

# Universidad de las Ciencias Informáticas

## Facultad 4



Interacciones gráficas del Componente creación de actividades de autoevaluación para la arquitectura basada en componentes del centro FORTES.

Trabajo de Diploma para optar por el título de Ingeniero Informático

**Autor(es):**

Yudismary Suárez Calero


Dinier Ordaz Sánchez

**Tutor(es):**

Ing. Lizardo Ramírez Tabuada

La Habana, junio de 2016

“Año 58 de la Revolución”



*“ En la tierra hacen falta personas que trabajen más y  
critiquen menos que construyan más y destruyan menos que  
digan mejor ahora que prometan menos y resuelvan más que  
esperen recibir menos y dar más que mañana ”* Che

## DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Centro de Tecnologías para la Formación FORTES, de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los \_\_\_\_ días del mes de \_\_\_\_\_ del año \_\_\_\_\_.

Autor(es)

\_\_\_\_Yudismary Suárez Calero\_\_\_\_\_

\_\_\_\_Dinier Ordaz Sánchez\_\_\_\_\_

Tutor(es)

\_\_\_\_Ing. Lizardo Ramírez Tabuada\_\_\_\_\_

## Datos de Contacto

Ing. Lizardo Ramírez Tabuada

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: [ltabuada@uci.cu](mailto:ltabuada@uci.cu)

## Agradecimientos

"[Insertar agradecimientos (opcional)]"

## Dedicatoria

"[Insertar dedicatoria (opcional)]"

## **RESUMEN**

En la actualidad las Tecnologías de la Información y las Comunicaciones (TIC) actúan vertiginosamente sobre nuestra sociedad, lo que motiva y acelera los procesos de cambio, esto modifica radicalmente las formas de trabajo, el acceso a los conocimientos, las formas de comunicación e incluso de aprender, incidiendo directamente sobre el ámbito educativo y fortaleciendo el proceso de enseñanza-aprendizaje. El uso de las tecnologías ha propiciado el uso de los Sistemas de Gestión de Aprendizaje (LMS), los cuales suelen incluir una herramienta de evaluación que permite la creación de exámenes y la manera de evaluarlos de forma automática. En el Centro de Tecnologías para la Formación (FORTES), de la Universidad de las Ciencias Informáticas (UCI), se está desarrollando un marco tecnológico llamado Xalix el cual carece de las herramientas para la gestión de ejercicios que puedan intercambiarse con otros sistemas. El presente trabajo tiene como propósito el desarrollo de las interacciones gráficas de un componente que permite gestionar ejercicios en el marco anteriormente mencionado sobre la arquitectura basada en componentes para la plataforma Zera 2.0, la cual surge dada la necesidad de contar con una plataforma de gestión del aprendizaje para la publicación de Cursos Masivos Abiertos en Línea. Como solución se desarrollan las interacciones gráficas de aplicación independiente que será la encargada de la gestión y el intercambio de dichos ejercicios. Para su implementación se utilizaron diferentes herramientas y tecnologías como el framework de desarrollo Symfony, el servidor web Apache, el gestor de base de datos PostgreSQL, entre otras. Para guiar el proceso de desarrollo se usó la metodología AUP en su versión UCI. Para la validación del sistema se realizaron pruebas unitarias utilizando la librería PHPUnit y pruebas de caja negra teniendo en cuenta el método de partición equivalente.

## **PALABRAS CLAVE**

Aprendizaje, ámbito educativo, ejercicios.

Tabla de Contenidos

RESUMEN.....	II
INTRODUCCIÓN.....	1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.....	6
1.1 Introducción .....	6
1.2 Conceptos asociados al dominio del problema. ....	6
1.3 Estándares en el e-learning .....	8
1.4 Sistemas Similares.....	10
1.5 Interacciones.....	14
1.6 Metodología de desarrollo de software.....	18
1.7 Lenguaje de modelado.....	21
1.8 Herramienta para el modelado.....	21
1.9 Lenguajes de desarrollo.....	25
1.10 Frameworks del lado del servidor.....	27
1.11 Conclusiones del capítulo .....	28
CAPÍTULO 2: PROPUESTA DE SOLUCIÓN. ANÁLISIS Y DISEÑO DEL SISTEMA.....	29
2.1 Introducción .....	29
2.2 Modelo del dominio.....	29
2.2.1 Definición de clases del Modelo del dominio.....	29
2.3 Descripción de sistema propuesto .....	31
2.4 Requisitos del sistema .....	32
2.4.1 Requisitos funcionales (RF) .....	33
2.6 Historias de Usuario.....	35
2.7 Modelo de análisis .....	40

2.8 Diagrama de clases del análisis .....	40
2.9 Diagrama de colaboración del análisis .....	41
2.10 Patrón arquitectónico Modelo Vista Controlador en Symfony2 .....	41
2.11 Patrones de diseño utilizados .....	43
2.12 Modelo del diseño .....	45
2.13 Diagrama de clases del diseño .....	45
2.14 Diagrama de secuencia del diseño .....	46
2.15 Diseño de la base de datos .....	47
2.15.1 Descripción de las tablas .....	48
2.16 Conclusiones del capítulo .....	49
<b>CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA .....</b>	<b>51</b>
3.1 Introducción .....	51
3.2 Modelo de implementación .....	51
3.3 Diagrama de componente .....	51
3.4 Pruebas de software .....	53
3.4.1 Niveles de prueba .....	53
3.4.2 Tipos de pruebas .....	54
3.4.3 Métodos de prueba .....	54
3.5 Diseño de casos de prueba.....	56
3.6 Resultados obtenidos.....	63
3.7 Conclusiones del capítulo .....	65
<b>CONCLUSIONES .....</b>	<b>66</b>
<b>RECOMENDACIONES .....</b>	<b>67</b>
<b>BIBLIOGRAFÍA.....</b>	<b>68</b>



GLOSARIO.....72

## INTRODUCCIÓN

En la actualidad las Tecnologías de la Información y las Comunicaciones (TIC), desempeñan un rol fundamental en la evolución de la sociedad. Las TIC's son el conjunto de tecnologías que permiten el acceso, producción, tratamiento y comunicación de información presentada en diferentes códigos (texto, imagen, sonido, etc.), son una especialización dentro del ámbito de la didáctica y de otras ciencias aplicadas de la educación, refiriéndose especialmente al diseño, desarrollo y aplicación de recursos en procesos educativos, no únicamente en los procesos instructivos, sino también en aspectos relacionados con la educación social y otros campos educativos. Estos recursos se refieren, en general, especialmente a los recursos de carácter informático, audiovisual, tecnológicos, del tratamiento de la información y los que facilitan la comunicación (1).

Las nuevas tecnologías ofrecen al docente grandes oportunidades, tanto a la hora de implantar mejoras en su metodología didáctica como a la hora de aumentar la eficiencia de sus propuestas actuales optimizando y simplificando las labores de enseñanza, tutorización y evaluación. Siendo el proceso enseñanza-aprendizaje la actividad central de cualquier centro educativo y las TIC's una realidad que ofrece un sin fin de nuevas oportunidades a este proceso.

Para la construcción de las actividades de formación, la plataforma virtual de tele formación conocida como Sistemas para la Gestión de Aprendizaje (*del inglés Learning Management System (LMS)*) constituye el elemento central, que no es más que una aplicación instalada en un servidor, que administra, distribuye y controla las actividades de formación de una institución u organización, teniendo como principales funcionalidades gestionar usuarios, recursos, así como materiales y actividades de formación, administrar el acceso, controlar y hacer seguimiento del proceso de aprendizaje, realizar evaluaciones, generar informes y gestionar servicios de comunicación. Los LMS son el marco que se encarga de todos los aspectos del proceso de aprendizaje. Un LMS es la infraestructura que ofrece y gestiona contenidos de instrucción, identifica y evalúa el aprendizaje individual, sigue el progreso hacia el logro de los objetivos y recoge y presenta datos para supervisar el proceso de aprendizaje (2).

Dentro de las funcionalidades de los LMS juega un papel fundamental la evaluación que incluye la gestión y realización de exámenes o ejercicios. Facilitando al profesor el trabajo para realizar los exámenes y corregirlos de forma automática, evaluando así la capacidad y rendimiento de cada uno de los estudiantes.

Para realizar el intercambio de estos ejercicios entre los diferentes sistemas, se hace necesario el uso de los IMS (del inglés *IMS Question and Test Interoperability (IMS-QTI)*). IMS define un marco de trabajo y arquitectura base para tráfico de voz, datos, video, servicios e imágenes conjuntamente a través de infraestructura basada en el ruteo de paquetes mediante direcciones IP. Esto permite incorporar en una red todo tipo de servicios de voz, multimedia y datos en una plataforma accesible a través de cualquier medio con conexión a internet, ya sea fija, o móvil; permitiendo así representar preguntas individuales y gestionar evaluaciones o exámenes completos. (2).

Dentro de la Universidad de las Ciencias Informáticas (UCI), la facultad número cuatro cuenta con el Centro de Tecnologías para la Formación (FORTES), donde se aplica un modelo de producción basado en Líneas de Productos de Software (LMS). LMS busca justamente lograr promover la reutilización sistemática de artefactos de los cuales la arquitectura es uno de los más importantes. Este enfoque busca tener distintos beneficios asociados a la reutilización como pueden ser la reducción del tiempo de desarrollo (pues ya no se tienen que desarrollar ciertas partes del sistema), y la mejora de la calidad (pues se incorporan partes que ya han sido verificadas previamente). Existen aplicaciones como la herramienta de autor web CRODA y el sistema para la gestión del aprendizaje ZERA, ambas permiten el diseño de ejercicios.

Existen actualmente dos versiones de CRODA las cuales brindan la posibilidad de desarrollar un sistema de auto evaluación que soporta la especificación IMS-QTI, pero estas soluciones están realizadas de manera independiente propiciando problemas de compatibilidad entre ellas y con otros sistemas; por lo que se hace necesario homogeneizar dichas versiones soportado sobre la arquitectura basada en componentes. Está en desarrollo la plataforma para cursos masivos abiertos en línea (Zera 2.0), la cual incluye la personalización básica e intermedia de diseño visual y módulos, dando la posibilidad de crear, administrar, almacenar, distribuir y gestionar las actividades de formación virtual. Esta plataforma es creada con el objetivo de aumentar la incorporación de estudiantes a la educación mediante el uso de cursos Zera 2.0, apoyar la formación docente en tecnologías educativas, brindar nuevas soluciones para incentivar el estudio en las personas, difundir las tecnologías de la información y las comunicaciones en la educación y aumentar la producción, distribución y actualización de cursos para los diferentes programas incluidos en esta modalidad.

Sobre esta plataforma se está desarrollando un componente para la creación de actividades de autoevaluación donde se cuenta con varios tipos de ejercicios, al cual se le está integrando otras tipologías de ejercicios como las interacciones gráficas.

Teniendo en cuenta la situación planteada anteriormente se define como **problema a resolver**: ¿Cómo dotar al componente de actividades de autoevaluación con la integración de las interacciones gráficas sobre la arquitectura basada en componentes para la plataforma Zera 2.0?

Partiendo del problema anteriormente planteado se define como **objeto de estudio** el proceso de gestión e intercambio de ejercicios con interacciones gráficas.

Tomando como **campo de acción** el proceso de gestión e intercambio de ejercicios con interacciones gráficas sobre la arquitectura basada en componentes.

Para dar solución al problema anteriormente se establece como **objetivo general**, desarrollar las interacciones gráficas del componente Creación de actividades de autoevaluación para la arquitectura basada en componentes del Centro de Tecnologías para la Formación (FORTES).

Para dar cumplimiento al objetivo general se consideran los siguientes **objetivos específicos**:

- Elaborar el marco teórico de la investigación mediante el estudio de las tendencias actuales y las principales herramientas que soportan la especificación de actividades de autoevaluación en diferentes escenarios.
- Realizar el análisis y diseño de las interacciones gráficas del componente creación de actividades de autoevaluación.
- Desarrollar las interacciones gráficas del componente creación de actividades de autoevaluación para la arquitectura basada en componentes.
- Validar mediante pruebas las interacciones gráficas del componente de autoevaluación.

Se espera como **posibles resultados** de esta investigación que el desarrollo de las interacciones gráficas del componente permita:

- Lograr el intercambio de ejercicios con interacciones gráficas entre distintos sistemas sobre la arquitectura basada en componentes.
- Permitir la gestión de ejercicios con interacciones gráficas basadas en la especificación IMS-QTI sobre la arquitectura basada en componentes.

Los métodos científicos que se utilizan para el desarrollo de la investigación son:

**Métodos teóricos:**

**Analítico-Sintético:** Se aplica para identificar los elementos más importantes y dar solución al problema planteado tras haber realizado un estudio de las fuentes bibliográficas existentes referentes al tema.

**Histórico-Lógico:** Para el estudio del desarrollo y evolución de sistemas similares así como la evolución de los motores de evaluación y las tendencias actuales de la interoperabilidad para el intercambio de ejercicios entre LMS.

**Inductivo-Deductivo:** Para el estudio de las principales iniciativas de motores de evaluación, permitiendo el arribo a conclusiones a partir del estudio de los conceptos y características generales y/o particulares de temas relacionados con la investigación.

**Modelación:** Para la generación de los artefactos que permitan lograr una mejor comprensión entre el equipo de desarrollo y las personas relacionadas.

El presente trabajo se encuentra dividido en tres capítulos con la siguiente organización:

### **Capítulo 1: Análisis del estado del arte.**

Se analizan y se exponen los enfoques teóricos, las principales investigaciones que anteceden a la presente, profundizando en el estudio de la especificación IMS-QTI, así como el de las soluciones existentes semejantes a las interacciones gráficas del componente que se desarrolla, para así tener una medida de los cambios a realizar sobre el trabajo en esta nueva arquitectura. Además, se enfatiza el estudio de las metodologías de desarrollo, las tecnologías y lenguajes de programación que se utilizan en el desarrollo de estas interacciones gráficas.

### **Capítulo 2: Propuesta de solución. Análisis y diseño del sistema.**

En este capítulo se especifica la propuesta de solución para las interacciones gráficas del componente a desarrollar, se describen las características del sistema así como parte del proceso de desarrollo. Se presenta el modelo del dominio y se especifican los requisitos a cumplir. Se definen los requisitos funcionales y no funcionales del sistema y se describen detalladamente mediante historias de usuario. Además se realiza el análisis y diseño de las interacciones gráficas del componente, realizando el modelado de múltiples diagramas como: diagramas de clases del análisis, los diagramas de colaboración, los diagramas de clases del diseño, diagramas de secuencia y el modelo de datos, permitiendo así tener una visión clara de las tipologías que se desean desarrollar.

### **Capítulo 3: Implementación y prueba.**

En este capítulo se describe el proceso de implementación a partir del resultado obtenido del diseño realizado en el capítulo anterior. Se definen los tipos y niveles de pruebas a utilizar para liberar un componente con calidad. Además se realiza el diagrama de componente para mostrar la organización de los componentes y las relaciones existentes entre ellos.

## **CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA**

### **1.1 Introducción**

En el presente capítulo se precisan un conjunto de elementos para conformar el marco teórico con los aspectos definidos en el objeto de estudio. Se examina el proceso de evaluación en los sistemas *e-learning* y las herramientas informáticas más utilizadas en la actualidad para apoyarlos, dichos sistemas constituyen una modalidad que enriquece la educación a distancia, por tanto juega un papel fundamental en el proceso de enseñanza-aprendizaje. Entre las herramientas que brindan apoyo a estos medios se destacan los LMS, los cuales proponen la estandarización de sus procesos, para lograr la integración con otros sistemas y mayor interoperabilidad de los contenidos generados.

### **1.2 Conceptos asociados al dominio del problema.**

#### **Sistema de gestión de aprendizaje**

Entre las herramientas más utilizadas para los ambientes o sistemas *e-learning* están los Sistemas de Administración de Aprendizaje (LMS). Son aplicaciones web que integran un conjunto de herramientas para el proceso de enseñanza-aprendizaje en línea, permitiendo una enseñanza no presencial (*e-learning*) y/o enseñanza mixta (*blended-learning*). (2)

Los LMS facilitan la creación de entornos de enseñanza-aprendizaje integrando materiales didácticos, herramientas de comunicación, colaboración y gestión educativa. (3)

Las principales funciones de los LMS son: gestionar usuarios, recursos, así como materiales y actividades de formación, administrar el acceso, controlar y hacer seguimiento del proceso de aprendizaje, realizar evaluaciones, generar informes, gestionar servicios de comunicación como de discusión, videoconferencias, entre otros. Un LMS generalmente no incluye posibilidades de autoría (crear sus propios contenidos), pero se focaliza en gestionar contenidos creados por fuentes diferentes. En sus funcionalidades juega un papel fundamental la evaluación, debido a que permite la creación, edición y realización de ciertos tipos de test de trabajos, la autocorrección o la corrección (con realimentación), la calificación y publicación de calificaciones y la visualización de información estadística sobre los resultados y, también, el progreso de cada alumno (3). En la evaluación radica una de las funciones pedagógicas principales del empleo de los LMS, debido a que facilitan la creación de ejercicios de evaluación y autoevaluación.

A partir de los conceptos anteriormente planteados se establece para la presente investigación que los LMS son sistemas basados en la web que facilitan la interacción entre los docentes y los estudiantes. Los mismos cuentan con las herramientas educativas para una correcta asimilación de los conocimientos por parte de los estudiantes y permiten el seguimiento y la evaluación.

## **E-learning**

Una traducción literal para el *e-learning* se denominaría como una educación virtual a distancia a través de canales electrónicos, refiriéndose en un sentido amplio a un tipo de proceso de enseñanza-aprendizaje realizado a través del uso de las nuevas tecnologías. Varias han sido las definiciones planteadas por profesionales y especialistas, ejemplo de ello lo constituyen las definiciones planteadas a continuación:

Según la autora Maricarmen González Videgaray presenta el concepto de *e-learning* como “aprendizaje basado en tecnologías de la información y de la comunicación con interacciones pedagógicas entre estudiante y contenidos, estudiante y estudiante, y estudiante e instructor” (4).

Según los autores Manuel Area Moreira y Jordi Adell Segura el *e-learning* “es una modalidad de enseñanza-aprendizaje que consiste en el diseño, puesta en práctica y evaluación de un curso o plan formativo desarrollado a través de redes de ordenadores y puede definirse como una educación o formación ofrecida a individuos que están geográficamente dispersos o separados o que interactúan en tiempos diferidos del docente empleando los recursos informáticos y de telecomunicaciones” (5).

