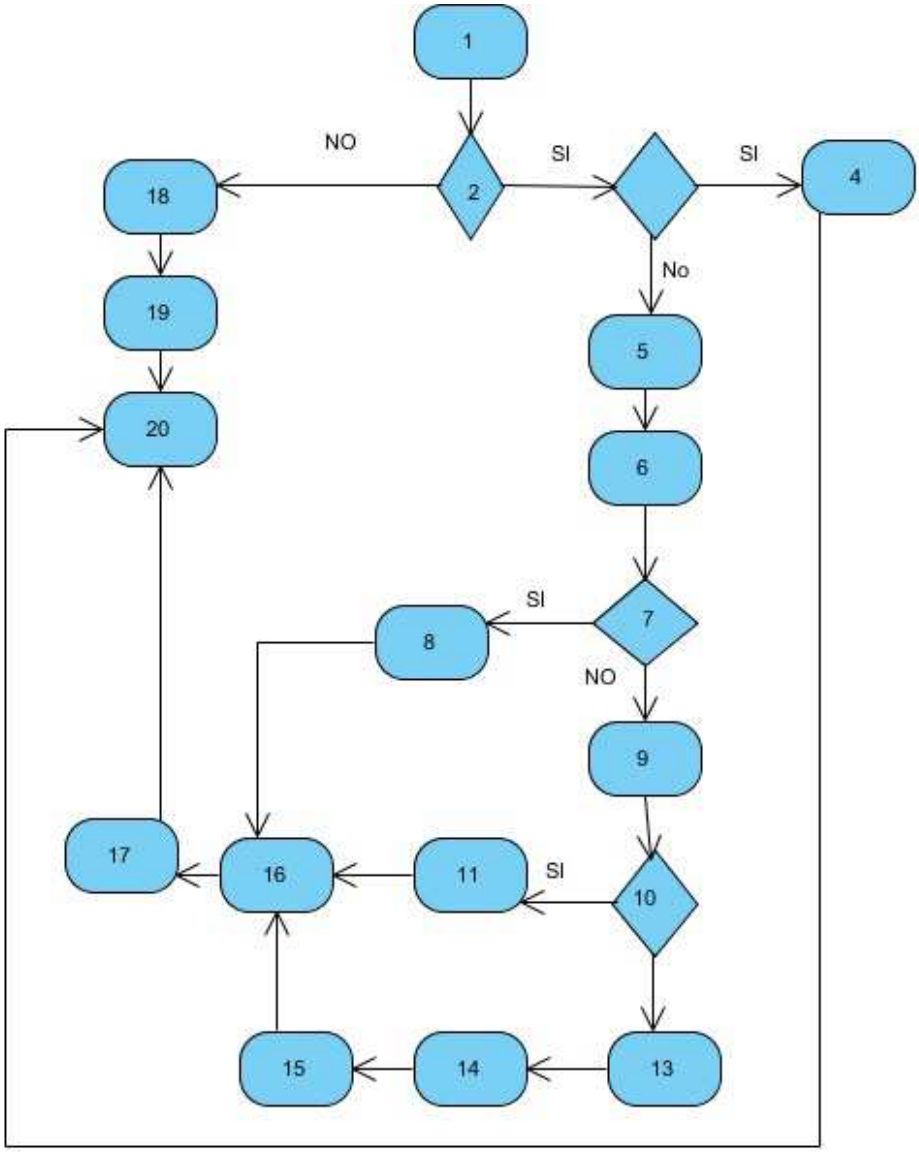


Imagen 29. Diagrama de Caja Blanca

Nº de caminos = Nº regiones = Aristas – Nodos + 2
Nº de caminos = 22 – 19 +2 = 5

- **Camino 1**: 1, 2, 3, 18, 19, 20
- **Camino 2**: 1, 2, 3, 4, 20
- **Camino 3**: 1, 2, 3, 5, 6, 7, 8, 16, 17, 20
- **Camino 4**: 1, 2, 3, 5, 6, 7, 9, 10, 11, 16, 17, 20
- **Camino 5**: 1, 2, 3, 5, 6, 7, 9, 10, 13, 14, 15, 16, 17, 20

CAMINO	existe_Usuario	Usuario=Admin	nivel=Nivel_1	nivel=Nivel_2
Camino 1	FALSO	FALSO	FALSO	FALSO
Camino 2	VERDADERO	VERDADERO	FALSO	FALSO
Camino 3	VERDADERO	FALSO	VERDADERO	FALSO
Camino 4	VERDADERO	FALSO	FALSO	VERDADERO
Camino 5	VERDADERO	FALSO	FALSO	FALSO

Tabla 13. Caminos Caja Blanca

4.1.2. Pruebas Caja Negra

Las cajas negras son un tipo de pruebas de software que se realizan desde el punto de vista de las entradas que recibe y las salidas que produce, sin tener en cuenta el funcionamiento interno. Por tanto, la caja negra analiza la forma de interactuar de la aplicación con el medio que le rodea. Se atiende de esta forma a QUE HACE sin importar COMO LO HACE, comprobando que la salida esperada coincide con la real.

