Universidad de las Ciencias Informáticas
Facultad 10

Desarrollo de una herramienta de Autor Web para el diseño flexible de objetos de aprendizaje.

Trabajo diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores: *Annis Parra Rodríguez*
Rafael Lorente Salgueiro

Tutoras: *Ing. Dunia María Colomé*
Msc. Daymy Tamayo Ávila

Co-tutora: *Roxana Cañizares González*

Ciudad de la Habana, Junio 2009
"Año del 50 aniversario del triunfo de la Revolución"

Cada trecho recorrido enriquece al peregrino y lo acerca un poco más a hacer realidad sus sueños.

-Paulo Cohelo

Declaración de autoría

Por este medio declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmamos la presente a los __ días del mes de __ del 2009.

Autores

Annis Parra Rodríguez	Rafael Lorente Salgueiro

Tutoras

Ing. Dunia María Colomé Cedeño	MSc. Daymy Tamayo Ávila

Co-tutora

Ing. Roxana Cañizares González

Agradecimientos

De Annis:

A mis padres por su apoyo incondicional durante todos estos años.

A nuestras tutoras Dunia, Roxana y Daymy que nos apoyaron aun estando en momentos difícil del desarrollo de este trabajo, sin ustedes esto no se hubiese logrado.

A todos los amigos que me brindaron su apoyo en estos maravillosos años de alegrías y tristezas, en especial a Ale y Chuchi. A Jorge por ayudarnos tanto, gracias por siempre estar preocupado. Un agradecimiento especial para mi hermano quien ha sido la fuente de inspiración en toda mi carrera estudiantil.

Agradecimientos

De Rafael:

A todos los amigos que me brindaron su apoyo. Un agradecimiento especial para mi madre quien ha sido la fuente de inspiración en toda mi carrera estudiantil.

Dedicatoria

Annis

A mis padres, por ser motivo de inspiración de mi trabajo como estudiante, y porque sin ellos no haría podido hacer realidad mi sueño. A mi hermano por ser la meta que deseo algún día alcanzar.

Rafael

A mi madre por todo su amor y entrega.

Resumen

La educación a distancia ha ocupado un lugar significativo a nivel mundial desde que se comenzaron a utilizar las Tecnologías de la Informática y las Comunicaciones (TICs) en este proceso, de ésta forma se dio inicio al concepto de e-learning, se hizo visible la necesidad de utilizar herramientas capaces de editar y crear las unidades básicas de ése tipo de enseñanza, conocidas como Objetos de Aprendizaje (OA). En el presente trabajo se da continuidad al desarrollo de la Herramienta Autor para la creación de contenidos reutilizables, con basamento en las especificaciones SCORM 1.2 (Sharable Content Object Reference Model). El objetivo de éste trabajo es proponer una aplicación Web interoperable capaz de diseñar y crear OA, además de poseer características que le permitan interactuar con las demás aplicaciones que conforman una plataforma de aprendizaje a distancia.

Para su implementación se realizó un estudio de la todas las Herramientas de Autor utilizadas para la edición y creación de los OA, que ayudarán a remozar dicha aplicación.

Palabras Claves: Objetos de Aprendizaje, Interoperabilidad, SCORM, E-learning, Herramienta de Autor.

Índice

Índice de Figuras

Índice de Tablas

INTRODUCCIÓN

El siglo XX ha traido consigo una revolución tecnológica en lo que respecta a las Tecnologías de la Información y las Comunicaciones (TIC), las cuales han propiciado una nueva modalidad educativa, la educación a distancia; en la que no es necesaria la presencia física del estudiante para adquirir los conocimientos.

La educación a distancia representa una alternativa para enfrentar la compleja y costosa tarea de impartir cursos de formación presencial, debido al material que implica, por ejemplo: local en el que los estudiantes reciban dichos cursos, el mobiliario necesario, además del personal docente que requiere. Razón por la que se precisa la creación de una estrategia de trabajo que permita el uso de las más proliferantes tecnologías con las corrientes educativas actuales, dando paso al desarrollo del e-learning. (Quevedo, 2000)

El término e-learning describe *"conjunto de tecnologías, aplicaciones y servicios orientados a facilitar la enseñanza y el aprendizaje a través de Internet/Intranet, que facilitan el acceso a la información y la comunicación con otros participantes"* (Red TTnet, 2005). Una de las principales ventajas del e-learning es la facilidad de acceso. La formación puede llegar a más personas, puesto que desaparecen las barreras espacio-temporales, viabilizando un acercamiento al plano de la enseñanza a personas que con anterioridad tenían dificultades para estar en contacto continuo con los procesos de formación.

Con el objetivo de abrir las puertas a la enseñanza y aprendizaje a distancia, se requieren sistemas para la administración, distribución y gestión de los contenidos que forman parte de cada evento de aprendizaje. Entre dichos sistemas, se encuentran los Learning Management System (LMS) y los Learning Content Management System (LCMS). Los LCMS están enfocados en la creación y administración de contenidos, a diferentes niveles, permitiendo de esa manera reestructurar la información y los objetivos de los contenidos, de manera dinámica, para crear y modificar OA que atiendan a necesidades y estilos de aprendizaje específicos. Los LMS por su parte, se definen como aplicaciones de software viabilizadas a través de la web, empleadas para administrar, distribuir y controlar las actividades de formación online[1] de una organización o institución. Estas posibilitan la gestión de usuarios, recursos, así como administración de acceso y seguimiento de la evolución del estudiante. Sin embargo, dichas plataformas manejan como mínima unidad didáctica el curso.

[1] Conectado a una red o sistema mayor.

En la actualidad existe una tendencia a granular los contenidos en pequeñas unidades llamadas Objetos de Aprendizaje (OA), rigiéndose por el paradigma orientado a objetos y que tienen como objetivo maximizar el número de contextos donde puedan ser utilizados. (Tamayo, 2007)

Explícitamente, no existe una única definición del concepto de OA. Dada la amplitud de conceptos y definiciones, los cuales provocan poca claridad con respecto al término, en el presente trabajo se considerará como OA: *"cualquier recurso con una intención formativa, compuesto de uno o varios elementos digitales, descrito con metadatos, que pueda ser utilizado y reutilizado dentro de un entorno e-learning"* (López, 2005)

Con el objetivo de facilitar la búsqueda y localización de los OA con vista a su reutilización, es necesario que los mismos se encuentren adecuadamente descritos a través de metadatos que le permitan ser identificados. Se entiende por metadatos, para los OA: *"información relativa a los propios datos que facilitan su catalogación.*" (Ayllón, 2007). Para ello, los metadatos abordan cuestiones relacionadas con la autoría, los derechos de la misma, la descripción del propio OA, su nombre, las versiones por las cuales ha transitado, así como la localización, cadena URL (Uniform Resource Locator)[2] utilizada para acceder al OA entre otros. (ADL, 2001)

Con vista a su uso, los OA deben ser compatibles con diversos ambientes y sistemas de administración de aprendizaje, fáciles de migrar de una plataforma a otra, posibles de localizar, acceder, archivar y reutilizar.(López, 2005) Estas exigencias conllevan a que en una aplicación informática es imposible cumplir con todos esos requerimientos, y es común que para atenderlos existan diversos software, de ahí la necesidad que dichos sistemas puedan intercambiar y compartir información tanto para el almacenaje, búsquedas de los propios OAs entre otras funciones imprescindibles para el manejo de los mismos; producto a esto, actualmente constituye una tendencia el consumo y publicación de servicios como la búsqueda y el almacenaje de los recursos compartiendo de esa forma funcionalidades entre sistemas, lo cual constituye una práctica de desarrollo para lograr la interoperabilidad, definida por la IEEE (Institute of Electrical and Electronics Engineers) como: "*habilidad de dos o más sistemas o componentes para intercambiar información y para usar la información que ha sido intercambiada*".

[2] Dirección global de cualquier documento o recurso.

Para garantizar interoperabilidad, además de la reusabilidad, durabilidad y accesibilidad de los OA es necesaria la aplicación de estándares y especificaciones. *"Un estándar figura un modelo que se sigue para realizar un proceso, es un patrón, una tipificación o una norma de cómo realizar algo"* (AulaGlobal, 2007). Los estándares o especificaciones son propuestos por varias organizaciones, ejemplo de ellas son: Aviation Industry Computer-Based Training Comitee (AICC), Advanced Distributed Learning (ADL), Institute of Electrical and Electronics Engineers/Learning Technology Standard Committee (IEEE/LTSC), World Wide Web Consortium (W3C), además de IMS Global Consortium Inc, entre otras.

Entre los múltiples estándares que existen con vista a lograr la interoperabilidad se puede encontrar SQI (Simple Query Interface) e IMS-DRI (Digital Repositories Interoperability). SQI define una capa para facilitar las búsquedas de contenidos, especifica además un estándar y un API (Application Programming Interface) para resolver la problemática de las búsquedas de contenidos digitales en entornos heterogéneos permitiendo la comunicación sincrónica y asincrónica de búsquedas y resultados (Agrega, 2008). IMS-DRI facilita el acceso a los contenidos para contextos educativos (Riley & Mckell, 2003). Ambas especificaciones son de gran utilidad para lograr la reutilización de los OA.

SCORM (Sharable Content Object Reference Model) es el modelo de referencia de mayor uso a nivel mundial para lograr la interoperabilidad de los OA. SCORM especifica qué estándares o especificaciones se deben utilizar y cómo hacerlo. El mismo utiliza como modelo de empaquetamiento el Content Packaging Specification (IMS-CP) de IMS Global Learning Consortium que provee la funcionalidad para describir y empaquetar materiales de aprendizaje. Recurre además como modelo de metadatos al LOM (Learning Object Metadata) de IEEE-LTSC (Learning Technology Standards Committee), que tiene como objetivo la creación de descripciones estructuradas de recursos educativos, donde su modelo de datos especifica qué aspectos de un OA deben ser descritos y qué vocabularios se pueden utilizar en dicha descripción para la conformación del mismo en las Herramientas de Autor (HA).

Las HA posibilitan la creación de contenidos estandarizados de tal forma que los mismos puedan ser exportados hacia ROA[3]. En la actualidad existen distintas HA que permiten crear contenidos de aprendizaje conformes a la especificación SCORM. Algunas, como ReLOAD sólo posibilitan importar

[3] Repositorio de objetos de aprendizaje.

contenidos creados con anterioridad, otras, simplemente, facilitan la labor de crear los archivos manifiesto[4], ejemplo de ello es la herramienta Manifest Generator Pro.

Existen además HA web como Autore, en la cual los OA presentan una estructura rígida imposibilitando el diseño flexible de los mismos; WeLoad, donde se reproducen las funcionalidades de ReLOAD Editor pero incorporando a la misma pocos metadatos, y además ésta utiliza un servidor web apache portátil impidiendo así el acceso a través de la red de los recursos de dicha herramienta.

En la Universidad de Ciencias Informáticas (UCI) se ha desarrollado una herramienta web nombrada *"ROXS",* la misma se basaba en LOM como modelo de metadatos, el cual guía el proceso de identificación de los contenidos para el aprendizaje. Esta herramienta básicamente empaquetaba los OA con el objetivo de que formara parte de un curso y de esta forma ofrecerlo a los estudiantes a través de una plataforma de aprendizaje con soporte SCORM 1.2. Pero dicha aplicación no poseía un diseño del OA flexible, debido a la variedad conceptual del término, además no permitía la creación de contenido incluyendo la conformación de preguntas tanto de verdadero o falso, de completamiento o selección múltiple; funcionalidades claves en una HA, tampoco existía comunicación interna entre los usuarios que la utilizaban; no admitía la búsqueda o el almacenaje de los OA en los repositorios con vista a su reutilización.

Por todo lo anteriormente expuesto se plantea como **problema científico** ¿Cómo garantizar una herramienta de autor web interoperable que posibilite el diseño flexible de objetos de aprendizaje, a partir de la interpretación que se de a este término?

Como **objeto de estudio** el diseño de objetos de aprendizaje y el **campo de acción** lo constituye las tecnologías que posibilitan el diseño de los OA además de los sistemas que los manejan.

De acuerdo al problema planteado anteriormente se propone como **objetivo** desarrollar una herramienta de autor web interoperable que posibilite el diseño flexible de objetos de aprendizaje haciendo uso de estándares.

[4] Archivo XML que contenga los metadatos que deberá llamarse IMSMANIFEST.XML.

Objetivos específicos:

- ✓ Revisión teórica de las principales bibliografías y proyectos relacionados con el tema.
- ✓ Análisis y diseño de una herramienta de autor web interoperable para el diseño flexible de contenidos educativos reutilizables.
- ✓ Implementación de una herramienta de autor web interoperable para el diseño flexible de contenidos educativos reutilizables.

Idea a defender: El diseño de OA reutilizables puede lograrse mediante la utilización de una herramienta de autor Web interoperable haciendo uso de estándares.

Para dar cumplimiento al objetivo enunciado se han planteado las siguientes **tareas**:

- ✓ Realizar un análisis acerca de los principales proyectos relacionados con la creación y estandarización de OA a nivel nacional e internacional.
- ✓ Realizar un estudio de metodologías utilizadas para el desarrollo de aplicaciones web y seleccionar una que cumpla los lineamientos mínimos de calidad establecidos en la UCI.
- ✓ Realizar el análisis y diseño de la Herramienta de Autor para el diseño flexible de OA reutilizables.
- ✓ Implementar Herramienta de Autor para el diseño flexible de los OAs, partiendo del análisis y diseño ya realizados.

Para dar cumplimiento a estas tareas se recurrirá a los **métodos teóricos**: **Analítico-Sintético**, al identificar los conceptos y definiciones más importantes relacionados con el tema que permita generar luego, una propuesta adecuada a la situación planteada y la tecnología estudiada; el **Histórico-Lógico** que permite llevar a cabo un estudio de las diferentes herramientas de autor existentes actualmente, sus características, ventajas así como su evolución. **Modelación,** el cual posibilita el esbozo de los diferentes diagramas y modelos del proceso de desarrollo de la aplicación. También se utilizan los **métodos empíricos** como la **Observación** ya que mediante este se puede observar el comportamiento de los sistemas existentes y permite extraer información de su modo de procesar los OA.

El presente trabajo consta de una introducción, cuatro capítulos, conclusiones generales, recomendaciones, referencias bibliográficas utilizadas durante el desarrollo del trabajo y por último, los anexos que complementan el cuerpo del trabajo.

Capítulo I Fundamentación teórica: Se tratan de forma general los principales conceptos que cimientan la investigación, entre los que se encuentran OA, metadatos, herramientas de autor (HA); metodología de desarrollo de software, las tecnologías a utilizar, así como normativas y estándares para el diseño y creación de OA.

Capítulo II Características del Sistema: se detalla el flujo existente de los procesos involucrados en el campo de acción, estudiando cómo se ejecutan actualmente dichos procesos; además se reseña el objeto de automatización, la propuesta del sistema, modelo de dominio, se determinan los requerimientos funcionales y no funcionales de la aplicación y se describen los casos de uso y los actores del sistema.

Capítulo III Análisis y Diseño del Sistema: en este capítulo se presentan los diagramas de clases del domino, de interacción, de clases del diseño, diseño de la base de datos y el diagrama de casos de uso del sistema en cuestión.

Capítulo IV Construcción de la solución propuesta: Los artefactos generados en el capítulo anterior proporcionan la entrada a la implementación. Es aquí donde se generan los diagramas de clases, componentes y paquetes de implementación. Además se tratan las pruebas realizadas para comprobar las funcionalidades del sistema, quedando construido el software propuesto al terminar el capítulo.

Capítulo I Fundamentación Teórica

1.1 Introducción

En el presente capítulo se precisan elementos teóricos que sustentan la investigación y desarrollo de la herramienta. Se tratan los principales conceptos y definiciones relacionadas con el diseño y creación de los OA, las normativas y estándares internacionales para la descripción de metadatos e interoperabilidad.

Se profundizará de igual forma vitales conceptos que fundan la investigación, tales como: OA, metadatos, herramientas de autor (HA). Además se realiza un estudio de un conjunto de HA que se utilizan en el mundo. Se fundamenta la selección de metodología de desarrollo de software, lenguaje de programación, tecnología a utilizar en la web, servidores tanto web como de base de datos (BD) y framework de desarrollo.

1.2 Fundamentos del e-learning

En la sociedad del conocimiento las TIC desempeñan un papel esencial. Como resultado de la aplicación de esas tecnologías al ámbito de la educación y la formación surge el e-learning, término descrito anteriormente como: "*conjunto de tecnologías, aplicaciones y servicios orientados a facilitar la enseñanza y el aprendizaje a través de Internet/Intranet, que facilitan el acceso a la información y la comunicación con otros participantes*." (Red TTnet, 2005).

En un entorno e-learning el diseño de los contenidos debe ser realizado en respuesta a las necesidades y posibilidades del estudiante, respondiendo además a la cantidad y calidad de la información, interactividad y una estructura apropiada. Las herramientas de comunicación viabilizan la interacción entre los diferentes actores en el proceso de enseñanza-aprendizaje. Otro de los componentes es la plataforma, la cual se define como *"un entorno de hardware y software diseñado para automatizar y gestionar el desarrollo de actividades formativas"* (ATICA_SOCRATES 2007).

Sin dudas el papel de los contenidos en una solución de e-learning tiene un lugar significativo, de ahí la importancia de utilizar herramientas de autor adecuadas para su diseño y creación, teniendo presente las tendencias que en este sentido existen en la actualidad.

1.3 Objetos de Aprendizaje

Los OA son una pieza fundamental en el desarrollo del e-learning. En la búsqueda de una definición estable a lo largo de la investigación se encuentran una serie de conceptos referidos al mismo; definiciones, que en su totalidad no abarcan el significado global de los mismos, por ejemplo: *"los objetos de aprendizaje son archivos de texto, ilustraciones, vídeos, fotografías, animaciones y otros tipos de recursos digitales"* (González, 2005); *"Un objeto es cualquier entidad digital o no digital que puede ser usada, re-usada o referenciada para el aprendizaje soportado en tecnología"* (IEEE, 2001); *"cualquier recurso digital que puede ser reutilizado para apoyar el aprendizaje"* (Wiley, 2000).

Estos conceptos no revelan elementos claves para la conformación de un OA, como el esclarecimiento de que no es cualquier archivo, es necesario aclarar su intención formativa. Por esa razón, para el desarrollo de este trabajo, se adoptará la siguiente definición enunciada en la introducción de la investigación para los OA: *"cualquier recurso con una intención formativa, compuesto de uno o varios elementos digitales, descrito con metadatos, que pueda ser utilizado y reutilizado dentro de un entorno e-learning."* (López, 2005)

Los beneficios de los OA en un contexto educativo están dados por la accesibilidad, durabilidad, interoperabilidad y reutilización, característica notable en las diferentes definiciones de los OA es la reutilización. Dicho concepto, expone el manejo de elementos previamente desarrollados para generar un nuevo producto. Dada la modularidad de los OA y su independencia de otros recursos, el uso de dichos recursos en diferentes aplicaciones es una de sus bondades, evitando duplicidad de esfuerzos para el desarrollo de contenidos.

Con el objetivo de alcanzar la reutilización de OA, es imprescindible que el mismo esté descrito a partir de metadatos que permitan su localización y búsqueda, así como la categorización y calificación pedagógica del OA. Como se ha expuesto anteriormente los metadatos se definen como: *"información relativa a los propios datos que facilitan su catalogación y además proporcionan información asociada."* (Ayllón, 2007).

Los metadatos se encuentran organizados por categorías y subcategorías donde se recoge información referente a: título, autor, fecha de creación, descripción entre otros, los mismos se pueden clasificar en (MANJÓN 2006):

- ✓ Descriptivos: Describen e identifican recursos de información, además de permitir a los usuarios la búsqueda y recuperación de información.

- ✓ Estructurales: Facilitan la navegación y la presentación de los recursos. Proporcionan información sobre la estructura interna de los documentos, así como la relación entre ellos.
- ✓ Administrativos: Facilitan la gestión de conjuntos de recursos. Incluyen la gestión de derechos y control de acceso y uso.

A pesar de que los autores conocen la importancia de catalogar los contenidos educativos que crean para su posterior búsqueda y localización, esta tarea les resulta un tanto tediosa, debido a la gran cantidad de metadatos que exigen algunos esquemas, razón por la cual, con la intención de facilitar esta actividad a los autores, la herramienta ROXS autocompletará algunos de estos metadatos a partir de información que obtenga en la autenticación del propio autor en la aplicación o del mismo sistema.

Entre los requisitos para los OAs por parte de los estándares internacionales se encuentra la estructura o esquema para crear los metadatos en el formato XML (Extensible Markup Language). Algunos de estos indican que los metadatos deberán de estar en un paquete junto con el OA. En la figura 1 se muestra la estructura de un paquete según el Modelo de referencia SCORM, (ADL, 2004), el cual se abordará más adelante.

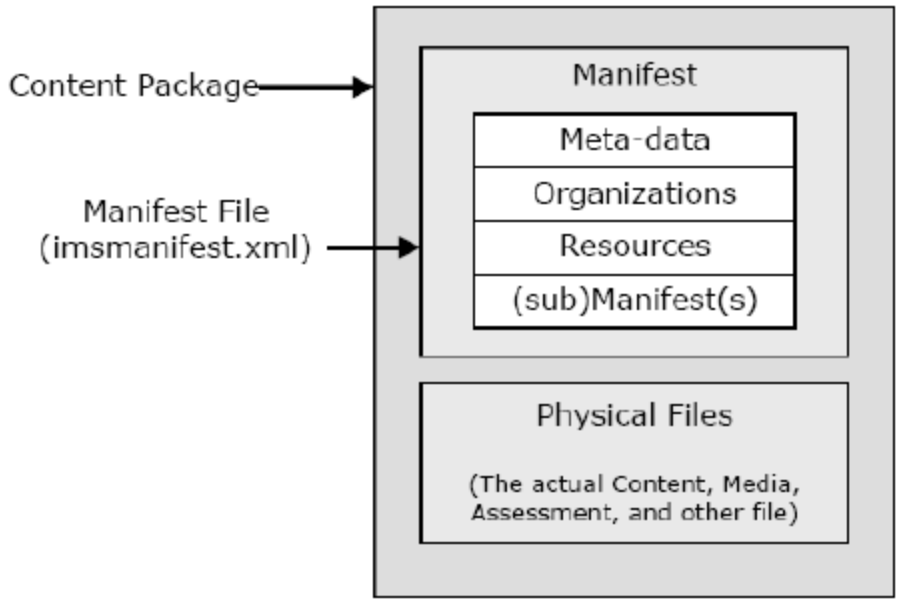

Figura 1.1. Estructura del Paquete.

El paquete será en formato ZIP[5]; donde el contenido del propio paquete según el modelo de referencia SCORM será: el OA, el archivo XML que contenga los metadatos (que deberá llamarse IMSMANIFEST.XML y que es denominado archivo manifiesto) además de una organización. El archivo manifiesto contiene un inventario estructurado del contenido del paquete. Dicho archivo comprende los metadatos, la organización de los metadatos los recursos y el OA denotado como archivo físico o pueden conformarse paquetes que comprendan la descripción de los metadatos en un XML externo, el cual será incluido en el archivo manifiesto (Priego & López, 2005).

Según el estándar LOM los metadatos se dividen en nueve categorías que son: General, Ciclo de Vida, Meta-Información, Técnica, Educativa, Derechos, Relación, Anotación y Clasificación. En este estándar cada metadato es opcional. La lista de elementos requeridos difiere dependiendo del componente al cual describa.

En general, se puede afirmar que el mundo de los metadatos está en constante desarrollo y que es inevitable su aplicación y dominio para lograr mejores resultados en la recuperación de una información más precisa e independientemente. Razón por la que se hace necesaria la existencia de normativas y estándares que guíen este proceso.

1.4 Normativas y estándares internacionales en el diseño y creación de OA

El uso de Internet como medio en alza para la formación ha provocado que numerosas organizaciones hayan fracasado en la implementación y difusión de sistemas de enseñanza. Esto ha supuesto la aparición de multitud de sistemas y recursos educativos, lo que conlleva a la necesidad de establecer recomendaciones y estándares que permitan su uso eficiente. Como anteriormente se expuso, *"Un estándar es un patrón, una tipificación o una norma de cómo realizar algo."* (AulaGlobal, 2007).

En la actualidad existen una serie de grupos y organizaciones que desarrollan especificaciones. Hasta la fecha, ninguna de éstas, ha sido formalmente adoptada en la industria del e-learning, aunque dichas especificaciones no dejan de ser recomendaciones, que por el momento la industria trata de seguir. A continuación, algunas organizaciones con sus iniciativas:

[5] **ZIP** o **Zip** es un formato de almacenamiento sin pérdida, muy utilizado para la compresión de datos como imágenes, música, programas o documentos.

1.4.1 Organizaciones e Iniciativas de estándares e-learning

1.4.1.1 IEEE Learning Technologies Standards Comitee (LTSC)

Se trata de un organismo que promueve la creación de una norma ISO, una normativa estándar real de amplia aceptación. El LTSC se encarga de preparar normas técnicas, prácticas y guías recomendadas para el uso informático de componentes y sistemas de educación y de formación, en concreto, los componentes de software, las herramientas, las tecnologías y los métodos de diseño que facilitan su desarrollo, despliegue, mantenimiento e interoperación.

1.4.1.2 IMS Global Learning Consortium

El trabajo de la IEEE fue recogido por esta corporación privada creada por algunas de las empresas más importantes del sector. Su objetivo fue la creación de un formato que pusiese en práctica las recomendaciones de crecidas compañías como la IEEE. (IEEE, 2002)

Estableció un tipo de fichero XML para la descripción de los contenidos de los cursos. De tal modo que cualquier LMS pudiese, leyendo su fichero de configuración IMSMANIFEST.XML, cargar el contenido. Algunas de las principales iniciativas de este consorcio son:

- **IMS Learning Object Metadata (LOM)**

IMS Learning Object Metadata entrega una guía sobre cómo los contenidos deben ser identificados o etiquetados, incluyendo organizar la información de los alumnos de manera que se puedan intercambiar entre los distintos servicios involucrados en un sistema de gestión de aprendizaje (LMS).