Según el Dr. Francisco José García Peñalvo, el *e-learning* es la “capacitación no presencial que, a través de plataformas tecnológicas, posibilita y flexibiliza el acceso y el tiempo en el proceso de enseñanza-aprendizaje, adecuándolos a las habilidades, necesidades y disponibilidades de cada discente, además de garantizar ambientes de aprendizaje colaborativos mediante el uso de herramientas de comunicación síncrona y asíncrona, potenciando en suma el proceso de gestión basado en competencias” (6).

Después de haber realizado un análisis de los diferentes conceptos antes expuestos, se deduce que hay gran diversidad de criterios entre los diferentes autores con respecto a la definición de *e-learning*, mientras que unos lo consideran como un proceso de aprendizaje a distancia, otros lo tratan como una modalidad de enseñanza-aprendizaje, pero todos coinciden en que se realiza haciendo uso de las nuevas tecnologías.

Por lo que para la siguiente investigación se asumen los sistemas *e-learning* como una modalidad educativa, no solo de formación a distancia, sino también como apoyo a clases presenciales o incluso un modelo mixto



de clases semi presencial. Es considerado como un proceso de enseñanza-aprendizaje que facilita la reutilización de materiales docentes ya desarrollados dentro de las aplicaciones y entre ellas.

### **Retroalimentación**

La retroalimentación o *feedback* se refiere a un proceso de autorregulación pluridireccional de los sistemas finalistas que permite compartir observaciones, resultados y/o sugerencias, con la intención de recolectar información, a nivel individual o colectivo, para intentar mejorar el funcionamiento de cualquier grupo formado por seres humanos. La retroalimentación le posibilita al profesor reorientar el proceso de enseñar ya que, a partir de la información resultante de los exámenes (calificaciones, nivel de aprobación, indicadores de recursado, etc.), se hace posible reorientar las estrategias de enseñanza, los recursos empleados en los procesos de enseñar y la reflexión sobre las modalidades e instrumentos de evaluación empleados. (7).

El concepto de retroalimentación que se defiende en la presente investigación, es el que define la retroalimentación como “un producto resultante de la revisión y el análisis por parte del profesor a la actividad, aportación o proyecto que ha enviado el estudiante. (8).

### **1.3 Estándares en el e-learning**

Los estándares son acuerdos internacionales documentados o normas establecidas por consenso mundial. Contienen las especificaciones técnicas y de calidad que deben reunir todos los productos y servicios para cumplir satisfactoriamente con las necesidades para las que han sido creadas y para poder competir internacionalmente en condiciones de igualdad. (9)

En el campo técnico, la estandarización es el proceso por el cual se establecen normas comúnmente aceptadas que permiten la cooperación de diferentes empresas o instituciones sin menoscabar su posibilidad de competir. Una especificación es un conjunto de declaraciones detalladas y exactas de los requisitos funcionales y particularidades de algo que quiere construirse, instalarse o manufacturarse. (10)

Entre los principales objetivos que persigue la aplicación de un estándar en el *e-learning* se encuentran los siguientes (11):

**Durabilidad:** Que la tecnología desarrollada con el estándar evite la obsolescencia.

**Interoperabilidad:** Que se pueda intercambiar información a través de una amplia variedad de LMS.

**Accesibilidad:** Que un usuario pueda acceder al contenido apropiado en el momento justo y en el dispositivo correcto.

**Reusabilidad:** Que los distintos objetos de aprendizaje puedan ser reutilizados con diferentes herramientas y en distintas plataformas.

**Adaptabilidad:** Que se facilite la adaptación o personalización del entorno de aprendizaje.

**Productividad:** Si los proveedores de tecnología *e-learning* desarrollan sus productos siguiendo estándares comúnmente aceptados, la efectividad de *e-learning* se incrementa significativamente y el tiempo y costos se reducen.

Actualmente, el principal promotor y desarrollador de especificaciones abiertas orientadas a la enseñanza electrónica es IMS3 *Global Learning Consortium*. Consorcio internacional que ha propuesto un conjunto de especificaciones sobre distintos aspectos que interviene en el modelado del aprendizaje en *e-learning*. Entre las especificaciones disponibles se encuentran IMS-QTI y el sistema de paquetes de contenidos de IMS (del inglés *IMS Content Packaging (IMS-CP)*). La primera ha sido uno de los intentos más reconocidos por lograr la interoperabilidad de ejercicios, entre sistemas gestores de aprendizaje y la segunda fue desarrollada para describir el modo en el que se debe empaquetar el contenido educativo para que pueda ser procesado por otro LMS. (11)

El empleo de los estándares en *e-learning* trae consigo múltiples beneficios por diversas razones. Los estándares garantizan la viabilidad futura de la inversión en productos del *e-learning*, impidiendo que sean dependientes de una única tecnología. Por otra parte, facilita la aparición de herramientas estándar para la creación de contenidos, permitiendo que las organizaciones puedan desarrollar contenidos sin la necesidad de acudir a especialistas en *e-learning*. (11)

### **IMS-QTI**

IMS-QTI es un formato estándar que permite crear preguntas individuales y evaluaciones completas. El objetivo principal de esta especificación es permitir el intercambio de preguntas, evaluaciones y resultados entre distintas herramientas (10). Con este propósito IMS QTI plantea un modelo en el que se definen los componentes principales que intervienen en el proceso de evaluación y, adicionalmente a este modelo, se proporciona un formato de contenido para almacenar las preguntas de manera independientemente del sistema o herramienta de autoría utilizada para crearlas. Es válido para herramientas de autor, *e-learning*, encuestas, etc. Es un formato de intercambio entre distintas plataformas otorgando interoperabilidad entre

ellas; en su primera versión se centraba en crear preguntas individuales, ahora también permite la creación de exámenes completos y bancos de preguntas.

Se basa en un modelo de datos descrito en el lenguaje unificado de modelado (del inglés *Unified Modeling Language* (UML)) y con una traslación directa al lenguaje de marcado extensible (del inglés *Extensible Markup Language* (XML)) para su intercambio. (10)

### Historia y versiones IMS-QTI

IMS QTI es una de las especificaciones en las que el consorcio IMS trabajó más tempranamente (marzo 1999) con la versión 0.5. En la historia de IMS QTI hay 3 versiones de la misma que han tenido una gran repercusión: IMS QTI versión 1.2, IMS QTI versión 2.0 y finalmente IMS QTI versión 2.1. IMS QTI versión 1.2 (aparece en 2002) es la última versión finalizada completamente en la que se abarcan tanto preguntas individuales como exámenes completos.

Con el desarrollo de nuevas especificaciones como el *IMS Content Packaging*, *IMS Simple Sequencing* y *IMS Learning Design* surgió la necesidad de hacer compatible la especificación IMS QTI a las nuevas iniciativas, así apareció la versión 2.0 (2005), dicha versión introduce como novedad las preguntas adaptativas que permiten al alumno realizar varios intentos sobre dicha pregunta. Esto permite, por ejemplo, que el alumno pueda alterar su respuesta debido a la realimentación, por ejemplo, la presentación de alguna pista en caso de que la respuesta sea incorrecta o parcialmente incorrecta, o que se le planteen preguntas adicionales en función de su respuesta actual.

IMS QTI versión 2.1 (aparece en 2006) está actualmente en proceso de evolución en modo borrador sobre el que la comunidad, tanto educativa como técnica, puede opinar. El objetivo de esta nueva versión es seguir con el proceso de simplificación y evolución de la especificación, esta vez dando soporte a los exámenes completos y al intercambio de los resultados de los mismos. (10)

En esta investigación solo se hará alusión a las dos últimas versiones.

Luego de realizar un estudio de las versiones actuales de la especificación IMS-QTI se ha decidido utilizar la versión 2.1 puesto que, la versión 2.0 no permite la creación y el intercambio de exámenes completos.

### 1.4 Sistemas Similares

	Utiliza IMS-QTI	Exporta utilizando QTI	Importa utilizando QTI	Exporta varios ejercicios	Utiliza IMS-CP
--	-----------------	------------------------	------------------------	---------------------------	----------------

<b>CRODA</b>	No	No	No	No	No
<b>ZERA</b>	Sí	Sí	Sí	No	Sí
<b>Moodle</b>	Sí	Sí	Sí	Sí	Sí
<b>Claroline</b>	Sí	Sí	Sí	No	Sí

Tabla 1 Sistemas similares.

## **CRODA**

CRODA es una herramienta de autor web que pone a disposición de los docentes, una aplicación libre que posibilita crear Objetos de Aprendizaje de forma individual y colaborativa, así como Diseños de Aprendizaje estandarizados (12).

Principales funcionalidades: (12)

- Creación de Objetos y Diseños de Aprendizaje estandarizados.
- Gestión de plantillas.
- Gestión de metadatos.
- Creación de actividades de autoevaluación.
- Integración con el Repositorio de Objetos de Aprendizaje RHODA.
- Mensajería interna y notificaciones.
- Gestión de usuarios y roles.

## **ZERA**

ZERA es una plataforma educativa integral que une características de Sistemas de Gestión Académica, Gestión de Contenidos y del Aprendizaje. Se basa en una concepción pedagógica cubana de hiperentornos de aprendizaje. Integra elementos esenciales desde el punto de vista psicológico, didáctico y estándares educativos utilizados a nivel mundial en plataformas de aprendizaje. Contiene módulos para las prácticas y simulaciones, lo que facilita el trabajo de docentes y estudiantes en la creación y desarrollo de actividades de evaluación y autoevaluación (13).

Principales funcionalidades (13):

- Gestión de instituciones educativas (escuelas).
- Internacionalización al español e inglés. Posibilidad de incluir soporte para otros idiomas.

- Gestión de usuarios (estudiantes, docentes, tutores, directores de las escuelas, administradores locales, centrales, y editores de contenidos).
- Gestión de sistemas educativos (sistema curricular, programas de estudio, sistema evaluativo).
- Gestión de grupos académicos.
- Trabajo con Foros
- Gestión de materias e índices de contenido.
- Gestión de recursos de tipo:
  - Multimedia
  - Interactivos
  - Estructurales
  - Materiales
- Gestión de ejercicios de tipo
  - Selección simple.
  - Selección múltiple.
  - Completar por escritura.
  - Completar por desplazamiento (solo texto).
  - Ordenar procedimiento.
  - Ordenar media (solo imagen).
  - Enlazar (texto-texto, texto-imagen, imagen-imagen).
  - Clasificar texto. i. Clasificar media (solo imagen).
  - Selección de puntos en una imagen. k. Identificar áreas en una imagen.
- Integración con el estándar SCORM 2004 3ra Edición, importar y exportar paquetes.
- Integración con el repositorio XAUCE RHODA.

## **Moodle**

Moodle es un software diseñado para ayudar a los educadores a crear cursos en línea de alta calidad y entornos de aprendizaje virtuales. Tales sistemas de aprendizaje en línea son algunas veces llamados VLEs (*Virtual Learning Environments*) o entornos virtuales de aprendizaje. Una de las principales características de Moodle sobre otros sistemas es que está hecho en base a la pedagogía social constructivista, donde la comunicación tiene un espacio relevante en el camino de la construcción del conocimiento. Siendo el objetivo generar una experiencia de aprendizaje enriquecedora (14).

## IMS QTI en Moodle

Otra especificación del consorcio IMS que está (parcialmente) soportada dentro de Moodle, es la especificación IMS *Question and Test Interoperability* descrita previamente. En el caso de la especificación QTI está soportada parte de la versión 2.0 de la misma, permitiendo a un profesor la posibilidad de poder exportar las preguntas que ha creado utilizando Moodle al formato IMS QTI (15).

## Moodle y la interoperabilidad

Para soportar la integración fluida y el uso de contenido proveniente de diferentes orígenes y múltiples proveedores, la plataforma Moodle está diseñada para intercambiar datos empleando estándares abiertos de la industria para desplegarse en web, y soporta:

### **Autenticación** usando:

- LDAP, el protocolo estándar más ampliamente usado para Autenticación.
- Búsqueda directa en la Base de Datos (por ejemplo, en una Base de Datos Oracle externa), o en el protocolo Shibboleth, o en forma alterna, usando IMAP, NNTP, CAS o *FirstClass*.

### **Inscripción** usando:

- Servidor LDAP (*Active Directory*)
- IMS *Enterprise standard* (mediante un plugin descargable).

**Contenido** que usa la importación de Objetos de Aprendizaje reutilizables, empacados de acuerdo a los estándares de Empaquetamiento de Contenido SCORM/AICC/IMS.

- Moodle 1.9.5 está certificado como compatible con SCORM 1.2.
- Moodle soporta Importar y Exportar IMS *Common Cartridge*
- El uso de XML para importación/exportación de contenido (estándar en Moodle). El método de intercambio de datos con otros sistemas mediante servicios web (por ejemplo, mediante SOAP o XML-RPC) todavía no es estándar - pero está en desarrollo activo.

**Preguntas de examen** vía importar y exportar, usando una variedad de formatos.

## **Claroline**

Permite tanto a docentes como a instituciones educativas crear y administrar cursos en la web. Posibilita al docente gestionar y editar pruebas de evaluación y autoevaluación que permiten dar seguimiento a los

resultados obtenidos de los estudiantes. Se encuentra ubicada dentro de las plataformas que más herramientas de información compartida ofrecen. Permite evaluar los cuestionarios y actividades que realizan los alumnos. (16).

La versión 2.1 de QTI define los tipos de interacciones que puede realizar el usuario, y permite crear nuevas interacciones para poder extender el modelo y realizar nuevos tipos de preguntas.

### **Zera 2.0 vs Moodle. Interacciones gráficas**

Cada uno de estos sistemas similares brinda la posibilidad de crear ejercicios, ayudando así tanto a docentes como a instituciones educativas en el proceso de evaluación en la educación. De todos los sistemas comparados se demostró en la presente investigación que Moodle era el más completo con respecto al componente a desarrollar en la plataforma Zera 2.0, ya que cumplía con las características principales que se definieron en la tabla anteriormente expuesta. Moodle brinda la posibilidad de crear varias tipologías de ejercicios dentro de las que se encuentran las interacciones gráficas, pero el trabajo con este tipo de interacciones es bien complejo, sobre todo a la hora de trabajar sobre las imágenes de fondo que utiliza esta tipología, mientras que la plataforma Zera 2.0 con el desarrollo de estas interacciones gráficas para el componente tiene como objetivo facilitar el trabajo con las imágenes a utilizar a la hora de señalar cualquier área sobre la misma, marcando las coordenadas automáticamente sin que el usuario tenga que introducirlas de forma manual. Esto se convierte en la principal ventaja de las interacciones gráficas de Zera 2.0 sobre las interacciones gráficas de Moodle ya que facilita el trabajo al usuario para interactuar con el sistema, propiciando un mejor entendimiento con el mismo y mayor motivación para el uso de estas interacciones sobre la plataforma.

### **1.5 Interacciones**

Las preguntas individuales en QTI son auto-contenidas, es decir, incluyen toda la información necesaria para su presentación al estudiante y su corrección automática, es por esto que el término pregunta es reemplazado por “interacción”. Una interacción permite al profesor especificar puntualmente la herramienta que tendrá el estudiante para poder construir una respuesta. Así como existen múltiples tipos de preguntas, existen múltiples tipos de interacciones. A continuación se describen los tipos de interacciones posibles dentro de una pregunta junto a un ejemplo gráfico derivado de la documentación de la especificación.

#### **Interacciones Simples**

Las interacciones simples son aquellas interacciones en las que la corrección de las mismas se realiza en base a la selección de una opción o varias opciones disponibles. Dentro de estas interacciones se encuentran las siguientes (17):

- **choiceInteraction:** Esta interacción muestra al estudiante un conjunto de posibles opciones. El estudiante podrá seleccionar una o varias posibles opciones como respuesta. Es posible indicar que el conjunto de posibles opciones sea barajado entre distintos intentos del estudiante.
- **orderInteraction:** En esta interacción el objetivo del estudiante es reordenar el conjunto de soluciones proporcionadas. Además, es posible un número mínimo y un número máximo de opciones que conforman la solución, de manera que se realizaría una selección sobre las opciones disponibles y posteriormente se realizaría una ordenación de los elementos de dicha selección.
- **associateInteraction:** Esta interacción presenta al estudiante un conjunto de opciones y permite crear asociaciones por parejas entre dichas opciones. Es posible indicar el número mínimo y máximo de asociaciones que deben crearse como parte de la respuesta. Además, también es posible indicar el número mínimo y máximo de veces que una de las opciones puede aparecer dentro de una asociación.
- **matchInteraction:** Esta interacción presenta al estudiante dos conjuntos de opciones y le permite crear pares de asociaciones entre ellas. Al igual que en la interacción anterior es posible indicar el número mínimo y máximo de asociaciones posibles o el número mínimo y máximo de apariciones de una de las opciones en las asociaciones creadas.
- **gapMatchInteraction:** Esta interacción permite definir un conjunto de espacios en blanco dentro del enunciado de la pregunta a mostrar al estudiante. Además se permitirá al estudiante asociar a cada uno de los espacios una de las posibles opciones de respuesta. Hay que destacar que las opciones posibles son compartidas por todos los espacios. Como posibles respuestas es posible utilizar texto o también es posible utilizar imágenes. Además es posible restringir el número mínimo y máximo de veces que es utilizada cada una de las posibles opciones del conjunto de respuestas.

### **Interacciones de Texto**

En esta categoría se encuentran las interacciones en las que la respuesta que construirá el estudiante puede ser una única palabra, una frase corta o un párrafo de texto completo. Estas interacciones permiten que durante el proceso de corrección se tenga en cuenta la respuesta en forma de texto que ha construido el estudiante. Dentro de estas interacciones se encuentran las siguientes (17):



- **inlineChoiceInteraction:** Esta interacción está pensada para definir un espacio donde se permitirá al usuario escoger entre un conjunto de opciones, donde cada una de estas opciones es una palabra o frase corta. A diferencia de **gapMatchInteraction**, esta interacción está ideada para que cada uno de los espacios pueda tener un conjunto de opciones independientes. Es posible definir que las respuestas sean barajadas entre distintos intentos del estudiante.
- **textEntryInteraction:** Al igual que la interacción anterior, esta interacción tiene como objetivo crear un espacio donde se permitirá escribir una palabra o frase corta para poder construir la respuesta. Cuando se define una pregunta con esta interacción es posible especificar la longitud del texto que se espera que el estudiante introduzca.
- **extendedTextInteraction:** Esta interacción está pensada para que el estudiante construya como respuesta un párrafo de texto. Es posible indicar el número mínimo y máximo de líneas de texto esperadas, junto con la longitud máxima de cada una de ellas.
- **hotTextInteraction:** El objetivo de esta interacción es que el estudiante seleccione partes de texto que estarán resaltadas en el enunciado de la pregunta. Es posible indicar el número mínimo y máximo de elecciones que debe realizar el estudiante, siendo por defecto 0, el valor mínimo, y 1, el valor máximo, de selecciones.