- **Prueba 1:**

Clases Afectadas	Entrada	Descripción	Salida Esperada	Salida Real
Autentificacion.java	Al autentificarse se dejan los campos vacíos.	Autentificación del usuario para acceder al agente	Debía indicar que hay que introducir un usuario o contraseña	Aparece un mensaje que dice que se debe de introducir un usuario y contraseña

Tabla 14. Prueba Caja Negra

Resultado Correcto.

- **Prueba 2:**

Clases Afectadas	Entrada	Descripción	Salida Esperada	Salida Real
Autentificacion.java	Al autentificarse se introducen datos erróneos.	Autentificación del usuario para acceder al agente	Debía indicar que hay que los datos introducidos no corresponden a ningún usuario	Se muestra un cuadro dónde indica que el nombre de usuario y contraseña no son validos

Tabla 15. Prueba Caja Negra

Resultado Correcto.

- **Prueba 3:**

Clases Afectadas	Entrada	Descripción	Salida Esperada	Salida Real
AñadirPreguntas.java	Al añadir una pregunta no seleccionar cualquiera de las dos opciones de vídeo	Insertar una pregunta nueva en el fichero Temario.xml donde no aparecerá la ruta del vídeo	Se inserta la pregunta en el xml correctamente	Se inserta la pregunta en el XML con la etiqueta video vacía. Reconocerá que hay un vídeo a reproducir pero dará error porque no hay ruta especificada.

Tabla 16. Prueba Caja Negra

Resultado Incorrecto. Para solucionar este problema se puso como condición que para poder guardar una pregunta debía de estar marcada una de las dos opciones de video, o un nuevo video seleccionado por el profesor o un video mudo por defecto.

- **Prueba 4:**

Clases Afectadas	Entrada	Descripción	Salida Esperada	Salida Real
ChatBox.java Conversation.java	El usuario utiliza el agente.	El alumno vuelve a iniciar una sesión o continua repitiendo o avanzando un curso.	El agente continuará por donde se quedó o empezando un nuevo nivel	Aparecía Shamael volviéndose a presentar y preguntando al alumno su nombre de nuevo

Tabla 17. Prueba Caja Negra

Resultado Incorrecto. Para solucionar este problema se incluyeron las etiquetas referentes al saludo entre las etiquetas <presentación> para que sólo se leyeran la primera vez que se accede a la aplicación y el vídeo de presentación sólo se reproduce en esa ocasión.

- **Prueba 5:**

Clases Afectadas	Entrada	Descripción	Salida Esperada	Salida Real
EliminarProfesor.java	Se introduce el código de profesor correspondiente al admin para eliminarlo	Prueba realizada para comprobar que no se puede eliminar el administrador.	No se puede eliminar el administrador.	Aparece un mensaje de error. Impidiendo borrar ese usuario.

Tabla 18. Prueba Caja Negra

Resultado Correcto.

- **Prueba 6:**

Clases Afectadas	Entrada	Descripción	Salida Esperada	Salida Real
ChatBox.java ModificarXMLs.java	Durante la conversación, el alumno responde correctamente mezclando mayúsculas y minúsculas	Comprobar si Shamael diferencia entre mayúsculas y minúsculas	La respuesta es correcta	La respuesta es correcta

Tabla 19. Prueba Caja Negra

Resultado Correcto

- **Prueba 7:**

Clases Afectadas	Entrada	Descripción	Salida Esperada	Salida Real
ChatBox.java ModificarPreguntas.java EliminarPreguntas.java	Añadir una imagen, un sonido o un vídeo a una pregunta	Comprobar que el contenido multimedia se reproducía en el agente	La imagen, sonido y video se reproducen sin problema.	Al cambiar de ordenador las rutas introducidas para estos elementos ya no eran válidas y no se mostraban.

Tabla 20. Prueba Caja Negra

Resultado Incorrecto: Este problema ha sido solucionado utilizando rutas relativas. Así no habrá problemas con las rutas si se ejecuta el agente desde otro lugar.

4.2. Pruebas de Integración

En este apartado se comprueba que al interrelacionar todos los módulos, éstos funcionan correctamente.

Esta parte se ha ido desarrollando de manera sistemática, desde el inicio del proyecto comprobando que la unión entre los distintos módulos que componen la aplicación eran accesibles y mediante lenguaje natural, para luego comprobar que las acciones a realizar dentro de cada modulo son realizadas correctamente.