LOM es un estándar que define metadatos para OA, detallando un esquema conceptual de datos que puntualiza la estructura de una instancia de metadatos para el propio OA. Este esquema especifica los elementos de datos de los que se compone una instancia de metadatos del OA. *Las cualidades relevantes de los objetos de aprendizaje que se describen incluyen: título, idioma, tipo de objeto, autor, propietario, términos de distribución, formato, copyright, y cualidades pedagógicas, tales como estilo de la enseñanza o de la interacción.* (IEEE, 2002)

La estructura y organización de metadatos que se propone en esta especificación se basa íntegramente en el estándar propuesto por IEEE-LTSC. A continuación, se describen las 9 categorías originales en base a las cuales se agrupa y organiza el esquema de metadatos LOM:

1. La categoría ***General*** (<General>) agrupa la información general que describe un OA de manera global.
2. La categoría ***Ciclo de Vida*** (<Life Cycle>) agrupa las características relacionadas con la historia y el estado actual del OA, y aquellas que le han afectado durante su evolución.
3. La categoría ***Meta-Metadatos*** (<Meta-metadata>) agrupa la información sobre la propia instancia de metadatos (quién es el responsable de la documentación del OA, cuándo, etc.).
4. La categoría ***Técnica*** (<Technical>) agrupa los requerimientos y características técnicas del OA.
5. La categoría ***Uso Educativo*** (<Educational>) agrupa las características educativas y pedagógicas del OA.
6. La categoría ***Derechos*** (<Rights>) agrupa los derechos de propiedad intelectual y las condiciones para el uso del OA.
7. La categoría ***Relación*** (<Relation>) agrupa las características que definen la relación entre un OA y otros OAs relacionados.
8. La categoría ***Anotación*** (<Annotation>) permite incluir comentarios sobre el uso educativo del OA e información sobre cuándo y por quién fueron creados dichos comentarios.
9. La categoría ***Clasificación*** (<Classification>) describe el OA en relación a un determinado sistema de clasificación.

Este estándar no define símbolos para los nombres de los elementos o los valores de los vocabularios. Dentro del esquema base LOM, el orden de las categorías y de las subcategorías es a título informativo. Una instancia del esquema base LOM debe preservar el anidamiento de las categorías, pero no precisa ordenar las categorías o los subitems dentro de una categoría o subcategoría. Por ejemplo la categoría *5: Uso Educativo* puede aparecer antes que la categoría *1: General* y dentro de la categoría general el *ítem 1.3: General*.

Con el objetivo de garantizar interoperabilidad, los usuarios de este estándar deben procurar utilizar los metadatos de acuerdo con los elementos de datos más adecuados de esta especificación. Por ejemplo no se debería introducir un elemento nuevo de datos "*nombre*" para remplazar a *1.2: General. Título,* debido a que cuando se vaya a realizar una búsqueda del OA al cual pertenece el elemento *"nombre"* introducido con vista a su reutilización, el mismo no aportará ninguna descripción acerca del contenido real del propio elemento. Otra de las especificaciones de la IMS es IMS Content Packaging (CP).

- **IMS Content Packaging (CP)**

Esta especificación provee la funcionalidad para describir y empaquetar material de aprendizaje, ya sea un curso individual o una colección de cursos, en paquetes portables e interoperables. El empaquetamiento de contenidos está vinculado a la descripción, estructura, y ubicación de los materiales de aprendizaje on-line, y a la definición de algunos tipos particulares de contenidos. La idea es que el contenido desarrollado bajo este estándar sea utilizado en una variedad de sistemas de gestión de aprendizaje (*LMS*).

El estándar define un sistema estandarizado de estructuras para permitir interoperabilidad entre las herramientas de creación de contenidos, los sistemas de administración de contenidos (LMS) y los ambientes en tiempo de ejecución. La especificación proporciona un formato común de la entrada-salida que cualquier sistema puede soportar e incluye lo siguiente (CUDI, 2006):

- **Contenido Empaquetado IMS (Content Package):** Un paquete es una unidad de contenido usable (y reutilizable), debe estar disponible, y por lo tanto debe contener toda la información y componentes necesarios para utilizar el contenido de aprendizaje cuando se desempaquete.
- **Modelo de Información de Empaquetamiento de Contenido (Content Packaging Information Model):** El modelo de la información define los elementos de datos disponibles para la construcción del paquete SCORM.
- **"Encuadernado" en XML para el Empaquetamiento de Contenidos (Content Packaging XML Binding):** El XML referenciado es un una plantilla (llamada Schema) para representar un paquete de contenido SCORM a través de un archivo XML. Un archivo XML es un archivo de texto estructurado que contiene información de cómo presentar el contenido además contiene las instrucciones para reestructurarlo en tiempo de ejecución.

Un contenido empaquetado IMS contiene dos componentes importantes, como se observa en la figura anterior (Figura 1):

- Un archivo manifiesto con el nombre imsmanifest.xml, el cual describe el contenido y la organización además de enumerar los recursos contenidos en el paquete.
- Los archivos físicos referidos en el manifiesto.

Para transportar archivos entre los sistemas, los componentes se pueden poner en un archivo comprimido "ZIP". El manifiesto contiene los metadatos, una sección de las organizaciones, una lista de recursos, y las referencias a sub-manifiesto si estos existieran. Los metadatos describen a los paquetes, organizaciones y recursos en un manifiesto IMS, consorcio de prestigio mundial, ya que sus

estándares y modelos han sido adoptados por grandes organizaciones como la ADL (Advanced Distributed Learning) para su modelo de referencia SCORM (Shareable Content Object Reference Model).

1.4.1.3 ADL (Advanced Distributed Learning).

La iniciativa ADL (Advanced Distributed Learning), es un programa del Departamento de Defensa de los Estados Unidos y de la Oficina de Ciencia y Tecnología de la Casa Blanca para desarrollar principios y guías de trabajo necesarias para el desarrollo y la implementación eficiente, efectiva y en gran escala, de formación educativa sobre nuevas tecnologías Web. Este organismo recogió lo mejor de las anteriores iniciativas, el sistema de descripción de cursos en XML de la IMS, y el mecanismo de intercambio de información, mejorándolas para su propio modelo, el SCORM.

- ***SCORM (Shareable Content Object Reference Model)***

SCORM proporciona un marco de trabajo y una referencia de implementación detallada que permite a los contenidos y a los sistemas usarlo para el intercambio con otros sistemas, alcanzando de esa forma interoperabilidad, reusabilidad y adaptabilidad.

No obstante, como se ha expuesto con anterioridad, SCORM ha producido varias versiones, SCORM 1.1 el cual usa un formato de estructura de curso donde el archivo de XML se basa en las especificaciones de AICC (Aviation Industry Computed Based-Training Comitee) para describir la estructura de contenido, pero carece de un empaquetamiento robusto; SCORM 1.2, que fue la primera versión que se adoptara con seguridad, utilizada en la actualidad por los LMS, aunque con dificultades de adopción por los autores, debido a la complejidad de los metadatos y la comunicación con el LMS y SCORM 2004 o 1.3, versión actual, está basado en las nuevas normas de IEEE para el API (Application Programming Interface), con muchas ambigüedades de versiones anteriores ya resueltas. Incluye la habilidad del secuenciamiento de actividades que emplean los OA, además de compartir y reutilizar la información de los mismos.

La mera agrupación de ficheros en recursos simples y compuestos es interesante desde el punto de vista de la interoperabilidad, pero no representa ningún avance en cuanto a la complejidad del proceso de aprendizaje planteado. Por ello, los SCOs (Sharable Content Object) representan el núcleo del modelo SCORM, éstos, se definen como una colección de uno o más recursos que aprovecha el Run-Time Environment de SCORM para comunicarse con un LMS.

El modelo de contenido SCORM 2004 define cómo los recursos de nivel inferior se complementan con otros y a su vez se organizan en unidades de un nivel mayor de instrucción, dicho modelo se compone además de las actividades. "*Una actividad puede describirse como una unidad de instrucción, la cual puede proporcionar un SCO y a la vez puede estar compuesta por sub-actividades representadas en una Organización de Contenido"* (Content Aggregation Model, 2006), no existe límite en el número de niveles de anidación. Si bien una taxonomía específica de aprendizaje puede estar asociada con los niveles jerárquicos de actividades, por ejemplo, capítulo, módulo, etc., esto no es un requisito.

Cada actividad está comprendida dentro de una Organización de Contenido, con la que se puede referenciar metadatos para búsquedas en los repositorios permitiendo la reutilización de las mismas. "*Una Organización de Contenido es una representación o mapa que define el uso de los contenidos estructurados a través de unidades de instrucción, las actividades* "(Content Aggregation Model, 2006), se puede describir también con metadatos, permitiendo así oportunidades para facilitar la reutilización y el mantenimiento.

"*La secuencia de actividades se define como parte de la Organización de Contenido, mediante la estructuración de las actividades* "(Content Aggregation Model, 2006). Los LMS son los responsables de la interpretación de la información de secuencia descrita en dicha organización y la aplicación de la secuencia de comportamientos para el control de la sucesión real de los recursos de aprendizaje en tiempo de ejecución.

Producto a la estabilidad que aporta SCORM 1.2, además de la adopción por varias herramientas de ésta versión ofreciendo seguridad en cuanto a su utilización, se decidió usarlo como modelo de referencia en la implementación de la herramienta *"ROXS".*

1.5 Estándares para la interoperabilidad entre sistemas

Más allá de lo complicado que pueda resultar introducir modelos pedagógicos complejos en la enseñanza a través de Internet, la interoperabilidad de los sistemas que lo soportan resulta aún más problemática debido a la riqueza del campo a tratar. Para poder introducir diseños pedagógicos complejos (habitualmente denominados diseños instruccionales), que involucren simultáneamente a distintos usuarios con disímiles roles de un modo que además sea interoperable, resulta necesario desarrollar especificaciones que formalicen de

manera precisa los elementos básicos de estos diseños para así poder trasladarlos de un sistema a otro sin pérdida de información. A continuación se describe brevemente las que se hará uso en el desarrollo de la herramienta.

IMS-DRI (IMS Digital Repository Interoperability)

IMS-DRI tiene como objetivo facilitar el acceso a los contenidos en los repositorios para contextos educativos, con los LMS y los LCMS, aunque también con otros sistemas como los portales de búsqueda. Entre sus principales funciones, se reconocen cinco combinaciones como actividades principales: Buscar/Exponer, Colectar/Exponer, Enviar/Almacenar, Pedir/Entregar y Alertar/Exponer. En la Tabla 1 se resumen las recomendaciones tecnológicas que la especificación hace para cada una de las funciones principales. (Riley & Mckell, 2003)

Función	Descripción	Recomendación tecnológica
Buscar/Exponer (Search/Expose)	Ejecuta la búsqueda de metadatos asociados con los contenidos que el repositorio expone.	XQuery para colecciones con metadatos en XML.
Colectar/Exponer (Gather/Expose)	Define la solicitud de metadatos que el repositorio expone, la agregación de los metadatos para utilizarse en búsquedas subsecuentes y la agregación de metadatos para crear nuevos repositorios.	No hay recomendación específica.
Enviar/Almacenar (Submit/Store)	Se enfoca a la manera en la que un objeto se mueve a un repositorio desde un sitio accesible por red y cómo el objeto será representado en el repositorio para su acceso.	Se recomienda el uso de paquetes IMS a través de SOAP.

Pedir/Entregar (Request/Deliver)	Permite, que una vez que el usuario a localizado los metadatos en la función Buscar, pueda solicitar al repositorio el acceso al recurso.	Recomendación general para utilizar HTTP, y FTP, para diferentes tipos de recursos.
Alertar/Exponer (Alert/Expose)	Función clave, en la que a través de correo electrónico se notifica a los usuarios sobre eventos en el repositorio, pero no está contemplada todavía en esta versión de la especificación.	No hay recomendación específica ya que esta función sale del alcance de esta especificación.

Tabla 1.1 Funciones principales de IMS DRI

De forma general IMS DRI provee recomendaciones para la interoperabilidad de las funciones más comunes entre repositorios. En el nivel más general, define los repositorios digitales como colecciones de recursos con acceso a través de una red, sin conocimiento previo de la estructura de la colección. Los repositorios pueden contener los OA o los metadatos que los describen y no importa si los OA y los metadatos se encuentran en diferentes repositorios.

SQI (Simple Query Interface)

Esta especificación define una capa para facilitar las búsquedas de contenidos. Especifica un estándar y un API para resolver la problemática de las búsquedas de contenidos digitales en entornos heterogéneos como Internet, donde existen múltiples repositorios con recursos orientados a la educación que residen en entornos dispares y hacen complicado su acceso para usuarios externos.

El API define una serie de primitivas que permiten a los usuarios considerar los repositorios digitales como “peers” (pares) de una comunicación en la que un repositorio puede actuar como fuente de consultas y otro como objetivo de las mismas.

La interfaz SQI no establece la forma en la que los repositorios realizan la catalogación ni almacenan los metadatos de sus contenidos, ni define la manera en la que resuelven las consultas. Pero detalla, en el API, el mecanismo necesario para configurar tanto el lenguaje utilizado en la búsqueda, como el lenguaje de la respuesta. (AGUIRRE 2004).

1.6 Herramientas de Autor

En el proceso de producción de un curso a distancia basado en el uso de las TIC, convergen conocimientos y habilidades que tienen su origen en el campo pedagógico, informático, del diseño gráfico, el video, el trabajo con el sonido, etc.; por lo cual resulta bastante complejo la publicación de un curso, exigiendo del profesor cambios en su desempeño. Esta dificultad puede ser reducida con el empleo de softwares que automatizan parte del proceso, llamados herramientas de autor (HA).

Las HA fueron desplegadas para la creación de materiales educativos y publicación de cursos; son aplicaciones que permiten un trabajo constructivista para generar un entorno de aprendizaje dinámico, dentro de las funcionalidades que este tipo de herramientas presentan se puede destacar la posibilidad de crear actividades o pequeñas aplicaciones desde la misma herramienta (Carrodeguas & Ricardo, 2008).

Desde el punto de vista del autor los problemas referentes a la estandarización y los OAs son muy similares a otros relacionados con la introducción de las tecnologías en la educación, como las propias TICs; que en un inicio fueron catalogadas como la solución a los problemas educativos de su época. Las HA pueden ser aplicadas al desarrollo de contenidos educativos, pues permiten la creación de aplicaciones en las que, de forma sencilla y rápida se tiene la posibilidad de intercambiar el flujo de la información según las necesidades del alumno, relacionar palabras, incluir cuestionarios y marcadores que evalúen los conocimientos alcanzados, activar animaciones y vídeos explicativos, incorporar sonidos y lenguaje hablado, entre otros.

De esta forma se ha desarrollado una nueva clase de software cuyo objetivo es facilitar la creación, publicación y gestión de los materiales educativos digitales a utilizar en la teleformación. Los materiales educativos están compuestos por uno o varios OAs por lo que el nivel de habilidades y conocimientos que el autor de los materiales educativos requiere, puede variar considerablemente de OA a OA. Los intentos para utilizar herramientas digitales en la creación de materiales educativos han sido muchos y han respondido a varios ejes de clasificación. Uno principal ha sido considerar la creación de materiales como una actividad estandarizable o, por el contrario, como una actividad creativa y completamente personal (Herrero, 2002).

Entre ambos extremos se han situado muchas opciones. En la actualidad una concepción importante es la referida a los OA, así como a la estandarización de materiales educativos y de su descripción (basándose en SCORM).

En la actualidad existe una gran cantidad HA con características similares que no tienen un progreso funcional distintivo y donde los conflictos fundamentales radican en su flexibilidad y diseño complejo a la hora de crear el OA. A continuación se describe algunas de estas herramientas:

Autore

Está concebida para ser integrada en un LCMS y ser utilizada por el usuario a través de un navegador. El conjunto Autore-LCMS permite tanto producir OA como buscar objetos de otros autores, importarlos, exportarlos y publicarlos en varios formatos. Es un programa para la creación y visualización de materiales docentes que ofrece la posibilidad al usuario final aprender de una manera interactiva. (Romo, 2002)

El autor de los contenidos, generalmente un profesor, utilizará Autore para crear un documento que en su mínima dimensión contendrá un único OA, que estará compuesto de tres partes, la primera de las cuales existirá obligatoriamente, forzando al profesor a realizar un diseño impuesto a partir de estos tres elementos, visualizándose en la parte izquierda de la herramienta la siguiente estructura:

- ✓ Idea: es la parte obligatoria de todo OA de Autore, se entiende como la manifestación del concepto o tema a tratar, dicho de otra forma, sería algo análogo al enunciado de un problema.
- ✓ Desarrollo: no es obligatoria pero generalmente estará presente en la mayoría de los OAs de Autore, pues es donde se despliega el tema y la explicación.
- ✓ Evaluación: también es optativa para el autor del OA de Autore, en esta parte se introducirán las cuestiones que el estudiante deberá responder para asegurarse de la adquisición de las habilidades y competencias planteadas en la Idea y desarrolladas en la Descripción.

eXelearning

Es un redactor de XHTML para generar recursos y contenidos para e-learning. EXeLearning no es LMS, sino es un ambiente para crear el contenido aprendizaje basado en web. Con esta herramienta los usuarios pueden desarrollar las estructuras de capacitación que satisfagan sus necesidades.

Los contenidos educativos pueden exportarse como páginas Web, paquetes de contenido con formato estándar SCORM 1.2 e IMS Content Packaging; ambas soluciones le permitirán elaborar automáticamente el referido fichero de tipo ZIP, aunque ésta herramienta no se acoge a las versiones actuales de estas especificaciones desaprovechando los beneficios ofrecidos por las mismas.

EXelearning sigue una estructura de directorios y ficheros denominada *contorno*, que organiza la información en un árbol sobre la base de tres elementos básicos: tópico, sección y unidad, los cuales pueden contener esa misma organización recursivamente. Este sistema de estructuración es una aproximación hacia el de la herramienta Autore, pues proporciona una estructura común, facilitando el acceso a la información aunque estos sistemas de organización de la información, son poco flexibles ante modificaciones y actualizaciones. (Universidad de Valencia, 2004)

Reload

Los desarrolladores de esta aplicación han desplegado varias herramientas, entre ellas el Reload Editor y Reload Scorm Player, ambas de código abierto. Reload Editor es un editor de metadatos y empaquetador de contenidos, a través del cual se puede crear, importar, editar y exportar paquetes de contenido. Reload posee una organización un tanto compleja para aquellos profesores sin conocimiento previo de informática, pues su diseño consta de una estructura la cual nombran manifiesto con organizaciones contenidas, que a su vez está compuestas por ítems, a los cuales se le puede agregar metadatos, estructura válida, pero sin ningún tipo de interés por parte del profesor, mostrándose deficiente en cuestiones organizativas.

La versión 2.5.5 incluye soporte para IEEE LOM, SCORM 2004, IMS CP últimas versiones. Un inconveniente de dicha herramienta es su editor, el cual está en inglés, y aunque posee la elección de cambiar a otros idiomas, es limitado, modifica sólo algunos términos. Además de necesitar de un navegador Web para ver el contenido de los paquetes.(GONZÁLEZ 2005).

1.7 Metodología de desarrollo del software

Las metodologías de desarrollo de software abarcan todo el ciclo de vida del software, y se definen como *"un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar un nuevo software"* (Patón, 2006), adoptando la misma se obtiene un producto más predecible y permite ciertas características deseables como: existencias de reglas bien

definidas, verificaciones intermedias, planificación y control, comunicación efectiva y utilización sobre un abanico amplio de proyectos.

En cualquier empresa de desarrollo de software, un rasgo fundamental es el conocimiento a plenitud que tenga su personal de desarrollo, pero esto no solo sirve para alcanzar el objetivo deseado, la comunicación en equipo que permite el empleo de una metodología de desarrollo de software se hace muy necesaria, pues brinda un ambiente disciplinado y organizado a la hora de obtener un producto eficaz y de calidad.

Algunas de las metodologías de desarrollo de software más conocidas son: eXtreme Programming (XP), Microsoft Solution Framework (MSF) la cual se vincula a proyectos de cualquier dimensión y tecnología, Feature Driven Development (FDD) y Rational Unified Process (RUP).

Extreme Programming (XP)

La simplicidad es la base de la Extreme Programming (XP). Se simplifica el diseño para agilizar el desarrollo y facilitar el mantenimiento. Un diseño complejo del código junto a sucesivas modificaciones por parte de diferentes desarrolladores hace que la complejidad aumente exponencialmente. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. Se inserta dentro de la familia de metodologías ligeras, tratando de obtener métodos sencillos de obtener software de calidad. Uno de sus principios básicos es la mejora de la comunicación con los usuarios y los desarrolladores, la simplicidad, al desarrollar y codificar los módulos del sistema.

Microsoft Solution Framework (MSF)

Esta es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. MSF se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas. MSF tiene como características que es adaptable, siendo su uso limitado a un específico lugar; escalable, donde puede organizar equipos de 3 o 4 personas como también proyectos que requieren 50 personas o más, es flexible, utilizado en el ambiente de desarrollo de cualquier cliente y su tecnología es agnóstica porque puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología.(SANCHEZ, 2004)

Feature Driven Development (FDD)

Es un proceso de desarrollo de software reiterativo e incremental. Es uno de varios métodos Ágiles para el software en vías de desarrollo y parte de los formularios de la Alianza Ágil. FDD mezcla varias prácticas de las mejores industrias. Además este proceso de desarrollo permite realizar cambios de último momento debido a nuevos requerimientos y a las necesidades del negocio. Sin embargo, presenta su talón de Aquiles debido a la necesidad de tener en el equipo, miembros con experiencia que marquen el camino a seguir desde el principio, con la elaboración del modelo global, puesto que no es tan ágil como podría serlo XP. Su propósito principal es entregar el software tangible, activo repetidamente de una manera oportuna.

RUP (Rational Unified Process)

Rational Unified Process (RUP), es un proceso de Ingeniería de Software (ISW) propuesto por Rational Software Corporation para la construcción completa del ciclo de ISW. Permite la productividad en equipo y la realización de mejores prácticas de software a través de plantillas y herramientas que lo guían en todas las actividades de desarrollo del software. Es un producto que unifica las disciplinas en lo que a desarrollo de software se refiere, incluyendo modelado de negocio, manejo de requerimientos, componentes de desarrollo, ingeniería de datos, manejo y configuración de cambios, y pruebas, cubriendo todo el ciclo de vida de los proyectos basado en la construcción de componentes y maximizando el uso del UML (Unified Modeling Language). Pueden resumirse tres aspectos definitorios: está dirigido por casos de uso, centrado en la arquitectura, e iterativo e incremental. Lo que hace único al proceso.

RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización. Está basado en 5 principios clave que son:

1. El proceso deberá adaptarse a las características propias del proyecto u organización. El tamaño del mismo, así como su tipo o las regulaciones que lo condicionen, influirán en su diseño específico. También se deberá tener en cuenta el alcance del proyecto.
2. Los requerimientos de los diversos participantes pueden ser diferentes, contradictorios o disputarse recursos limitados. Debe encontrarse un balance que satisfaga los deseos de todos. Debido a este balanceo se podrán corregir desacuerdos que surjan en el futuro.

3. Los proyectos se entregan, aunque sea de un modo interno, en etapas iteradas. En cada iteración se analiza la opinión de los inversores, la estabilidad y calidad del producto, y se refina la dirección del proyecto así como también los riesgos involucrados
4. Este principio dominante motiva el uso de conceptos reutilizables tales como patrón del software, marcos de trabajo (frameworks) por nombrar algunos. Esto evita que los ingenieros de software vayan directamente de los requisitos a la codificación de software a la medida del cliente, sin saber con certeza qué codificar para satisfacer de la mejor manera los requerimientos y sin comenzar desde un principio pensando en la reutilización del código. Un alto nivel de abstracción también permite discusiones sobre diversos niveles y soluciones arquitectónicas. Éstas se pueden acompañar por las representaciones visuales de la arquitectura, por ejemplo con el lenguaje UML (Unified Modeling Language).
5. El control de calidad no debe realizarse al final de cada iteración, sino en todos los aspectos de la producción. El aseguramiento de la calidad forma parte del proceso de desarrollo y no de un grupo independiente.

RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en los distintas actividades. Las primeras iteraciones, en las fases de Inicio y Elaboración, se enfocan hacia la comprensión del problema y la tecnología, la delimitación del ámbito del proyecto, la eliminación de los riesgos críticos, y al establecimiento de una línea base de la arquitectura.

Durante la fase de inicio las iteraciones hacen mayor énfasis en actividades de modelado del negocio y de requerimientos. En la fase de elaboración, las iteraciones se orientan al desarrollo de la línea base de la arquitectura, abarcan más los flujos de trabajo de requerimientos, modelo de negocios (refinamiento), análisis, diseño y una parte de implementación orientado a la línea base de la arquitectura. En la fase de construcción, se lleva a cabo la construcción del producto por medio de una serie de iteraciones. En la fase de transición se pretende garantizar que se tiene un producto preparado para su entrega a la comunidad de usuarios.

Para cada iteración se selecciona algunos Casos de Uso, se refina su análisis y diseño y se procede a su implementación y pruebas. Se realiza una pequeña cascada para cada ciclo. Se realizan tantas iteraciones hasta que se termine la implementación de la nueva versión del producto.

Se decide la utilización de la metodología RUP, por todas las ventajas de organización que brinda, además de que el resto metodologías estudiadas presentan ciertas debilidades que, a juicio del equipo de trabajo, representan riesgos considerables, como es el caso de una posible captura de requisitos no adecuada. Además de que su desarrollo como método iterativo le permite reducir riesgos y dividir los proyectos en pequeños ciclos o iteraciones a través de cada una de las fases.

1.8 Lenguajes de Programación en la Web

Desde los inicios de Internet, las tecnologías fueron desarrollándose y surgieron nuevos problemas a solucionar. Esto dio lugar a desarrollar lenguajes de programación para la web dinámica, que permitieran interactuar con los usuarios y utilizaran sistemas de Bases de Datos. Actualmente existen diferentes lenguajes de programación para desarrollar en la web. Estos han surgido debido a las tendencias y necesidades de las plataformas.