### **Interacciones Gráficas**

Las interacciones gráficas tienen como elemento principal una imagen que se utilizará como fondo del enunciado y sobre la que se realizarán todas las acciones permitidas en las interacciones para que el usuario construya la respuesta. Dentro de estas interacciones se encuentran las siguientes (17):

- **hotspotInteraction:** El objetivo de esta interacción es presentar al estudiante un conjunto de hotspots (puntos críticos) sobre una imagen utilizada como fondo del enunciado. El estudiante deberá seleccionar uno o varios de estos puntos críticos para construir la respuesta. Es posible especificar el número mínimo y máximo de selecciones que debe realizar el estudiante, siendo 0, el valor mínimo, y 1, el valor máximo, de selecciones por defecto.
- **selectPointInteraction:** El objetivo de esta interacción es que el usuario seleccione uno o varios puntos de una imagen utilizada como fondo del enunciado. Al contrario que en la interacción anterior, no se le presentará al estudiante ninguna zona resaltada.
- **graphicOrderInteraction:** Esta interacción mostrará un conjunto de puntos críticos sobre una imagen que será utilizada como fondo del enunciado. El objetivo es que el usuario realice una ordenación de estos puntos críticos. Al igual que la interacción **orderInteraction**, es posible definir el número mínimo y máximo

de opciones que formarán parte de la respuesta, de manera que el usuario primero seleccionará las opciones y posteriormente realizará la ordenación de las mismas.

- **graphicAssociateInteraction:** Esta interacción mostrará un conjunto de zonas seleccionables o puntos críticos sobre una imagen que será utilizada como fondo del enunciado. El objetivo de la interacción es permitir al estudiante la creación de pares de asociación entre los puntos críticos. Es posible indicar el número máximo de asociaciones que el estudiante puede crear. Asimismo también es posible indicar el número mínimo y máximo de cada uno de los puntos críticos dentro de las asociaciones creadas.
- **graphicGapMatchInteraction:** Esta interacción mostrará un conjunto de puntos críticos sobre una imagen que será utilizada como fondo del enunciado, además se proporcionará al usuario un conjunto de opciones. El objetivo es que el usuario construya parejas entre los puntos críticos y las opciones que le son proporcionadas. Hay que destacar que el conjunto de opciones disponibles es compartido por todos los puntos críticos. Asimismo, es posible definir el número mínimo y máximo de veces que puede aparecer una de las opciones en una de la parejas creadas por el usuario.
- **positionObjectInteraction:** En esta interacción el estudiante colocará una imagen que le será proporcionada sobre alguna zona de otra imagen que será utilizada como fondo del enunciado. Esta interacción es similar a la interacción **selectPointInteraction**, ya que ambas tienen como objetivo seleccionar puntos de la imagen que se utiliza como fondo. En el caso particular de la interacción **positionObjectInteraction** la posición seleccionada se marcará con la imagen que se le proporciona al usuario. Es posible definir el número mínimo y máximo de posibles selecciones que puede realizar el usuario.

### **Otros tipos de Interacciones**

En esta categoría se encuentran interacciones relativamente avanzadas. (17).

- **sliderInteraction:** Esta interacción muestra al estudiante una barra deslizante que permitirá al usuario seleccionar la respuesta correcta.
- **mediaInteraction:** Esta interacción está pensada para ser utilizada de manera conjunta con alguna otra interacción. El objetivo de esta interacción es permitir controlar el número de veces que un usuario visualiza un material multimedia, de esta manera es posible incluir esta interacción con un video como enunciado, e incluir alguna otra interacción que realmente realice una pregunta acerca del enunciado. El número de veces que el usuario visualice el vídeo puede utilizarse para cambiar la interacción o simplemente puede ser utilizada para posteriormente realizar un estudio estadístico acerca del número de veces que es necesario visualizar el vídeo para poder responder a la pregunta.

- drawingInteraction: Esta interacción tiene como objetivo permitir al usuario pintar sobre una imagen proporcionada en el enunciado. El resultado de la pregunta será la propia imagen modificada.
- uploadInteraction: Esta interacción tiene como objetivo permitir al usuario crear una respuesta a partir de un archivo que será enviado a la herramienta desde el computador del usuario.
- customInteraction: Esta interacción tiene como objetivo servir como base para la creación de interacciones particulares de cada herramienta que no se puedan ajustar a ninguna de las interacciones descritas anteriormente.

Las interacciones seleccionadas para desarrollar son hotspotInteraction, selectPointInteraction, graphicOrderInteraction, graphicAssociateInteraction, graphicGapMatchInteraction y positionObjectInteraction. Se realizó dicha selección debido a la adecuada representación gráfica de estas preguntas, estas facilitan el trabajo al estudiante durante el proceso de evaluación ya que contienen imágenes de apoyo que motivan al usuario a interactuar y trabajar con el sistema que contiene los ejercicios de interacciones gráficas.

## **1.6 Metodología de desarrollo de software**

Para el desarrollo de un buen software se hace necesaria la elección de la mejor metodología. Un proceso de software detallado y completo suele denominarse metodología. Una metodología es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar un software. Las metodologías de desarrollo de software pueden clasificarse en tradicionales/pesadas o ágiles. Las primeras centran su atención en llevar una documentación exhaustiva de todo el proyecto y en cumplir con un plan de proyecto, definido todo esto, en la fase inicial del desarrollo. Mientras que la segunda se basa en dos aspectos puntuales, el retrasar las decisiones y la planificación adaptativa; permitiendo potenciar aún más el desarrollo de software a gran escala. (18)

A continuación se realiza un estudio de las metodologías *Extreme Programming (XP)*, *Rational Unified Process (RUP)* y *Agile Unified Process (AUP)* siendo estas las más utilizadas durante el proceso de desarrollo de software.

### **Extreme Programming (XP)**

XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, está concebida para dirigir las necesidades específicas del desarrollo de software conducido por equipos pequeños. Esta metodología es más adecuada para proyectos con requisitos

imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. Es considerada ligera, flexible, predecible, de bajo riesgo, y no por ello menos científica. (19)

### **Rational Unified Process (RUP)**

RUP es una metodología tradicional, además, es un proceso formal que provee un acercamiento disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo.

Su objetivo es asegurar la producción de software de alta calidad que satisfaga los requerimientos de los usuarios finales. Puede ser adaptado y extendido para satisfacer las necesidades de la organización que lo adopte. (18). RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en un número variable según el proyecto y en las que se hace un mayor o menor hincapié en las distintas actividades. Las cuatro fases del ciclo de vida son:

- Inicio: Se desarrolla una descripción del producto visual a partir de una buena idea y se presenta el análisis de negocio para el producto (determinar la visión del proyecto).
- Elaboración: Se especifican en detalle la mayoría de los casos de usos del producto y se diseña la arquitectura del sistema, mediante modelos de casos de uso, análisis, diseño, implementación y despliegue (determinar la arquitectura óptima).
- Construcción: Se crea el producto (obtener la capacidad inicial).
- Transición: Se obtiene el producto acabado y definido.

### **Agile Unified Process (AUP)**

AUP es una versión simplificada del Proceso Unificado de Rational (RUP). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. (20)

AUP establece cuatro fases al igual que RUP:

- Inicio: El objetivo de esta fase es obtener una comprensión común cliente-equipo de desarrollo del alcance del nuevo sistema y definir una o varias arquitecturas candidatas para el mismo.
- Elaboración: El objetivo es que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema y en validar la arquitectura.

- Construcción: Durante la fase de construcción el sistema es desarrollado y probado al completo en el ambiente de desarrollo.
- Transición: El sistema se lleva a los entornos de preproducción donde se somete a pruebas de validación y aceptación y finalmente se despliega en los sistemas de producción.

### **Agile Unified Process (AUP) versión UCI**

Al no existir una metodología de software universal y como cada metodología debe ser adaptada a cada uno de los proyectos de la Universidad de las Ciencias Informáticas (UCI), se decide hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la Universidad. Esta metodología se apoya en el Modelo CMMI-DEV v1.3, el cual constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora. De las cuatro fases que propone AUP (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola, a la que llamaremos Ejecución y se agrega la fase de Cierre.

AUP UCI establece tres fases (20):

- Inicio: Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
- Ejecución: En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.
- Cierre: En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

### **Selección de la metodología a utilizar**

Después de haber realizado un análisis sobre estas metodologías se definió como metodología a utilizar *Agile Unified Process* (AUP) en su versión UCI ya que esta se adapta al ciclo de vida definido por la actividad productiva de la UCI, convergiendo en una única metodología que cubre las particularidades de cada una de ellas, esta metodología se adapta a lo que la Universidad ha estado proponiendo como ciclo de vida de

los proyectos sin alejarse de lo que se ha trabajado hasta el momento, logrando así estandarizar el proceso de desarrollo de software y dar cumplimiento de las buenas prácticas que define CMMI-DEV v1.3, así como hablar un lenguaje común en cuanto a fases, disciplinas, roles y productos de trabajo.

### **1.7 Lenguaje de modelado**

*Unified Modeling Language* (UML), es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. Permite a los creadores de sistemas generar diseños que capturen sus ideas en una forma convencional y fácil de entender para comunicárselas a otras personas. (21)

Características de UML:

- Gran capacidad para el soporte de diferentes tipos de diagramas.
- Se basa fundamentalmente en sus esquemas de apoyo de diseño, documentación, construcción e implantación de sistema.
- Posee gran flexibilidad para admitir cambios no previstos durante el diseño o el rediseño.
- Se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real.
- Se utiliza UML para la modelación de la solución propuesta por las características que ofrece, pues admite diseño desde inicio a fin, diseño inverso o rediseño y diseño vice-versa, con esquemas amplios para documentar detalladamente los procesos.

### **1.8 Herramienta para el modelado**

#### **Visual Paradigm**

Visual Paradigm es una herramienta CASE: Ingeniería de Software Asistida por Computación. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas y constituye una herramienta privada disponible en varias ediciones, cada una destinada a satisfacer diferentes necesidades. (22)

Características principales:

- Disponibilidad en múltiples plataformas (Windows, Linux).

- Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, con diferentes especificaciones.
- Generación de código para Java y exportación como HTML.
- Soporte de UML versión 2.1.
- Ingeniería inversa Java, C++, Esquemas XML, XML, NET exe/dll, CORBA IDL.
- Importación y exportación de ficheros XMI.

La herramienta anteriormente definida constituye un elemento fundamental para llevar a cabo la investigación ya que esta nos permite representar todos los diagramas que propone la metodología a utilizar *Agile Unified Process (AUP)*. La documentación que propone contribuye a lograr un mejor entendimiento del sistema por parte del equipo de desarrollo.

### **Gestor de Base de Datos**

Un Sistema Gestor de Base de Datos (SGBD), en inglés *DataBase Management System (DBMS)* es un sistema de software que permite la definición de bases de datos; así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación. (23)

### **MySQL**

MySQL es un sistema gestor de bases de datos relacionales rápido, sólido y flexible. Es idóneo para la creación de bases de datos con acceso desde páginas web dinámicas, así como para la creación de cualquier otra solución que implique el almacenamiento de datos, posibilitando realizar múltiples y rápidas consultas. Posee excelente documentación, ofrece soporte de alta calidad para sus productos, incluyendo un servicio que permite a los desarrolladores de MySQL iniciar sesión en el servidor para corregir problemas de forma proactiva y ayudar con la optimización. Está disponible para una gran variedad de arquitecturas de computadoras y muchos sistemas operativos diferentes. (24)

### **PostgreSQL**

PostgreSQL es un Sistema Gestor de Bases de Datos Relacionales Orientadas a Objetos. Utiliza un modelo cliente/servidor y multiprocesos en vez de multihilos para garantizar la estabilidad del sistema, por lo que un

fallo en uno de los procesos no afecta el resto. Es un gestor de bases de datos de código abierto, brinda un control de concurrencia multi-versión (MVCC por sus siglas en inglés) que permite trabajar con grandes volúmenes de datos; soporta gran parte de la sintaxis SQL y cuenta con un extenso grupo de enlaces con lenguajes de programación. Posee una integridad referencial e interfaces nativas para lenguajes como ODBC, JDBC, C, C++, PHP, PERL, TCL, ECPG; PYTHON y RUBY. Funciona en todos los sistemas operativos Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), y Windows. Debido a la liberación de la licencia, PostgreSQL se puede usar, modificar y distribuir de forma gratuita para cualquier fin, ya sea privado, comercial o académico. Durante su desarrollo, la estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características más significativas (25).

La selección de PostgreSQL como SGBD se basó en la necesidad garantizar la estabilidad del sistema ya que usa multiprocesos en lugar de multihilos, funciona muy bien ante una alta concurrencia de usuarios, y permite el manejo de grandes volúmenes de información. Se utilizará la versión 9.3.5.

### **Servidor web**

Es un programa que gestiona cualquier aplicación en el lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente generando una respuesta en cualquier lenguaje o aplicación en el lado del cliente. El código recibido por el cliente suele ser compilado y ejecutado por un Navegador Web. Para la transmisión de todos estos datos se utiliza algún protocolo. Generalmente se utiliza el protocolo HTTP para estas comunicaciones, perteneciente a la capa de aplicación del Modelo OSI (26).

### **Apache**

Apache es un servidor de páginas web de código abierto multiplataforma y modular. Está diseñado para ser un Servidor Web potente y flexible que pueda funcionar en la más amplia variedad de plataformas y entornos. Apache se ha adaptado siempre a una gran variedad de entornos a través de su diseño modular. Este diseño permite a los administradores de Sitios Web elegir qué características van a ser incluidas en el servidor seleccionando, que módulos se van a cargar, ya sea al compilar o al ejecutar el servidor. Este es el más común y más utilizado en todo el mundo. Además permite desplegar aplicaciones de PHP y brinda soporte para bases de datos (26).

### **Entorno de Desarrollo Integrado**



Un entorno de desarrollo integrado en inglés *Integrated Development Environment* (IDE), es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica.

Algunos ejemplos de entornos integrados de desarrollo (IDE) son los siguientes:

- Eclipse
- NetBeans
- IntelliJ IDEA
- JBuilder de Borland
- JDeveloper de Oracle
- KDevelop
- Anjuta
- Clarion
- MS Visual Studio
- Visual C++

Un IDE debe tener las siguientes características:

- Multiplataforma.
- Soporte para diversos lenguajes de programación.
- Integración con Sistemas de Control de Versiones.
- Reconocimiento de Sintaxis.
- Extensiones y Componentes para el IDE.
- Integración con Framework populares.
- Depurador.
- Importar y Exportar proyectos.
- Múltiples idiomas.
- Manual de Usuarios y Ayuda.

Se define como Entorno de Desarrollo Integrado (IDE) NetBeans.

### **NetBeans IDE**

NetBeans IDE es un entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje

de programación. Existe además un número importante de módulos para extender el NetBeans-IDE. NetBeans-IDE es un producto libre y gratuito sin restricciones de uso. Facilita bastante el diseño gráfico asociado a aplicaciones Java. Ofrece todas las funciones de los entornos de desarrollo integrado avanzados como diseño de interfaces, asistentes para la conexión con bases de datos, creación automática de propiedades y clases, etc. (27)

La versión a utilizar será la 8.0 la cual cuenta con comandos para el uso de Symfony2, que pueden ejecutarse desde el mismo IDE sin necesidad de instalar extensiones adicionales y brinda soporte al control de versiones y trabajo colaborativo.

### **1.9 Lenguajes de desarrollo**

Un lenguaje de programación es el conjunto de sentencias que sirven para decirle a una computadora qué es lo que tiene que hacer. Mediante los lenguajes de programación se pueden crear y modificar programas y comandos, para que una computadora u otro dispositivo digital puedan realizar diversas funciones lógicas. (28)

#### **Lenguajes de desarrollo del lado del cliente**

**Hoja de estilo en cascada** o **CSS** (siglas en inglés *cascading style sheets*) es un lenguaje usado para definir y crear la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). Es utilizado para definir el aspecto de cada elemento: color, tamaño, tipo de letra, separación vertical y horizontal entre elementos, posición de cada elemento dentro de la página (29). La versión a utilizar será la 3.0

#### **Limitaciones de usar CSS**

- Los selectores no pueden usarse en orden ascendente según la jerarquía del DOM (hacia padres u otros ancestros).
- Dificultad para el alineamiento vertical; así como el centrado horizontal se hace de manera evidente en CSS2.1, el centrado vertical requiere de diferentes reglas en combinaciones no evidentes, o no estándares.
- Ausencia de expresiones de cálculo numérico para especificar valores.
- Las pseudo-clases dinámicas no se pueden controlar o deshabilitar desde el navegador, lo que las hace susceptibles de abuso por parte de los diseñadores en banners, o ventanas emergentes.

#### **Ventajas de usar CSS**

- Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.
- Optimización del ancho de banda de la conexión, pues pueden definirse los mismos estilos para muchos elementos con un sólo selector; o porque un mismo archivo CSS puede servir para una multitud de documentos.
- Mejora en la accesibilidad del documento, pues con el uso del CSS se evitan antiguas prácticas necesarias para el control del diseño (como las tablas), y que iban en perjuicio de ciertos usos de los documentos, por parte de navegadores orientados a personas con algunas limitaciones sensoriales.

**Lenguaje de Marcado de Hipertexto** (por sus siglas en inglés *Hypertext Markup Language* (HTML)) es el lenguaje con el que se escriben las páginas web. Es un lenguaje de hipertexto, es decir, un lenguaje que permite escribir texto de forma estructurada, y que está compuesto por etiquetas, que marcan el inicio y el fin de cada elemento del documento. Además, permite crear documentos de tipo multimedia, es decir, que contengan información más allá de la simplemente textual como por ejemplo: imágenes, videos, sonido, entre otros (30). La versión a utilizar será la 5.0.