La unión entre módulos se ha comprobado siguiendo el esquema de la aplicación mostrado en el Diagrama de módulos de alto nivel (imagen 24)

4.3. Pruebas de Validación

En las pruebas de validación se revisa que la aplicación desarrollada cumple con las especificaciones, determinando si los requisitos iniciales se cumplen.

RF1.El usuario podrá autentificarse en el sistema introduciendo su usuario y contraseña.

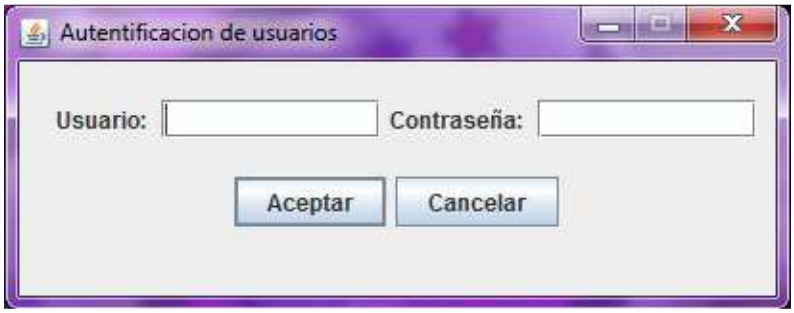


Imagen 30. Ventana Autentificación Usuario

RF2: El programa leerá las preguntas de un archivo XML y las mostrará al usuario alumno por la pantalla de la interfaz tipo messenger.

RF3: El sistema controlará los turnos de pregunta-respuesta con el alumno.

RF4: El sistema mostrará los vídeos del agente según los estados de la conversación.

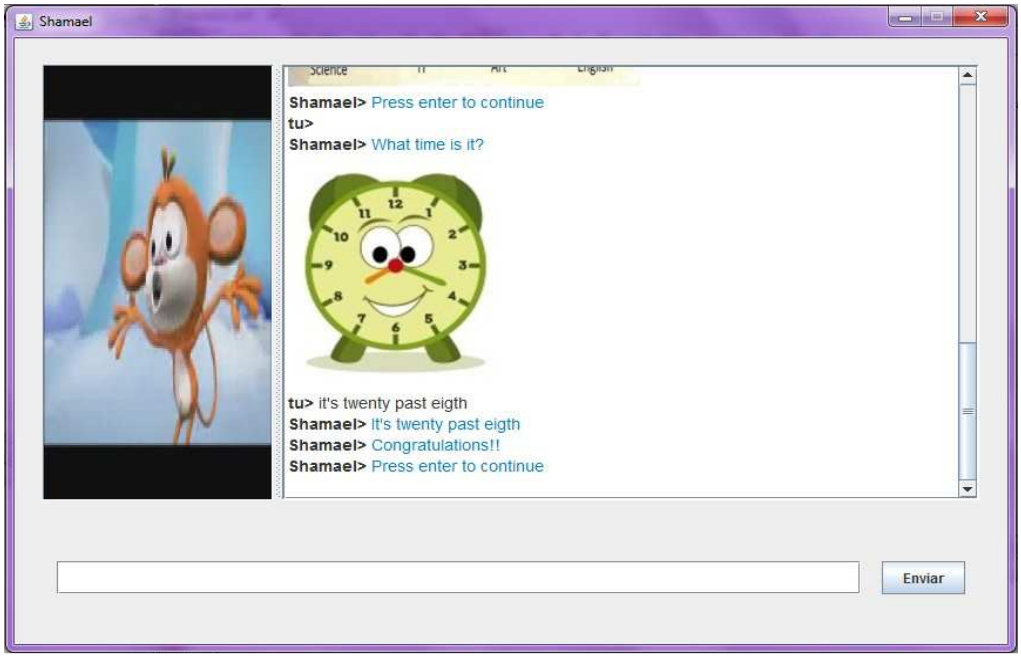


Imagen 31. Interfaz Shamael Preguntas-Respuestas

RF5: El sistema irá creando un log compuesto por la conversación agente-alumno.

RF6: El sistema enviará a un correo electrónico el log con la información de la sesión.