Los lenguajes de programación web pueden dividirse en dos partes fundamentales: los lenguajes del lado del Servidor y los lenguajes del lado del Cliente, donde ambos reconocen la filosofía de la arquitectura Cliente/Servidor para las plataformas Web.

Entre los lenguajes pertenecientes al lado del servidor los más significativos son: ASP, PERL, Java, JSP y PHP. Ellos se caracterizan por implementar la lógica del negocio dentro del Servidor, además de encargarse de la tarea del acceso a las Bases de Datos y del tratamiento de la Información. A continuación una reseña de los lenguajes más significativos.

ASP (Active Server Pages)

ASP es una tecnología desarrollada por Microsoft para la crear páginas web de contenido dinámico apoyándose en scripts ejecutados en el servidor. Básicamente una página ASP es una mezcla entre una página HTML y un programa que da como resultado una página HTML que es enviada al cliente (navegador). Estos scripts o programas pueden en ASP ser escritos en uno de estos dos lenguajes de programación VBScript o JavaScript, pero el más extendido es VBScript.

Perl

Perl puede funcionar en cualquier plataforma que tenga un intérprete para el lenguaje. Una de las grandes ventajas del Perl es la comunidad de usuarios donde hay listas de correo, grupos de noticias, y foros en la Web, donde plantear y resolver las dudas que surjan en el desarrollo de una aplicación.

Perl es además ideal para desarrollo rápido de aplicaciones, aunque su modularidad permite que se desarrollen también aplicaciones más complejas. Ésta licenciado bajo la licencia artística y la GNU (General Public License).

Java

Java implementa la tecnología básica del lenguaje C++ con algunas mejoras y elimina algunas cosas para mantener el objetivo de la simplicidad del lenguaje. Java trabaja con sus datos como objetos y con interfaces a esos objetos. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo. Las plantillas de objetos son llamadas, como en C++, clases y sus copias, instancias. Estas instancias, como en C++, necesitan ser construidas y destruidas en espacios de memoria.

JSP (Java Server Pages)

Es una tecnología orientada a crear páginas Web con programación en Java, que se ejecutan en variados servidores Web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Los componentes JSP son reusables en distintas plataformas (Unix, Windows) Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar scripts de servidor en sintaxis Java.

PHP (Hypertext Preprocessor)

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas, usado principalmente en interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

PHP tiene asequible estructura de programación, la facilidad de llevar a cabo sentencias SQL, además de permitir la posibilidad de correr en diferentes tipos de servidores, entre ellos Apache. Además ofrece la integración con las varias bibliotecas externas, que permiten que el desarrollador haga casi cualquier cosa, desde generar documentos en PDF (Portable Document Format) hasta analizar código XML.

Debido a la complejidad del problema se selecciona como lenguaje de programación PHP por todas las ventajas mencionadas, agregando que es el más rápido de todos los analizados portando gran facilidad de uso, aprendizaje y utilización.

1.9 Tecnologías a utilizar en la Web

Como se expresó con anterioridad los lenguajes pueden dividirse en los lenguajes del lado del Servidor y los lenguajes del lado del Cliente, donde ambos reconocen la filosofía de la arquitectura Cliente/Servidor para las plataformas Web. En el lado del cliente existen tecnologías como: Visual Basic Script (VBScript) y JavaScript (JScript) además de CSS (Cascading Style Sheets) y HTML (HyperText Markup Language), destinados a ofrecer dinamismo a la aplicación donde se halla utilizado contribuyendo a la no recarga en servidores, proporcionando rapidez y optimización en los canales de comunicación. (Carrodeguas & Ricardo, 2008) Aunque también existen técnicas de desarrollo para la web como lo es Ajax (Asynchronous JavaScript And XML).

Ajax (Asynchronous JavaScript and XML)

Ajax está conformado con el objetivo de crear aplicaciones interactivas o RIA (Rich Internet Applications), es una tecnología asíncrona, en el sentido que los datos adicionales se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página. Además de ser una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores dado que está basado en estándares abiertos como JavaScript y Document Object Model (DOM).

VBScript (Visual Basic Script)

Su sintaxis refleja su origen como variación del lenguaje de programación Visual Basic. Ha logrado un apoyo significativo por parte de los administradores de Windows como herramienta de automatización, pues, paralelamente a las mejoras introducidas en los sistemas operativos Windows donde opera fundamentalmente, permite mayor margen de actuación y flexibilidad.

JScript (Javascript)

Es un lenguaje de programación interpretado con sintaxis semejante a Java y C, al igual que Java, JavaScript es orientado a objetos propiamente dicho, pues dispone de herencia, esta se realiza siguiendo el paradigma de programación basada en prototipos, debido a que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad. Todos los navegadores

modernos interpretan el código JavaScript integrado dentro de las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del DOM.

CSS (Cascading Style Sheets)

Las hojas de estilo en cascada (Cascading Style Sheets) son un lenguaje formal usado para definir la presentación estética de un documento estructurado y escrito en HTML. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura y el contenido de la presentación estética en un documento. Esto permite un control mayor del documento y sus atributos convirtiendo al HTML en un documento versátil y liviano.

HTML (HyperText Markup Language)

Se traduce al español como Lenguaje de Etiquetas de Hipertexto. Es el lenguaje de marcado predominante para la construcción de páginas web, diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas Web. Permite además códigos en lenguajes de programación extendiendo su capacidad y funcionalidad. Es el estándar usado en el World Wide Web. HTML ofrece los medios a través de los cuales se publica online documentos con cabeceras, texto, fotos, tablas además de incluir hojas de cálculo y otras aplicaciones directamente en los documentos.

Para el desarrollo de la Herramienta se ha decidido utilizar Javascript, Ajax, HTML y CSS.

1.10 Servidores Web

Un servidor no es más que una computadora incluida en una red, que le ofrece ya sea servicios o información a otras PCs, llamadas clientes. Existen varios tipos de servidores como: los servidores de correo, de chat, de fax, de Web.

Los servidores Web son ordenadores que a través del protocolo HTTP (Hypertext Transfer Protocol), los archivos para cada sitio de Internet se almacenan y se ejecutan en el servidor. A modo conceptual, *"un servidor web es un programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten usando el protocolo HTTP o el protocolo HTTPS (Hypertext Transfer Protocol Secure)"* (Cibernetia, 2009)

Como servidor Web destacado se puede señalar a Internet Information Services (IIS), que ofrece una serie de servicios para los ordenadores como: FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), NNTP (Network News Transport Protocol) y HTTP/HTTPS. Se basa en varios módulos que le dan capacidad para procesar distintos tipos de páginas, por ejemplo Microsoft incluye los de Active Server Pages (ASP) y ASP.NET. También pueden ser incluidos los de otros fabricantes, como PHP o Perl, aunque este servidor solo funciona con sistema operativo Windows, por lo que es propiedad de Microsoft Corporation.
Otro de los servidores Web más utilizados a nivel mundial es el servidor HTTP Apache, el cual es multiplataforma, extensible y de código abierto, puede soportar un sin número de sistemas operativos. HTTP Apache está conformado para ser un servidor Web flexible, atributo que le permite personalizar las necesidades de cada sitio.

Se utilizará como Servidor Web Apache debido a las características y ventajas que presenta. Es un software Libre que presenta una gran seguridad, lo que lo hace especial para el sistema, debido a que otros servidores como IIS presentan una mayor probabilidad de contener vulnerabilidades.

1.11 Gestores de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD) se define como el conjunto de programas que administran y gestionan la información contenida. Permite a los usuarios definir, crear y mantener la Base de Datos (BD) proporcionando acceso controlado a la misma. Dentro de los sistemas gestores más distribuidos a nivel mundial se encuentran: Oracle, MySQL, PostgreSQL.

Oracle (Relational Data Base Management System)

Oracle se destaca por su estabilidad y escalabilidad, además de ser multiplataforma. En el desarrollo de páginas Web pasa lo mismo al ser un sistema de alto costo no está extendido como otras bases de datos como SQL Server, Access o MySQL.

MySQL

MySQL además de ser muy popular, es un sistema de gestión de base de datos relacional, multihilo y multiusuario. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la

empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C[6].

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y el copyright del código está en poder del autor individual, MySQL es propietario y está patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

PostgreSQL

Es un poderoso manejador de bases de datos de código abierto si costos de licencia, diseñado para administrar grandes cantidades de datos. Es robusto, confiable y mantiene la integridad de los datos. Se ejecuta en la mayoría de los Sistemas Operativos más utilizados en el mundo incluyendo, Linux, varias versiones de UNIX y en Windows. (Carrodeguas & Ricardo, 2008)

PostgreSQL soporta funciones que donde la salida puede tratarse como un conjunto de valores que pueden ser tratados igual a una fila retornada por una consulta (query en inglés). Bloques de código que se ejecutan en el servidor. Pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos ofrece, desde las operaciones básicas de programación, tales como bifurcaciones y bucles, hasta las complejidades de la programación orientada a objetos o la programación funcional.

Basado en los argumentos anteriormente expuestos se considera que la mejor decisión que se tomó es la de utilizar PostgreSQL, pues este es un software de código abierto, provee servicio y procesamiento a múltiples usuarios simultáneamente, permite a una aplicación realizar varias tareas concurrentemente y realiza un buen trabajo junto a la programación PHP. Es eficiente a la hora de trabajar con bases de datos, por lo que resolvió disponer de este sistema gestor de bases de datos en la realización de la Herramienta de Autor.

1.12 Framework de Desarrollo

A partir del desarrollo alcanzado en el mundo de las tecnologías ha sido necesaria la creación de marcos de trabajo (framework) con el objetivo de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel para proveer un sistema funcional.

[6] La primera estandarización del lenguaje C fue en ANSI, con el estándar X3.159-1989. El lenguaje que define este estándar fue conocido vulgarmente como ANSI C.

Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Proporciona además una estructura al código fuente, forzando al desarrollador a crear código más legible y fácil de mantener. Los framework facilitan la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

Aunque los frameworks existen desde hace décadas, el lanzamiento de Ruby On Rails[7] en 2004 supuso una revolución en el desarrollo de las aplicaciones web que aún hoy continúa, dando origen a la gran diversidad de frameworks que en la actualidad existen como:

- **CodeIgniter**

CodeIgniter es un poderoso y a la vez ligero framework de PHP, diseñado para desarrolladores de PHP que necesitan un simple y elegante conjunto de herramientas para crear cualquier tipo de aplicaciones Web.

- **Zend Framework**

Zend Framework se centra en el desarrollo de aplicaciones más robustas, confiables, modernas para la Web 2.0 y los servicios web, consumiendo una amplia gama de APIs disponibles de los principales proveedores Google, Amazon, Yahoo!, Flickr, StrikeIron and ProgrammableWeb

- **Akelos**

El Framework de PHP Akelos, es una plataforma de desarrollo de aplicaciones web basada en el patrón de diseño MVC (Modelo – Vista - Controlador), que se enfoca en buenas prácticas que permiten programar Vistas que emplean AJAX de forma fácil, controlar las peticiones y las respuestas a través de las Controladoras, manejar aplicaciones estandarizadas internacionalmente, modelos de comunicación y de base de datos utilizando convenciones sencillas.

- **Symfony**

Primera versión publicada en octubre de 2005 por Fabien Potencier. Symfony es un completo framework diseñado para optimizar el desarrollo de las aplicaciones web. Para empezar, emplea el

[7] Entorno de programación (Rails) que se apoya en el lenguaje Ruby. Goza de gran popularidad para el desarrollo de aplicaciones de tipo Web 2.0.

tradicional patrón de diseño Modelo Vista Controlador (MVC)[8] para separar las distintas partes que forman una aplicación web, la lógica de negocio, la lógica de servidor y la presentación. 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 constituye un framework lo suficientemente maduro en el desarrollo de aplicaciones para empresas y posee además una gran cantidad de documentación, lo que propicia un mejor aprendizaje por parte de los desarrolladores. Por estas razones la dirección de arquitectura del proyecto escoge Symfony para la adecuada implementación del sistema informático *"ROXS".*

[8] Patrón arquitectural muy difundido en uso en aplicaciones de entorno Web.

1.13 Conclusiones

Como resultado de la investigación y el análisis bibliográfico realizado, a lo largo de este capítulo, han sido expuestos los principales puntos de interés relacionados con el diseño flexible de los OA, su reutilización, estándares para lograr interoperabilidad con el repositorio ROA, los aspectos generales de los metadatos que describen los OA, etc. Se plantea también la necesidad de contar con una Herramienta de Autor y objetivos de ésta. Se hace alusión a la solución informática llamada *"ROXS"*. En el capítulo donde se expone además la metodología de desarrollo de software, las herramientas, tecnologías, gestor de base de datos y framework a utilizar, así como el lenguaje para la implementación del sistema.

Capítulo II Características del Sistema.

2.1 Introducción.

Conforme se inicia y progresa el proceso de creación del OA en las HA desarrolladas, disminuye la flexibilidad del contenido, limitando la personalización del autor y el uso de dichos OA para problemas específicos. Mientras que un OA con diseño flexible, puede utilizarse en múltiples contextos, y a su vez el autor logra reutilizarlo con más facilidad, además de que su actualización y gestión se realiza de forma mucho más sencilla.

La escasez de la flexibilidad en el diseño del OA viene dada por la insuficiencia tecnológica que soportan las HA existentes, como eXelearning. A partir de dicha problemática se decide desarrollar en el área temática "Herramientas para la Teleformación", la herramienta *"ROXS".*

2.2 Objeto de Automatización.

Durante el ciclo de desarrollo de la herramienta *"ROXS"* existen varios procesos que deben ser automatizados, pues su ejecución de forma manual resulta tediosa y propensa a errores, además de consumir una valiosa porción de tiempo de los autores o personal que trabaje con la herramienta.

Muchos de estos procesos son de vital importancia para lograr un nivel de calidad óptimo en la creación de los OAs. Producto a esto en el sistema se automatizarán los procesos de diseño de los OAs mediante una aplicación Web, la cual ofrecerá plantillas prediseñadas con el objetivo de proponer al autor un esbozo fácil y atractivo de la estructura de su futuro OA, aunque si lo prefiere dicho autor tendrá la posibilidad de crear su propia plantilla. Este sistema contará con la funcionalidad de realizar búsquedas en el repositorio ROA con vista a la reutilización de dichos OAs además de almacenar los OAs en el mismo. En la herramienta se manejará información referente a las plantillas creadas por los autores, las cuales pueden ser propuestas a publicación, proceso que se realizará manualmente por los revisores, el revisor contará con la responsabilidad para agregar esas plantillas propuestas que hayan pasado por revisión al apartado de plantillas públicas.

2.3 Propuesta del Sistema.

Se presenta una aplicación Web basada modelos y estándares para el diseño de los OA, soportada sobre bases de datos en PostgreSQL. La aplicación se desarrollará con lenguaje PHP 5.0 debido a sus potencialidades como software libre la cual se ejecuta sobre un servidor Apache, la aplicación permitirá la automatización de procesos propios de los autores, revisores o el propio administrador del sistema. Después de consumar un estudio de varias metodologías se decide recurrir a RUP, pues permite

alcanzar calidad y madurez en el análisis y diseño de la aplicación contando con popularidad y uso extensivo a nivel mundial. Como lenguaje de modelado se utiliza el UML y la herramienta para representar los artefactos es el Visual Paradigm en su versión 5.3 por ser de libre distribución, open source y ofrecer facilidades en materia de desarrollo colaborativo. La herramienta será capaz de realizar de manera eficiente y amigable los procesos relacionados con la gestión y diseño de los OA, mejorando potencialmente la calidad y la agilidad del trabajo.

Ofrecidos los elementos generales del trabajo en la presente investigación, se considera necesario abordar los aspectos técnicos de la misma.

2.3.1 Modelo de Dominio.

Debido al bajo nivel de estructuración que presenta el negocio sobre el que se está investigando y que está altamente centrado en tecnologías informáticas, se propone un modelo conceptual o modelo de dominio, ya que de manera visual permite mostrar los principales conceptos que se manejan en el sistema en desarrollo. Para capturar correctamente los requisitos y poder construir la herramienta correcta se necesita tener un claro y pleno conocimiento del funcionamiento del objeto de estudio. Este modelo contribuirá a identificar algunas clases que se identificarán en el sistema.

Durante el desarrollo de la herramienta no se pudo contactar procesos bien definidos en el entorno del negocio. Se hizo difícil determinar los elementos más importantes del sistema y sus interconexiones, así como el establecimiento de las reglas de funcionamiento. Sin embargo se pueden identificar personas, eventos, transacciones y objetos involucrados en ese entorno que no está bien delimitado, por lo que se hizo necesario un modelado del domino perteneciente a la solución.

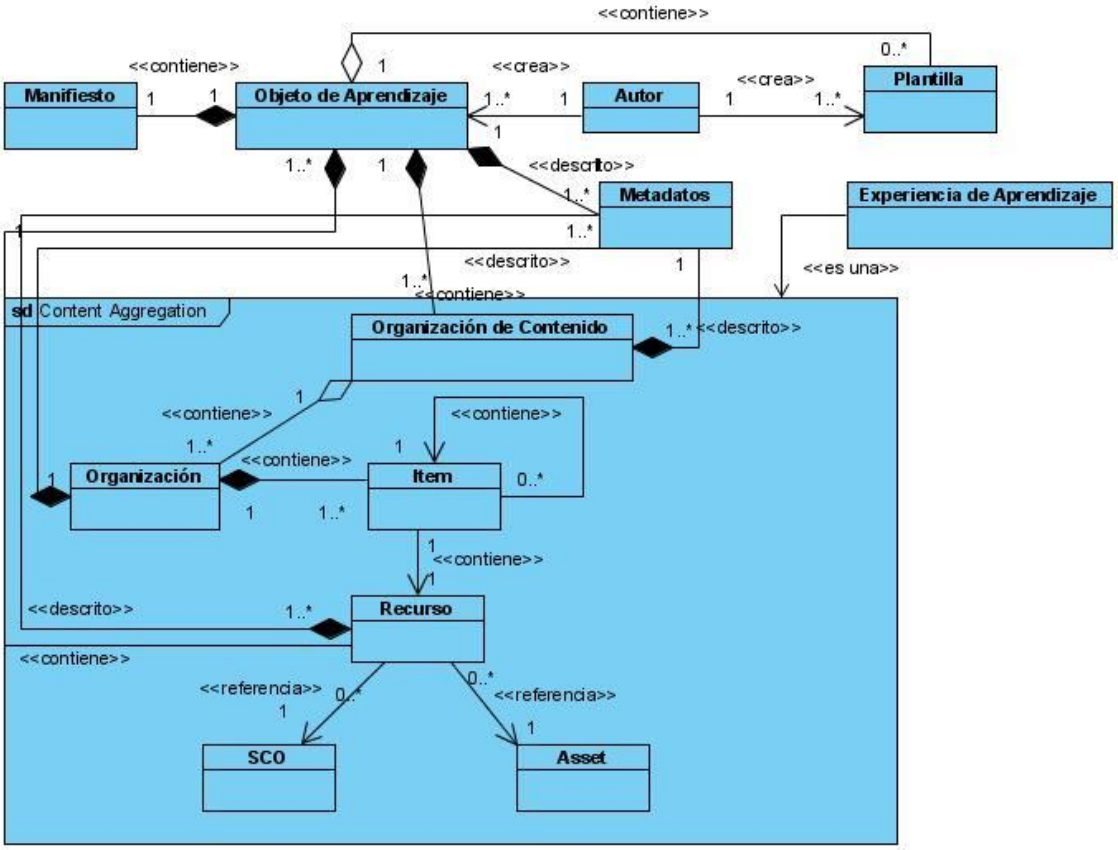

Figura 2.1. Diagrama del Modelo del Dominio.

Definición de las clases del modelo del dominio

Experiencia de aprendizaje: es aquello que se crea a través de la cual el estudiante adquiere conocimientos.

Plantilla: es una estructura que crea el autor a través de la cual se organizan los contenidos de la experiencia de aprendizaje.

Autor: es la persona encargada de crear la experiencia de aprendizaje.

Manifiesto: archivo XML que describe la organización del contenido creado, para que cualquier sistema pueda entenderla.

Contenido: recursos creados por el autor.

Content Aggregation: es usada para entregar el contenido y fijar la estructura del contenido.

Organización de contenido: estructura que presenta el contenido.

Organización: nombre que se le da una determinada distribución del contenido.

Item: elementos que conforman a la organización.

Recurso: referencia realizada desde los ítems a los recursos.

Asset: representación electrónica de medios como: textos, imágenes, videos, etc.

SCO (Shareable Content Object): es una colección de uno o más Asset.

Metadatos: los metadatos son un conjunto de atributos o elementos necesarios para describir un recurso, con el objetivo de facilitar su búsqueda.

2.4 Especificación de Requisitos.

2.4.1 Requerimientos Funcionales.

R 1. Autenticar usuario.

R 2. Crear plantilla.

R 2.1 Incorporar metadatos de plantilla.

R 2.2 Completar metadatos de plantilla.

R 3. Guardar plantilla privada.

R 4. Enviar plantilla propuesta al Revisor.

R 5. Modificar plantilla.

R 6. Revisar plantilla.

R 7.1 Publicar plantilla propuesta.

R 6.1.1 Enviar notificación al autor.

R 6.2 Denegar plantilla propuesta.

R 6.2.1 Enviar mensaje al autor.

R 8. Eliminar plantilla pública.

R 9. Eliminar mensaje enviado.

R 10. Eliminar mensaje recibido.

R 11. Listar objetos de aprendizaje en edición.

R 12. Listar plantillas privadas y públicas.

R 13. Eliminar plantilla privada.

R 18. Aplicar plantilla.

R 20. Guardar objeto de aprendizaje en repositorio.

R 21. Descargar objeto de aprendizaje.

R 22. Buscar objeto de aprendizaje en repositorio.

R 22.1 Realizar búsqueda general.

R 22.2 Realizar búsqueda avanzada.

R 23. Modificar texto de inicio.

R 24. Establecer metadatos a autocompletar.

R 25. Establecer número máximo de plantillas privadas.

2.4.2 Requerimientos no Funcionales.

Los requerimientos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con la toda la funcionalidad requerida, las propiedades no funcionales, como cuán usable, seguro, conveniente y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

Existen múltiples categorías para clasificar a los requerimientos no funcionales, siendo las siguientes representativas de un conjunto de aspectos que se deben tener en cuenta, aunque no limitan a la definición de otros.

Software:

· El sistema se implementará con tecnología PHP 5.0.

· Se requiere PostgreSQL 8.1 como servidor de bases de datos.

· Servidor Web Apache.

Hardware:

· Microprocesador 200 MHz.

· 32 MB de memoria RAM.

· 4 GB de disco duro.

Apariencia o interfaz externa:

· La interfaz del sistema será amigable a los usuarios finales.

· La aplicación estará estructurada de forma clara y comprensible.

· El diseño responderá a la ejecución de acciones de una manera rápida, minimizando los pasos a dar en cada proceso.

· Todos los textos y mensajes de la aplicación aparecerán en idioma español.

Seguridad:

· Necesidad de autenticación para acceder a las funcionalidades de la aplicación y que no sean utilizadas por personas que no tienen permisos.

· Protección contra acciones no autorizadas o que puedan afectar la integridad de los datos.

Soporte:

· Por parte del cliente se requiere un navegador capaz de interpretar JavaScript y CSS.

· El sistema debe dar la posibilidad de incorporarle nuevos servicios en caso de ser necesarios.

· Las pruebas realizadas al sistema deben permitir evaluar sus ventajas y funcionalidades, además de detectar los errores que presenta.

Rendimiento

· La herramienta propuesta debe ser rápida y el tiempo de respuesta debe ser el mínimo posible, al igual que la velocidad de procesamiento de la información.

Portabilidad:

· La aplicación debe funcionar en varias plataformas, siendo posible su acceso a través de cualquier navegador Web.

2.5 Definición de Actores y Casos de Uso.

2.5.1 Definición de Actores del Sistema.

Actor	Justificación
Autor	Es el encargado del diseño y creación de plantillas.
Revisor	Rol tiene acceso para publicar o denegar plantillas propuestas a publicación por el autor.
Administrador	Persona encargada de modificar el texto de la página de inicio de la aplicación así como configurar el protocolo de transferencia segura (https).

Tabla 2.1. Actores del Sistema.

2.5.2 Definición de Casos de Uso.

CU-2	**Crear plantilla.**
Actor	Autor
Descripción	El autor accede al sistema para crear plantillas.
Referencia	

Tabla 2.2. Caso de Uso - Crear plantilla.

CU-2.2	**Completar metadatos de plantilla.**
Actor	Autor, Sistema.
Descripción	El autor accede al sistema para crear una plantilla, a partir de esta acción el sistema completa algunos metadatos de forma automática.
Referencia	

Tabla 2.3. Caso de Uso - Completar metadatos de la plantilla.

CU-3	**Guardar plantilla privada.**
Actor	Autor
Descripción	El caso de uso inicia cuando el Autor decide guardar una plantilla privada que haya creado o alguna que haya modificado.
Referencia	R4, R6

Tabla 2.4. Caso de Uso - Guardar plantilla privada.

CU-4	**Enviar plantilla propuesta al Revisor.**
Actor	Autor
Descripción	El caso de uso inicia cuando el Autor decide proponer para el área de *"Plantillas públicas"* alguna plantilla que haya creado o modificado.
Referencia	R4, R6

Tabla 2.5. Caso de Uso - Enviar plantilla propuesta al Revisor.

CU-5	**Modificar plantilla.**
Actor	Autor
Descripción	El caso de uso inicia cuando el Autor decide modificar una plantilla para la creación de objetos de aprendizaje o con el objetivo de proponerla para publicación.
Referencia	R4, R6

Tabla 2.6. Caso de Uso - Modificar plantilla.

CU-6	**Revisar plantilla.**
Actor	Autor
Descripción	El caso de uso se refiere a la revisión de una plantilla propuesta por el autor, para su publicación o denegación.
Referencia	R4, R6

Tabla 2.7. Caso de Uso - Revisar plantilla.

CU-8	**Eliminar plantilla pública.**
Actor	Autor
Descripción	El caso de uso inicia cuando el Revisor decide eliminar una plantilla publicada en la herramienta.
Referencia	R4, R6

Tabla 2.8. Caso de Uso - Eliminar plantilla pública.