**Java Script** es un sub-lenguaje o “dialecto” enfocado a los navegadores web, si bien se basa en el lenguaje Java parte de su diseño, sintaxis y estructura posee similitudes con el lenguaje C, aunque poseyendo sus características propias. Es un lenguaje moderno, sencillo, muy útil, barato pues solo se necesita un bloc de notas y un navegador, además, es visual debido a que permite la moderna programación visual, cuenta con una navegación simple y agradable para el usuario (31). La versión a utilizar será la 1.8.

## **Lenguajes de desarrollo del lado del servidor**

### **Hypertext Pre-processor**

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico, se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hasta el día de hoy. Es un lenguaje de programación sencillo, de sintaxis cómoda y similar a otros lenguajes. Es rápido, interpretado, orientado a objetos y multiplataforma (32). La versión a utilizar será la 5.4.13.

## **Frameworks del lado del cliente**

### **JQuery**

jQuery es un framework JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Ofrece una infraestructura con la que se obtendrá mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Posibilita agregar efectos a las páginas haciéndola más interactiva (33). La versión a utilizar será la 1.10.2.

## **Bootstrap**

Bootstrap es un framework o conjunto de herramientas de software libre para diseño de sitios y aplicaciones web. Es un framework que posee numerosos componentes web. Además, permite un ahorro significativo de esfuerzo y tiempo debido a que facilita la creación de interfaces que se adapten a cualquier navegador. Bootstrap es modular y consiste esencialmente en una serie de hojas de estilo LESS (lenguaje de hojas de estilo) que implementan la variedad de componentes de la herramienta. Por otra parte, es capaz de integrarse con las principales librerías JavaScript, por ejemplo, jQuery (34). La versión a utilizar será la 3.0.

## **1.10 Frameworks del lado del servidor**

### **Symfony**

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. Symfony está desarrollado completamente con PHP, es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas \*nix (Unix, Linux, etc.) como en plataformas Windows (35).

Entre sus principales características se encuentran:

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y \*nix estándares).
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.

- Basado en la premisa de "*convenir en vez de configurar*", en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

La versión a utilizar será la 2.7.9.

### **1.11 Conclusiones del capítulo**

En este capítulo se estudiaron los estándares más utilizados, para garantizar la interoperabilidad en *e-learning*, donde se demostró que IMS-QTI es una de las especificaciones más usadas y se empleará para representar los ejercicios en el desarrollo de la propuesta de solución. El estudio de los sistemas similares evidenció a pesar que el entorno virtual de aprendizaje Moodle satisface las características específicas para la solución propuesta, las interacciones gráficas en este no alcanzan los niveles de usabilidad deseados para el componente a desarrollar. Para el desarrollo de las interacciones gráficas se escogieron determinadas tecnologías y herramientas que le aportaron eficiencia y efectividad al desarrollo del software entre las que se encuentran el Visual Paradigm para la generación de artefactos, Apache como servidor web, PostgreSQL como gestor de base de datos y NetBeans como herramienta de desarrollo. Todo el proceso de desarrollo del software será guiado y controlado mediante la metodología de desarrollo AUP en su versión UCI.

## **CAPÍTULO 2: PROPUESTA DE SOLUCIÓN. ANÁLISIS Y DISEÑO DEL SISTEMA.**

### **2.1 Introducción**

En este capítulo se realiza una descripción de la solución propuesta con el objetivo de garantizar un mejor entendimiento del sistema. Se representa el diagrama del modelo del dominio con la correspondiente descripción de cada uno de los elementos, se especifican los requisitos funcionales y no funcionales con los que debe cumplir el sistema a desarrollar, así como las historias de usuario para encapsular cada uno de los requisitos funcionales. Además se lleva a cabo los flujos de trabajo análisis y diseño que sustentan la propuesta de solución, realizándose el modelo del análisis y el modelo del diseño, generándose varios artefactos como: los diagramas de clases, los diagramas de interacción y el modelo de datos.

### **2.2 Modelo del dominio**

Un modelo de dominio es una representación de las clases conceptuales del mundo real, no de componentes de software. Constituye el artefacto más importante que se crea durante el análisis orientado a objetos. Mediante la notación UML, se representa un conjunto de diagramas de clases en los que no se define ninguna operación. En el mismo se muestran los objetos del dominio y las asociaciones entre las clases conceptuales, facilita la comprensión de los sistemas debido a que permite clasificar y agrupar objetos, disminuyendo la complejidad y el número de elementos en el modelado (36).

En la presente investigación se realiza la confección de un modelo de dominio ya que el mismo ayuda a comprender los conceptos relacionados a cada uno de los procesos que se van a desarrollar. Luego de haber realizado un análisis del problema se identificaron dichos conceptos, se confecciona el modelo de dominio, representando mediante un marco conceptual las relaciones entre las diferentes definiciones.

#### **2.2.1 Definición de clases del Modelo del dominio**

Con el objetivo de lograr una mejor comprensión del diagrama del modelo del dominio se hace necesario definir los conceptos relacionados con dicho modelo, los cuales se presentan a continuación:

**Usuario:** Persona que realiza las tareas de administración, gestión e intercambio de ejercicios en el sistema.

**Ejercicio:** Preguntas formuladas que permiten comprobar el nivel de asimilación del estudiante de un contenido dado.

**Imagen:** Representación gráfica que se utiliza como fondo del ejercicio para elaborar la respuesta.

**Tipología:** Tipo de interacción que se utiliza dentro de un ejercicio.

**Retroalimentación:** Mensaje donde se le brinda información al estudiante, indicando si la respuesta de un ejercicio está correcta o incorrecta, proporcionándole una ayuda para mejorar próximas evaluaciones.

**Complejidad:** Nivel de complejidad de un ejercicio, puede ser muy fácil, fácil, difícil, muy difícil.

### Diagrama del modelo del dominio

En el siguiente diagrama se muestran las relaciones existentes entre los conceptos definidos anteriormente.

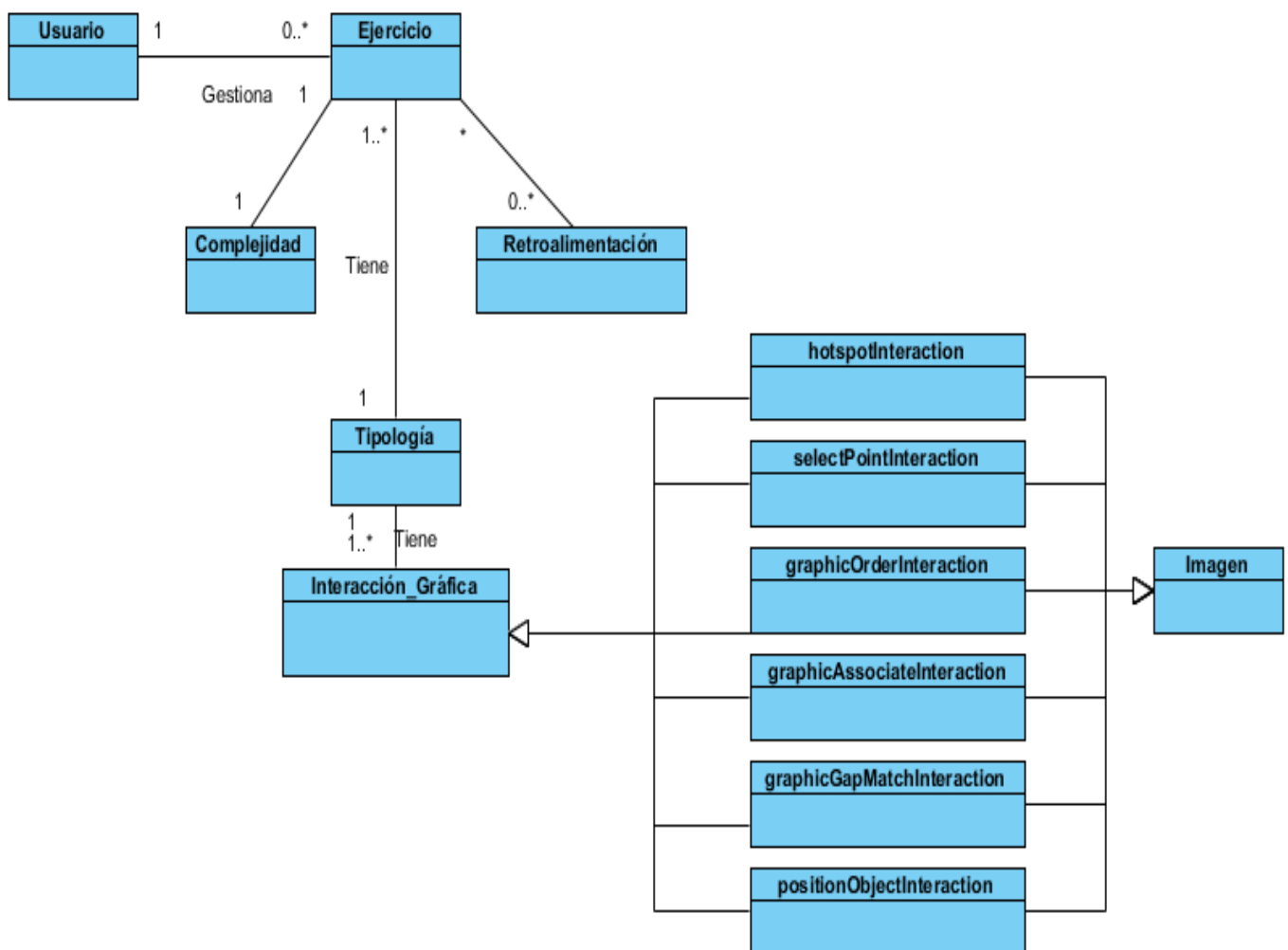


Ilustración 1 Representación del modelo del dominio.

### 2.3 Descripción de sistema propuesto

El componente que se desea desarrollar tiene como objetivo principal la gestión de ejercicios, siendo así de vital importancia para el proceso de evaluación en la educación. Estos ejercicios podrán ser creados utilizando algunas de las interacciones definidas por IMS-QTI, dentro de estas se encuentran las interacciones gráficas que contienen dentro hotspotInteraction, selectPointInteraction, graphicOrderInteraction, graphicAssociateInteraction, GraphicGapMatchInteraction y positionObjectInteraction. Esta tipología de ejercicio contiene imágenes asociadas y retroalimentación con el objetivo de ayudar al estudiante en su evaluación. Luego de haber gestionado el ejercicio el sistema podrá realizar intercambio de estos con sistemas similares siguiendo el formato IMS-QTI.

A continuación se muestran las descripciones de cada una de las interacciones que se van a desarrollar en el componente (17):

- hotspotInteraction: El objetivo de esta interacción es presentar al estudiante un conjunto de hotspots (puntos críticos) sobre una imagen utilizada como fondo del enunciado. El estudiante deberá seleccionar uno o varios de estos puntos críticos para construir la respuesta. Es posible especificar el número mínimo y máximo de selecciones que debe realizar el estudiante, siendo 0, el valor mínimo, y 1, el valor máximo, de selecciones por defecto.
- selectPointInteraction: El objetivo de esta interacción es que el usuario seleccione uno o varios puntos de una imagen utilizada como fondo del enunciado. Al contrario que en la interacción anterior, no se le presentará al estudiante ninguna zona resaltada.
- graphicOrderInteraction: Esta interacción mostrará un conjunto de puntos críticos sobre una imagen que será utilizada como fondo del enunciado. El objetivo es que el usuario realice una ordenación de estos puntos críticos. Al igual que la interacción orderInteraction, es posible definir el número mínimo y máximo de opciones que formarán parte de la respuesta, de manera que el usuario primero seleccionará las opciones y posteriormente realizará la ordenación de las mismas.
- graphicAssociateInteraction: Esta interacción mostrará un conjunto de zonas seleccionables o puntos críticos sobre una imagen que será utilizada como fondo del enunciado. El objetivo de la interacción es permitir al estudiante la creación de pares de asociación entre los puntos críticos. Es posible indicar el número máximo de asociaciones que el estudiante puede crear. Asimismo también es posible indicar el número mínimo y máximo de cada uno de los puntos críticos dentro de las asociaciones creadas.



- **graphicGapMatchInteraction:** Esta interacción mostrará un conjunto de puntos críticos sobre una imagen que será utilizada como fondo del enunciado, además se proporcionará al usuario un conjunto de opciones. El objetivo es que el usuario construya parejas entre los puntos críticos y las opciones que le son proporcionadas. Hay que destacar que el conjunto de opciones disponibles es compartido por todos los puntos críticos. Asimismo, es posible definir el número mínimo y máximo de veces que puede aparecer una de las opciones en una de la parejas creadas por el usuario.
- **positionObjectInteraction:** En esta interacción el estudiante colocará una imagen que le será proporcionada sobre alguna zona de otra imagen que será utilizada como fondo del enunciado. Esta interacción es similar a la interacción **selectPointInteraction**, ya que ambas tienen como objetivo seleccionar puntos de la imagen que se utiliza como fondo. En el caso particular de la interacción **position Object Interaction** la posición seleccionada se marcará con la imagen que se le proporciona al usuario. Es posible definir el número mínimo y máximo de posibles selecciones que puede realizar el usuario.

## **2.4 Requisitos del sistema**

La obtención y análisis de los requisitos del sistema es una de las etapas de mayor importancia en el proceso de ingeniería de requisitos. Para la obtención de los mismos deben ser analizadas las ideas del cliente y de los miembros del equipo del proyecto.

Los requisitos son clasificados en funcionales (RF) y no funcionales (RNF), un requisito funcional (RF) no es más que la característica requerida del sistema que expresa una capacidad de acción del mismo, una funcionalidad; generalmente expresada en una declaración en forma verbal, mientras que un requisito no funcional (RNF) es la característica requerida del sistema, del proceso de desarrollo, del servicio prestado o de cualquier otro aspecto del desarrollo, que señala una restricción del mismo (21).

Para definir los requisitos funcionales del sistema existen varias técnicas de captura de requisitos:

- Entrevista.
- Prototipado.
- Tormenta de ideas.

En la presente investigación se utilizó como técnica de captura de requisitos la entrevista donde el cliente especificó cada una de las funcionalidades con las que debe cumplir el sistema.

### **2.4.1 Requisitos funcionales (RF)**

#### **Proceso de gestionar ejercicio hotspotInteraction**

R1: crear ejercicio de hotspotInteraction.

R2: listar ejercicio de hotspotInteraction.

R3: modificar ejercicio de hotspotInteraction.

R4: visualizar ejercicio de hotspotInteraction.

#### **Proceso de gestionar ejercicio selectPointInteraction**

R5: crear ejercicio de selectPointInteraction.

R6: listar ejercicio de selectPointInteraction.

R7: modificar ejercicio de selectPointInteraction.

R8: visualizar ejercicio de selectPointInteraction.

#### **Proceso de gestionar ejercicio graphicOrderInteraction**

R9: crear ejercicio de graphicOrderInteraction.

R10: listar ejercicio de graphicOrderInteraction.

R11: modificar ejercicio de graphicOrderInteraction.

R12: visualizar ejercicio de graphicOrderInteraction.

#### **Proceso de gestionar ejercicio graphicAssociateInteraction**

R13: crear ejercicio de graphicAssociateInteraction.

R14: listar ejercicio de graphicAssociateInteraction.

R15: modificar ejercicio de graphicAssociateInteraction.

R16: visualizar ejercicio de graphicAssociateInteraction.

#### **Proceso de gestionar ejercicio graphicGapMatchInteraction**

R17: crear ejercicio de graphicGapMatchInteraction.

R18: listar ejercicio de graphicGapMatchInteraction.

R19: modificar ejercicio de graphicGapMatchInteraction.

R20: visualizar ejercicio de graphicGapMatchInteraction.

### **Proceso de gestionar ejercicio positionObjectInteraction**

R21: crear ejercicio de positionObjectInteraction.

R22: listar ejercicio de positionObjectInteraction.

R23: modificar ejercicio de positionObjectInteraction.

R24: visualizar ejercicio de positionObjectInteraction.

## **2.4.2 Requisitos no funcionales (RNF)**

### **Interfaz**

RNF1: Debe contener un diseño sencillo, donde sea fácil el trabajo para el usuario aunque no tenga conocimientos informáticos.

RNF2: Debe mostrar claridad y buena organización de la información, permitiendo la interpretación correcta de esta.

### **Usabilidad**

RNF3: El sistema desarrollado podrá ser utilizado por cualquier persona con conocimientos básicos informáticos.

### **Portabilidad**

RNF4: El usuario podrá acceder a la aplicación desde cualquier sistema operativo.

### **Confiabilidad**

RNF5: Solo los usuarios con un usuario y contraseña válida tendrán acceso a la aplicación.

### **Eficiencia**

RNF6: Los tiempos de respuestas de cualquier petición del usuario deben estar por debajo de los 6 segundos.

### **Software**

RNF7: El componente debe poder visualizarse en los siguientes navegadores: Mozilla Firefox 28, Internet Explorer 9, Chrome 24, o superiores.

## **Hardware**

RNF8: Se necesita como mínimo un procesador Pentium 4, con 512 megabytes de memoria RAM y capacidad de almacenamiento de 20 gigabyte.

## **Restricciones en el diseño y la implementación**

RNF9: Lenguaje de programación PHP 5.4.13.

RNF10: Framework de desarrollo Symfony v2.7.9.

RNF11: Lenguaje del lado del cliente HTML5.

RNF12: Librería CCS: Bootstrap 3.0.

RNF13: Librería de JavaScript: JQuery v1.10.2.

RNF14: SGBD: PostgreSQL v9.3.5.

## **2.6 Historias de Usuario**

Una historia de usuario (o *user story* en inglés) describe una funcionalidad que, por sí misma, aporta valor al usuario. Se compone de (37):

- Una descripción escrita de la historia usada como recordatorio y para planificar. (Debe ser breve).
- Conversaciones acerca de la historia que sirven para aclarar los detalles.
- Un criterio de aceptación (idealmente automatizado) que permita determinar cuándo la historia ha sido completada.