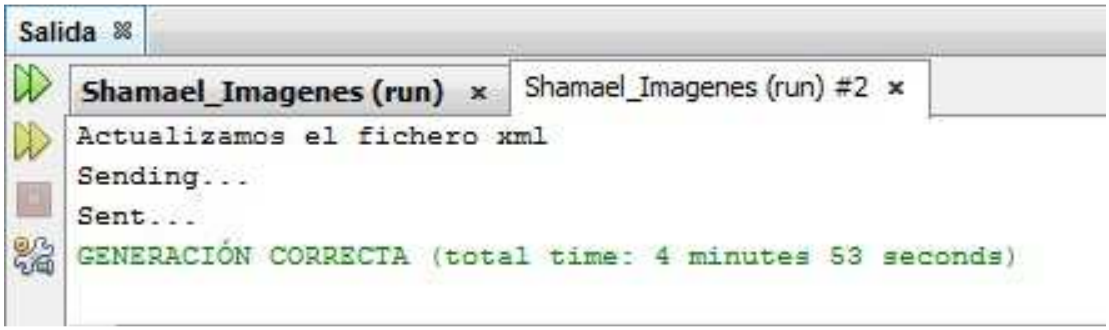


Imagen 32. Envío fichero log.xml - Comsola

RF7: El usuario administrador podrá añadir preguntas nuevas al agente que el sistema almacenará en el archivo XML.

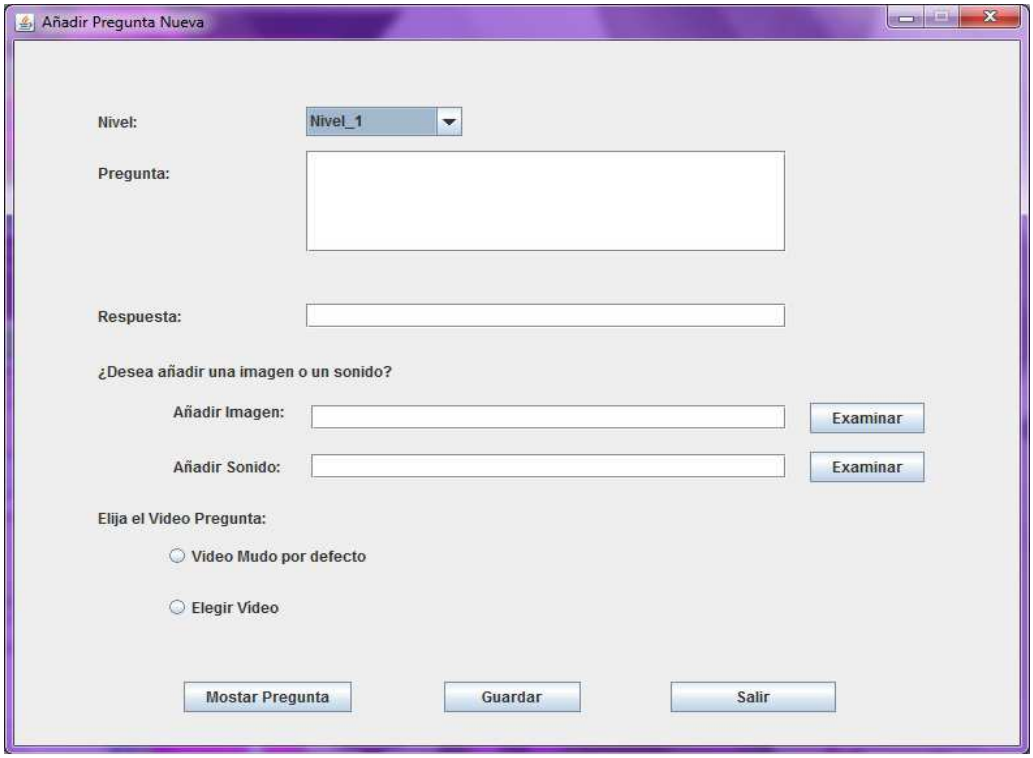


Imagen 33. Ventana Añadir Pregunta

RF8: El administrador podrá cargar las preguntas del fichero XML para visualizarlas y las podrá eliminar o modificar. El sistema registrará los cambios en el fichero XML.