CU-16	**Crear objeto de aprendizaje.**
Actor	Autor
Descripción	El caso de uso se inicia cuando el autor escoge la opción *"Crear objeto de aprendizaje".*
Referencia	R4, R6

Tabla 2.9. Caso de Uso - Crear Objeto de Aprendizaje.

CU-20	**Guardar objeto de aprendizaje en repositorio.**
Actor	Autor
Descripción	El caso de uso inicia cuando el Autor decide guardar un objeto de aprendizaje creado anteriormente con vista a la reutilización del propio objeto de aprendizaje.
Referencia	R4, R6

Tabla 2.10. Caso de Uso - Guardar Objeto de Aprendizaje en repositorio.

CU-21	**Descargar objeto de aprendizaje.**
Actor	Autor
Descripción	El caso de uso se inicia cuando el usuario escoge la opción *"Descargar objeto de aprendizaje"* ofreciéndole la posibilidad de almacenar el mismo en su máquina tanto desde la interfaz de creación del propio objeto de aprendizaje como del área de trabajo donde se encuentran los que están en edición.
Referencia	R4, R6

Tabla 2.11. Caso de Uso - Descargar Objeto de Aprendizaje.

CU-22	**Buscar objeto de aprendizaje en repositorio.**
Actor	Autor
Descripción	El caso de uso inicia cuando el Autor decide reutilizar un objeto de aprendizaje creado anteriormente y almacenado en el repositorio con vista a la creación de uno nuevo.
Referencia	R4, R6

Tabla 2.12. Caso de Uso - Buscar Objeto de Aprendizaje en repositorio.

2.5.3 Diagrama de Casos de Uso del Sistema.

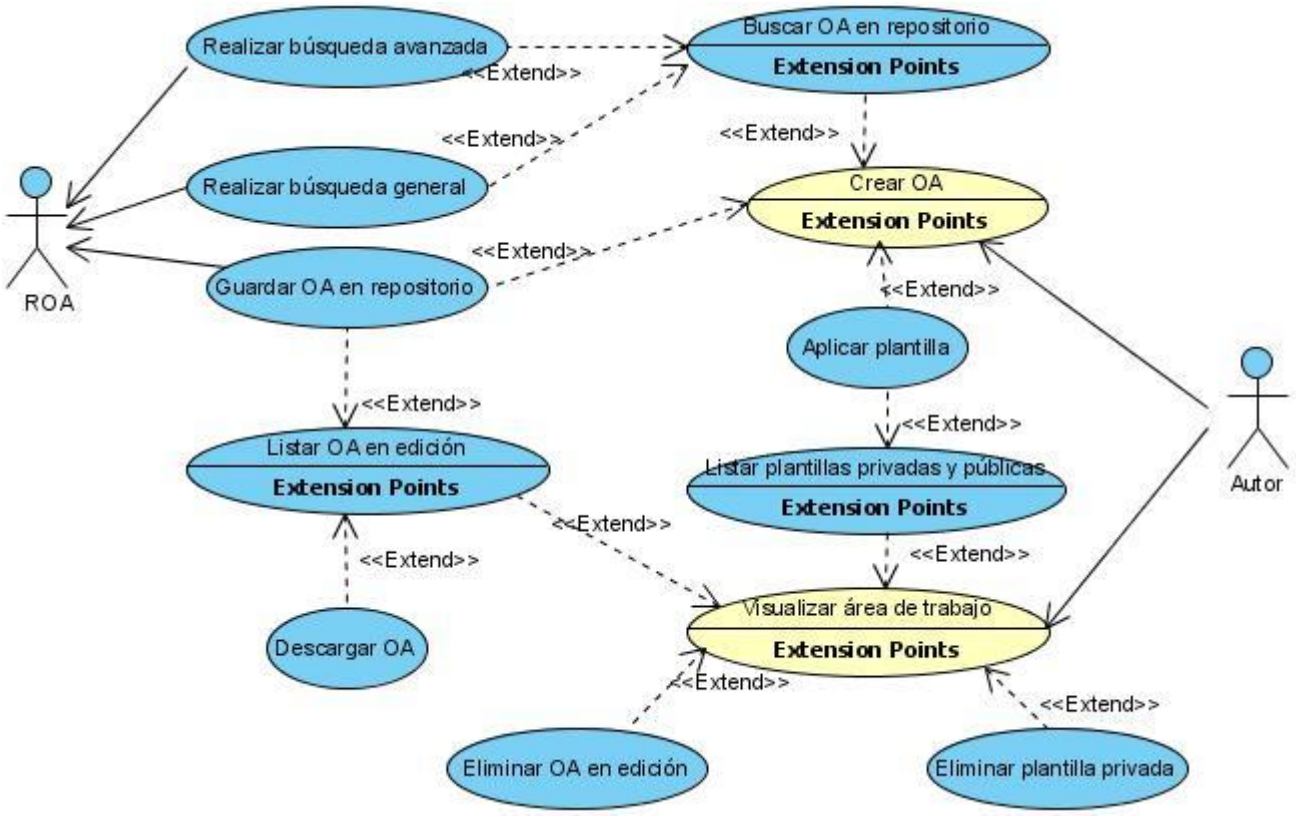

Figura 2.1. Diagrama de Casos de Uso del Subsistema (Buscar y almacenar en repositorio).

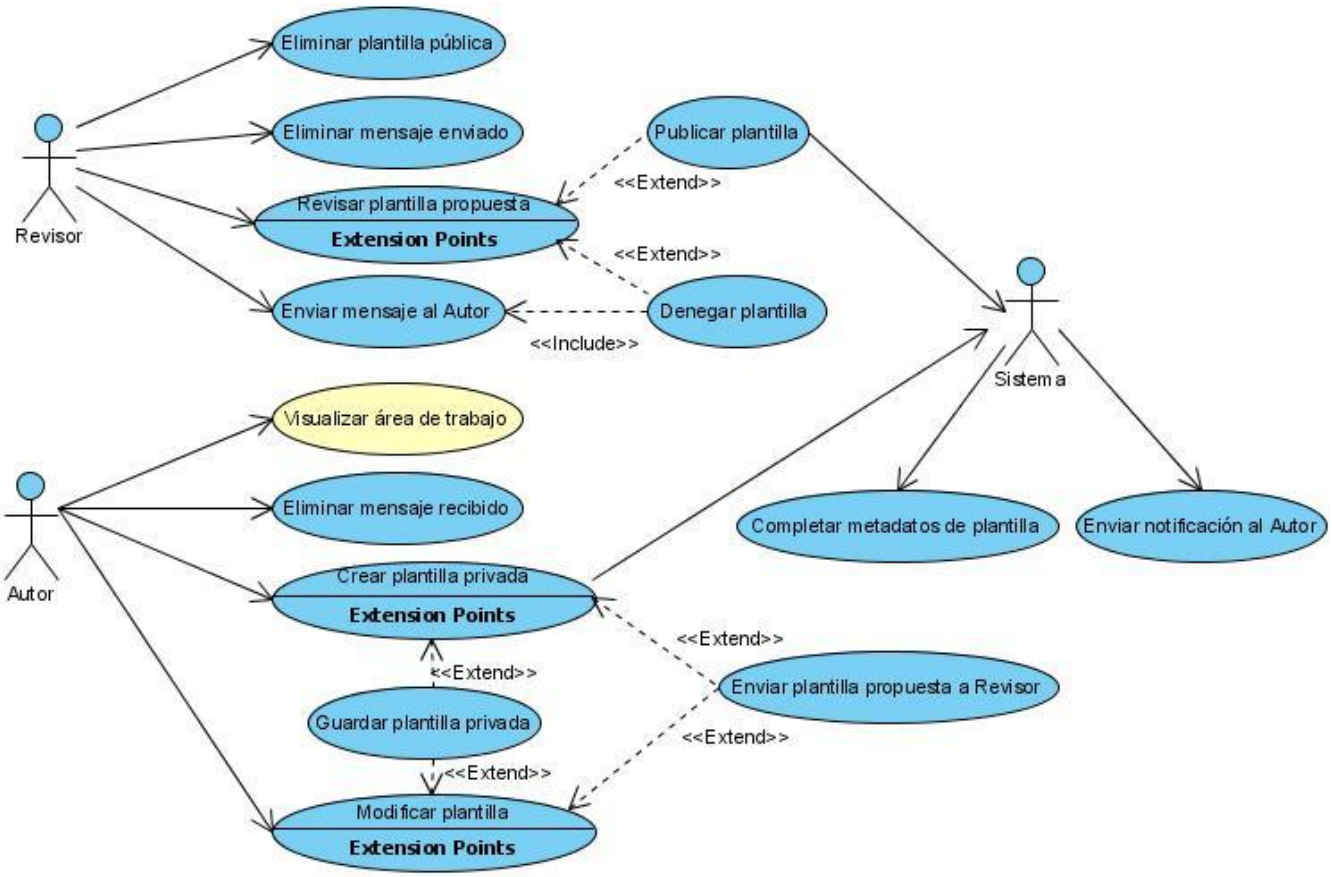

Figura 2.2. Diagrama de Casos de Uso del Subsistema (Crear y revisar OA).

2.6 Descripción Textual de Casos de Uso del Sistema.

R 3. Crear plantilla.

Caso de Uso	Crear Plantilla.
Actores	Autor.
Resumen	El caso de uso inicia cuando el Autor decide crear una nueva plantilla.
Referencia	
Precondiciones	El Autor debe estar autenticado en el sistema.
Poscondiciones	Queda conformada la plantilla diseñada por el autor.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El Autor selecciona la opción *"Nueva plantilla"*.	1.1- La herramienta solicita el nombre y descripción de la plantilla a crear.

2- El Autor introduce los datos requeridos (Nombre y descripción).	
3- El autor selecciona la opción *"Siguiente".*	3.1- La herramienta verifica que los datos se hayan completado. 3.2- La aplicación comprueba si el nombre asignado a la plantilla coincide con otro ya existente. 3.3- Guarda los datos en la Base de Datos. 3.4- Visualiza el área donde definir la estructura de la plantilla, brindando en la misma tres opciones, que se mostraran a través de un clic derecho sobre el nivel al cual se desee o agregarle un nuevo, o eliminarlo o cambiar su nombre. a) Agregar nivel, ver **Sección 1**. b) Eliminar nivel, ver **Sección 2**. a) Cambiar el nombre del nivel, ver **Sección 3**
Sección "Agregar nivel"	
4- El Autor decide agregar un nivel accediendo mediante un clic derecho, y selecciona la opción *"Agregar nivel".*	4.1- La herramienta adiciona el nuevo nivel. 4.2- La herramienta actualiza el manifiesto (XML) donde se almacena la estructura de la plantilla.
Sección "Eliminar nivel"	
4.- El Autor decide eliminar un nivel, para lo cual accede mediante un clic derecho, y selecciona la opción *"Eliminar nivel".*	4.1- La herramienta elimina el nivel. 4.2- La herramienta actualiza el manifiesto (XML) donde se almacena la estructura de la plantilla.
Sección "Cambiar el nombre del nivel"	
4.- El Autor decide cambiar el nombre a un nivel, para lo cual, lo selecciona a través de un clic derecho, y escoge la opción *"Cambiar nombre".*	4.1- La herramienta habilita el campo nombre del nivel para que el autor pueda modificarlo.

5.- El Autor introduce el nombre que desee.	
6- El Autor da clic en cualquier área de la ventana.	6.1- La herramienta cambia el nombre del nivel de organización. 6.2- La herramienta actualiza el manifiesto (XML) donde se almacena la estructura de la plantilla.
Flujos Alternos-Sección "Agregar nivel"	
Acción del Actor	**Respuesta del Sistema**
5.- El Autor no agrega ningún nivel.	5.1-La herramienta conserva la estructura inicial de la plantilla conformada por una organización y un ítem.
Flujos Alternos-Sección "Cambiar el nombre del nivel"	
Acción del Actor	**Respuesta del Sistema**
5.- El Autor no introduce ningún nombre.	5.1-La herramienta conserva el nombre que tenía dicho nivel de organización.
Prioridad	Crítico

Tabla 2.13. Descripción Textual (CU-Crear plantilla).

R 2.2 Completar metadatos de plantilla.

Caso de Uso	Completar metadatos de plantilla.
Actores	Sistema, Autor.
Resumen	El caso de uso inicia cuando el Autor decide crear una plantilla.
Referencia	
Precondiciones	El autor debe estar autenticado en el sistema y haber creado una plantilla.
Poscondiciones	Se automatiza el completamiento de los metadatos.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El Autor selecciona la opción *"Crear plantilla".*	1.1- La herramienta procede de forma interna con el completamiento de metadatos como el identificador, asignando un valor numérico, contribuidor (autor de la plantilla) asumiendo como autor al usuario autenticado en el sistema, además de catálogo, nivel de agregación,

	versión de la plantilla, entidad a la que pertenece, estructura, contexto educacional, contribución, esquema de metadatos, y localización. 1.2- La herramienta actualiza la base de datos.
Prioridad	Auxiliar

Tabla 2.14. Descripción Textual (CU-Completar metadatos de plantilla).

R 3. Guardar plantilla privada.

Caso de Uso	Guardar plantilla privada.
Actores	Autor
Resumen	El caso de uso inicia cuando el Autor decide guardar una plantilla privada que haya creado o alguna que haya modificado.
Referencia	
Precondiciones	El autor debe estar autenticado en el sistema y haber creado o modificado una plantilla.
Poscondiciones	Se archiva la plantilla en *"Plantillas privadas".*
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El Autor selecciona la opción *"Guardar plantilla".*	1.1- La herramienta almacena dicha plantilla, en la sección perteneciente al autor *"Plantillas privadas".* 1.2- La herramienta muestra un mensaje de éxito en el almacenamiento de dicha plantilla, *"Su plantilla ha sido almacenada con éxito"*.
Prioridad	Auxiliar

Tabla 2.15. Descripción Textual (CU-Guardar plantilla privada).

R 4. Enviar plantilla propuesta al Revisor.

Caso de Uso	Enviar plantilla propuesta a Revisor.
Actores	Autor.
Resumen	El caso de uso inicia cuando el Autor decide proponer para el área de *"Plantillas públicas"* alguna plantilla que haya creado o modificado.
Referencia	
Precondiciones	El autor debe haber creado o modificado una plantilla.

Poscondiciones	Se almacena la plantilla en *"Plantillas privadas"* y se envía al revisor.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El Autor se dirige a la opción "*Enviar al revisor*".	1.1- La herramienta almacena dicha plantilla en la sección perteneciente al autor, *"Plantillas privadas"* y envía dicha plantilla a la lista de plantillas propuestas para el Revisor. 1.2- La herramienta muestra un mensaje de éxito en el almacenamiento y envío de dicha plantilla, *"Su plantilla ha sido guardada y enviada con éxito"*.
Prioridad	Auxiliar

Tabla 2.16. Descripción Textual (CU-Enviar plantilla propuesta al Revisor).

R 5. Modificar plantilla.

Caso de Uso	Modificar Plantilla.
Actores	Autor.
Resumen	El caso de uso inicia cuando el Autor decide Modificar una plantilla para la creación de objetos de aprendizaje o con el objetivo de proponerla para publicarla.
Referencia	
Precondiciones	El Autor debe estar autenticado en el sistema y deben existir plantillas públicas o privadas en la herramienta.
Poscondiciones	Se realizan los cambios deseados por el Autor.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El Autor selecciona la opción *"Modificar plantilla".*	1.1- La herramienta muestra un formulario con todas las plantillas creadas por el autor además de las públicas que ofrezca la herramienta.
2- El Autor selecciona la plantilla que desea modificar.	2.1- La herramienta muestra el árbol de organización de contenido de la plantilla. 2.2- La herramienta solicita el *nombre y descripción* de la

	plantilla a modificar.
3- El Autor introduce los datos requeridos (*nombre y descripción*).	3.1 La herramienta comprueba si el nombre asignado a la plantilla coincide con otro ya existente. 3.2 Guarda los datos en la Base de Datos. 3.3 Visualiza el área donde definir la estructura de la plantilla, brindando en la misma tres opciones: a) *"Agregar nivel"*, ver **Sección 1.** b) *"Eliminar nivel"*, ver **Sección 2.** c) *"Cambiar el nombre del nivel"*, ver **Sección 3.**
Sección "Agregar nivel"	
4- El Autor decide agregar un nivel accediendo mediante un clic derecho, y selecciona la opción *"Agregar nivel".*	4.1- La herramienta adiciona el nuevo nivel. 4.2- La herramienta actualiza el manifiesto (XML) donde se almacena la estructura de la plantilla.
Sección "Eliminar nivel"	
4.- El Autor decide eliminar un nivel, para lo cual accede mediante un clic derecho, y selecciona la opción *"Eliminar nivel".*	4.1- La herramienta elimina el nuevo nivel. 4.2- La herramienta actualiza el manifiesto (XML) donde se almacena la estructura de la plantilla.
Sección "Cambiar el nombre del nivel"	
4.- El Autor decide cambiar el nombre a un nivel, para lo cual, lo selecciona a través de un clic derecho, y escoge la opción *"Cambiar nombre".*	4.1- La herramienta muestra ofrece la posibilidad de cambiar el nombre al nivel.
5.- El Autor introduce el nombre que desee.	5.1- La herramienta cambia el nombre del nivel de organización. 5.2- La herramienta actualiza el manifiesto (XML) donde se almacena la estructura de la plantilla.
Flujos Alternos-Sección "Agregar nivel"	

Acción del Actor	Respuesta del Sistema
5.- El Autor no agrega ningún nivel.	5.1-La herramienta conserva la estructura inicial de la plantilla conformada por una organización y un ítem.
Flujos Alternos-Sección "Cambiar el nombre del nivel"	
Acción del Actor	**Respuesta del Sistema**
5.- El Autor no introduce ningún nombre.	5.1-La herramienta conserva el nombre que tenía dicho nivel de organización.
Prioridad	Crítico

Tabla 2.17. Descripción Textual (CU-Modificar plantilla).

R 7. Revisar plantilla.

Caso de Uso	**Revisar plantilla.**
Actores	Revisor.
Resumen	El caso de uso se refiere a la revisión de una plantilla propuesta por el autor, para su publicación o denegación.
Referencia	
Precondiciones	El Revisor debe estar autenticado en el sistema y debe existir alguna plantilla propuesta por algún autor para revisar.
Poscondiciones	Se publica o deniega la plantilla propuesta.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1-El Revisor selecciona la plantilla a revisar.	1.1-La herramienta visualiza dicha plantilla en detalle y brinda dos opciones: *a) "Publicar plantilla"*, ver **Sección 1**. *b) "Denegar plantilla",* ver **Sección 2**.
Sección "Publicar Plantilla"	

2. El Revisor selecciona la opción *"Publicar plantilla"*.	2.1 La herramienta cambia el estado de la plantilla en la Base de Datos (de propuesta a pública). 2.2- La herramienta muestra el siguiente mensaje al Revisor *"La plantilla ha sido publicada".* 2.3- La herramienta envía un mensaje predeterminado informando al autor sobre la publicación de dicha plantilla. 2.4- La herramienta visualiza el formulario actualizado del resto de las plantillas a revisar.
Sección "Denegar Plantilla"	
3. El Revisor selecciona la opción *"Denegar plantilla".*	3.1. La herramienta muestra un formulario de edición de un mensaje para el autor con un campo para el destinatario (Autor de la plantilla propuesta), una caja de texto, en la cual el Revisor explica las causas de la denegación, un botón *"Enviar"* y otro *"Cancelar".*
4. El Revisor redacta el mensaje. 5. El Revisor accede al botón *"Enviar".*	5.1. La herramienta envía el mensaje al autor. 5.2. Visualiza el formulario actualizado del resto de las plantillas a revisar.
Flujos Alternos-"Sección Denegar Plantilla"	
Acción del Actor	**Respuesta del Sistema**
5. El revisor accede al botón *"Cancelar"*	5.1. **Ir al paso 1.1.**
Prioridad	Crítico

Tabla 2.18. Descripción Textual (CU-Revisar plantilla).

R 14. Eliminar plantilla pública.

Caso de Uso	**Eliminar plantilla pública.**
Actores	Revisor.
Resumen	El caso de uso inicia cuando el Revisor decide eliminar una plantilla publicada en la herramienta.
Referencia	
Precondiciones	El Revisor debe estar autenticado en el sistema.
Poscondiciones	Se elimina la plantilla pública.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El Revisor *accede al botón "Eliminar"* a través de un clic derecho el cual visualiza la opción.	1.1- El sistema elimina la plantilla del apartado de plantillas públicas. 1.2- El sistema actualiza la base de datos.
Prioridad	Auxiliar

Tabla 2.19. Descripción Textual (CU-Eliminar plantilla pública).

R 26. Guardar objeto de aprendizaje en repositorio.

Caso de Uso	**Guardar objeto se aprendizaje en repositorio.**
Actores	Autor.
Resumen	El caso de uso inicia cuando el Autor decide guardar un objeto de aprendizaje creado anteriormente con vista a la reutilización del propio objeto de aprendizaje.
Referencia	
Precondiciones	El autor debe estar autenticado en el sistema y haber creado o modificado el objeto de aprendizaje.
Poscondiciones	Se archiva el objeto en el repositorio.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El Autor se dirige a la opción *"Guardar en repositorio".*	1.1- La Herramienta se comunica a través de un servicio web con el repositorio, el cual le retorna la petición de usuario y contraseña del repositorio al Autor.
2- El Autor introduce usuario y contraseña.	2.1- La herramienta envía los datos al repositorio, este verifica su validez y retorna la petición de la ubicación del paquete a subir.
3- El Autor introduce la dirección del objeto de aprendizaje.	3.1- La herramienta envía el objeto de aprendizaje al repositorio. 3.2- La herramienta muestra un mensaje de correcto envío hacia el repositorio. *"Su objeto de aprendizaje ha sido almacenado con éxito"*
Flujos Alternos	
2- El Autor introduce incorrectamente los	2.1- La Herramienta se comunica a través de un servicio web con el repositorio, el cual le retorna el mensaje *"Su usuario o contraseña*

datos de usuario y contraseña.	*no es correcta"*. **Ir al paso 1.1**
Prioridad	Auxiliar

Tabla 2.20. Descripción Textual (CU-Guardar OA en repositorio).

R 27. Descargar objeto de aprendizaje.

Caso de uso	Descargar objeto de aprendizaje.
Actores	Autor.
Resumen	El caso de uso se inicia cuando el usuario escoge la opción Descargar objeto de aprendizaje ofreciéndole la posibilidad de almacenar el mismo en su máquina y finaliza luego de haber descargado el objeto de aprendizaje.
Referencias	
Precondiciones	El usuario debe haberse autenticado y encontrarse creando o editando el objeto de aprendizaje o seleccionarlo de su lista de objetos de aprendizaje en edición.
Poscondiciones	Se almacena el objeto de aprendizaje en la computadora del autor.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El Autor escoge la opción *"Descargar objeto de aprendizaje".*	1.1 El sistema le solicita al autor la dirección del lugar donde desea guardar el objeto de aprendizaje.
2. El Autor introduce la dirección.	2.1 El sistema notifica el resultado de la operación, *"Su objeto de aprendizaje ha sido descargado exitosamente".*
Prioridad	Auxiliar

Tabla 2.21. Descripción Textual (CU-Descargar OA).

R 22. Buscar objeto de aprendizaje en repositorio.

Caso de Uso	Buscar en el repositorio.
Actores	Autor.
Resumen	El caso de uso inicia cuando el Autor decide reutilizar un objeto de aprendizaje creado anteriormente y

	almacenado en el repositorio con vista a la creación de uno nuevo.
Referencia	
Precondiciones	El autor debe estar autenticado en el sistema y debe encontrarse creando o editando un objeto de aprendizaje.
Poscondiciones	Se lista los objetos de aprendizaje que cumplen con el criterio de búsqueda.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El Autor selecciona la opción *"Búsqueda en repositorio".*	1.1- La herramienta ofrece 2 opciones de búsqueda: a) *"Búsqueda general"*, ver **sección 1.** b) *"Búsqueda avanzada"*, ver **sección 2.**
Sección "Búsqueda general"	
3- El Autor elije la opción *"Búsqueda general".*	3.1- El sistema se comunica con el repositorio a través de un servicio web el cual retorna una petición del nombre del objeto de aprendizaje que desea el autor.
4- El Autor introduce el nombre del objeto de aprendizaje.	4.1- El sistema envía el nombre del objeto al repositorio, este realiza la búsqueda y le retorna el objeto u objetos de aprendizaje relacionados con el nombre que introdujo el autor.
Sección "Búsqueda avanzada"	
5- El Autor elije la opción *"Búsqueda avanzada".*	3.1- El sistema se comunica con el repositorio a través de un servicio web el cual retorna una interfaz con los campos por los cuales se puede realizar la búsqueda.
6- El Autor escoge los campos por los cuales realizara la búsqueda introduciendo los datos.	6.1 El sistema envía dichos datos al repositorio y este devuelve el resultado de la búsqueda.
Prioridad	Auxiliar

Tabla 2.22. Descripción Textual (CU-Buscar OA en repositorio).

2.7 Conclusiones

En este capítulo se mostraron las principales clases del dominio para una mayor comprensión del negocio, se justificó la selección de los actores y trabajadores que intervienen; además se analizaron las características y funciones fundamentales del sistema para el diseño flexible de los OA así como su reutilización a través de la búsqueda y almacenaje en el repositorio ROA, los cuales se representaron mediante los diagramas de casos de uso de los subsistemas *"Crear y Revisar OA"* y *"Buscar y Almacenar OA"*, arribando de esta forma a las descripciones detalladas de cada caso de uso del sistema. Una vez realizado esto es posible comenzar a realizar el análisis y diseño de la aplicación teniendo en cuenta los requerimientos especificados en el capítulo.

Capítulo III Análisis y Diseño del Sistema.