Además, las Historias de Usuario deben cumplir las siguientes características para que puedan realizar su función de manera correcta:

- Independientes. Deben ser atómicas en su definición. Es decir, se debe intentar que no dependa de otras historias para poder completarla.
- Negociables. Como he dicho anteriormente, son entidades vivas. Deben ser ambiguas en su enunciado para poder debatirlas, dejando su concreción a los criterios de aceptación.
- Valoradas. Deben ser valoradas por el cliente. Para poder saber cuánto aporta al Valor de la aplicación y junto con la estimación convertirse en un criterio de prioridad.

- Estimables. Aunque sea siempre un poco como leer de una bola de cristal, deben poder ser estimadas. Tener su alcance lo suficientemente definido como para poder suponer una medida de trabajo en la que pueda ser completarla.
- Pequeñas. Para poder realizar una estimación con cierta validez y no perder la visión de la Historia de Usuario, se recomienda que sean mayores de dos días y menores de dos semanas.

### Descripción de las historias de usuario

A continuación se presentan las historias de usuario de cada uno de los requisitos funcionales definidos con sus descripciones.

<b>Número:</b> 1	<b>Nombre del requisito:</b> Crear ejercicio de hotspotInteraction.
<b>Programador:</b> Yudismary Suárez Calero y Dinier Ordaz Sánchez.	<b>Iteración Asignada:</b> 1ra
<b>Prioridad:</b> Alta	<b>Tiempo Estimado:</b> 1 semana
<b>Riesgo en Desarrollo:</b>	<b>Tiempo Real:</b>
<p><b>Descripción:</b>  <b>El sistema debe permitir crear ejercicio de hotspotInteraction. Para crear un ejercicio hotspotInteraction hay que:</b>  Introducir o seleccionar los siguientes datos:</p> <ul style="list-style-type: none"> <li>• Título: nombre del ejercicio. Admite cualquier carácter.</li> <li>• Descripción: descripción de ejercicio. Admite cualquier carácter.</li> <li>• Recurso de apoyo: permite adjuntar cualquier tipo de archivo como recurso auxiliar.</li> <li>• Compartir: te da la opción de compartir o no el ejercicio que se está creando.</li> <li>• Complejidad: complejidad del ejercicio que estará dada en muy fácil, fácil, difícil y muy difícil.</li> <li>• Evaluación: permite otorgar una evaluación al estudiante ya sea automática o dada por el profesor.</li> <li>• Orientación: enunciado del ejercicio. Admite cualquier carácter.</li> <li>• Imagen: imagen de fondo obligatoria que se utiliza en el ejercicio para dar una solución. Permite archivos de tipo jpg/jpeg/png.</li> </ul>	

- ✓ El sistema debe cumplir con la especificación IMS-QTI, permitiendo el intercambio de preguntas, evaluaciones y resultados entre distintas herramientas.
- ✓ El sistema debe presentar al estudiante un conjunto de puntos calientes sobre una imagen seleccionada como fondo del enunciado, permitiendo la selección de uno o varios puntos para construir su respuesta y mostrándosele una zona resaltada.
- ✓ Al crear un ejercicio debe mostrar un mensaje de información en caso de que no se llenen todos los campos obligatorios, mostrando un indicador sobre los campos obligatorios.
- ✓ Cuando el usuario incluye y/o selecciona correctamente los datos necesarios para crear un ejercicio de hotspotInteraction y selecciona la opción Guardar, se crea un nuevo ejercicio y el sistema muestra un mensaje de información.
- ✓ En caso de que existan datos incompletos muestra un mensaje de error, dando la posibilidad al usuario de realizar nuevamente la acción en cuestión.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**

*Tabla 2 HU Crear ejercicio de hotspotInteraction.*

<b>Número:</b> 2	<b>Nombre del requisito:</b> Listar ejercicio de hotspotInteraction.
<b>Programador:</b> Yudismary Suárez Calero y Dinier Ordaz Sánchez.	<b>Iteración Asignada:</b> 1ra
<b>Prioridad:</b> Media.	<b>Tiempo Estimado:</b> 2 días
<b>Riesgo en Desarrollo:</b>	<b>Tiempo Real:</b>
<b>Descripción:</b>	
El sistema debe permitir listar el ejercicio de hotspotInteraction. Para listar un ejercicio de hotspotInteraction hay que:	
<ul style="list-style-type: none"> <li>✓ Mostrar el ejercicio con los siguientes datos:</li> </ul>	

- Título: nombre del ejercicio. Admite cualquier carácter.
- Descripción: descripción de ejercicio. Admite cualquier carácter.
- Recurso de apoyo: permite adjuntar cualquier tipo de archivo como recurso auxiliar.
- Compartir: te da la opción de compartir o no el ejercicio que se está creando.
- Complejidad: complejidad del ejercicio que estará dada en muy fácil, fácil, difícil y muy difícil.
- Evaluación: permite otorgar una evaluación al estudiante ya sea automática o dada por el profesor.
- Orientación: enunciado del ejercicio. Admite cualquier carácter.
- Imagen: imagen de fondo obligatoria que se utiliza en el ejercicio para dar una solución. Permite archivos de tipo jpg/jpeg/png.
- ✓ El sistema debe permitir mostrar un listado de los ejercicios de hotspotInteraction.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**

*Ilustración 2 Listar ejercicio de hotspotInteraction*

<b>Número:</b> 3	<b>Nombre del requisito:</b> Modificar ejercicio de hotspotInteraction.
<b>Programador:</b> Yudismary Suárez Calero y Dinier Ordaz Sánchez.	<b>Iteración Asignada:</b> 1ra
<b>Prioridad:</b> Media.	<b>Tiempo Estimado:</b> 3 días
<b>Riesgo en Desarrollo:</b>	<b>Tiempo Real:</b>
<b>Descripción:</b>	
<p><b>El sistema debe permitir modificar el ejercicio de hotspotInteraction. Para modificar un ejercicio de hotspotInteraction hay que:</b></p> <ul style="list-style-type: none"> <li>✓ Mostrar el ejercicio permitiendo modificar los siguientes datos: <ul style="list-style-type: none"> <li>• Título: nombre del ejercicio. Admite cualquier carácter.</li> <li>• Descripción: descripción de ejercicio. Admite cualquier carácter.</li> <li>• Recurso de apoyo: permite adjuntar cualquier tipo de archivo como recurso auxiliar.</li> </ul> </li> </ul>	

- Compartir: te da la opción de compartir o no el ejercicio que se está creando.
- Complejidad: complejidad del ejercicio que estará dada en muy fácil, fácil, difícil y muy difícil.
- Evaluación: permite otorgar una evaluación al estudiante ya sea automática o dada por el profesor.
- Orientación: enunciado del ejercicio. Admite cualquier carácter.
- Imagen: imagen de fondo obligatoria que se utiliza en el ejercicio para dar una solución. Permite archivos de tipo jpg/jpeg/png.
- ✓ Cuando el usuario modifica correctamente los datos necesarios para crear un ejercicio de hotspotInteraction y selecciona la opción Guardar, se modifica el ejercicio y el sistema muestra un mensaje de información.
- ✓ Al modificar un ejercicio debe mostrar un mensaje de información en caso de que no se llenen todos los campos obligatorios, mostrando un indicador sobre los campos obligatorios.
- ✓ En caso de que existan datos incompletos muestra un mensaje de error.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**

*Tabla 3 HU Modificar ejercicio de hotspotInteraction*

<b>Número:</b> 4	<b>Nombre del requisito:</b> Visualizar el ejercicio de hotspotInteraction.
<b>Programador:</b> Yudismary Suárez Calero y Dinier Ordaz Sánchez.	<b>Iteración Asignada:</b> 1ra
<b>Prioridad:</b> Alta.	<b>Tiempo Estimado:</b> 1 semana
<b>Riesgo en Desarrollo:</b>	<b>Tiempo Real:</b>
<b>Descripción:</b>	
<p><b>El sistema debe permitir visualizar el ejercicio de hotspotInteraction. Para visualizar un ejercicio de hotspotInteraction hay que:</b></p> <ul style="list-style-type: none"> <li>✓ Mostrar el ejercicio con los siguientes datos: <ul style="list-style-type: none"> <li>• Título: nombre del ejercicio. Admite cualquier carácter.</li> <li>• Descripción: descripción de ejercicio. Admite cualquier carácter.</li> </ul> </li> </ul>	



- **Recurso de apoyo:** permite adjuntar cualquier tipo de archivo como recurso auxiliar.
- **Compartir:** te da la opción de compartir o no el ejercicio que se está creando.
- **Complejidad:** complejidad del ejercicio que estará dada en muy fácil, fácil, difícil y muy difícil.
- **Evaluación:** permite otorgar una evaluación al estudiante ya sea automática o dada por el profesor.
- **Orientación:** enunciado del ejercicio. Admite cualquier carácter.
- **Imagen:** imagen de fondo obligatoria que se utiliza en el ejercicio para dar una solución. Permite archivos de tipo jpg/jpeg/png.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**

*Tabla 4 HU Visualizar ejercicio de hotspotInteraction.*

## **2.7 Modelo de análisis**

El modelo de análisis es la primera representación técnica de un sistema. Utiliza una mezcla de formatos en texto y diagramas para representar los requisitos del software, las funciones y el comportamiento. De esta manera se hace mucho más fácil de comprender dicha representación, ya que es posible examinar los requisitos desde diferentes puntos de vista aumentando la probabilidad de encontrar errores, de que surjan debilidades y de que se descubran descuidos (21).

## **2.8 Diagrama de clases del análisis**

Los diagramas de clases del análisis se utilizan para mostrar clases y sus relaciones, así como subsistemas e interfaces. Las clases de análisis representan abstracciones de clases, se centran en el tratamiento de los RF y son evidentes en el contexto del dominio del problema debido a que representan conceptos y relaciones del dominio. Las del análisis siempre encajan en uno de los tres estereotipos básicos: clase de interfaz, de control y de entidad (21).

A continuación se presenta el diagrama de clases del análisis (DCA) del proceso gestionar ejercicio de hotspotInteraction. Para el estudio de los demás diagramas de clases del análisis remitirse a los Anexos.

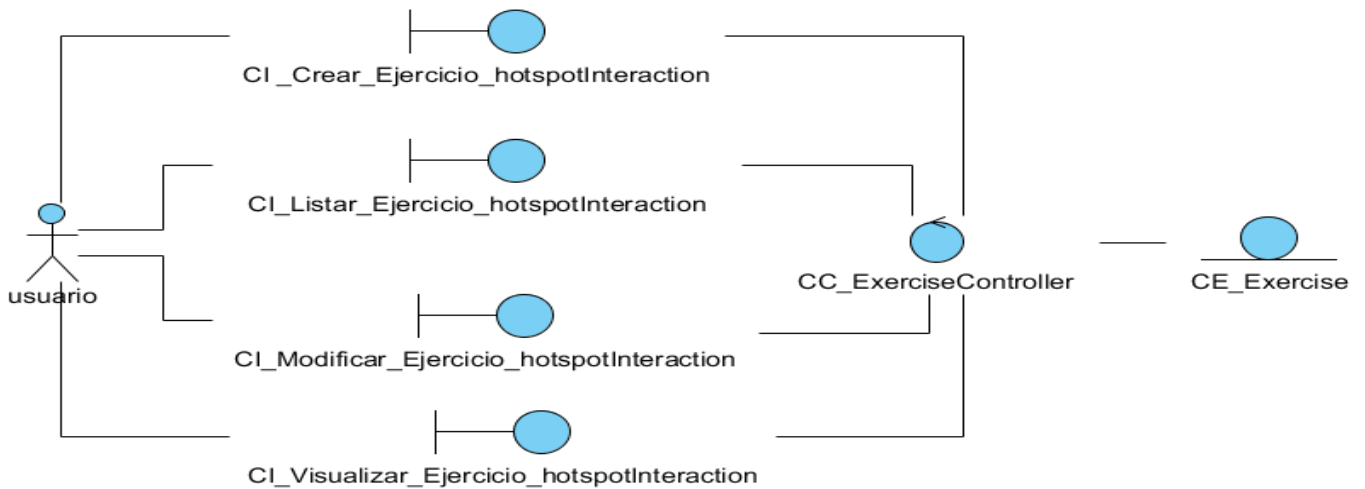


Ilustración 3 Diagrama de clases del análisis. Gestionar ejercicio de hotspotInteraction.

## 2.9 Diagrama de colaboración del análisis

El diagrama de colaboración es un tipo de diagrama de interacción cuyo objetivo es describir el comportamiento dinámico del sistema de información mostrando cómo interactúan los objetos entre sí, es decir, con qué otros objetos tiene vínculos o intercambia mensajes un determinado objeto (38).

A continuación se presenta el diagrama de clases del diseño correspondiente a la HU Crear ejercicio de hotspotInteraction. Para el estudio de los demás diagramas de colaboración del análisis remitirse a los Anexos.

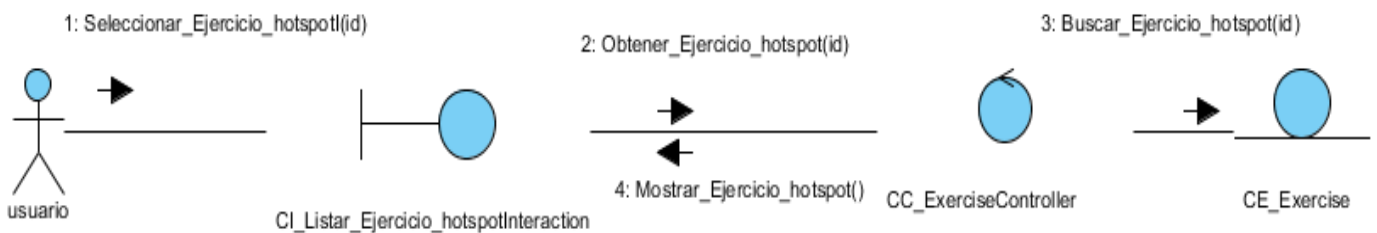


Ilustración 4 Diagrama de colaboración del análisis: HU Crear ejercicio hotspotInteraction.

## 2.10 Patrón arquitectónico Modelo Vista Controlador en Symfony2

El uso del framework que utiliza MVC obliga a dividir y organizar el código de acuerdo a las convenciones establecidas por el framework. El código de la presentación se guarda en la vista, el código de manipulación de datos se guarda en el modelo y la lógica de procesamiento de las peticiones constituye el controlador.

Symfony toma lo mejor de la arquitectura MVC y la realiza de modo que el desarrollo de aplicaciones sea rápido y sencillo. En el controlador se encuentran las acciones, las cuales son el núcleo de la aplicación, pues contienen toda la lógica de la aplicación. Estas acciones utilizan el modelo y precisan las variables para la vista. Al realizarse una petición web en una aplicación Symfony, la URL define una acción y los parámetros de la petición. La vista es la encargada de originar las páginas que son mostradas como resultado de las acciones, donde se encuentra el layout, que es común para todas las páginas de la aplicación.

La vista en Symfony está conformada por varias partes, preparadas cada una de ellas especialmente para ser fácilmente transformable por la persona que normalmente trabaja con cada aspecto del diseño de las aplicaciones. En el Modelo se encuentran las clases, que son generadas de forma automática según la estructura de la BD. En Symfony, el acceso y la modificación de los datos que se almacenan en la base de datos, se realiza mediante objetos. Propel es el motor generador que se encarga de esta generación automática para construir sus clases, creando la estructura y generando el código de las mismas. A medida que el desarrollo de un proyecto va avanzando, puede ser necesario agregar métodos y propiedades personalizadas en los objetos del modelo, esto trae consigo que se aumenten las tablas o columnas. Asimismo, cada vez que se modifica se deben regenerar las clases del modelo de objetos. Si se añaden los métodos personalizados en las clases que se generan, cada vez que se vuelvan a generar esas clases estos métodos se borrarían. (39)

A continuación se muestra la arquitectura del patrón MVC.

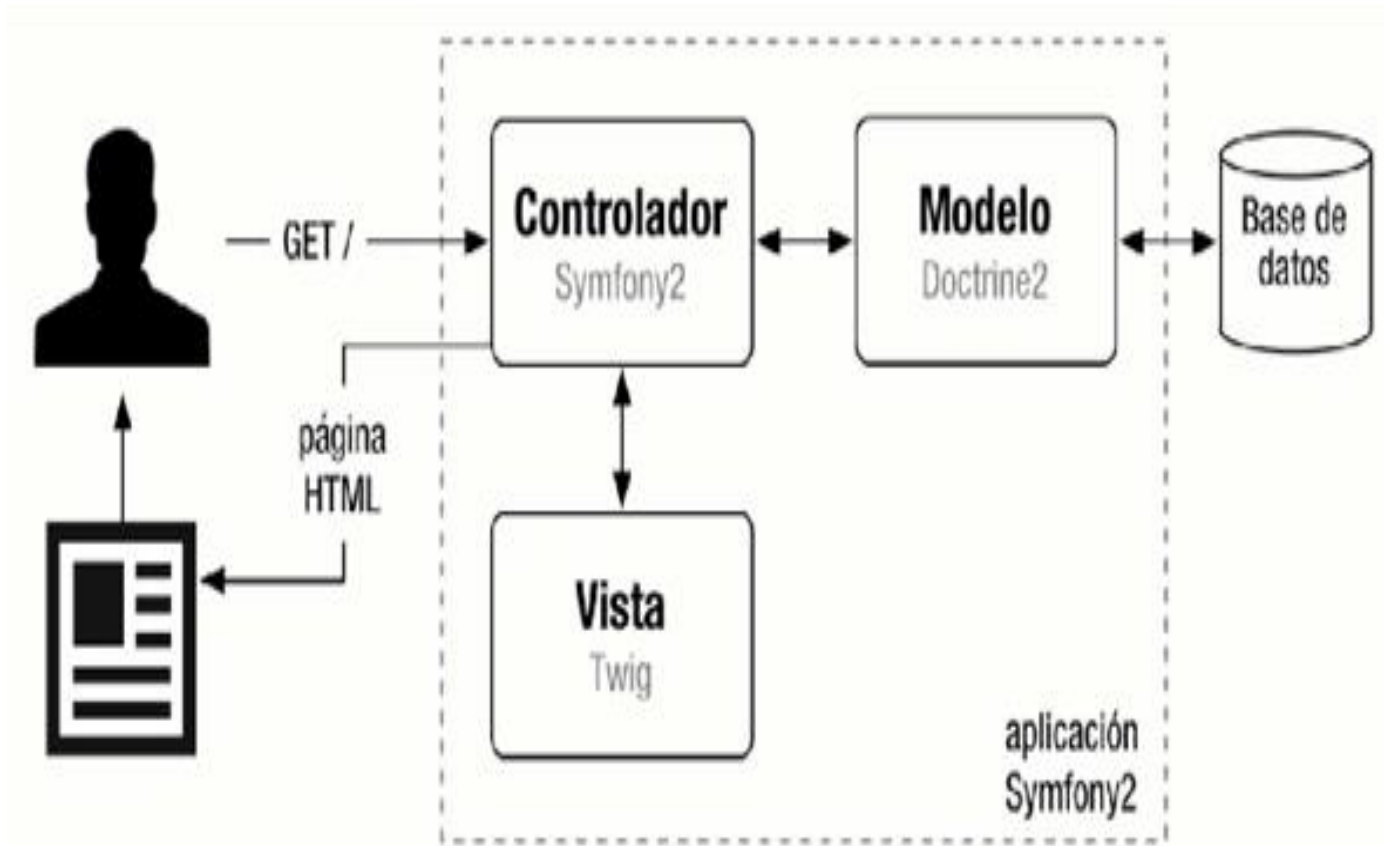


Ilustración 5 Arquitectura del patrón MVC.