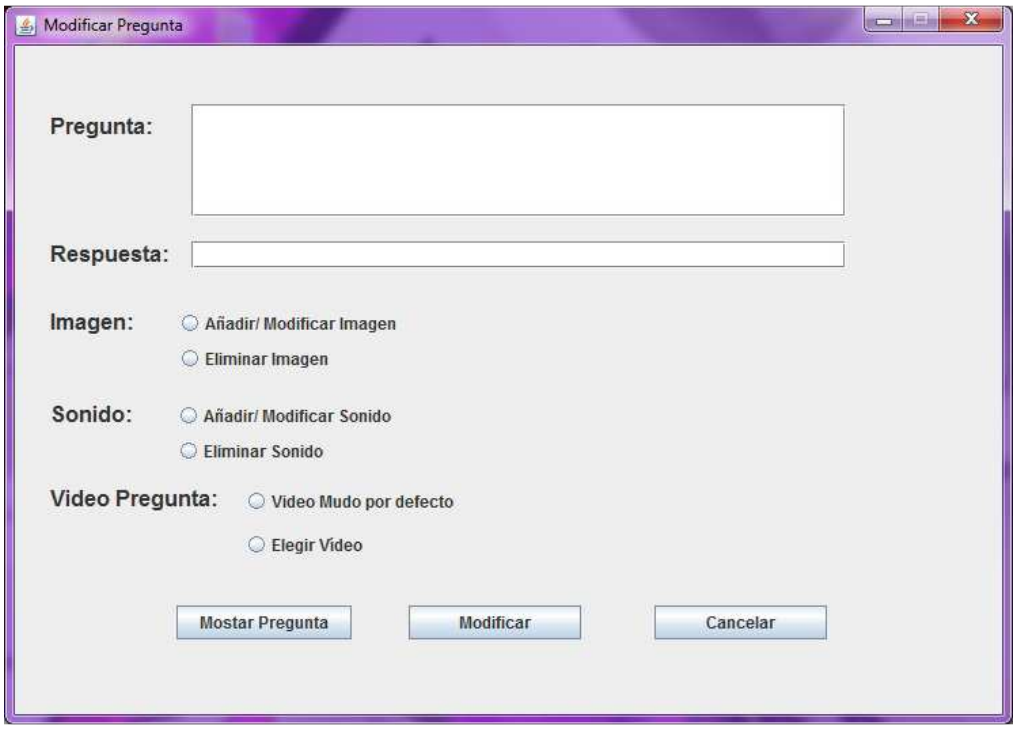


Imagen 34. Ventana Modificar Pregunta

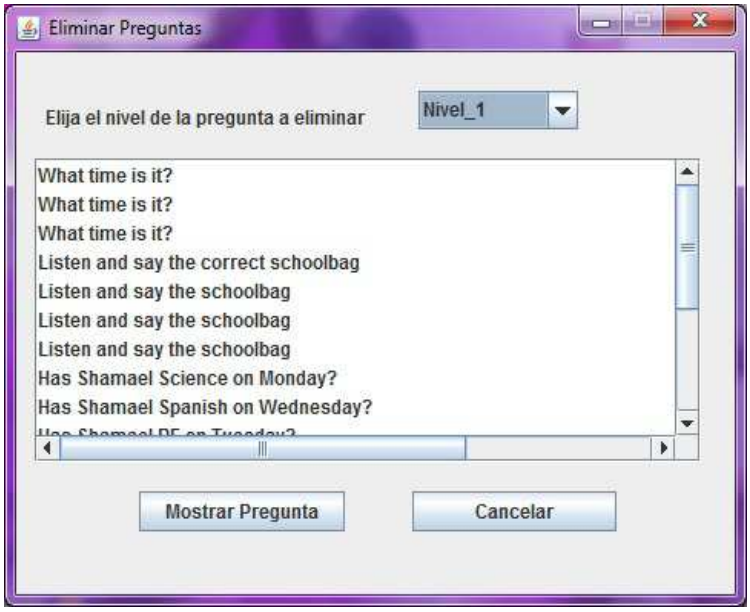


Imagen 35. Ventana Eliminar Pregunta

RF9: El administrador podrá añadir nuevos profesores para que puedan acceder a la aplicación. El sistema almacenará los nuevos datos en un fichero XML de usuarios.

Imagen 36. Ventana Registro Profesores

RF10: El administrador podrá eliminar a un profesor del fichero XML de usuarios.

RF11: El administrador podrá modificar los datos de cualquier profesor. El sistema realizará los cambios en fichero de usuarios.

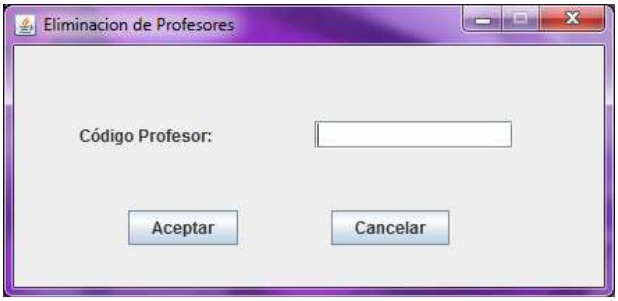


Imagen37. Ventana Eliminar Profesor

Imagen 38. Ventana Modificar Profesor

RF12: El administrador podrá modificar los datos de acceso del usuario alumno. El sistema lo modificará en el fichero XML de usuarios.

Imagen 39. Ventana Modificar Acceso Alumnos

4.4. Pruebas de Aceptación

Para la realización de estas pruebas se ha contado con la colaboración de 10 personas con conocimientos muy básicos de informática. El estudio se ha dividido en dos partes: la parte del Agente dirigida al alumno y la parte de la zona del Administrador dirigido a los profesores. Los datos reflejados se han obtenido del un cuestionario que se realizó posteriormente a los usuarios (Anexo A).