3.1 Introducción.

El análisis y diseño constituyen partes fundamentales dentro del proceso de desarrollo de software. El objetivo del análisis es identificar un esbozo preliminar del comportamiento requerido a partir de los elementos de modelación del sistema. El diseño por su parte transforma este esbozo preliminar y algunas veces idealizado en un conjunto de elementos del modelo que serán posteriormente implementados. Las clases del análisis son con frecuencia lo suficientemente fluidas, cambiables y evolucionan satisfactoriamente antes de fraguarse en actividades del diseño. En el presente capítulo se plantea el análisis y el diseño del sistema, utilizando para su modelación el UML y quedando plasmados los diagramas de clases del análisis, de clases del diseño y de interacción como artefactos fundamentales en esta fase. Los mismos han sido separados por casos de uso para facilitar su comprensión.

3.2 Modelo de Análisis.

El modelo de análisis contiene clases del análisis incluyendo artefactos asociados. Dicho modelo puede ser un artefacto temporal, razón por la que el mismo evoluciona hacia el modelo de diseño o utilizándose como un modelo conceptual general del sistema.

Encontrar un conjunto candidato de clases del análisis es el primer paso en la transformación del sistema desde el mero estado de comportamiento requerido a una descripción de cómo el sistema trabajará. En este esfuerzo, las clases del análisis son usadas para representar los roles de los elementos del modelo, los cuales proveerán el comportamiento necesario para satisfacer los requerimientos funcionales especificados por los casos de uso y los no funcionales especificados por los requerimientos adicionales.

Las clases *interfaz* y de *control* típicamente evolucionan a elementos de diseño de la capa de aplicación, mientras que las clases *entidad* evolucionan a elementos de diseño específicos del dominio. Más allá de ofrecer una guía específica a la hora de encontrar clases, estos estereotipos (*interfaz, control y entidad*) resultan en un modelo de objeto robusto, pues los cambios en el modelo tienden a afectar solamente un área específica. Los cambios en la interfaz de usuario, por ejemplo, afectarán solamente las clases interfaz. Los cambios en el flujo de control afectarán solamente las

clases de control, mientras que los cambios en la información a largo plazo solamente afectarán las clases entidades.

3.2.1 Diagrama de Clases del Análisis.

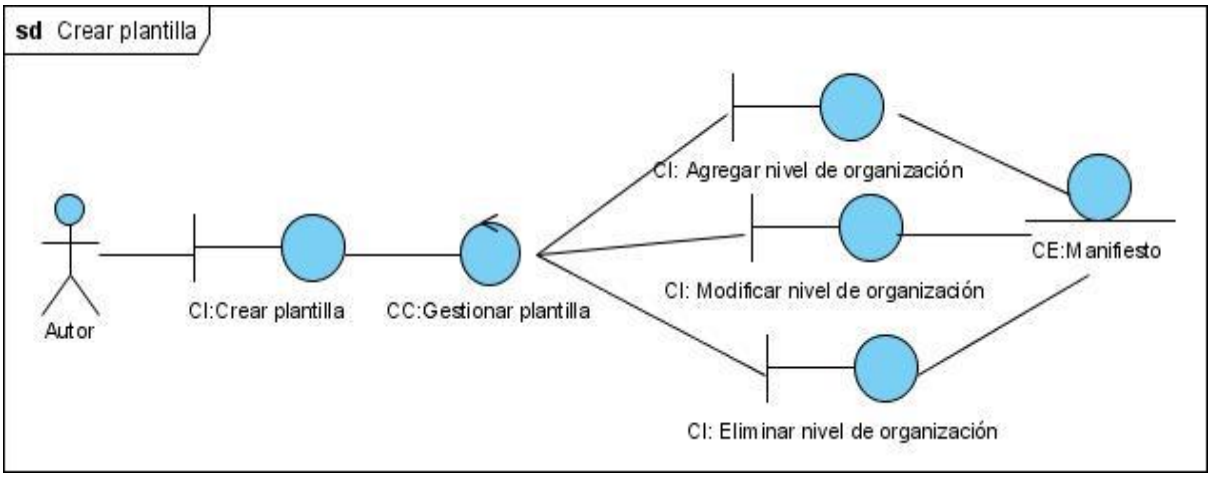

Figura 3.1. Diagrama de Clases del Análisis (CU-Crear Plantilla).

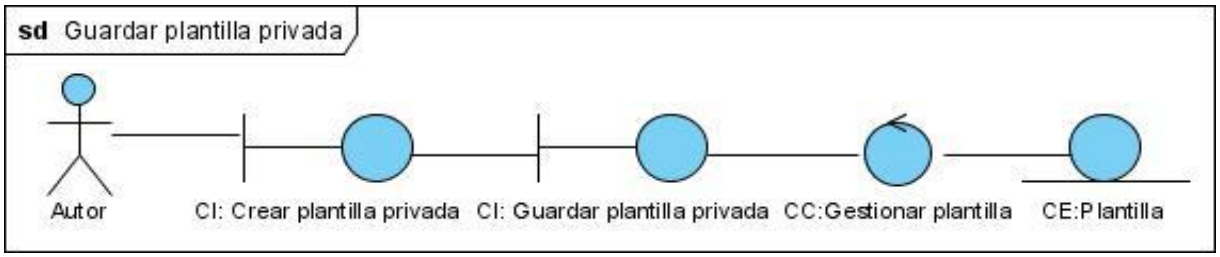

Figura 3.2. Diagrama de Clases del Análisis (CU-Guardar plantilla privada).

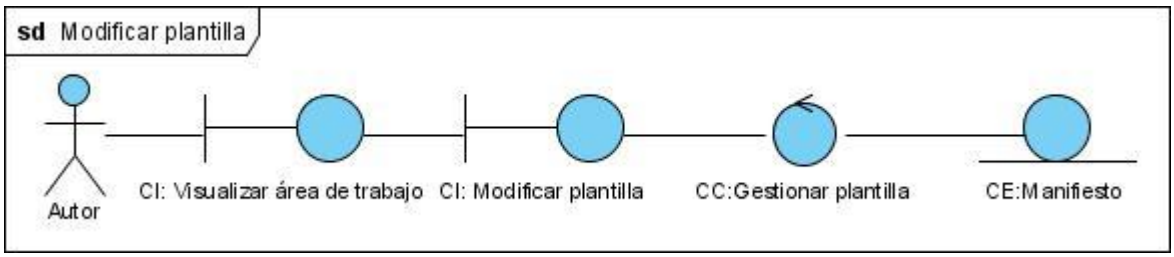

Figura 3.3. Diagrama de Clases del Análisis (CU-Modificara plantilla).

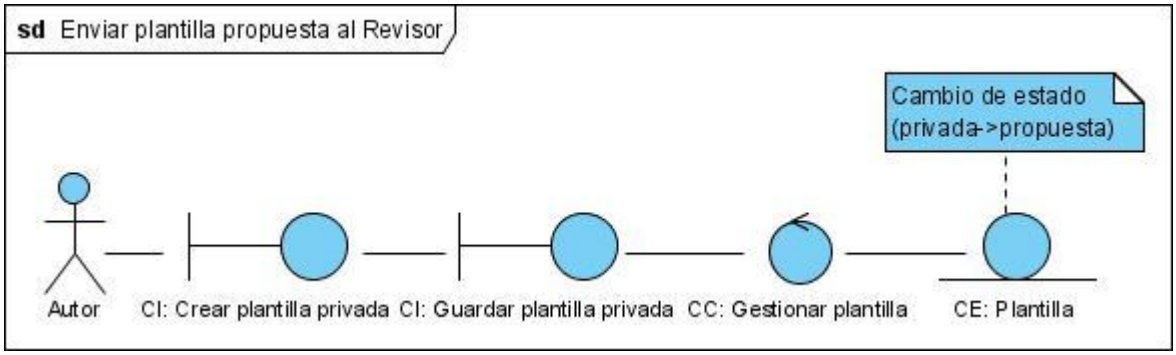

Figura 3.4. Diagrama de Clases del Análisis (CU-Enviar plantilla propuesta al Revisor).

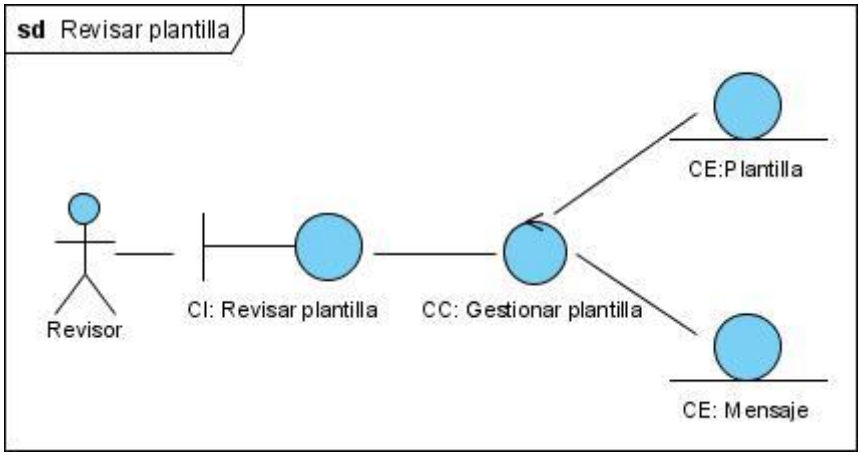

Figura3.5. Diagrama de Clases del Análisis (CU-Eliminar plantilla pública).

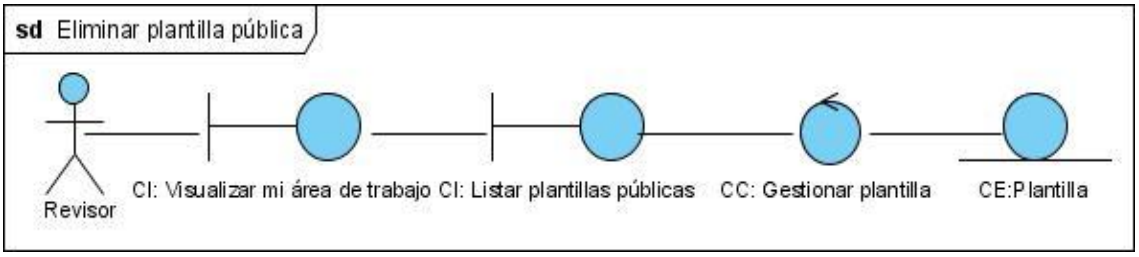

Figura 3.6. Diagrama de Clases del Análisis (CU-Eliminar plantilla pública).

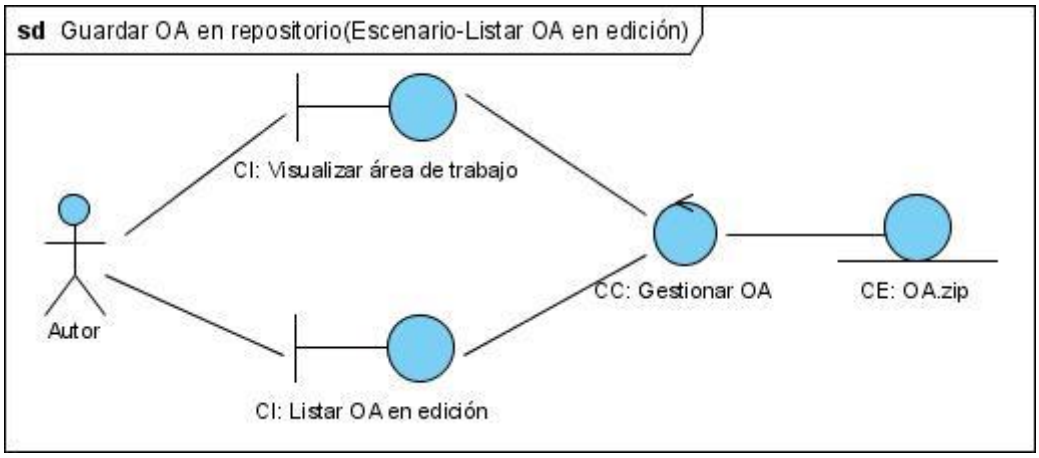

Figura 3.7. Diagrama de Clases del Análisis (CU-Guardar OA en repositorio-Escenario-Listar OA en edición).

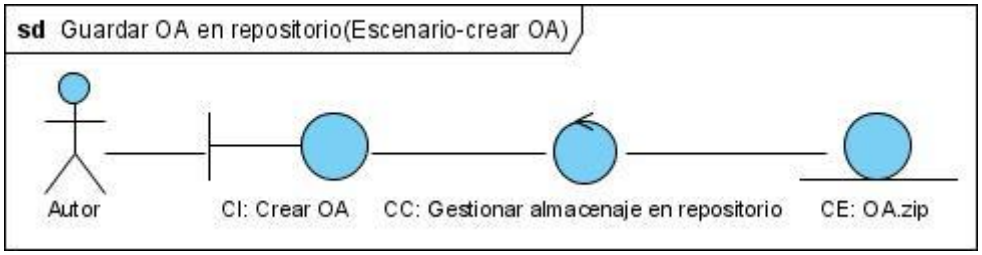

Figura 3.8. Diagrama de Clases del Análisis (CU-Guardar OA en repositorio-Escenario-Crear OA).

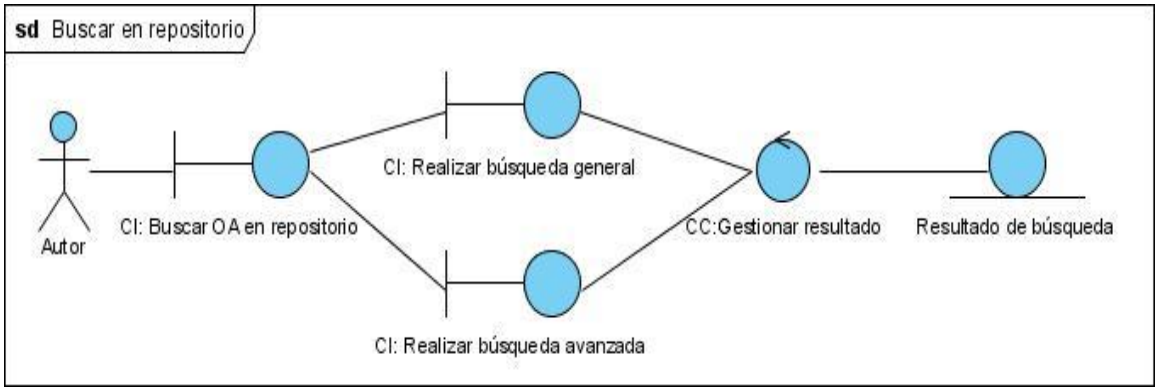

Figura 3.9. Diagrama de Clases del Análisis (CU-Buscar en repositorio).

3.3 Diseño.

El diseño tiene el propósito de formular los modelos que se centran en los requisitos no funcionales y en el dominio de la solución y que prepara para la implementación y prueba del sistema. Pretende crear un plano del modelo de implementación, por lo que el grueso del esfuerzo está en las últimas iteraciones de elaboración y las primeras de construcción.

3.3.1 Diagramas de interacción.

Los diagramas de interacción se utilizan para modelar los aspectos dinámicos de un sistema, lo que conlleva a modelar instancias concretas de clases interfaces, componentes y nodos, junto con los mensajes enviados entre ellos, todo en el contexto de un escenario que ilustra un comportamiento. (Fernández, 2001) En el contexto de las clases describen la forma en que grupos de objetos colaboran para proveer un comportamiento.

En los diagramas de interacción se muestra un patrón de interacción entre objetos. Existen dos tipos de diagramas de interacción, ambos basados en la misma información, pero cada uno enfatizando un aspecto particular: Diagramas de Secuencia y Diagramas de Colaboración. En este trabajo se utilizará para la modelación del diseño de la herramienta los diagramas de secuencia, debido al nivel de detalle que describen, proporcionando un mayor flujo de información para el desarrollo de la herramienta.

3.3.1.1 Diagramas de Secuencia.

El diagrama de secuencia es uno de los diagramas más efectivos para modelar interacción entre objetos en un sistema. Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso del sistema, este además contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes pasados entre los objetos. A continuación las figuras de los distintos diagramas de secuencia de la herramienta *"ROXS".*

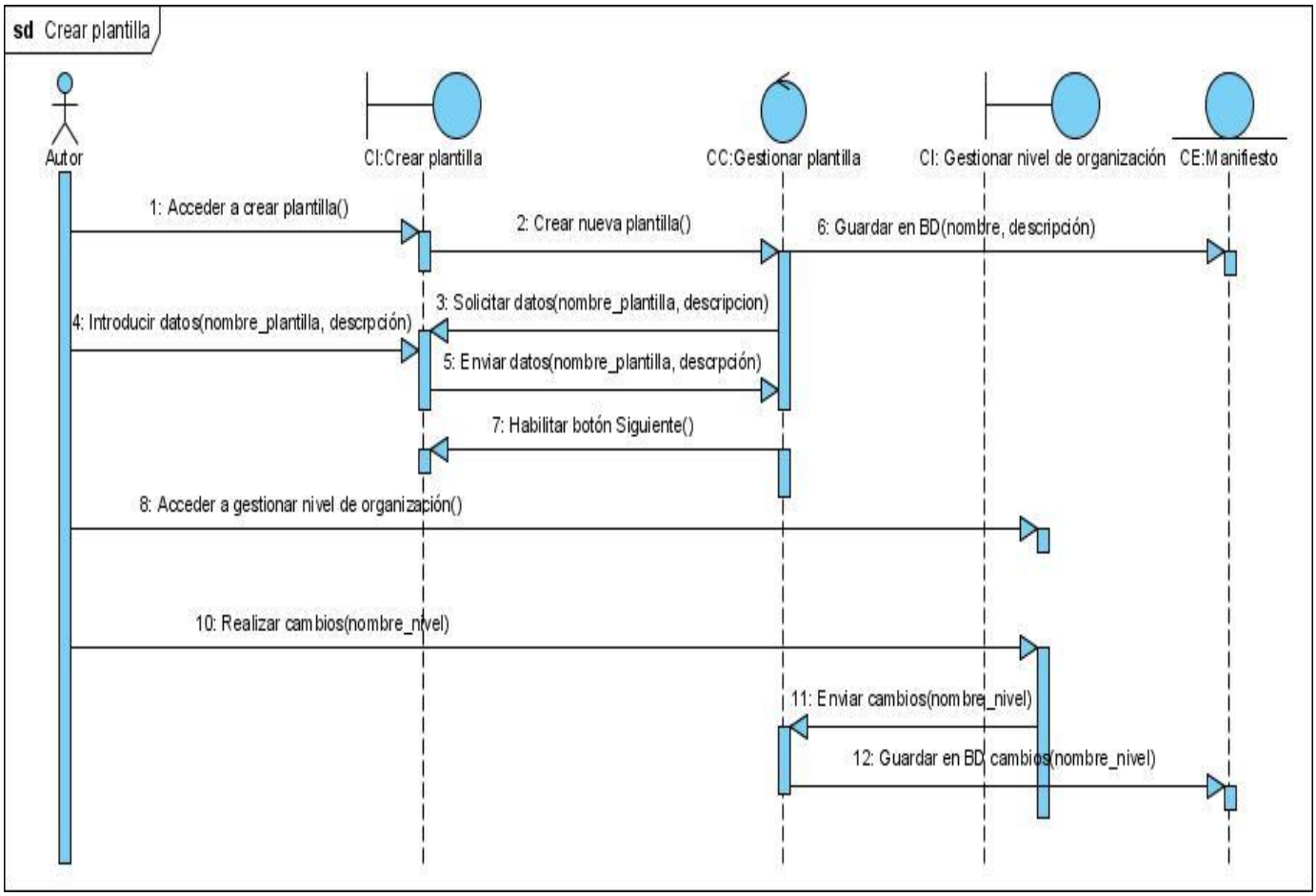

Figura 3.10. Diagrama de Secuencia (CU-Crear plantilla).

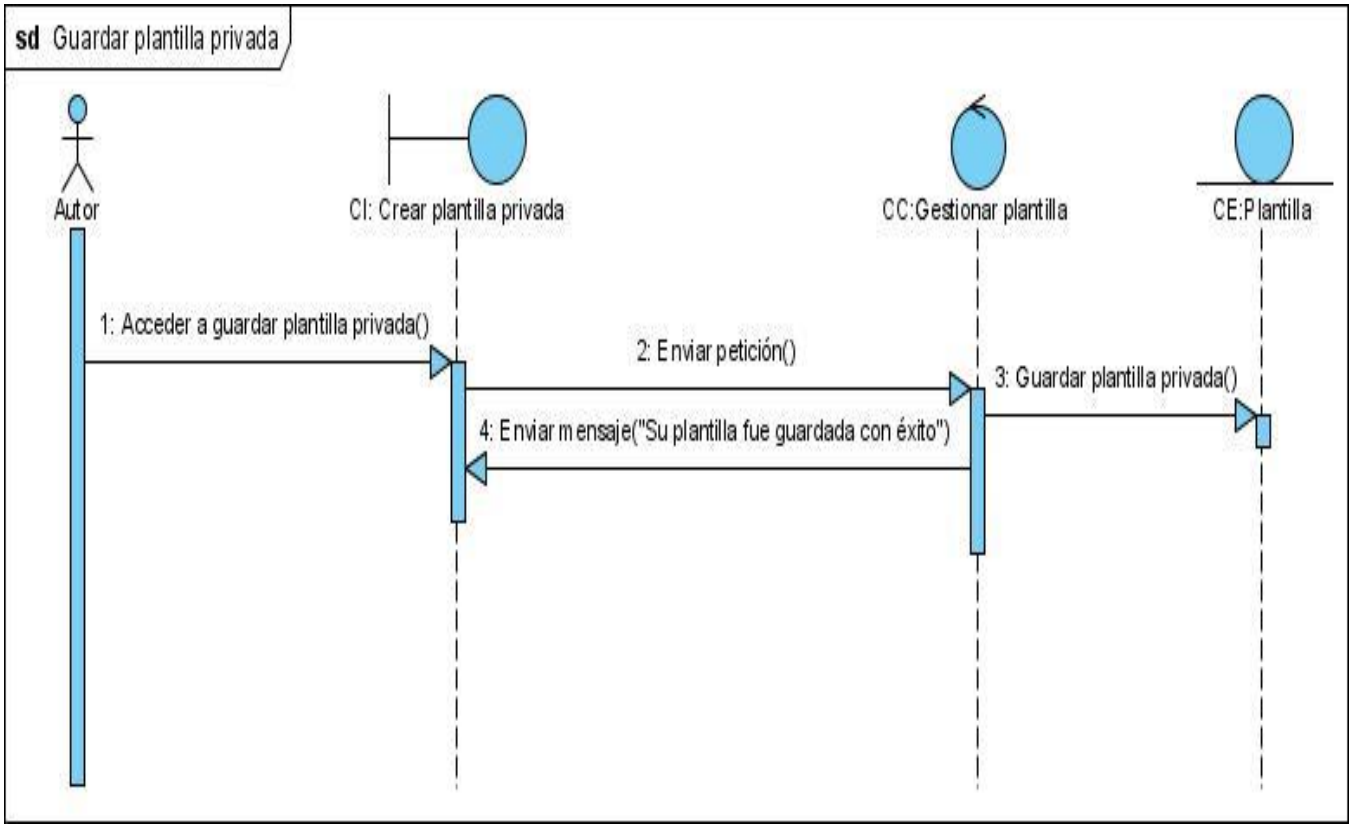

Figura 3.11. Diagrama de Secuencia (CU-Guardar plantilla privada).

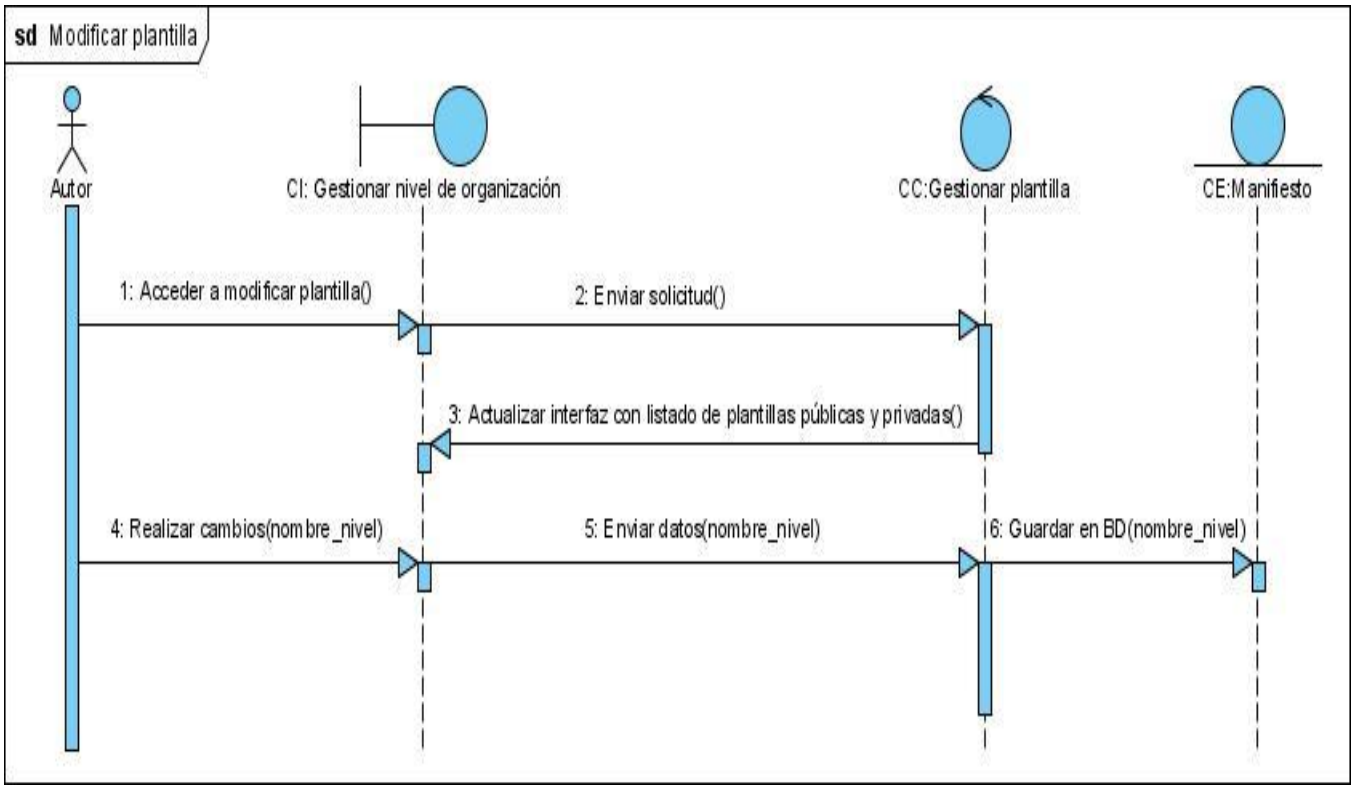

Figura 3.12. Diagrama de Secuencia (CU-Modificar plantilla).