En las interacciones gráficas del componente propuesto, se pone de manifiesto en el bundle Exercise donde se encuentra la clase controladora ExerciseController; en el directorio Entity se sitúa la clase entidad que representa el modelo Exercise y en el directorio Resources están contenidas los archivos JavaScript, CSS, las imágenes y las plantillas .twig.

### 2.11 Patrones de diseño utilizados

Para la implementación de Symfony se utilizan varios patrones, situándolos en las capas de Modelo y Control que plantea el patrón arquitectónico MVC. Estos describen formas de solucionar problemas frecuentes en el desarrollo. Estos patrones son: Patrones Generales de Software para Asignar Responsabilidades (del inglés *General Responsibility Assignment Software Patterns* (GRASP)) y Grupo de 4 (del inglés *Gang of Four* (GoF)) (40).

## **Patrones GRASP:**

**Experto:** Es uno de los patrones que más se utiliza cuando se trabaja con Symfony, con la inclusión de la librería Propel para mapear la Base de Datos. Symfony utiliza esta librería para realizar su capa de abstracción en el modelo, encapsular toda la lógica de los datos y generar las clases con todas las funcionalidades comunes de las entidades, las clases de abstracción de datos (Peer del Modelo) poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan y contienen la información necesaria de la tabla que representan. Para el caso de las interacciones gráficas del componente que se van a desarrollar la clase Exercise sería la experta pues esta posee toda la información referente a un ejercicio.

**Creador:** En la clase Actions se encuentran las acciones definidas para el sistema y se ejecutan en cada una de ellas. En dichas acciones se crean los objetos de las clases que representan las entidades, lo que evidencia que la clase Actions es “creador” de dichas entidades. Este patrón asigna la responsabilidad de crear instancias a una determinada clase. En la clase controladora ExerciseController encuentran las acciones definidas para el sistema con el uso de los métodos Actions. En cada una de estas acciones se crean objetos de la clase entidad Exercise, lo que evidencia que la clase controladora es “creadora” de los objetos de la clase entidad.

**Alta Cohesión:** Symfony permite la organización del trabajo en cuanto a la estructura del proyecto y la asignación de responsabilidades con una alta cohesión. Un ejemplo de ello es la clase Actions, la cual está formada por varias funcionalidades que están estrechamente relacionadas, siendo la misma la responsable de definir las acciones para las plantillas y colaborar con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades. Un ejemplo de esto se muestra en la clase ExerciseController, donde están contenidas las operaciones relacionadas a la entidad Exercise.

**Bajo Acoplamiento:** La clase Actions hereda únicamente de sfActions para alcanzar un bajo acoplamiento de clases. Las clases que implementan la lógica del negocio y de acceso a datos se encuentran en el modelo, las cuales no tienen asociaciones con las de la vista o el controlador, lo que proporciona que la dependencia en este caso sea baja. Un ejemplo de este patrón se refleja en la separación que existe entre entidad, controlador y vista de modo que cada una contiene solo la información necesaria lo cual implica que un cambio en una no afecta o afecta lo menos posible a las restantes.

**Controlador:** Todas las peticiones Web son manipuladas por un solo controlador frontal (sfActions), que es el punto de entrada único de toda la aplicación en un entorno determinado. Se encarga de asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase que represente.

En el caso de las interacciones gráficas a realizar del componente la clase controladora es ExerciseController.

## **Patrones GoF**

**Singleton:** Este patrón asegura que una clase tiene solo una instancia y proporciona un punto de acceso global a la misma.

**Decorador:** Este método pertenece a la clase view, padre de todas las vistas, que contienen un decorador para permitir agregar funcionalidades dinámicamente. El archivo nombrado layout.php.twig es el que contiene el Layout de la página. Este archivo, conocido también como plantilla global, guarda el código HTML que es usual en todas las páginas del sistema, para no tener que repetirlo en cada página. El contenido de la plantilla se integra en el layout, o si se mira desde el otro punto de vista, el layout decora la plantilla. Este procedimiento es una implementación del patrón Decorador.

### **2.12 Modelo del diseño**

En el diseño se modela el sistema y se encuentra la forma de que soporte todos los requisitos, incluyendo los requisitos no funcionales y cualquier otra restricción. El modelo de diseño crea un punto de partida para las actividades de implementación subsiguientes. Permite descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo. Da una forma al sistema mientras que intenta preservar la estructura definida por el modelo de análisis (21).

### **2.13 Diagrama de clases del diseño**

El diagrama de clases del diseño muestra la estructura interna del componente. Son los encargados de mostrar las clases participantes, subsistemas y sus relaciones, así como los atributos y operaciones correspondientes a cada clase (40).

A continuación se presenta el diagrama de clase del diseño (DCD) correspondiente al proceso Gestionar ejercicio de hotspotInteraction. Para el estudio de los demás diagramas de clases del diseño (DCD) remitirse a los Anexos.

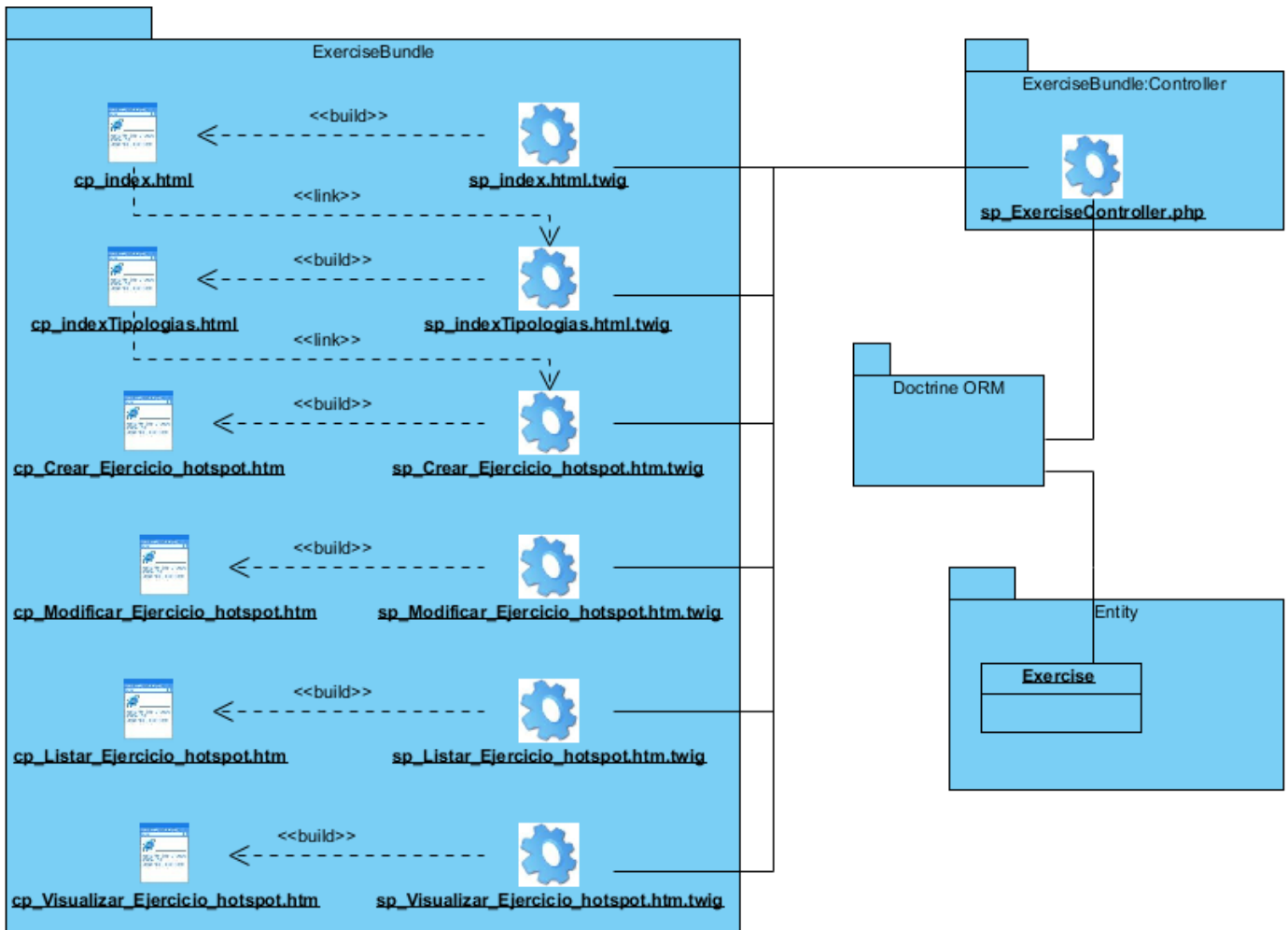


Ilustración 6 Diagrama de clases del diseño. Gestionar ejercicio de hotspotInteraction.

## 2.14 Diagrama de secuencia del diseño

Los diagramas de secuencia muestran las interacciones entre objetos mediante transferencia de mensajes entre objetos o subsistemas. El nombre del mensaje debería indicar una operación del objeto que recibe la invocación o de una interfaz que el objeto proporciona (41).

A continuación se presenta el diagrama de secuencia del diseño (DSD) correspondiente a la HU Crear ejercicio de hotspotInteraction. Para el estudio de los demás diagramas de secuencia del diseño (DSD) remitirse a los Anexos.

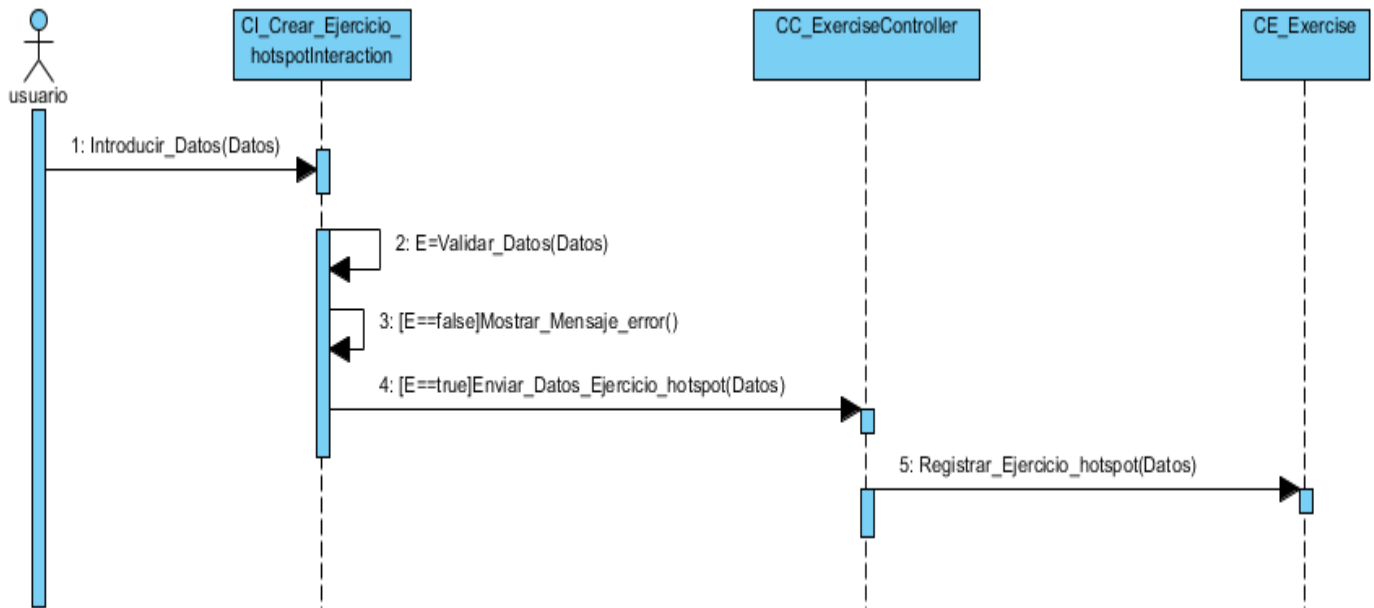


Ilustración 7 Diagrama de secuencia del diseño (DSD). HU Crear ejercicio de hotspotInteraction.

## 2.15 Diseño de la base de datos

El modelo de datos se utiliza para describir la representación lógica y física de la información manejada por el sistema. El diseño de la base de datos es un proceso que requiere extremo cuidado, el principal aspecto a tener en cuenta para la creación de las tablas de una base de datos es la definición correcta de cada uno de sus atributos.

A continuación se presenta el diseño del modelo de base de datos propuesto para la investigación.



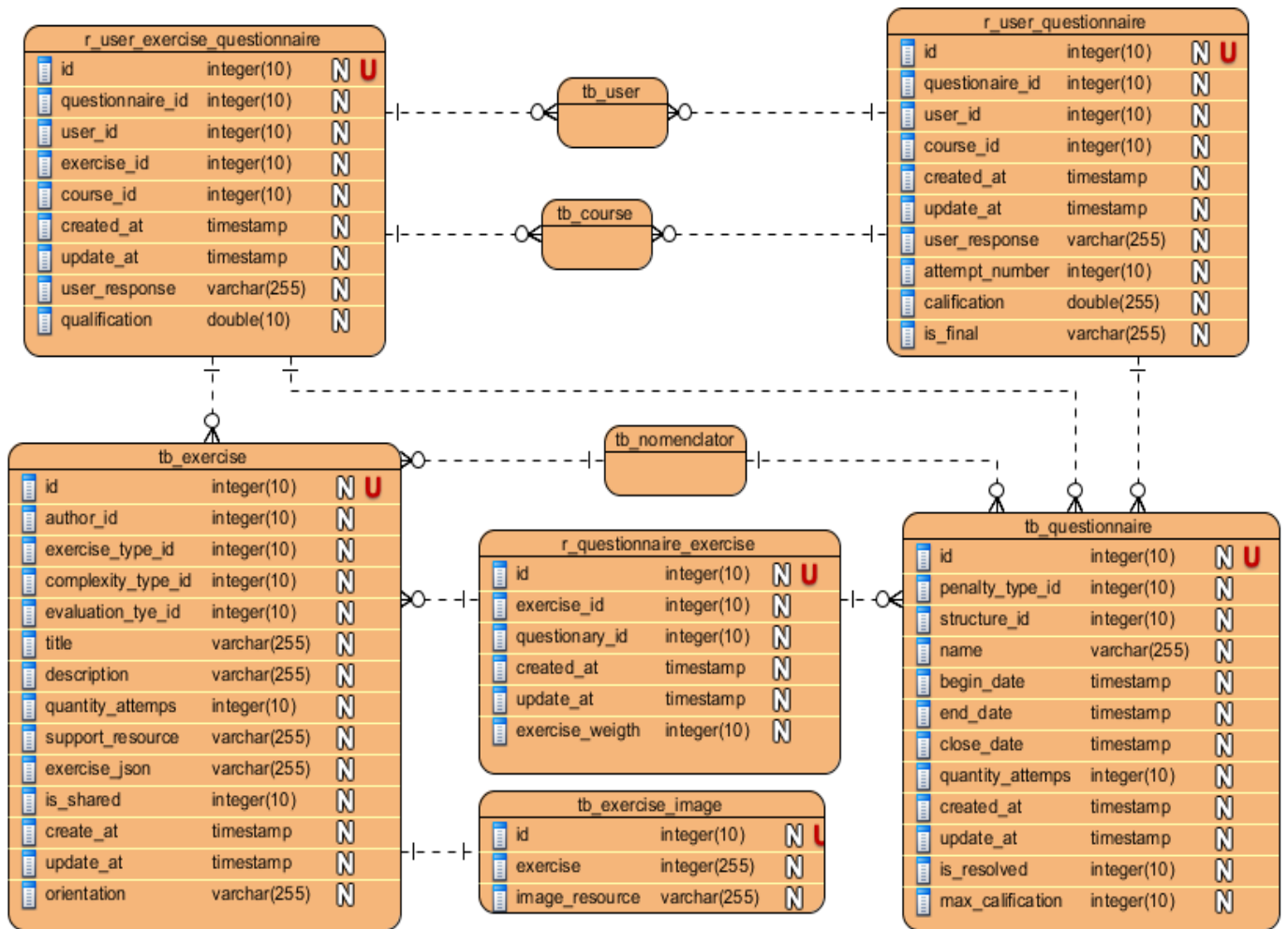


Ilustración 8 Modelo de datos.

### 2.15.1 Descripción de las tablas

A continuación se describe la tabla que contiene información referente al ejercicio.

Tabla 5 Descripción de la tabla tb\_exercise.

<b>tb_exercise</b>		
<b>Atributo</b>	<b>Tipo</b>	<b>Descripción</b>
id	integer	Etiqueta única que identifica al objeto (tb_exercise) en la tabla.

author_id	integer	Almacena el identificador del autor del ejercicio.
exercise_type_id	integer	Almacena el identificador del tipo de ejercicio.
complexity_type_id	integer	Almacena el identificador de la complejidad del tipo de ejercicio.
evaluation_type_id	integer	Almacena el identificador de la evaluación del tipo de ejercicio.
title	varchar	Almacena el título del ejercicio.
description	varchar	Almacena la descripción del ejercicio.
quantity_attempts	integer	Almacena la cantidad de intentos del ejercicio.
support_resource	varchar	Almacena el recurso de apoyo que puede tener el ejercicio.
exercise_json	varchar	Almacena la estructura del ejercicio en formato json.
is_shared	boolean	Almacena la información si es compartido el ejercicio.
created_at	timestamp	Almacena la fecha en que el usuario crea el ejercicio.
update_at	timestamp	Almacena la fecha en que el usuario modifica el ejercicio.