Parte 1 – Evaluación Agente Shamael

Los factores que se han evaluado en esta parte han sido:

- Aplicación intuitiva y usable:

  Todos los usuarios han comprendido desde el primer momento la dinámica del programa tras una breve introducción. La interacción pregunta-respuesta les ha parecido muy sencilla al asociarla con el "Messenger"
- Tiempo de Respuesta:

  El tiempo de respuesta ha sido el esperado. Las preguntas- respuestas llevaban el ritmo adecuado a la conversación. A pesar de tardar a veces los videos en cargar no superaban los dos segundos.
- Interfaz del programa:

  Sencilla no lleva lugar a equivocaciones. Zona del agente, zona del alumno y pantalla donde aparece la conversación.

- Inserción de vídeos en el avatar:

  Es lo que más le ha gustado a los usuarios. Les ha parecido original y divertido, sobre todo la parte de las felicitaciones al acertar una pregunta.

  No creen que el vídeo pueda distraer la atención del alumno ya que son videos cortos que no necesitan de demasiada atención y sin embargo dinamizan y entretienen y animan a continuar con el curso.
- Utilización de imágenes y sonidos en las preguntas:

  Es un aspecto que han valorado muy positivamente porque ayuda a desarrollar las capacidades visuales y auditivas y hacen más interactivas las preguntas que si solamente fuera pregunta respuesta.
- Lo que más ha gustado de Shamael:

  En general lo que más ha gustado del programa es la utilización de vídeos, imágenes y sonidos en las preguntas, hacen que el agente se convierta en un juego para el alumno y esté aprendiendo a la vez.

  Comentarios a destacar:

  > "es de fácil utilizar y, sobre todo, que al final de cada test te repitan sólo las preguntas que has fallado"
  >
  > "me parece un programa completo ya que hace que la persona que está realizando el test escuche, escriba, lea, se fije en las imágenes y preste atención a los audios"
- Lo que menos ha gustado y cambios a realizar

  La opinión general es que no ha gustado la poca flexibilidad que da el agente a la hora de evaluar las preguntas. Aunque al inicio de cada pregunta hay un ejemplo sobre como contestar la pregunta no tiene porque ser inválida otro tipo de respuesta que conteste lo mismo.

  Otro cambio que se propone realizar sería la introducción de preguntas tipo test con respuestas en las que sólo tengas que elegir la letra de la respuesta.

  El resto de comentarios sobre cosas que cambiarían están relacionados con la interfaz, posibilidad de maximizar, colores, etc.

Parte 2 – Evaluación Zona Administrador

- Aplicación intuitiva y usable:

  Tras una breve introducción e indagando se entiende perfectamente, los botones ayudan mucho.

- Tiempo de Respuesta:

  El tiempo de respuesta es el esperado no tarda cuando tiene que mostrar las preguntas ni al guardarlas.

- Interfaz del programa:

  Muy sencilla, llegando incluso a parecer un poco sobria, pero al ser ventanas como las de Windows resultan familiares y ayuda a moverte por la aplicación.

- Como herramienta para un profesor:

  Los usuarios lo han calificado como una herramienta que podría ser útil para un profesor ya que tiene un manejo total sobre las preguntas y resulta muy sencillo de utilizar. Podría realizar preguntas de repaso a los alumnos rápidamente.

- Lo que más ha gustado:

  Una vez comprendido la utilidad es fácil de manejar.

- Lo que menos ha gustado y cambios a realizar:

  El único comentario generalizado ha sido el incluir un botón de ayuda en el menú principal que te informe de cómo funciona la aplicación, ya que aunque con los botones es fácil ir de una ventana a otra sin una explicación inicial

Usuario	1	2	3	4	5	6	7	8	9	10
Edad	25	29	24	24	21	10	10	51	57	15
Valoración General	7	6	8	8	7	9	6	7	7	8