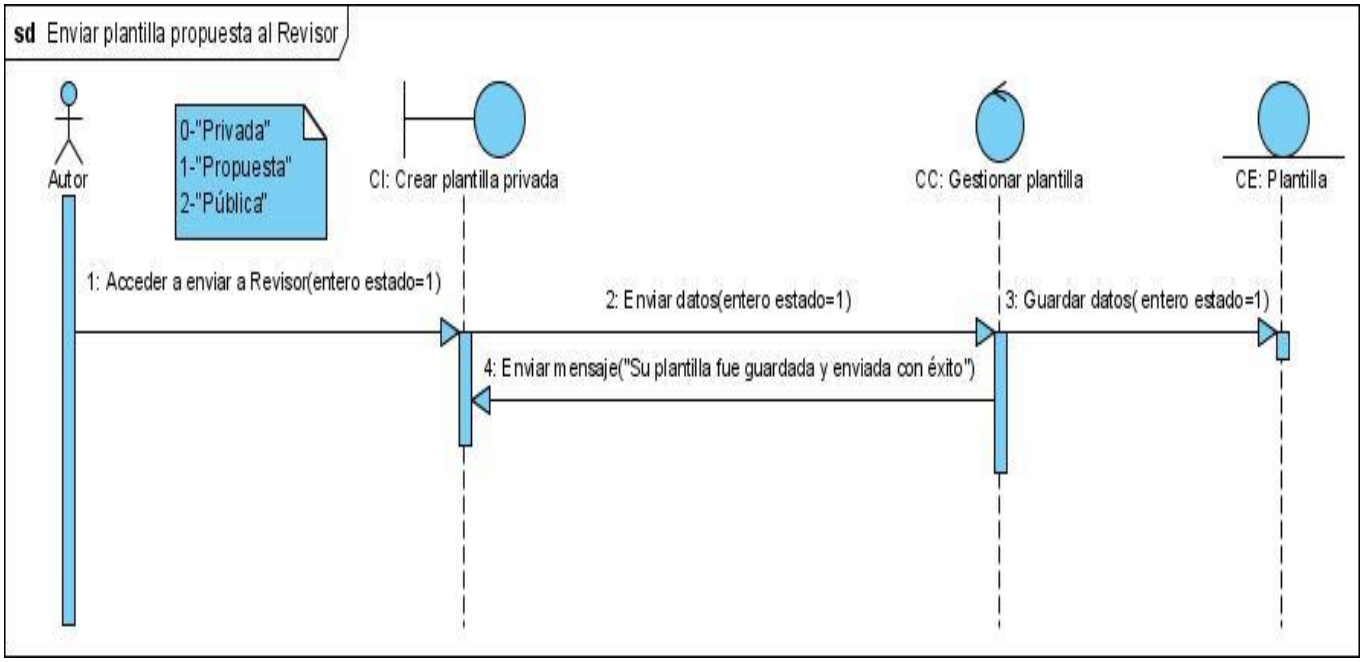

Figura 3.13. Diagrama de Secuencia (CU-Enviar plantilla propuesta al Revisor).

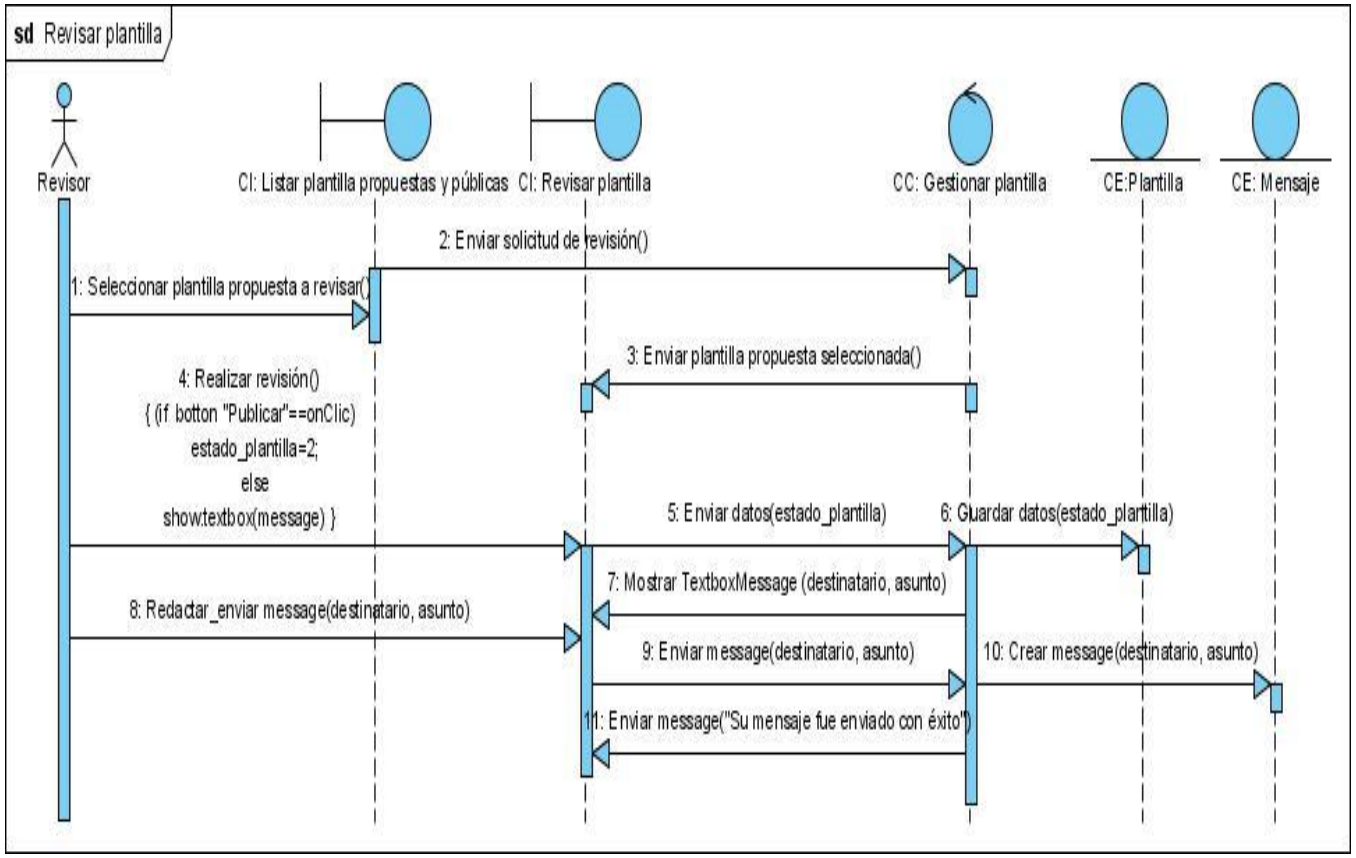

Figura 3.14. Diagrama de Secuencia (CU-Revisar plantilla).

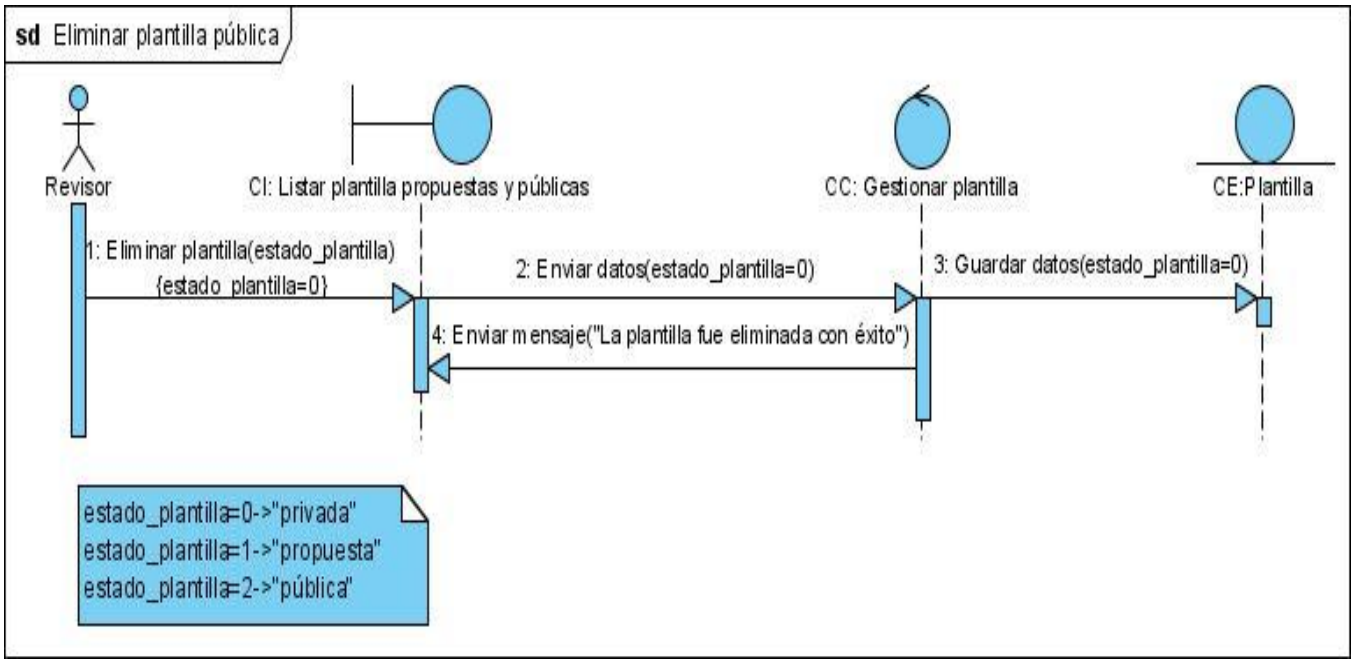

Figura 3.15. Diagrama de Secuencia (CU-Eliminar plantilla pública).

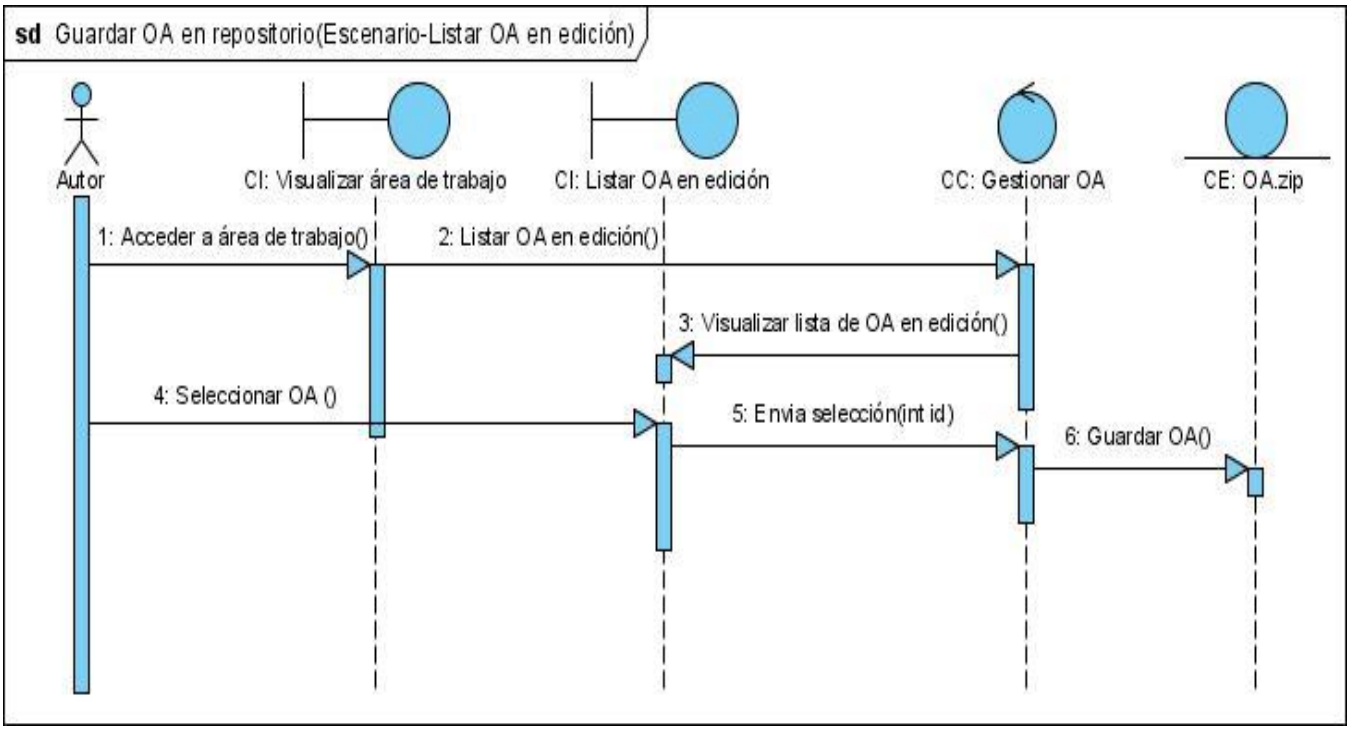

Figura 3.16. Diagrama de Secuencia (CU-Guardar OA en repositorio-Escenario-Listar OA en edición).

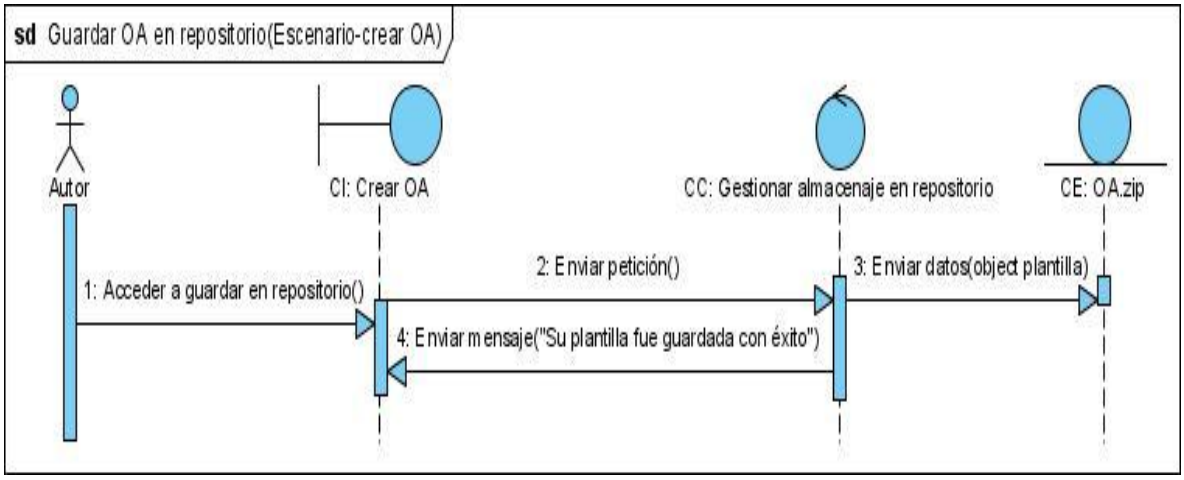

Figura 3.17. Diagrama de Secuencia (CU-Guardar OA en repositorio-Escenario-Crear OA).

3.3.2 Diagrama de clases.

Los diagramas de clases son diagramas de estructura estática que muestran las clases del sistema y las relaciones entre ellas. Los diagramas de clase son el pilar básico del modelado con UML, siendo utilizados tanto para mostrar lo que el sistema puede hacer, como para mostrar cómo puede ser construido. Cuando se crea un diagrama de clases, se está modelando una parte de los elementos y relaciones que configuran la vista de diseño del sistema.

El diseño del sistema a construir es un diseño basado en la Web puesto que la solución informática a implementar es una aplicación Web. Sin embargo, en la práctica existen problemas cuando se trata de utilizar el diagrama de clases tradicional para modelar aplicaciones Web. Si se mirara una clase como una página Web en el modelo, habría una confusión pues no se sabría distinguir cuáles atributos, operaciones y relaciones están activas en el servidor y cuáles están activas cuando el usuario está interactuando con la página en el navegador cliente. Por este motivo, no se puede mapear una página Web como una clase UML pues no se alcanzaría una correcta comprensión del sistema.

Actualmente el UML brinda solución para modelar aplicaciones Web, para esto tiene varios mecanismos de extensión los cuales están compuestos por estereotipos, valores etiquetados y restricciones. En la presente investigación se construye una propuesta para el diagrama de clases del diseño basándose en la implementación que realiza el framework Symfony de la arquitectura Modelo-Vista-Controlador, lo cual se evidencia en la siguiente figura:

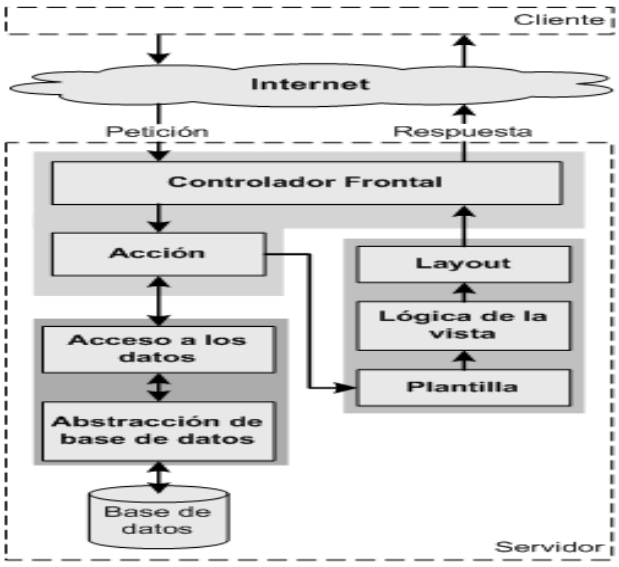

Figura 3.18. Implementación del patrón MVC utilizado por Symfony.

En la siguiente figura se muestra el diagrama de paquetes del diseño:

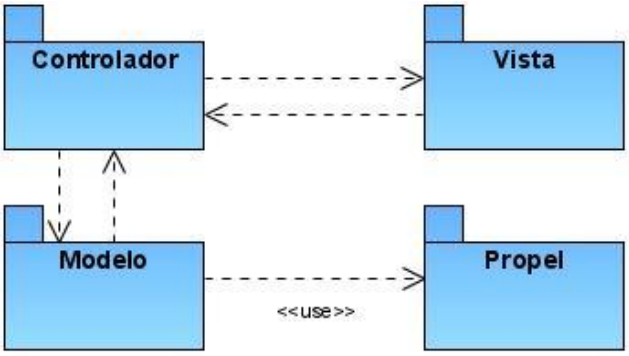

Figura 3.19. Diagrama de Paquetes.

Se presenta a continuación el diseño de la aplicación.

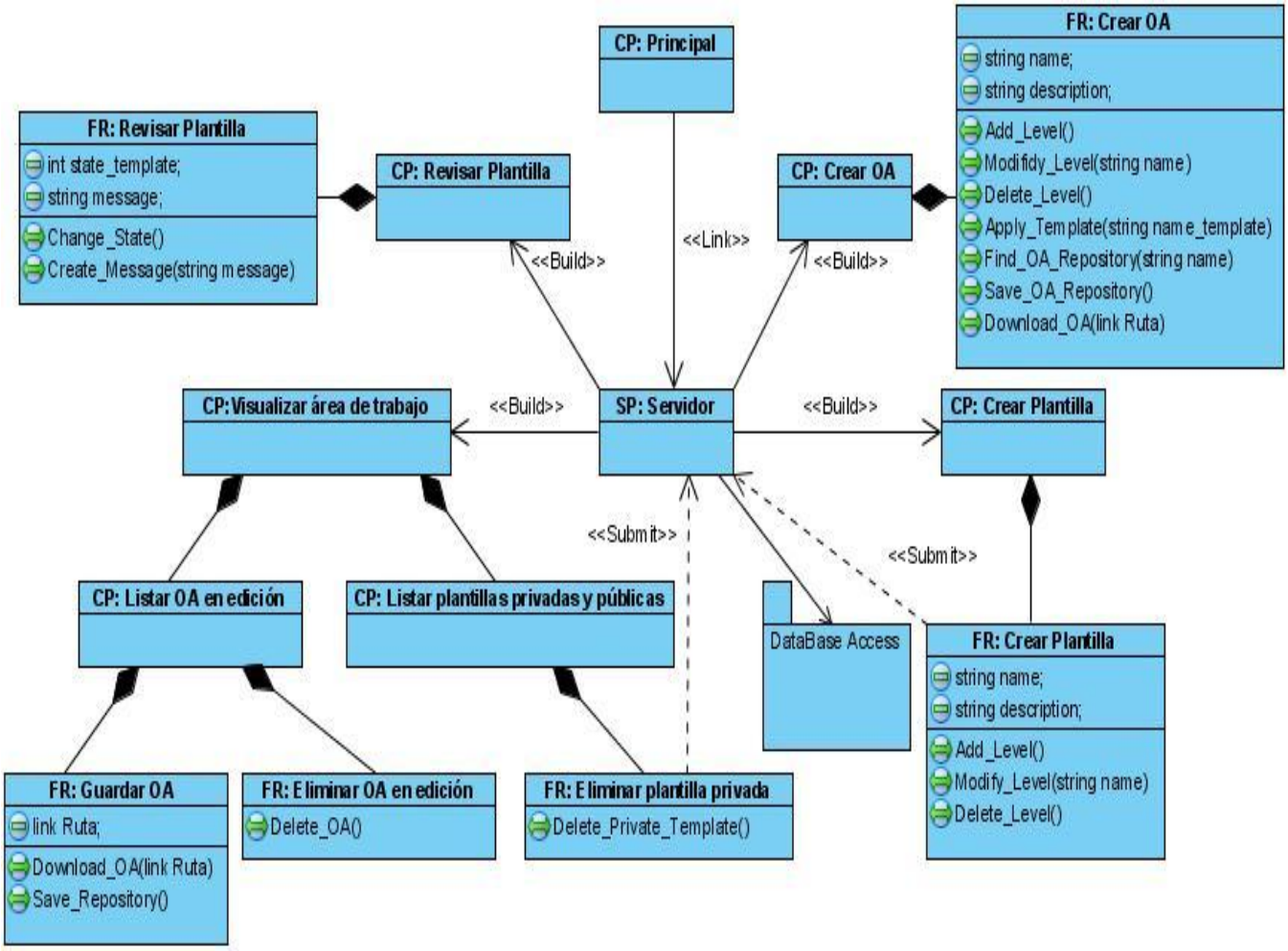

Figura 3.20. Diagrama de Clases.

3.3.2.1 Descripción de Clases

Capa Revisar plantilla

Propósito

Es la encargada de gestionar todo el proceso de revisión de la plantilla, conteniendo funciones como listar plantillas propuestas, que serian las plantillas a revisar, visualizar determinada plantilla seleccionada por el revisor con vista a publicarla o denegar la misma y además en esta se crea el mensaje para el autor con la razones expuestas en caso de que se denegara su plantilla.

Descripción

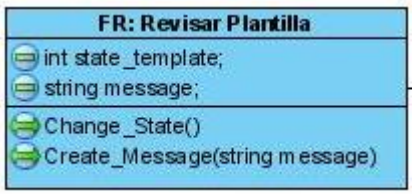

Figura. 3.20.1 Clase - Revisar plantilla.

Observaciones

- En caso de que la plantilla se publique el sistema automáticamente enviará una notificación al autor de la misma informándole la decisión.

Capa Crear OA

Propósito

Es la encargada de gestionar todo el proceso de creación de la estructura del árbol de organización de contenido del OA, además ofrece la posibilidad de almacenar el mismo en el repositorio ROA, buscar OA previamente creados y almacenados en dicho repositorio con vista a su reutilización, conteniendo funciones como aplicar plantilla, que consiste en copiar la estructura de la plantilla a la del OA.

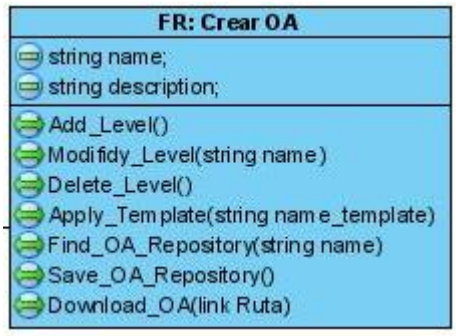

Figura. 3.20.2 Clase – Crear OA.

Observaciones

- En versión anterior ya se desarrollo todo el proceso de creación de contenido (creación de preguntas, expresiones matemáticas, importación de recursos, ect.)

Capa Crear plantilla

Propósito

Es la encargada de gestionar todo el proceso de creación de la plantilla para lo cual se podrá agregar el numero de niveles que se desee modificarlos o a su vez eliminar un determinado nivel quedando estructurada la plantilla.

Descripción

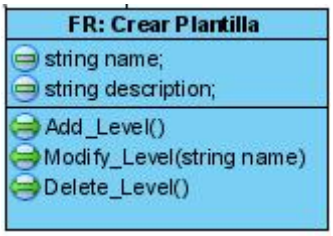

Figura. 3.20.3 Clase - Crear plantilla.

Observaciones

- Al culminar la creación de la plantilla, ésta podrá ser propuesta para revisión y posible publicación.

Capa Buscar OA en repositorio

Propósito

Se encargará de gestionar la búsqueda de OA en ROA, devolviendo los resultados de la misma según los campos introducidos.

Descripción

Figura. 3.20.4 Clase – Buscar OA en repositorio.

Capa Guardar OA en repositorio

Propósito

A través de esta clase se realiza el almacenaje del OA, el cual puede ser descargando el OA hacia la PC del Autor o enviándolo al repositorio.