## 2.16 Conclusiones del capítulo

El flujo de trabajo de análisis mediante la elaboración de los diagramas de clases de análisis y de colaboración por cada historia de usuario constituyó la base para la confección del modelo de diseño con sus respectivos diagramas de clases del diseño y de secuencia. Se define como arquitectura MVC por las facilidades que brinda, los patrones utilizados permitieron un mejor desarrollo de la solución y entendimiento del sistema. Además el diseño del modelo de datos ayudó al avance de la investigación, permitiendo

modelar cada una de las clases necesarias. Todos estos artefactos dieron paso a la fase de implementación y prueba.

## **CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA**

### **3.1 Introducción**

La implementación y prueba es la última parte de desarrollo de sistemas luego de haber realizado un análisis y diseño previo. Al implementar un sistema debemos asegurarnos que sea operacional o que funcione e acuerdo a los requerimientos del análisis y permitir que los usuarios puedan operarlos. Durante el proceso de implementación y prueba se deben poner en práctica todas las estrategias posibles para garantizar que el usuario inicial del sistema se encuentre libre de problemas. Para esto se realizan varios tipos de pruebas comprobando que el sistema cumple con las funcionalidades descritas en las historias de usuario y que satisface los requisitos funcionales definidos.

### **3.2 Modelo de implementación**

El modelo de implementación describe como los elementos del diseño se implementan en términos de componentes y cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modulación disponibles en el entorno de implementación y los lenguajes de programación utilizados, así como la dependencia entre los componentes (21).

### **3.3 Diagrama de componente**

El diagrama de componentes muestra como el sistema está dividido en componentes. Es utilizado para modelar la vista estática y dinámica de un sistema, mostrando la organización y las dependencias entre un conjunto de componentes. Muestra los elementos de un diseño de un sistema de software, permite visualizar la estructura de alto nivel del sistema y el comportamiento del servicio que estos componentes proporcionan y usan a través de interfaces. Puede utilizarse para describir un diseño que está implementado en cualquier idioma o estilo, solo es necesario identificar los elementos del diseño que interactúan con los otros elementos del diseño a través de un conjunto restringido de entradas y salidas, los componentes pueden ser de cualquier escala y pueden estar interconectados de cualquier manera (42).

A continuación se muestra el diagrama de componentes del sistema.

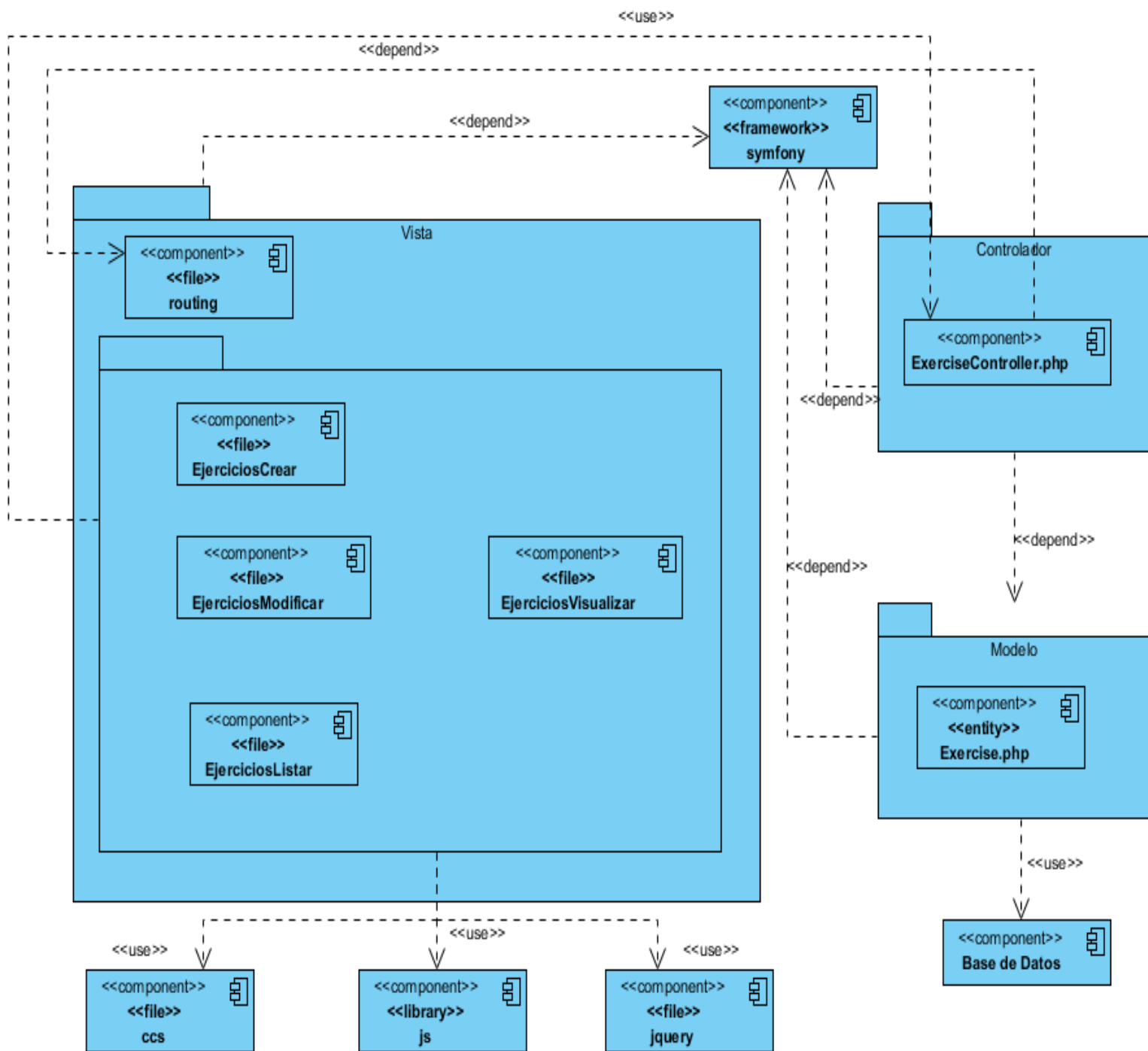


Ilustración 9 Diagrama de componentes

### 3.4 Pruebas de software

Las pruebas de software son las investigaciones empíricas y técnicas cuyo fin es proporcionar información objetiva e independiente sobre la calidad del producto. Esta actividad forma parte del proceso de control de calidad global. Las pruebas son básicamente un conjunto de actividades dentro del desarrollo de software y dependiendo del tipo de pruebas, estas actividades podrán ser implementadas en cualquier momento del proceso de desarrollo (43).

#### 3.4.1 Niveles de prueba

Para evaluar dinámicamente un sistema de software debemos comenzar por los componentes más simples y más pequeños e ir avanzando progresivamente hasta probar todo el software en su conjunto. Para ello se realizan los niveles de pruebas tales como (44):

- **Pruebas Unitarias:** Este tipo de pruebas son ejecutadas normalmente por el equipo de desarrollo, básicamente consisten en la ejecución de actividades que le permitan verificar al desarrollador que los componentes unitarios están codificados bajo condiciones de robustez, esto es, soportando el ingreso de datos erróneos o inesperados y demostrando así la capacidad de tratar errores de manera controlada. Adicionalmente, las pruebas sobre componentes unitarios, suelen denominarse pruebas de módulos o pruebas de clases, siendo la convención definida por el lenguaje de programación la que influye en el término a utilizar. Por último, es importante que toda la funcionalidad de cada componente unitario sea cubierta, por al menos, dos casos de prueba, los cuales deben centrarse en probar al menos una funcionalidad positiva y una negativa.
- **Pruebas de Integración:** Este tipo de pruebas son ejecutadas por el equipo de desarrollo y consisten en la comprobación de que elementos del software que interactúan entre sí, funcionan de manera correcta.
- **Pruebas de Sistema:** Este tipo de pruebas deben ser ejecutadas idealmente por un equipo de pruebas ajeno al equipo de desarrollo, una buena práctica en este punto corresponde a la tercerización de esta responsabilidad. La obligación de este equipo, consiste en la ejecución de actividades de prueba en donde se debe verificar que la funcionalidad total de un sistema fue implementada de acuerdo a los documentos de especificación definidos en el proyecto. Los casos de prueba a diseñar en este nivel de pruebas, deben cubrir los aspectos funcionales y no funcionales del sistema. Para el diseño de los casos de prueba en este nivel, el equipo debe utilizar como bases de prueba entregables tales como: requerimientos iniciales, casos de uso, historias de usuario, diseños, manuales técnicos y de usuario final, etc. Por último, es importante

que los tipos de pruebas ejecutados en este nivel se desplieguen en un ambiente de pruebas, ambiente de pre-producción cuya infraestructura y arquitectura sea similar al ambiente de producción, evitando en todos los casos utilizar el ambiente real del cliente, debido principalmente, a que pueda ocasionar fallos en los servidores, lo que ocasionaría indisponibilidad en otros servicios alojados en este ambiente.

- **Pruebas de Aceptación:** Independientemente de que se haya tercerizado el proceso de pruebas y así la firma responsable de estas actividades haya emitido un certificado de calidad sobre el sistema objeto de prueba, es indispensable, que el cliente designe a personal que haga parte de los procesos de negocio para la ejecución de pruebas de aceptación, es incluso recomendable, que los usuarios finales que participen en este proceso, sean independientes al personal que apoyó el proceso de desarrollo. Cuando las pruebas de aceptación son ejecutadas en instalaciones o ambientes proporcionados por la firma desarrolladora se les denominan pruebas Alpha, cuando son ejecutadas desde la infraestructura del cliente se les denomina pruebas Beta. En los casos en que las pruebas de aceptación del producto se hayan ejecutado en el ambiente del proveedor, el aplicativo no podrá salir a producción, sin que se hayan ejecutados las respectivas pruebas Beta en el ambiente del cliente, de lo anterior es importante concluir, que las pruebas Alpha son opcionales, pero las pruebas Beta son obligatorias.

### **3.4.2 Tipos de pruebas**

En cada uno de estos niveles de prueba, se podrán ejecutar diferentes tipos de prueba tales como: pruebas funcionales, no funcionales, de arquitectura y asociadas al cambio de los productos.

Las pruebas funcionales requieren el uso y la aplicación de una técnica de caja negra, siguiendo esta técnica se confeccionan los casos de pruebas para probar las condiciones de entrada tanto válidas como inválidas.

En la presente investigación se escogió como prueba a realizar la prueba de funcionalidad con el objetivo de verificar el cumplimiento de cada uno de los requisitos funcionales propuestos para el desarrollo del sistema.

### **3.4.3 Métodos de prueba**

Para el desarrollo de un buen sistema solo hay que tratar de diseñar pruebas que tengan la mayor probabilidad de encontrar el mayor número de errores con la mínima cantidad de esfuerzo y de tiempo para ello cualquier producto de ingeniería se puede probar de dos formas (45).

**Pruebas de caja negra:** Realizar pruebas de forma que se compruebe que cada función es operativa, que la entrada se acepta de forma adecuada y que se produce una salida correcta. Estas pruebas se llevan a cabo sobre la interfaz del software, obviando el comportamiento interno y la estructura del programa. Los casos de prueba de la caja negra pretenden demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma correcta.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

Las pruebas de caja negra pretenden encontrar estos tipos de errores:

- Funciones incorrectas o ausentes.
- Errores en la interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Los tipos de caja negra son:

- Prueba de partición equivalente.
- Prueba de análisis de valores límites.
- Prueba de grafo de causa efecto.
- Prueba de adivinación/suposición de errores.

**Pruebas de caja blanca:** Desarrollar pruebas de forma que se asegure que la operación interna se ajusta a las especificaciones, y que todos los componentes internos se han probado de forma adecuada. En esta prueba se realiza un examen minucioso de los detalles procedimentales, comprobando los caminos lógicos del programa, comprobando los bucles y condiciones, y examinado el estado del programa en varios puntos. La prueba de la caja blanca es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivar los casos de prueba.

Las pruebas de caja blanca intentan garantizar que:

- Se ejecutan al menos una vez todos los caminos independientes de cada módulo.
- Se utilizan las decisiones en su parte verdadera y en su parte falsa.



- Se ejecuten todos los bucles en sus límites.
- Se utilizan todas las estructuras de datos internas.

En el presente trabajo se realizaron pruebas de caja negra utilizando el método de particiones equivalentes ya que es uno de los más efectivos para examinar los valores válidos e inválidos de las entradas existentes en el sistema. Este método divide el dominio de entrada de un programa en un número finito de variables de equivalencia (46).

### **3.5 Diseño de casos de prueba**

Los casos de prueba son un conjunto de datos de entrada que permiten obtener un valor de salida del sistema, tienen como objetivo hallar cuál es el subconjunto de todos los casos de prueba que tiene una mayor probabilidad de detectar errores.

Los casos de prueba de caja negra pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta, así como que la integridad de la información externa se mantiene (45).

A continuación se presenta el caso de prueba (CP) del proceso Gestionar ejercicio de hotspotInteraction. Para el estudio de los demás CP remitirse a los Anexos.

<b>Diseño de Casos de Prueba: Gestionar ejercicio de hotspotInteraction</b>
---

SC 1 Crear ejercicio de hotspotInteraction

Tabla 6 Gestionar ejercicio de hotspotInteraction. SC 1 Crear ejercicio de hotspotInteraction.

Escenario	Descripción	Título	Descripción	Recurso de apoyo	Compartir	Complejidad	Evaluación	Orientación	Imagen	Respuesta del sistema	Flujo central
EC 1.1 Incluir ejercicio	Selecciona la opción de crear un ejercicio.									El sistema brinda la posibilidad de seleccionar los tipos de ejercicios que desea crear como: -Puntos calientes -Relacionar puntos. -Asociación gráfico. -Orden gráfico -Gráfico partido. -Posición del objeto.	Exercises/Crear ejercicio/new

EC 1.2	Incluir tipo de ejercicio	Selecciona el tipo de ejercicio Puntos calientes									Permite introducir o seleccionar los siguientes datos: -Título -Descripción -Recurso de apoyo -Compartir -Complejidad -Evaluación -Orientación -Imagen Y permite seleccionar la opción -Guardar -Cancelar	Exercises/Crear ejercicio/new/Puntos calientes
EC 1.3	Introducir los datos del ejercicio de Puntos calientes	Introduce los datos y selecciona la opción Guardar	V	V	N/A	N/A	V	V	V	V	Valida los datos. Registra los datos del ejercicio de Puntos calientes.	Exercises/Crear ejercicio/Puntos calientes/Guardar
EC 1.4	Cancelar creación del ejercicio Puntos calientes	Selecciona la opción Cancelar									Regresa al EC 1.1	Exercises/Crear ejercicio/Puntos calientes/Cancelar

EC 1.5 Datos incompletos	Existen datos incompletos	I	V	N/A	N/A	V	V	V	V	Muestra el mensaje de información "Este valor no debería estar vacío". Muestra un indicador sobre los campos obligatorios. Regresa al EC 1.3	Exercises/Crear ejercicio/Puntos calientes/Guardar
		V	I	N/A	N/A	V	V	V	V		
		V	V	N/A	N/A	I	V	V	V		
		V	V	N/A	N/A	V	I	V	V		
		V	V	N/A	N/A	V	V	I	V		
		V	V	N/A	N/A	V	V	V	I		

SC 2 Listar ejercicio de hotspotInteraction

Tabla 7 Gestionar ejercicio de hotspotInteraction. SC 2 Listar ejercicio de hotspotInteraction.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Listar ejercicios de puntos calientes existentes.	El sistema te muestra después de haberte autenticado un listado de todos los ejercicios de puntos calientes o vacíos en caso de que no se halla creado ningún ejercicio.	El sistema lista los ejercicios de puntos calientes existentes. El listado se debe mostrar en una vista general con los siguientes datos: - Título - Tipología - Complejidad -Evaluación Permite organizar los ejercicios según los datos anteriores. Además puedes realizar varias acciones como: -Editar -Ver	Exercises/list/Listado de ejercicios

SC 3 Modificar ejercicio de hotspotInteraction

Tabla 8 Gestionar ejercicio de hotspotInteraction. SC 3 Modificar ejercicio de hotspotInteraction.