Tabla 21. Valoración Pruebas Usuarios

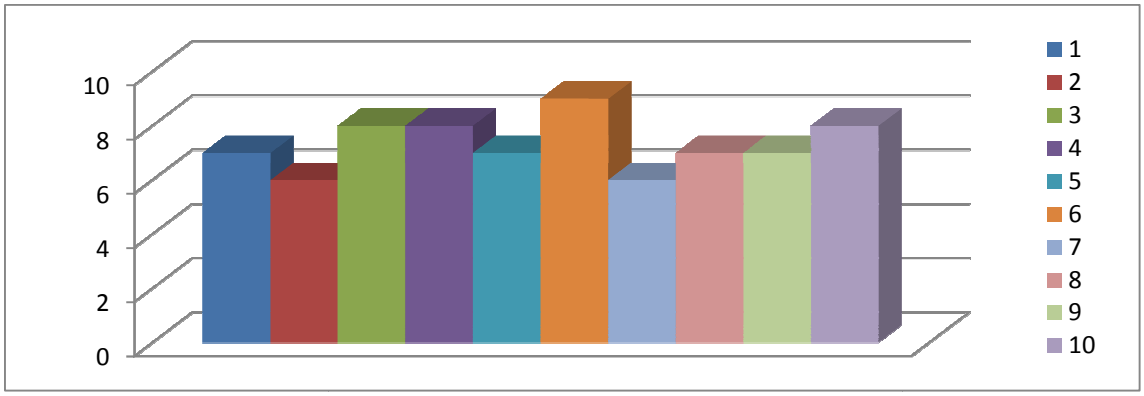


Imagen 40. Gráfica Satisfacción Usuarios

5. Conclusiones y Trabajos futuros

Los objetivos que se establecieron para la modificación del agente Shamael y hacer que incluyera una imagen dinámica que expresara emociones correspondientes a un esquema de diálogo basado en preguntas-respuestas se han logrado. Además se ha conseguido desarrollar una parte de administrador que ayuda a poder evolucionar al agente ya que puede gestionar las preguntas que presentará éste al alumno. El cumplimiento de los objetivos ha quedado reflejado en las pruebas de validación y por la valoración positiva de los usuarios que han probado el programa.

Se comenta a continuación cómo se han cumplido los requisitos establecidos:

- RF1: El usuario podrá autentificarse en el sistema introduciendo su usuario y contraseña.
- Esto se ha conseguido gracias al módulo de autentificación.

- RF2: El programa leerá las preguntas de un archivo XML y las mostrará al usuario alumno por la pantalla de la interfaz tipo messenger.
- RF3: El sistema controlará los turnos de pregunta-respuesta con el alumno.
- RF4: El sistema mostrará los vídeos del agente según los estados de la conversación.
- RF5: El sistema irá creando un log compuesto por la conversación agente-alumno.
- RF6: El sistema enviará a un correo electrónico el log con la información de la sesión.
- El modulo gestor de Archivos XML se ha encargado de cargar la información de el fichero temario.xml y mediante sus etiquetas ha podido mostrar las explicaciones y preguntas con todo su contenido por la interfaz, controlar los turnos de la conversación, crear el archivo log con la información que se iba generando y lo más importante ha ido controlando el estado del agente para reproducir el vídeo adecuado.

- RF7: El usuario administrador podrá añadir preguntas nuevas al agente que el sistema almacenará en el archivo XML.
- RF8: El administrador podrá cargar las preguntas del fichero XML para visualizarlas y las podrá eliminar o modificar. El sistema registrará los cambios en el fichero XML.
- RF9: El administrador podrá añadir nuevos profesores para que puedan acceder a la aplicación. El sistema almacenará los nuevos datos en un fichero XML de usuarios.
- RF10: El administrador podrá eliminar a un profesor del fichero XML de usuarios.

- RF11: El administrador podrá modificar los datos de cualquier profesor. El sistema realizará los cambios en fichero de usuarios.
- RF12: El administrador podrá modificar los datos de acceso del usuario alumno. El sistema lo modificará en el fichero XML de usuarios.

- El módulo Gestor del Administrador es el que dará al administrador todas estas opciones que realizará junto con el gestor de Archivos XML que permitirá realizar los cambios respecto a las preguntas, añadiendo, modificando o eliminando, en el fichero temario.xml y de igual manera actuarán sobre la información de los usuarios del archivo usuarios.xml en el que también podrán agregar, modificar o eliminar los datos de los usuarios.

En el caso de los requisitos no funcionales:

- RNF1: La aplicación deberá cumplir los principios de interacción usuario-ordenador, contará con una interfaz amena, visual, usable e intuitiva para favorecer la comunicación.

- Se ha comprobado la usabilidad y lo intuitivo que es el programa, ya que ha sido probado por usuarios con unos niveles básicos de informática, lo que corrobora que es una aplicación apta para cualquier usuario sin necesidad de tener grandes conocimientos de informática

- RNF2: El tiempo de respuesta de la aplicación debe ser óptimo.

- La interacción con el agente, la reproducción de los vídeos y los sonidos se ha producido en un tiempo de espera adecuado, en ningún momento la aplicación da la sensación de lentitud.

- RNF3: Soporte afectivo, el agente puede intentar animar al estudiante y mantener su atención.

- Esto se ha conseguido con la integración de vídeos que expresan el estado del agente.