Descripción

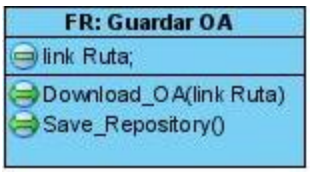

Figura. 3.20.5 Clase – Guardar OA.

3.4 Diseño de la Base de Datos.

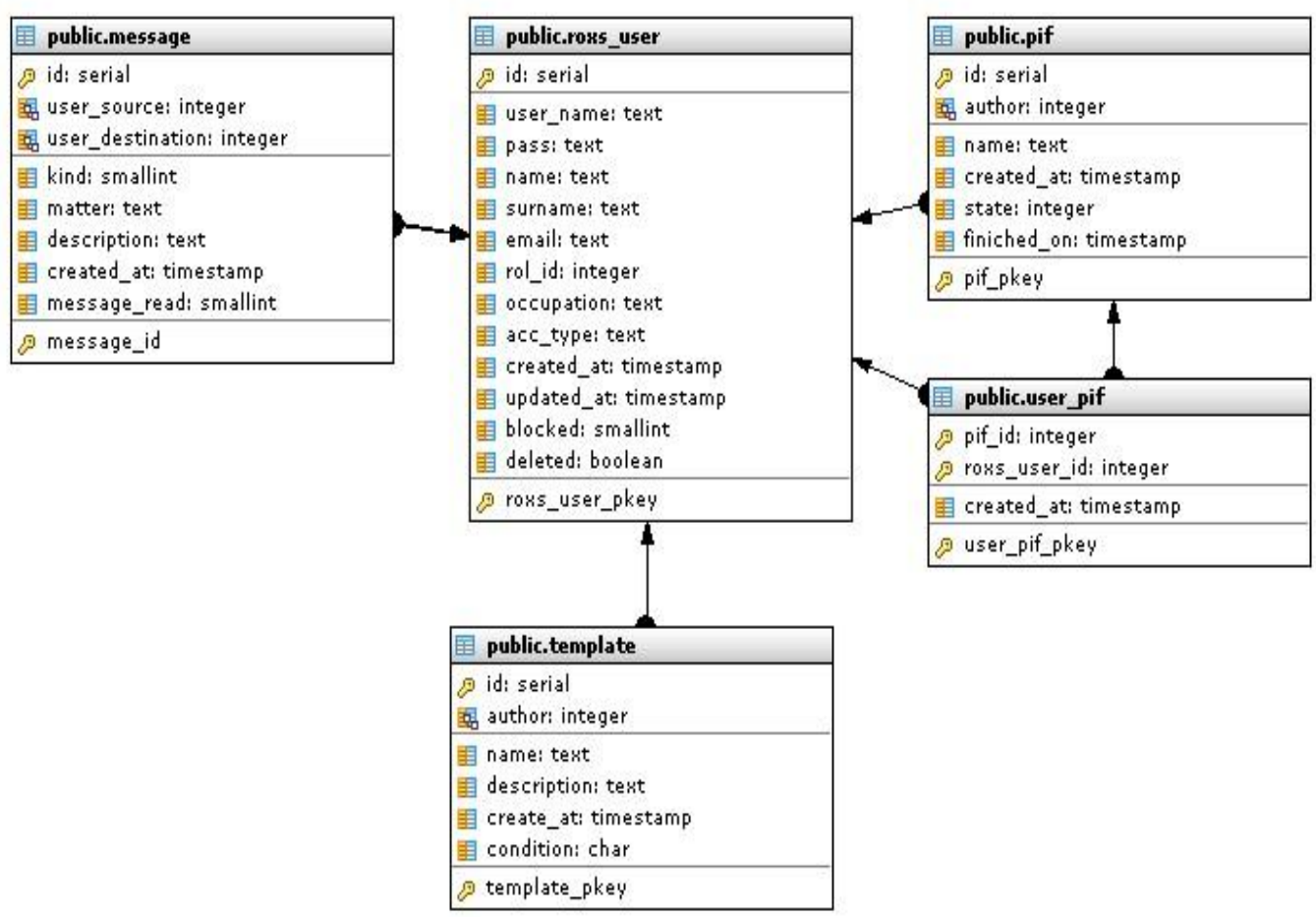

Figura 3.21. Diseño de la Base de Datos.

3.4.1 Descripción de las tablas de la Base de Datos.

Nombre: roxs_user.		
Descripción: tabla referente a los usuarios de la aplicación, la cual contiene los datos generales y específicos de los mismos, nombre de usuario, contraseña, nombre y apellidos, rol que ocupará en la aplicación (autor, revisor, administrador), el tipo de acceso a la herramienta, a través de LDAP o manual así como identificador del propio usuario.		
Atributo	**Tipo**	**Descripción**
• Id (llave primaria). • User_name. • Pass. • Name. • Surname. • E-mail. • Rol_id. • Occupation. • Acc_type. • Created_at. • Updated_at. • Blocked. • Deleted.	• Serial • Text. • Text. • Text. • Text. • Integer. • Text. • Text. • Timestamp. • Timestamp. • Smalling. • Boolean.	• Identificador del usuario. • Nombre de usuario. • Contraseña para acceder al sistema. • Nombre. • Apellidos. • Correo electrónico. • Identificador del rol. • Ocupación. • Tipo de acceso al sistema (LDAP-Manual). • Fecha de creación de usuario. • Fecha de actualización de cuenta de usuario. • Bloqueo en caso de contraseña no válida. • Atributo para la eliminación de usuario.

Tabla 3.1. Descripción de la tabla de la BD-roxs_user.

Nombre: pif.		
Descripción: tabla referente a los objetos de aprendizaje (OA) conformados en la aplicación, la cual contiene los datos generales y específicos de los mismos, nombre del objeto de aprendizaje, autor, fecha de creado, fecha de culminación, estado del objeto de aprendizaje, así como el identificador del mismo.		
Atributo	**Tipo**	**Descripción**
• Id (llave primaria). • Author. • State. • Name. • Created_at. • Finish_On.	• Serial. • Integer. • Integer. • Text. • Timestamp. • Timestamp.	• Identificador del OA. • Autor que conformó el OA. • Estado del OA (en edición). • Nombre del OA. • Fecha de creación del OA. • Fecha de la finalización de la creación del OA.

Tabla 3.2. Descripción de la tabla de la BD-pif.

Nombre: user_pif.		
Descripción: resultado de la relación de la tabla roxs_user con la pif.		
Atributo	**Tipo**	**Descripción**
• Pif_id (llave primaria). • Roxs_user_id. • Created_at.	• Integer. • Integer. • Timestamp.	• Identificador del OA. • Autor que conformó el OA. • Fecha de creación del OA.

Tabla 3.3. Descripción de la tabla de la BD-user_pif.

Nombre: template.		
Descripción: tabla referente a las plantillas creadas en la aplicación, la cual contiene nombre, autor y descripción de la misma, condición, fecha de creada y el identificador de la plantilla.		
Atributo	**Tipo**	**Descripción**
• Id (llave primaria). • Author. • Description. • Created_at. • Condition.	• Serial. • Integer. • Text. • Timestamp. • Char.	• Identificador de la plantilla. • Autor que conformó la plantilla. • Breve descripción de la plantilla. • Fecha de creación de la plantilla. • Estado de plantilla (privada, propuesta, pública).

Tabla 3.4. Descripción de la tabla de la BD-template.

Nombre: message.		
Descripción: tabla referente a los mensajes gestionados en el proceso de revisión de la plantilla, la cual contiene asunto, remitente y destinatario del mismo además de la fecha de enviado y leído.		
Atributo	**Tipo**	**Descripción**
• Id (llave primaria). • User_source. • User_destination. • Matter. • Message_read. • Created_at.	• Serial. • Integer. • Integer. • Text. • Smalling. • Timestamp.	• Identificador del mensaje. • Remitente del mensaje. • Destinatario del mensaje. • Asunto del mensaje. • Estado del mensaje (leído o no leído). • Fecha de creación del mensaje.

Tabla 3.5. Descripción de la tabla de la BD-message.

3.5 Conclusiones

En este capítulo se ha desarrollado uno de los flujos fundamentales en el ciclo de vida del software: el flujo de trabajo Análisis y Diseño. A partir de los casos de usos del sistema se modeló el análisis y el diseño con los artefactos principales que propone la metodología RUP, como son los diagrama de clases del análisis y del diseño, diagramas de interacción, modelo de datos y descripción de las clases.

Capítulo IV Construcción de la solución propuesta

4.1 Introducción

El flujo de trabajo de Implementación tiene como objetivo definir la organización del código teniendo en cuenta los subsistemas de implementación organizadas por capas, la implementación de los elementos de diseño en términos de ficheros fuentes, binarios, ejecutables y para poder integrar los diferentes componentes de desarrolladores o equipos y generar un ejecutable entregable o producto final.

Al finalizar, en la implementación deben quedar plasmados todos los requisitos recogidos en la fase de Requerimientos. Vale destacar que este flujo de trabajo está fuertemente regido por el flujo de Análisis y Diseño. En este capítulo se expondrá el artefacto diagrama de despliegue, que incluye la modelación del hardware utilizado en la implementación del sistema y las relaciones entre sus componentes, además del diagrama de componentes, el cual muestra las dependencias entre dichos componentes.

4.2 Diagrama de despliegue

El diagrama de despliegue muestra la configuración de los nodos de procesamiento en tiempo de ejecución, los links de comunicación entre ellos, y las instancias de los componentes y objetos que residen en ellos. El propósito del diagrama de despliegue es capturar la configuración de los elementos de procesamiento, y las conexiones entre estos elementos en el sistema. El diagrama consiste en uno o más nodos (elementos de procesamiento con al menos un procesador, memoria, y posiblemente otros dispositivos), dispositivos (nodos estereotipados con una capacidad de procesamiento en el nivel modelado de abstracción), y conectores, entre nodos, y entre nodos y dispositivos.

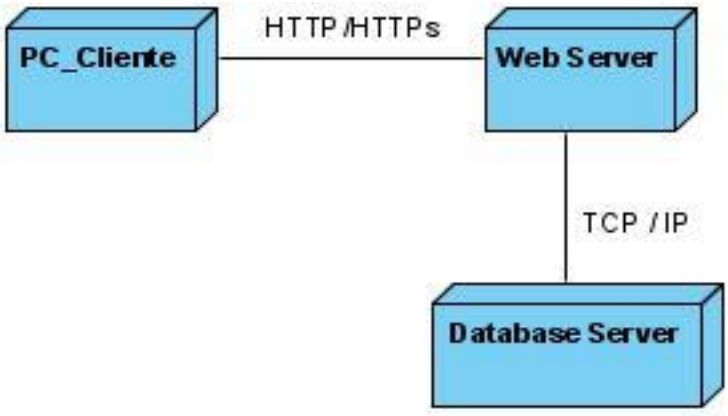

Figura 4.1. Diagrama de Despliegue.

4.3 Diagrama de componentes

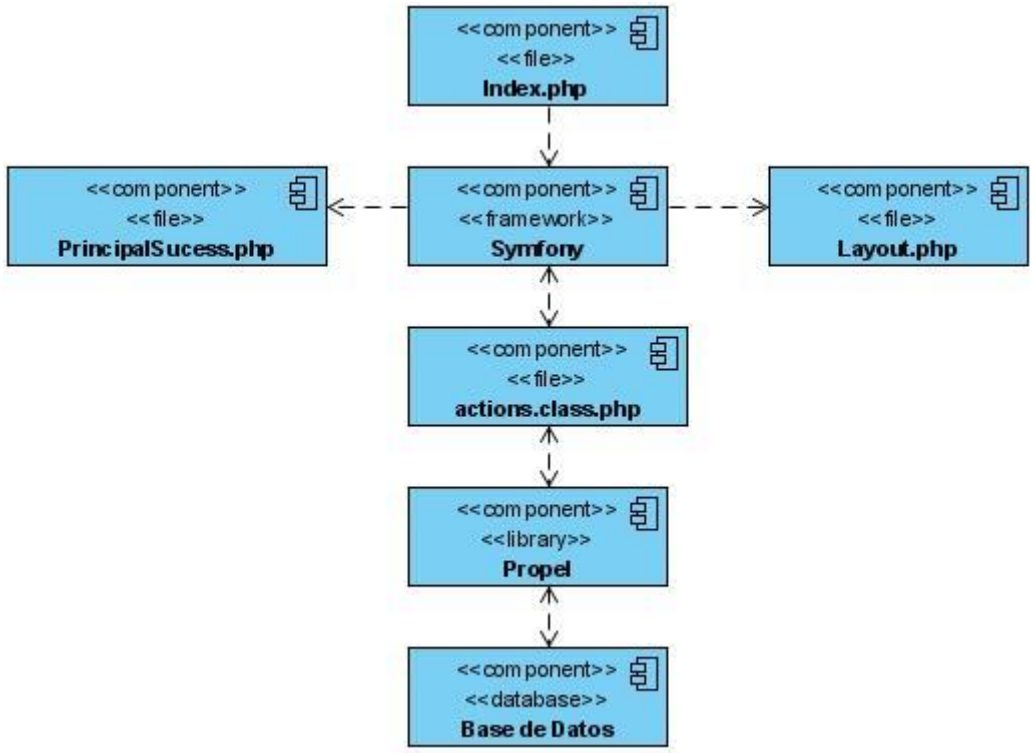

Figura 4.2. Diagrama de Componentes.

4.4 Prueba

Un instrumento adecuado para determinar el status de la calidad de un producto software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos.

4.4.1 Casos de Prueba

Un Caso de Prueba es el conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para un objetivo particular como, por ejemplo; ejercitar el camino concreto de un programa o verificar el cumplimiento de un requisito determinado.

4.4.1.1 Prueba de Caja Negra

Una Prueba de Caja Negra verifica el comportamiento observable externamente del sistema, o sea, se refiere a las pruebas que se llevan a cabo sobre la interfaz del software. Pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce

un resultado correcto, así como que la integridad de la información externa se mantiene. Especifica como probar un caso de uso o un escenario específico de un caso de uso.

4.4.2 Modelo de Caso de Prueba

Caso de Uso	**Crear plantilla.**	
Caso de Prueba	**Crear plantilla.**	
Entrada	**Resultados**	**Condiciones**
1. Se selecciona la opción *"Nueva Plantilla"*.	1.1 Se muestra una interfaz con el nombre y la descripción de la plantilla.	
2. Se introduce el nombre y descripción de la plantilla.	2.1. Se muestra la plantilla en detalle con el nombre y la descripción agregada para proceder con su estructuración.	
3. Se confecciona la estructura de la plantilla agregando, eliminando o modificando organizaciones o ítems.	3.1 La herramienta realiza los cambios a la estructura de la plantilla.	3.2 La plantilla contiene una organización con un ítem por defecto. 3.3 Culminado el proceso de estructuración se muestran las opciones *"Guardar plantilla"* o *"Proponer plantilla a publicación".*

Tabla 4.1.Modelo de Caso de Prueba del CU - Crear plantilla.

Caso de Uso	**Modificar plantilla.**	
Caso de Prueba	**Crear plantilla.**	
Entrada	**Resultados**	**Condiciones**
2. Se selecciona la opción *"Modificar Plantilla"*.	1.1 Se muestra una interfaz con el nombre y la descripción de la plantilla nuevamente.	

2. Se introduce el nuevo nombre y descripción de la plantilla.	2.1. Se muestra la plantilla en detalle con el nombre y la descripción agregada para proceder con su modificación.	
4. Se confecciona la estructura de la plantilla agregando, eliminando o modificando organizaciones o ítems.	3.1 La herramienta realiza los cambios a la estructura de la plantilla.	3.2 La plantilla contiene una organización con un ítem por defecto. 3.3 Culminado el proceso de estructuración se muestran las opciones *"Guardar plantilla"* o *"Proponer plantilla a publicación".*

Tabla 4.2.Modelo de Caso de Prueba del CU - Modificar plantilla.

Caso de Uso	**Revisar plantilla.**	
Caso de Prueba	**Revisar plantilla.**	
Entrada	**Resultados**	**Condiciones**
	1. Se listan plantillas propuestas.	
2. Se muestra la plantilla en detalle con el nombre y la descripción.		2.1 La plantilla tiene que aprobarse o denegarse
3. Se confecciona la estructura de la plantilla agregando, eliminando o modificando organizaciones o ítems.	3.1 La herramienta realiza los cambios a la estructura de la plantilla.	3.2 La plantilla contendrá al menos una organización.

Tabla 4.3.Modelo de Caso de Prueba del CU - Revisar plantilla.

Caso de Uso	**Buscar OA en repositorio.**	
Caso de Prueba	**Crear OA.**	
Entrada	**Resultados**	**Condiciones**

1. Se introduce los campos de búsqueda.	1.1 Se lista el resultado de la búsqueda, aunque pudiese en futuras versiones implementar una búsqueda más completa.	

Tabla 4.4.Modelo de Caso de Prueba del CU - Buscar OA en repositorio.

Caso de Uso	**Guardar OA en repositorio.**	
Caso de Prueba	**Crear OA.**	
Entrada	**Resultados**	**Condiciones**
	1. Se almacena el OA en el repositorio.	

Tabla 4.5.Modelo de Caso de Prueba del CU - Guardar OA en repositorio.

4.5 Estimación basada en puntos de casos de uso

A partir de determinados parámetros es posible estimar variables como el costo, el esfuerzo y el tiempo necesarios para obtener un software. Existen varias técnicas que posibilitan estimar estas variables. Una de las más utilizadas en la UCI es la estimación basada en puntos de casos de uso que es un método que permite evaluar el esfuerzo de un proyecto de desarrollo de software a partir de los casos de uso.

La estimación mediante el análisis de Puntos de Casos de Uso es un método propuesto originalmente por Gustav Karner de Objectory AB, y posteriormente refinado por muchos otros autores. Se trata de un método de estimación del tiempo de desarrollo de un proyecto mediante la asignación de *"pesos"* a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores.

Los casos de uso por sí mismos no permiten efectuar una estimación del tamaño que tendrá el sistema, ni del esfuerzo y el tiempo necesario para implementarlo. Estos permiten documentar los requerimientos del software de una manera compacta y precisa, luego con los puntos de función se puede estimar el tamaño del software a partir de los requerimientos obtenidos de los casos de uso.

La técnica puntos de función de casos de uso consiste en evaluar la complejidad de un sistema de software por medio de un método en el que se le asigna una cantidad de puntos de peso, que califican diferentes elementos que componen el sistema de software así como algunos factores del entorno,

para obtener una aproximación del tiempo requerido y la cantidad de esfuerzo necesario para la implementación del mismo. Este proceso se lleva a cabo mediante una serie de pasos que como se mencionó anteriormente evalúan cada factor, empezando por ponderar los casos de uso sin ajustar. A continuación se listan los pasos fundamentales en el proceso de estimación:

1. Calcular los puntos de caso de uso sin ajustar.
2. Calcular los puntos de casos de uso ajustados.
3. Calcular esfuerzo del flujo de trabajo de implementación.
4. Calcular esfuerzo de todo el proyecto.

4.5.1 Paso 1: Calcular los puntos de caso de uso sin ajustar.

El cálculo de los puntos de caso de uso sin ajustar puede servir para tener una idea un poco más precisa de la dificultad de los casos e interfaces, para lo cual se hace necesario calcular también el factor de peso de los actores sin ajustar y el factor de peso de los casos de uso sin ajustar. Este parámetro se calcula utilizando la siguiente ecuación:

UUCP = UAW + UUCW

UUCP: Puntos de Casos de Uso sin ajustar

UAW: Factor de Peso de los Actores sin ajustar

UUCW: Factor de Peso de los Casos de Uso sin ajustar

Factor de Peso de los Actores sin ajustar (UAW): Este valor se calcula mediante un análisis de la cantidad de actores presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los actores se establece teniendo en cuenta en primer lugar si se trata de una persona o de otro sistema, y en segundo lugar, la forma en la que el actor interactúa con el sistema.

Factor de peso de los actores sin ajustar.

Tipo de Actor	Descripción:	Cantidad de Actores:	Factor de Peso
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación.	1	1
Medio	Otro sistema que interactúa con el sistema a	1	2

	desarrollar mediante un protocolo o una interfaz basada en texto.		
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3	3

Tabla 4.6. Factor de peso de los actores sin ajustar.

UAW= ∑ (actores*Peso)

UAW= ∑ (1*1) + (1*2) + (2*3)

UAW= 9

Factor de Peso de los Casos de Uso sin ajustar (UUCW): Este valor se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia y está representada por uno o más pasos del flujo de eventos principal del Caso de Uso, pudiendo existir más de una transacción dentro del mismo Caso de Uso.

Factor de peso de los casos de uso sin ajustar.

Tipo de Actor	Descripción	Factor de Peso	Cantidad de Casos de Uso
Simple	El Caso de Uso contiene de 1 a 3 transacciones.	5	9
Medio	El Caso de Uso contiene de 4 a 7 transacciones.	10	6
Complejo	El Caso de Uso contiene más de 8 transacciones.	15	1

Tabla 4.7. Factor de peso de los CU sin ajustar.

UUCW = ∑ CU*Peso

UUCW = ∑ (9*5) + (6*10) + (1*15)

UUCW = 120

Después de obtenidos los valores de UAW y el UUCW podemos calcular el valor de los puntos de caso de uso sin ajustar.

UUCP = UAW + UUCW

UUCP = 9 + 120

UUCP = 129

4.5.2 Paso 2: Calcular los puntos de caso de uso ajustados.

Una vez que se tienen los Puntos de Casos de Uso sin ajustar, se debe ajustar éste valor mediante la siguiente ecuación:

UCP = UUCP x TCF x EF

UCP: Puntos de Casos de Uso ajustados

UUCP: Puntos de Casos de Uso sin ajustar

TCF: Factor de complejidad técnica

EF: Factor de ambiente

Por tanto para el cálculo de los puntos de caso de uso ajustados se hace necesario calcular también el factor de complejidad técnica y el factor de ambiente.

Factor de complejidad técnica (TCF): Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5, en dependencia de su influencia.

- ✓ 0-No presenta influencia
- ✓ 1-Influencia incidental
- ✓ 2-Influencia moderada
- ✓ 3-Influencia media
- ✓ 4-Influencia significativa
- ✓ 5-Influencia fuerte.

Factor	Descripción	Peso	Valor asignado	Observación
T1	Sistema distribuido	2	0	El sistema está centralizado.
T2	Objetivos de performance o tiempo de respuesta	1	4	El tiempo de respuesta debe ser mínimo.
T3	Eficiencia del usuario final	1	4	La eficiencia del usuario final de debe ser alta.
T4	Procesamiento interno complejo	1	5	Se efectuará un fuerte procesamiento interno.
T5	El código debe ser reutilizable	1	3	Se desea que el código sea reutilizable.
T6	Facilidad de instalación	0	0	Se requiere que el sistema sea fácil de instalar.
T7	Facilidad de uso	0.5	3	Se requiere que el sistema sea fácil de usar.
T8	Portabilidad	2	3	Se requiere que el sistema sea portable.
T9	Facilidad de cambio	1	3	Se requiere que el mantenimiento del sistema tengo un costo mínimo.
T10	Concurrencia	1	2	Concurrencia normal.
T11	Incluye objetivos especiales de seguridad	1	4	Se necesita una alta seguridad en el sistema.
T12	Provee acceso directo a terceras partes	1	0	No provee acceso directo a terceras partes.

T13	Se requieren facilidades especiales de entrenamiento a usuarios	1	1	Sistema fácil de usar.

El factor de complejidad técnica se puede calcular finalmente utilizando la siguiente ecuación:

TCF = 0.6 + 0.01 * Σ (peso * valor asignado)

TCF = 0.6 + 0.01 * 35

TCF = 0.6 + 0.35

TCF = 0.95

Factor de ambiente (EF): Las habilidades y el entrenamiento del grupo involucrado en el desarrollo tienen un gran impacto en las estimaciones de tiempo. Estos factores son los que se contemplan en el cálculo del Factor de ambiente. El cálculo del mismo es similar al cálculo del Factor de complejidad técnica, es decir, se trata de un conjunto de factores que se cuantifican con valores de 0 a 5.

Factor de ambiente.

Factor	Descripción	Peso	Valor asignado	Observaciones
E1	Familiaridad con el modelo de proyecto utilizado	1.5	5	Todo el equipo está familiarizado con el modelo utilizado.
E2	Experiencia en la aplicación	0.5	3	El equipo no tiene mucha experiencia con la aplicación.
E3	Experiencia en orientación a objetos	1	4	Los miembros del grupo dominan la programación orientada a objetos.

Tabla 4.8. Factor de complejidad técnica.

E4	Capacidad del analista líder	0.5	5	Los analistas están bien preparados.
E5	Motivación	1	5	El grupo tiene una alta motivación.
E6	Estabilidad de los requerimientos	2	2	Los requerimientos pueden

				variar
E7	Personal part-time	-1	3	Los miembros del equipo trabajan en el horario que les corresponde
E8	Dificultad del lenguaje de programación	-1	3	Se utilizará el lenguaje de programación PHP5.

Tabla 4.9. Factor ambiente.

El Factor de ambiente se puede calcular finalmente utilizando la siguiente ecuación:

EF = 1.4 - 0.03 * Σ (peso x valor asignado)

EF = 1.4 - 0.03 * 18.5

EF = 1.4- 0.555

EF = 0.85

Luego de haber obtenido los valores de UUCP, TCF y EF se calcula el valor de los puntos de casos de uso ajustados (UCP):

UCP = UUCP x TCF x EF

UCP = 129 * 0.95 * 0.85

UCP = 104.17

4.5.3 Paso 3: Calcular esfuerzo del flujo de trabajo de implementación.

El cálculo del esfuerzo se realiza utilizando la ecuación siguiente:

E = UCP * CF

E: esfuerzo estimado en horas-hombre

UCP: Puntos de Casos de Uso ajustados

CF: factor de conversión

Para el cálculo del esfuerzo horas-hombre (E) se hace necesario calcular el factor de conversión (CF), el mismo se obtiene contando cuántos valores de los que afectan el factor ambiente (E1-E6) están por debajo de la media (3), y los que están por arriba de la media para los restantes (E7, E8). Si el total es 2 o menos se utiliza el factor de conversión 20 Horas-Hombre / Punto de Casos de uso. Si el total es 3 o 4 se utiliza el factor de conversión 28 Horas-Hombre / Punto de Casos de uso. Si el total es mayor o igual que 5 se recomienda efectuar cambios en el proyecto ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

En este caso el factor de conversión es 20 (CF = 20), pues solo se encuentra un valor por debajo de la media en los que afectan el factor ambiente (E1-E6). Luego de obtenido el valor del factor de conversión se puede realizar el cálculo del esfuerzo.