Escenario	Descripción	Título	Descripción	Recurso de apoyo	Compartir	Complejidad	Evaluación	Orientación	Imagen	Respuesta del sistema	Flujo central
EC 3.1 Modificar datos del ejercicio de Puntos calientes	Se selecciona la opción editar los datos del ejercicio de Puntos calientes.									Muestra el ejercicio permitiendo modificar los siguientes: -Título -Descripción -Recurso de apoyo -Compartir -Complejidad -Evaluación -Orientación -Imagen Y permite seleccionar la opción: -Editar -Cancelar	Exercises/Lista ejercicios/Editar
EC 3.2 Modificar el ejercicio de	Modifica los datos necesarios y	V	V	N/A	N/A	V	V	V	V	Valida los datos. Actualiza los datos del	Exercises/Lista ejercicios/Editar/E

Puntos calientes	selecciona la opción Guardar.										ejercicio de Puntos calientes.		
EC 3.3 Cancelar la edición del ejercicio de Puntos calientes	Selecciona la opción Cancelar										Regresa al listado de ejercicios.	Exercises/Lista ejercicios/Editar/Editar/Cancelar	
EC 3.4 Datos incompletos	Existen datos incompletos	I	V	N/A	N/A	V	V	V	V	Muestra el mensaje de información "Este valor no debería estar vacío". Muestra un indicador sobre los campos obligatorios. Regresa al EC 1.2	Exercises/Lista ejercicios/Editar/E		
		V	I	N/A	N/A	V	V	V	V				
		V	V	N/A	N/A	I	V	V	V				
		V	V	N/A	N/A	V	I	V	V				
		V	V	N/A	N/A	V	V	I	V				
		V	V	N/A	N/A	V	V	V	I				

SC 4 Visualizar ejercicio de hotspotInteraction

Tabla 9 Gestionar ejercicio de hotspotInteraction. SC 4 Visualizar ejercicio de hotspotInteraction.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 4.1 Ver datos de un ejercicio de Puntos calientes	Seleccionar el ejercicio de Puntos calientes.	El sistema muestra los siguientes datos del ejercicio de Puntos calientes:	Exercises/Lista de ejercicios/

		<ul style="list-style-type: none"> <li>-Título</li> <li>-Descripción</li> <li>-Recurso de apoyo</li> <li>-Compartir</li> <li>-Complejidad</li> <li>-Evaluación</li> <li>-Orientación</li> <li>-Imagen</li> </ul> <p>Y permite seleccionar la opción:</p> <ul style="list-style-type: none"> <li>-Previsualizar</li> <li>-Cancelar</li> </ul>		
EC 4.2 Salir de ver datos del ejercicio de Puntos calientes	Selecciona la opción Cancelar.	Muestra el listado de todos los ejercicios creados.	Exercises/Lista ejercicios/Ver/Cancelar	

Descripción de las variables

*Tabla 10 Gestionar ejercicio de hotspot/Interaction. Descripción de las variables.*

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Título	Campo de texto	No	Nombre del ejercicio. Admite caracteres alfanuméricos.
2	Descripción	Campo de texto	No	Descripción del ejercicio. Admite cualquier carácter.
3	Recurso de apoyo	Campo de selección	Sí	Permite adjuntar cualquier tipo de archivo como recurso auxiliar.
4	Compartir	Campo de selección	Sí	Te da la opción de compartir o no.

				ejercicio que se creando.
5	Complejidad	Lista desplegable	No	Complejidad ejercicio que es dada en muy fácil, difícil y difícil.
6	Evaluación	Campo de selección	No	Permite otorgar evaluación estudiante ya automática o por el profesor.
7	Orientación	Campo de texto	No	Orden del ejercicio Admite cualquier carácter.
8	Imagen	Campo de selección	No	Imagen de fondo obligatoria que utiliza en el ejercicio para dar solución. Permite archivos de jpg/jpeg/png.

### 3.6 Resultados obtenidos

Durante el desarrollo de las interacciones gráficas del componente de autoevaluación se realizaron pruebas de unidad para comprobar el correcto funcionamiento del código de acuerdo con las especificaciones y que el módulo lógico es válido, este método se focaliza en ejecutar cada módulo o unidad mínima a ser probada lo que provee un mejor modo de manejar la integración de las unidades en componentes mayores. Estas pruebas se realizaron por desarrolladores durante la implementación y fueron haciéndose a medida que la solución iba desarrollándose, por lo que estas no se planificaron ni se registraron sus resultados.



Para comprobar la eficiencia de las interacciones desarrolladas del componente se realizaron varias iteraciones. Se realizaron los niveles de prueba, tipos de prueba y métodos de prueba con la utilización de técnicas definidas por estos métodos como se expuso anteriormente, todo esto realizado para cada una de las iteraciones.

En los resultados obtenidos luego de realizar cada una de estas pruebas se determinaron varias no conformidades (NC), considerándose como NC el incumplimiento de algún requisito definido para el desarrollo de sistema. Estas NC se clasifican de acuerdo al nivel de importancia en: (47)

- Significativas (Alta): Son aquellas que afectan la calidad del producto o servicio de manera visible, impidiendo o no el cumplimiento de algún requisito.
- No Significativas (Media): Son aquellas que resultan menos visibles, que no atentan el cumplimiento de algún requisito.
- Recomendaciones (Baja): Son aquellas que quedan en función de la apreciación del probador para oportunidades de mejoras del producto o servicio.

Iteraciones	Cantidad de caso de prueba	No conformidades			
		Alta	Media	Baja	Total
Primera	24	6	6	-	12
Segunda	24	4	3	1	8
Tercera	24	-	-	-	0

Las no conformidades encontradas en la primera iteración estuvieron relacionadas con guardar la imagen con la que se iba a trabajar como fondo del ejercicio en las interacciones gráficas y validaciones de los campos incompletos e incorrectos. En la segunda iteración las no conformidades estaban conformadas por los Previsualizar de los diferentes tipos de interacciones gráficas. Ya en una tercera iteración es sistema se encontró libre de errores para gran satisfacción del cliente y alto nivel de calidad.

### **3.7 Conclusiones del capítulo**

En este capítulo se generaron los diferentes artefactos que conforman el modelo de implementación, el diagrama de componentes desarrollado permitió la representación lógica de los elementos del diseño describiendo los factores físicos existentes en el sistema y sus relaciones. La realización de pruebas unitarias a las interacciones gráficas desarrolladas brindó en tres iteraciones eficiencia al código implementado. A su vez, la realización de las pruebas de caja negra evidenció que las interacciones gráficas desarrolladas satisfacen los requerimientos definidos para solución deseada.

## CONCLUSIONES

Después de desarrollada la presente investigación se arribaron a las siguientes conclusiones.

- El estudio realizado permitió definir las características fundamentales de la especificación IMS-QTI para su correcto uso en el desarrollo de las interacciones gráficas del componente, siendo esta especificación el estándar más utilizado a nivel mundial para garantizar la interoperabilidad entre aplicaciones *e-learning*.
- Se cumplieron todos los objetivos propuestos, obteniéndose un componente capaz de gestionar ejercicios incluyendo las interacciones gráficas, así como posibilitar el intercambio de estos con otros sistemas similares soportado sobre la arquitectura basada en componentes.
- Las pruebas realizadas a las interacciones gráficas desarrolladas evidenciaron un alto nivel de calidad del componente de actividades de autoevaluación soportado sobre la arquitectura basada en componentes para la plataforma Zera 2.0.

## **RECOMENDACIONES**

- Dada la importancia del componente para los LMS modernos se recomienda agregar al sistema una funcionalidad capaz de calcular el grado de aprendizaje de parte de los usuarios por cada tipología, con el objetivo de crear formularios más adaptados a cada grupo de usuarios.
- Por el impacto que han causado en la sociedad los teléfonos inteligentes, se recomienda la implementación de una aplicación para teléfonos con sistemas operativos Android capaz de crear ejercicios de forma offline y una vez conectados a la red poder subirlos a la plataforma.

## BIBLIOGRAFÍA

1. **Ortí, Consuelo Belloch.** *Las Tecnologías de la Información y la Comunicación(T.I.C)*. Valencia : s.n.
2. *Análisis Comparativo de las Plataformas Educativas Virtuales Moodle y Dokeos.* **Lizárraga, R. E., Colado, A. Z., and Garzón, J. F. P.** 2013, Revista Iberoamericana para la Investigación y el Desarrollo Educativo., pág. 10.
3. **Mayor, Alicia Cañellas.** LMS y LCMS Funcionalidades y beneficios. [En línea] 27 de Enero de 2014. [Citado el: 23 de Febrero de 2016.] <http://www.centrocp.com/lms-y-lcms-funcionalidades-y-beneficios/>.
4. **Videgaray, Maricarmen Gonzalez.** Evaluación de la reacción de alumnos y docentes en un modelo mixto de aprendizaje para educación superior. 2007.
5. *E-learning: Enseñar y Aprender en Espacios Virtuales.* **Segura, Manuel Area Moreira y Jordi Adell.** 1-29, 2009.
6. **Peñalvo, Francisco José García.** Estado actual de los sistemas e-learning. Teoría de la Educación: Educación y Cultura en la Sociedad de la Información. [En línea] [Citado el: 21 de 1 de 2016.] [http://campus.usal.es/~teoriaeducacion/rev\\_numero\\_06\\_2/n6\\_02\\_art\\_garcia\\_penalvo.htm..](http://campus.usal.es/~teoriaeducacion/rev_numero_06_2/n6_02_art_garcia_penalvo.htm..)
7. *El proceso de retroalimentación en la evaluación. Un aporte al aprendizaje significativo de los estudiantes universitarios.* **Vázquez, C., Cavallo, M., Sepliarsky, P., and Escobar, M. E.** 2010.
8. **Alvarado, M. A. G.** *Retroalimentación en educación en línea: una estrategia para la construcción del conocimiento.* 2014.
9. *Proyecto de evaluación de plataformas de tele formación para su implantación en el ámbito universitario.* **Carrera, D. R.** 0-124, 2003.
10. **Baltasar Fernández Manjón, Pablo Moreno Ger, José Luis Sierra Rodríguez, Iván Martínez Ortíz.** *Uso de estándares aplicados a Tic. Facultad de Informática de la Universidad Complutense de Madrid: s.n.* 2006.
11. *Estándares de e-learning: guía de consulta.* **González, J. R. H. and Marín, R. H.** 2010.
12. **Universidad de las Ciencias Informáticas, Centro FORTES.** *Herramienta de autor CRODA 2.0.* Cuba, Habana : s.n.
13. **Labañino Rizzo, César y del Toro Rodríguez, Mario.** *Plataforma Educativa ZERA 1.0.* 2001.

14. **Empresa de Sudamérica, Entornos Educativos.** Entornos Educativos. [En línea] [Citado el: 12 de 6 de 2016.] <http://www.entornos.com.ar/moodle>.
15. **Ministerio, de educación y ciencia.** Educación. IMS QTI en Moodle. [En línea] [Citado el: 12 de 6 de 2016.] <http://ares.cnice.mec.es/informes/16/contenido/52.htm>.
16. **Villar, Gabriela.** Plataformas virtuales. Búsqueda de información y navegación en distintas Plataformas. [En línea] 2007. [Citado el: 12 de 6 de 2016.] <http://www.unsam.edu.ar/profesores/gabrielavillar/Plataformas%20virtuales%20Gabriela%20Villar.doc>.
17. **Rutherford, Ricardo Enrique Krause.** *Mobile QTI: Una herramienta para interacciones gráficas en el aula, basado en el estándar IMS QTI, y su uso en dispositivos móviles en pantallas táctiles.* Valdivia-Chile : s.n., 2013.
18. *Metodologías tradicionales vs metodologías ágiles.* **Figuroa, R. G., Cabrera, A. A., and Solis, C. J.** 1-9, 2015.
19. **José H. Canós, Patricio Letelier y M. Carmen Penadés.** *Metodologías Ágiles en el Desarrollo de Software.* Valencia : s.n.
20. **Sánchez, Tamara Rodríguez.** *Metodología de desarrollo para la Actividad productiva de la UCI.* 2016.
21. **James Rumbaugh, Ivar Jacobson, Grady Booch, Addison Wesley.** *La referencia definitiva de UML escrita por sus creadores.* 2000.
22. **G, Susana.** Visual Paradigm. [En línea] 18 de 9 de 2012. [Citado el: 2 de 2 de 2016.] [http://www.ecured.cu/index.php/Visual\\_Paradigm..](http://www.ecured.cu/index.php/Visual_Paradigm..)
23. **ALEGSA, Corp.** Diccionario de Informática. Definición de SGBD. [En línea] [Citado el: 4 de 2 de 2016.] <http://www.alegsa.com.ar/Dic/sghbd.php..>
24. **Suehring, S.** *MySQL Bible.* Wiley Publishing, Inc., . s.l. : New York, 2002.
25. **Martínez, Rafael.** Sobre PostgreSQL. [En línea] 2 de 10 de 2010. [Citado el: 4 de 2 de 2016.] <http://www.postgresql.org.es/>.
26. **Foundation, The Apache Software.** Servidor Apache. [En línea] 25 de 11 de 2013. [Citado el: 4 de 2 de 2016.] <http://www.abcdatos.com/webmasters/programa/servidor-apache.html>.
27. **Corporation, Oracle.** NetBeans. [En línea] 2013. [Citado el: 4 de 2 de 2016.]

28. **Torrealday, Gustavo F.** Torrealday Informática. [En línea] [Citado el: 4 de 2 de 2016.] <http://www.torrealday.com.ar/articulos/articulo006.htm>.
29. **Perez, J. E.** *Introducción a CSS*. 2008.
30. **Cantón, Alejandro Castillo.** *Manual de HTML5 en español*.
31. **David Bruant, Vinícius Albuquerque, Nickolay.** Mozilla Developer Network. [En línea] 16 de 11 de 2013. [Citado el: 4 de 2 de 2016.] [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/About#JavaScript\\_history](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/About#JavaScript_history).
32. **Eguiluz, Javier.** Symfony. [En línea] 2 de 3 de 2012. [Citado el: 5 de 2 de 2016.] <http://symfony.es/noticias/2012/03/02/publicada-la-version-estable-de-php-5-4..>
33. **Alvarez, M. A.** *Manual de jQuery*. 2009.
34. **Mark Otto, Jacob Thornton, Traductor: Javier Eguiluz.** Bootstrap 3, manual oficial. [En línea] 2015. [Citado el: 5 de 2 de 2016.] [http://librosweb.es/bootstrap\\_3/index.html](http://librosweb.es/bootstrap_3/index.html).
35. **Eguiluz, Javier.** *Desarrollo web ágil con Symfony2*. 2013.
36. **Eriksson, H.-E. and Penker, M.** *Business Modeling with UML: Business Patterns at Work*. 2000.
37. **Quijano, Juan.** Genobeta, historias de usuario. [En línea] 28 de 2 de 2012. [Citado el: 15 de 3 de 2015.] <http://www.genbetadev.com/metodologias-de-programacion/historias-de-usuario-una-forma-natural-de-analisis-funcional>.
38. **Cillero, Manuel.** Diagrama de colaboración . [En línea] 2009. [Citado el: 16 de 3 de 2016.] <https://manuel.cillero.es/doc/metrica-3/tecnicas/diagrama-de-colaboracion/> .
39. **Eguiluz, Javier.** Maestros dela web. El framework Symfony, una introducción práctica (I parte). [En línea] 6 de 9 de 2007. [Citado el: 16 de 3 de 2016.] <http://www.maestrosdelweb.com/editorial/el-framework-symfony-una-introduccion-practica-i-parte/>.
40. **Larman, Craig.** *UML y Patrones 2da edición. Una introducción al análisis y diseño orientado a objetos y al proceso unificado*.
41. **Aramúndiz, Silvia.** *Diagrama de secuencia. Comportamiento de los sistemas*.
42. **Alva, Eduardo Rivera.** Arquitectura de software 2. Diagramas de Componentes y Despliegue. [En línea] [Citado el: 16 de 3 de 2016.] <http://es.scribd.com/doc/7884665/Arquitectura-de-Software-II-Diagrama-de-Componentes-y-Despliegue..>

43. **Calvo, Rafael.** Eleven Paths. QA: Pruebas para asegurar la calidad del producto software (I). [En línea] [Citado el: 16 de 3 de 2016.] <http://blog.elevenpaths.com/2014/09/qa-pruebas-para-asegurar-la-calidad-del.html>.
44. **Sánchez, Javier Zapata.** Pruebas del software, niveles de prueba del software. [En línea] [Citado el: 15 de 4 de 2016.] <https://pruebasdelsoftware.wordpress.com/>.
45. **Torres, M.** Técnicas de pruebas. [En línea] [Citado el: 20 de 4 de 2016.] <http://indalog.ual.es/mtorres/LP/Prueba.pdf>.
46. **Pressman, Roger S.** *“Ingeniería del Software. Un enfoque práctico”*, sexta edición. España : s.n., 2008. McGraw-Hill/Interamericana de España.
47. **Mares, Saúl Amézquita.** Gestión de No Conformidades. [En línea] 2007. [Citado el: 23 de 5 de 2016.] <http://www.mex.opsoms.org/contenido/tuberculosis/cdtaller/presentaciones/M%C3%B3dulo%20%20>.
48. **Macías, Diego.** *Plataformas de enseñanza virtual libres y sus características de extensión: Desarrollo de un bloque para la gestión de tutorías en Moodle*. Alcalá de Henares : Universidad de Alcalá : s.n., 2010.
49. **EcuRed. Plataformas Educativas.** [En línea] [http://www.ecured.cu/Plataformas\\_Educativas](http://www.ecured.cu/Plataformas_Educativas).



## GLOSARIO

**LMS:** Son aplicaciones web que integran un conjunto de herramientas para el proceso de enseñanza-aprendizaje en línea, permitiendo una enseñanza no presencial (*e-learning*) y/o enseñanza mixta (*blended-learning*).

**E-learning:** Es una modalidad de enseñanza y aprendizaje que puede representar todo o una parte del modelo educativo en el que se aplica, que explota los medios y dispositivos electrónicos para facilitar el acceso, la evolución y la mejora de la calidad de la educación y la formación.

**IMS:** Compañía global, sin fines de lucro, que se esfuerza por lograr un crecimiento y mayor impacto de las tecnologías de aprendizaje en la educación y los sectores empresariales de aprendizaje en todo el mundo. Su principal actividad es el desarrollo de estándares de interoperabilidad y las normas de adopción de prácticas para el aprendizaje distribuido, algunos de los cuales como IMS-QTI y Content Packaging son los más utilizados.

**IMS-QTI:** Especificación la cual proporciona un lenguaje XML para representar preguntas y exámenes de forma que permita la interoperabilidad del contenido con los sistemas gestores de aprendizaje.

**HTTP:** Protocolo de comunicaciones que permite la transferencia de documentos de lenguaje de marcas de hipertexto (HTML) desde servidores web a navegadores web.

**XML:** Es diseñado especialmente para los documentos de la web. Permite que los diseñadores creen sus propias etiquetas, permitiendo la definición, transmisión, validación e interpretación de datos entre aplicaciones y entre organizaciones.

**HTML:** Lenguaje de marcado de hipertexto utilizado para describir la estructura y el contenido de una página web.

**Java:** Lenguaje de programación, tiene como característica que es orientado a objetos.

**CSS:** Contienen un conjunto de etiquetas que definen el formato que se aplicará al contenido de las páginas web. Se llama cascada porque una hoja puede heredar los formatos definidos en otra hoja de forma de que no hace falta que vuelva a definirlos. Estas hojas permiten la separación entre el contenido y la presentación de un sitio web.

**JavaScript:** Lenguaje de programación interpretado, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java. Se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

**PHP:** Es un lenguaje multiplataforma, script (no se compila para conseguir códigos máquina sino que existe un intérprete que lee el código y se encarga de ejecutar las instrucciones que contiene éste código) para el desarrollo de páginas web dinámicas del lado del servidor, cuyos fragmentos de código se intercalan fácilmente en páginas HTML, debido a esto y a que es de código.