Trabajos Futuros

Las posibles líneas de trabajo futuro para esta aplicación serían las siguientes:

- La integración de vídeos con sistemas de reproducción de voz. De este modo ni cada pregunta ni cada afirmación o negación debería llevar un vídeo individual con la voz grabada de lo que está diciendo en ese momento, si no que el mismo vídeo valiera para cualquier pregunta pero fuera el sistema de reproducción de voz automático el que pusiera el sonido al vídeo. De esta manera sería mucho más fácil poder añadir nuevo temario al agente sin el inconveniente de tener grabado un vídeo para cada frase.

- La creación de un modo repaso en el cual el agente realice preguntas aleatorias de cualquiera de los niveles y el alumno las deba superar. Esto se podría realizar una vez ya superado el curso.

- Mejorar la forma de autoevaluación de las respuestas. Que la aplicación sea un poco más flexible y no sólo de cómo buena la respuesta si se escribió exactamente igual a cómo se definió.

- El administrador o profesor pueda cargar más módulos de teoría y pueda crear listas de preguntas y luego elegir cuál de ellas va a realizar al alumno.

6. Bibliografía

Referencias

[1] Pérez, Marín. D (2012). Aprende a diseñar y a construir tu propio Agente Conversacional Pedagógico, Universidad Rey Juan Carlos

[2] Artículo basado en entrevista con Jon Eyre, chief learning officer for Learning Voyage. Jon is expert in C++, ORB, CORBA, and Java, and teaches Oracle Application Server courses, as well as Java programming languages courses.

[3] ISO 12207-1 - Software Life Cycle Processes

Referencias Web

[HTTP1] portal.iteso.mx/portal/page/portal/.../FridaDiaz_paradigma.pdf

Último acceso 30 Mayo2012

[HTTP2] http://java.ciberaula.com/articulo/tecnologia_orientada_objetos/

Último acceso 30 Mayo 2012

[HTTP3] http://dev.mysql.com/doc/refman/5.0/es/what-is.html

Último acceso 31 Mayo 2012

[HTTP4] www.kybele.etsii.**urjc**.es/docencia/SI/.../[SI-2010-11]Tema4_SI.pdf

Último acceso 30 Mayo 2012

[HTTP5] http://java.ciberaula.com/articulo/tecnologia_orientada_objetos/

Último acceso 31 Mayo 2012

[HTTP6] http://www.latascadexela.es/2008/07/java-y-xml-dom-i.html

Último acceso 4 Junio 2012

Anexo A: Cuestionario de Satisfacción

Cuestionario de satisfacción

(Muchas Gracias por su colaboración)

Parte 1 – Evaluación Agente Shamael

1. ¿Le ha parecido un programa fácil de usar, intuitivo?
   -
2. ¿La interacción con el sistema le ha parecido fácil?
   -
3. ¿El tiempo de respuesta del programa ha sido el esperado?
   -
4. ¿La interfaz del programa le ha resultado atractiva? ¿Le han gustado las pantallas?
   -
5. ¿Qué opina de la inserción de vídeos que muestren el estado del avatar en vez de una imagen estática?
   -
6. ¿Cree que el uso de vídeos hacen del programa más entretenido o pueden desviar la atención del alumno del principal objetivo que es el aprendizaje?
   -
7. ¿Qué le parece el uso de imágenes y audio en las preguntas?
   -
8. ¿Qué es lo que más le ha gustado de esta aplicación?
   -
9. ¿Y lo qué menos?
   -
10. ¿Qué cambiaría?
    -

Parte 2 – Evaluación Zona Administrador

1. ¿Le ha parecido un programa fácil de usar, intuitivo?
   -

2. ¿Le parece una herramienta cómoda para un profesor?
   -
3. ¿La interacción con el sistema le ha parecido fácil?
   -
4. ¿El tiempo de respuesta del programa ha sido el esperado?
   -
5. ¿La interfaz del programa le ha resultado atractiva? ¿Le han gustado las pantallas?
   -
6. ¿Qué es lo que más le ha gustado de esta aplicación?
   -
7. ¿Y lo qué menos?
   -
8. ¿Qué cambiaría del programa?
   -

Parte 3 – Evaluación Aplicación en conjunto

1. ¿Le ha resultado fácil entender el fin de este programa?
   -
2. ¿Qué opina sobre la utilización de herramientas multimedia para el aprendizaje educativo?
   -
3. ¿Piensas que esta aplicación podría llegar a ser útil como complemento de aprendizaje?
   -
4. Usted se considera:
   a) Usuario sin conocimientos informáticos
   b) Usuario con conocimientos informáticos básicos
   c) Usuario con conocimientos informáticos avanzados

5. ¿Qué edad tiene usted?

6. Valore de 1-10 el conjunto de la aplicación.