E = UCP * CF

E = 104.17 * 20

E = 2083.4

4.5.4 Paso 4: Calcular esfuerzo de todo el proyecto.

El método anterior proporciona una estimación del esfuerzo en horas-hombre contemplando sólo el desarrollo de la funcionalidad especificada en los casos de uso. Para una estimación más completa de la duración total del proyecto, hay que agregar a la estimación del esfuerzo obtenida por los Puntos de Casos de Uso, las estimaciones de esfuerzo de las demás actividades relacionadas con el desarrollo de software.

Para ello se puede tener en cuenta el siguiente criterio, que estadísticamente se considera aceptable. El criterio plantea la distribución del esfuerzo entre las diferentes actividades de un proyecto. Como el valor de esfuerzo calculado anteriormente representa el esfuerzo del flujo de trabajo de implementación, por comparación salen el resto de los esfuerzo y la suma de ellos es el esfuerzo total (ET). En la siguiente tabla se muestra la distribución del esfuerzo horas-hombres en las diferentes etapas del proyecto.

Esfuerzo de todo el proyecto.

Actividad	Porcentaje	Horas-Hombres
Análisis	10.00 %	520.85

Diseño	20.00 %	1041.7
Programación	40.00 %	2083.4
Pruebas	15.00 %	781.28
Sobrecarga (otras actividades)	15.00 %	781.28
Total	100 %	5208.5

Tabla 4.10. Esfuerzo de todo el proyecto.

El esfuerzo total es de 5208.5 horas-hombre, si tenemos en cuenta que un mes laborable tiene 240 horas, el esfuerzo total sería entonces de 21.70 mes-hombre. Analizando el esfuerzo total se puede estimar que 2 hombres pueden realizar el sistema en aproximadamente 1 año y 2 meses.

4.6 Costos

A continuación se muestra una tabla resumen con el cálculo del costo total del proyecto, donde se asume como salario promedio mensual $100.00, pues es lo correspondiente con el estipendio estudiantil.

Resumen del costo del proyecto.

Costo del proyecto	
Esfuerzo Total (Horas-Hombres)	5208.5
Esfuerzo Total (Meses-Hombres)	21.70
Salario Promedio	$100
Cantidad de hombres	2
Costo (Mes-Hombre)	$200
Costo Total	$5652

Tabla 4.11. Resumen del costo del proyecto.

4.7 Beneficios tangibles e intangibles

4.7.1 Beneficios tangibles

Mencionar en la presente investigación beneficios económicos no es válido, teniendo en cuenta que la aplicación implementada no es un producto desarrollado inicialmente para la comercialización, sino como un aporte social a la *"Vice-rectoría de Formación"* de la UCI. Sin embargo, a pesar de que este proyecto no constituye un producto comercial y no aporta ganancias económicas, se puede exponer que el principal beneficio tangible social que representa la concreción de esta aplicación

Web, y es, que constituye una herramienta que mejora elevadamente la informatización de los procesos de gestión de objetos de aprendizaje.

4.7.2 Beneficios Intangibles

Una vez implementada y puesta en práctica la aplicación, la *"Vice-rectoría de Formación"* contará con un software calificado para la gestión rápida y fiable de los objetos de aprendizaje. La aplicación permitirá la incorporación de procesos que anteriormente no se tenían en cuenta, tales como la búsqueda y almacenaje de OA creados con anterioridad. También constituye un beneficio conceder a los autores acceso a las plantillas privadas y OAs que dicho autor ha creado. El software agilizará el proceso de creación del OA mediante la propuesta de plantillas.

4.8 Análisis de costos y beneficios

Se inicia este análisis teniendo en cuenta lo planteado en los epígrafes anteriores. Es indispensable mencionar que el desarrollo del sistema no requerirá gastos significativos, primeramente porque las herramientas que se utilizarán para confeccionar el mismo son libres y de código abierto, y son proporcionadas fácilmente por la universidad. No será necesaria la contratación de ninguna empresa que preste servicios informáticos, puesto que todos ellos serán gestionados dentro del proyecto productivo. Más, reiterando que este proyecto es resultado de una necesidad y que brinda una herramienta muy útil a la *"Vice-rectoría de Formación"*, se puede concluir que resulta factible el modelado y la implementación de dicha herramienta.

4.9 Conclusiones

A lo largo de este capítulo se revelaron los resultados de las disciplinas de Diseño e Implementación en el proceso de desarrollo del módulo "Presupuesto". Se desarrollaron los Diagramas de Clases y de Secuencia de los Casos de Uso fundamentales en el sistema, así como los tipos de clases que existen en el diseño de la solución y el Modelo de Datos. Se expuso también un grupo de componentes desarrollados para ser utilizados en el sistema, además de una serie de reglas que se asumieron para la codificación del software. Como complemento se brinda también el Diagrama de Componentes y el de Despliegue.

Conclusiones Generales

Luego de la investigación realizada se puede afirmar que se cumplieron los objetivos propuestos, arribando a las siguientes conclusiones:

- El análisis y diseño de las funcionalidades expuestas propició una vía de solución que guió el proceso de desarrollo de la herramienta “ROXS”.

- Con la realización de este trabajo en la Universidad de las Ciencias Informáticas se logró desarrollar una herramienta de autor web que posibilita el diseño flexible de objetos de aprendizaje, a partir de la interpretación del usuario con respecto a este concepto.

- Se presenta una aplicación capaz de diseñar plantillas de recursos educativos que podrán ser reutilizables por todos los autores del sistema en la confección de sus contenidos.

- El resultado de este trabajo proporciona una visión más amplia y detallada de los requerimientos funcionales de la herramienta de autor.

Por todo lo anterior se concluye que los objetivos propuestos para el presente trabajo han sido cumplidos satisfactoriamente. Se incluyen una serie de recomendaciones que deben tenerse en cuenta para el trabajo futuro.

Recomendaciones

A lo largo del desarrollo del presente trabajo, han ido surgiendo ideas que podrían implementarse en un futuro, de forma que se logre una aplicación más útil y efectiva, para lo cual se recomienda:

1- Añadir otras funcionalidades contempladas en el desarrollo de la investigación, con el objetivo de incrementar las facilidades que brinda la herramienta de autor.
2- Implementar las acciones descritas en el libro de Secuencia y Navegación del estándar SCORM 2004, para establecer una secuencia de navegación por los diferentes contenidos del recurso educativo.
3- Incorporar un módulo para el diseño instruccional.

Bibliografía.

Referencias bibliográficas

1. AULAGLOBAL. Estándares - SCORM, 2005. [Disponible en: http://www.aulaglobal.net.ve/observatorio/index.php?lng=es
2. TAMAYO, D. Herramientas para la reutilización de contenidos educativos, Universidad de las Ciencias Informáticas, 2007. p.
3. LÓPEZ, C. Los Repositorios de Objetos de Aprendizaje como soporte a un entorno e-learning. España, Salamanca, 2005. p.
4. AILLON, C. E-learning América Latina 16.01 2009. [Disponible en: http://www.elearningamericalatina.com/edicion/junio1_2004/tr_1.php
5. CAÑIZARES, R. Framework para la gestión de contenidos educativos, Universidad de las Ciencias Informáticas, 2008. p.
6. CARRODEGUAS, Y. Análisis y Diseño de una herramienta de autor Web Interoperable, 2008. p.
7. ATICA_SOCRATES. E-Learning, 2007.
8. González, D. Los repositorios de objetos de aprendizaje como soporte para los entornos e-learning. [Disponible en: http://www.biblioweb.dgsca.unam.mx/libros/repositorios/objetos_aprendizaje.htm
9. COAD & JILL. . Los repositorios de objetos de aprendizaje como soporte para los entornos e-learning. [Disponible en: http://www.biblioweb.dgsca.unam.mx/libros/repositorios/objetos_aprendizaje.htm
10. WILEY, A. E-learning América Latina 20.01 2009. [Disponible en: http://www.elearningamericalatina.com/edicion/junio1_2005/tr_2.php
11. MANJÓN, B. F. Especificaciones y estándares en e-learning Revista de Tecnologías de la Información y Comunicación Educativas, 2006, Nº 6: 2.
12. IEEE. IEEE Standards Association, 2002. [Disponible: http://www.ieee.org/web/standards/home/index.html
13. Content Aggregation Model. Content Aggregation Model, 2006.
14. PATÓN, M. Desarrollo de software, 2006.
15. SANCHEZ, E. Microsoft Solutions Framework, 2004.
16. CIBERNETIA, Servidores Web, 2009. [Disponible en: http://www.cibernetia.com/tesis_es/PEDAGOGIA/TEORIA_Y_METODOS_EDUCATIVOS/1

Bibliografía consultada

1. BARCO, A. SOA y los Servicios Web., 2006.
2. BONET, J. C. A. Metadatos y documentos XML/RDF para recuperación, 2007.
3. ADLNet. The SCORM Content Aggregation Model, Advanced Distributed Learning Initiative 2001. http://www.adlnet.org/. (06/03/09)
4. ADLNet. The SCORM Content Aggregation Model, Advanced Distributed Learning Initiative 2006. http://www.adlnet.org/. (06/03/09)
5. http://exelearning.org/files/Spanish_exe_DLE_A4.pdf. (30/03/09)
6. AulaGlobal. Objetos de Aprendizaje - SCO y Modelo de Contenidos. http://www.aulaglobal.net.ve/observatorio/articles.php?lng=es&pg=89. (27/02/06)
7. Cadíz, Universidad. Manual para la creación de Paquetes SCORM y su importación a Moodle. http://virtual.uca.es/portalFormacion/docs/portalFormacion/carpetaLocal2/manual_scorm.pdf?action=download. (21/03/09)
8. Dieguez, Jorge. Nuevas versiones de las herramientas Reload. http://www.elearningworkshops.com/modules.php?name=News&file=article&sid=368. (05/03/09)
9. Draft Standard for Learning Object Metadata. Learning Technology Standards Committee of IEEE, USA, 2002.
10. García, Juan Egea. Reload. http://www.um.es/atica/gat/tdm/reload/sesion1.pdf. (08/04/09)
11. Guerrero, L. A. Proceso Unificado de Desarrollo de Software. http://www.dcc.uchile.cl/~luguerre/cc51h/clase23.html. (04/04/09).
12. Lim, John. ADOdb. Database Abstraction Library for PHP (and Python). http://adodb.sourceforge.net/. (05/04/09)
13. Mariño, Olga. Objetos de aprendizaje, escenarios de aprendizaje y gestión de conocimiento. http://www.colombiaaprende.edu.co/html/docentes/1596/articles-81108_archivo.pdf. (6/04/09)
14. SCORM. http://www.ateneonline.net/datos/65_03_Romero_Daniel.pdf. (28/03/06)

Anexos.
Definición de elementos y categorías del perfil de LOM.

A continuación, se incluye una tabla descriptiva en la que se incluyen los diferentes elementos de datos agregados y simples que pertenecen a cada categoría junto con su definición correspondiente, señalando en color gris los elementos extendidos. Para cada elemento se indica:

El *Carácter* del elemento, pudiendo ser: obligatorio (ob.), recomendado (re.), optativo (op.) y/o condicionado (co.).

El *Tipo de* Datos, que en el caso de ser *Vocabulario,* se indica si se propone un vocabulario alternativo controlado LOM-ES.

El *Espacio de Valores* para los *Vocabularios* controlados LOM y LOM-ES.

El *Tamaño*, indicando el menor máximo permitido (m.m.p.) de elementos. Es decir, cualquier aplicación de catalogación que utilice este perfil debe incluir como mínimo la posibilidad de incluir ese número de instancias.

El *Orden,* donde se especifica si el orden de la lista de instancias para un elemento determinado es relevante o no (*No especificado*, *Ordenado* o *No ordenado*). Por ejemplo, en una lista de autores de una publicación, el primer autor normalmente se considera el autor más importante.

Categoría 1. General

Nº	Nombre	Carácter	Tamaño	Orden	Espacio de valores	Tipo de datos
1	General	(ob.)	1	No especificado		
1.1	Identificador	(ob.)	m.m.p. 10	No especificado		
1.1.1	Catálogo	(ob.)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 1000
1.1.2	Entrada	(ob.)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 1000
1.2	Título	(ob.)	1	No especificado		LangString 100
1.3	Idioma	(ob.)	m.m.p. 10	No ordenado	Código de 2 letras ISO 639-1988	CharacterString 100
1.4	Descripción	(ob.)	m.m.p. 10	No ordenado		LangString 2000
1.5	Palabra Clave	(re.)	m.m.p. 10	No ordenado		LangString 1000
1.6	Ámbito	(op.)	m.m.p. 10	No ordenado		LangString 1000
1.7	Estructura	(op.)	1	No especificado	atómica, colección, en red, jerárquica, lineal	Vocabulario
1.8	Nivel de Agregación	(ob.)	1	No especificado	1, 2, 3, 4	Vocabulario

Esta categoría agrupa la información general que describe este objeto digital educativo en su conjunto.

Identificador:

Una etiqueta, única que identifica este objeto educativo.

Catálogo:

El nombre o denominación del esquema de identificación o catalogación en base al cual se establecerá el identificador.

Ejemplo etiqueta en xml:

<identifier>

<catalog>DOI</catalog>

<catalog>ATENEX</catalog>

</identifier>

Categoría 2. Ciclo de vida

Nº	Nombre	Carácter	Tamaño	Orden	Espacio de valores	Tipo de datos
2	Ciclo de Vida	(re.)	1	No especificado		
2.1	Versión	(op.)	1	No especificado		LangString 50
2.2	Estado	(op.)	1	No especificado	borrador, final, revisado, no disponible	Vocabulario
2.3	Contribución	(re.)	m.m.p. 30	Ordenado		
2.3.1	Tipo	(ob.) si 2.3 (+)	1	Ordenado	autor, editor de publicación, iniciador, terminador, revisor, editor de contenido, diseñador gráfico, desarrollador técnico, proveedor de contenidos, revisor técnico, revisor educativo, guionista, diseñador educativo, experto en la materia	Vocabulario LOM-ES
2.3.2	Entidad	(ob.) si 2.3 (+)	m.m.p. 40	Ordenado	vCard tal y como se define en el IMC vCard 3.0 (RFC 2425, RFC 2426)	CharacterString 1000
2.3.3	Fecha	(ob.) si 2.3 (+)	1	No especificado		Fecha

Esta categoría describe la historia y estado actual de este ODE, así como aquellas entidades que han intervenido en su creación y evolución.

Versión:

La edición de este ODE.

§ Ejemplo etiqueta en xml:

<version>

<string language="es">v.1.0</string>

</version>

Estado:

El estado de completitud o condición de este objeto educativo.

§ Ejemplo etiqueta en xml:

<status>

<source uniqueElementName="source">LOM-ESv1.0</source>

<value uniqueElementName="value">final</value>

</status>

Categoría 3. Meta-Metadatos

Nº	Nombre	Carácter	Tamaño	Orden	Espacio de valores	Tipo de datos
3	Meta-Metadatos	(ob.)	1	No especificado		
3.1	Identificador	(op.)	m.m.p. 10	No especificado		
3.1.1	Catálogo	(ob.) si 3.1 (+)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 1000
3.1.2	Entrada	(ob.) si 3.1 (+)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 1000
3.2	Contribución	(re.)	m.m.p. 10	Ordenado		
3.2.1	Tipo	(ob.) si 3.2 (+)	1	No especificado	creador, revisor	Vocabulario
3.2.2	Entidad	(ob.) si 3.2 (+)	m.m.p. 10	Ordenado	vCard tal y como se define en el IMC vCard 3.0 (RFC 2425, RFC 2426)	CharacterString 1000
3.2.3	Fecha	(ob.) si 3.2 (+)	1	No especificado		Fecha
3.3	Esquema de Metadatos	(ob.)	m.m.p. 10	No ordenado	Repertorio del ISO/IEC 10646-1:2000. Actualmente "LOM-ES v.1.0".	CharacterString 30
3.4	Idioma	(ob.)	1	No especificado	Código de 2 letras ISO 639-1988. "ninguno" no es un valor aceptable porque la instancia de metadatos debe estar expresada en uno o varios idiomas humanos.	CharacterString 100

Esta categoría describe el propio registro de metadatos (en lugar del objeto educativo descrito por el registro de metadatos). Esta categoría describe como puede ser identificada esta instancia de metadatos, quién la creó, cómo, cuándo y con qué referencias. Esta no es, por tanto, información que describa el objeto educativo.

Identificador: Una etiqueta única globalmente que identifica este registro de metadatos.

Catálogo:

El nombre o denominación del esquema de identificación o catalogación para esta entrada. Un esquema de espacio de nombres.

§ Ejemplo etiqueta en xml:

<identifier>

<catalog>DOI-META</catalog>

<catalog>ATENEX-META</catalog>

</identifier>

Categoría 4. Técnica

Nº	Nombre	Carácter	Tamaño	Orden	Espacio de valores	Tipo de datos
4	**Técnica**	**(re.)**	**1**	**No especificado**		
4.1	Formato	(re.)	m.m.p. 40	No ordenado	Tipos MIME basado en el registro IANA (ver RFC2048). "no-digital" no es aceptable en LOM-ES v.1.0.	CharacterString 500
4.2	Tamaño	(re.)	1	No especificado	ISO/IEC 646:1991, pero sólo los dígitos "0"..."9"	CharacterString 30
4.3	Localización	(re.)	m.m.p. 10	Ordenado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 100
4.4	**Requisitos**	**(op.)**	**m.m.p. 40**	**No ordenado**		
4.4.1	**AgregadorOR**	**(ob.) si 4.4 (+)**	**m.m.p. 40**	**No ordenado**		
4.4.1.1	Tipo	(ob.) si 4.4.1 (+)	1	No especificado	sistema operativo, navegador	Vocabulario
4.4.1.2	Nombre	(ob.) si 4.4.1 (+) (co.)	1	No especificado	▪ Si Tipo='sistema operativo', entonces: pc-dos, ms-windows, linux, macos, unix, multi-so, ninguno. ▪ Si tipo='navegador' entonces: cualquiera, mozilla firefox, netscape communicator, ms internet explorer, opera, amaya	Vocabulario LOM-ES
4.4.1.3	Versión Mínima	(op.)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 30
4.4.1.4	Versión Máxima	(op.)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 30
4.5	Pautas de Instalación	(op.)	1	No especificado		LangString 1000
4.6	Otros Requisitos de Plataforma	(op.)	1	No especificado		LangString 1000
4.7	Duración	(op.)	1	No especificado		Duración

Esta categoría describe los requisitos y características técnicas de este objeto educativo.

Formato:

El(los) tipo(s) de datos de todos los componentes de este objeto digital. Este elemento de datos también debe ser utilizado para identificar el software necesario para acceder al objeto educativo.

NOTA 1: el listado actualizado de los tipos MIME basado en el registro IANA puede ser consultado en: http://www.iana.org/assignments/media-types/

Ejemplo etiqueta en xml:

<format>text/html</format>

Categoría 5. Uso educativo

Nº	Nombre	Carácter	Tamaño	Orden	Espacio de valores	Tipo de datos
5	**Uso Educativo**	**(ob.)**	**m.m.p. 100**	**No especificado**		
5.1	Tipo de Interactividad	(op.)	1	No especificado	activo, expositivo, combinado	Vocabulario
5.2	Tipo de Recurso Educativo	(ob.)	m.m.p. 10	Ordenado	▪ Media: fotografía, ilustración, video, animación, música, efecto sonoro, locución, audio compuesto, texto narrativo, hipertexto, grafismo, media integrado ▪ Sistema de representación de información y/o conocimiento: base de datos, tabla, gráfico, mapa conceptual, mapa de navegación , presentación multimedia, tutorial, diccionario digital, enciclopedia digital, publicación digital periódica, web/portal temático o corporativo, wiki, weblog ▪ Aplicación informática: herramienta de creación/edición multimedia, herramienta de creación/edición web, herramienta de ofimática, herramienta de programación, herramienta de análisis/organización de información/conocimiento, herramienta de apoyo a procesos/procedimientos, herramienta de gestión de aprendizaje/trabajo individual/cooperativo/ colaborativo ▪ Servicio: servicio de creación/edición multimedia, servicio de creación/edición web, servicio de ofimática, servicio de programación, servicio de análisis/organización de información/conocimiento, herramienta de apoyo a procesos/procedimientos, servicio de gestión de aprendizaje/trabajo individual/cooperativo/colaborativo ▪ Contenido didáctico: lecturas guiadas, lección magistral, comentario de texto-imagen, actividad de discusión, ejercicio o problema cerrado, caso contextualizado, problema abierto , escenario real o virtual de aprendizaje, juego didáctico, webquest, experimento, proyecto real, simulación, cuestionario, examen, autoevaluación	Vocabulario LOM-ES
5.3	Nivel de Interactividad	(op.)	1	No especificado	muy bajo, bajo, medio, alto, muy alto	Vocabulario

Esta categoría describe las características educativas y pedagógicas fundamentales de este objeto educativo. Concretamente, es la información didáctica esencial para aquellos agentes involucrados en una experiencia educativa de calidad. Algunos de estos agentes son: estudiantes, profesores, tutores y administradores.

Tipo de Interactividad:

El tipo de aprendizaje predominante soportado por este objeto educativo:

Aprendizaje “activo” (por ejemplo, aprendizaje participativo) es el soportado por aquellos contenidos que inducen a la participación directa por parte de los aprendices. Un objeto de aprendizaje activo solicita del aprendiz que interaccione e introduzca información semánticamente significativa, que tome decisiones o realice algún tipo de actividad productiva. Todo ello no necesariamente en el contexto del propio objeto educativo. Entre los objetos activos podemos mencionar los simuladores, cuestionarios y ejercicios.

Aprendizaje “expositivo” (por ejemplo, aprendizaje pasivo) es aquel en el que la tarea fundamental del aprendiz consiste en asimilar aquellos conceptos que le son expuestos (generalmente mediante textos,

imágenes o sonidos). Un objeto para aprendizaje expositivo muestra información al aprendiz sin solicitar de éste ningún tipo de acción por su parte semánticamente significativa. Entre los objetos expositivos se encuentran los ensayos, vídeos, todo tipo de material gráfico y los documentos hipertextuales.

Cuando un objeto educativo mezcla los tipos activo y expositivo, entonces su nivel de interactividad será "combinado"

Categoría 6. Derechos

Nº	Nombre	Carácter	Tamaño	Orden	Espacio de valores	Tipo de datos
6	Derechos	(ob.)	1	No especificado		
6.1	Coste	(op.)	1	No especificado	si, no	Vocabulario
6.2	Derechos de Autor y otras Restricciones	(ob.) (co.)	1	No especificado	▪ Si 5.2= a cualquier valor del grupo "Aplicación informática" entonces: licencia propietaria, licencia libre EUPL, licencia libre GPL, licencia libre dual GPL y EUPL, otras licencias libres, dominio publico ▪ Si 5.2= a cualquier valor del grupo "Servicio" entonces: no corresponde ▪ Si 5.2 ‡ a cualquier valor de los grupos "Aplicación informática' y/o "Servicio" entonces: licencia propietaria, creative commons: reconocimiento, creative commons: reconocimiento - sin obra derivada, creative commons: reconocimiento - sin obra derivada - no comercial, creative commons: reconocimiento - no comercial, creative commons: reconocimiento - no comercial - compartir igual, creative commons: reconocimiento - compartir igual, licencia GFDL, dominio público	Vocabulario LOM-ES
6.3	Descripción	(re.)	1	No especificado		LangString 1000
6.4	Acceso	(ob.)	1	No especificado		
6.4.1	Tipo de acceso	(ob.)	1	No especificado	universal, no universal	Vocabulario LOM-ES
6.4.2	Descripción	(ob.)	1	No especificado		LangString 1000

Esta categoría describe los derechos de propiedad intelectual y las condiciones de uso aplicables a este objeto digital educativo.

NOTA 1: la intención es reutilizar trabajos procedentes de las comunidades del Derecho de la Propiedad Intelectual y el comercio electrónico. Esta categoría proporciona actualmente el nivel mínimo de detalle.

Coste:

Indicación de si este objeto educativo requiere pago.

§ Ejemplo etiqueta en xml:

<cost>
<source uniqueElementName="source">LOM-ESv1.0</source>
<value uniqueElementName="value">no</value>
</cost>

Categoría 7. Relación

Nº	Nombre	Carácter	Tamaño	Orden	Espacio de valores	Tipo de datos
7	Relación	(op.)	m.m.p. 100	No ordenado		
7.1	Tipo	(ob.)	1	No especificado	Basado en Dublin Core: es parte de, tiene parte, es versión de, tiene versión, es formato de, tiene formato, referencia, es referenciado por, se basa en, es base para, requiere, es requerido por	Vocabulario
7.2	Recurso	(ob.)	1	No especificado		
7.2.1	Identificador	(ob.)	1	No especificado		
7.2.1.1	Catálogo	(ob.)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 1000
7.2.1.2	Entrada	(ob.)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 1000
7.2.2	Descripción	(op.)	m.m.p. 10	No especificado		LangString 1000

Esta categoría describe las relaciones existentes, si las hubiese, entre este objeto educativo y otros. Para definir relaciones múltiples deben utilizarse varias instancias de esta categoría. Si existen varios objetos educativos con los cuales éste está relacionado, cada uno de ellos tendrá una instancia propia de esta categoría.

Tipo:

Naturaleza de la relación entre este objeto y el objeto educativo objetivo identificado por 7.2. Relación. Recurso

§ Ejemplo etiqueta en xml:

<kind>
<source uniqueElementName="source">LOM-ESv1.0</source>
<value uniqueElementName="value">es parte de</value>
<source uniqueElementName="source">LOM-ESv1.0</source>
<value uniqueElementName="value">es referenciado por</value>
</kind>

Categoría 8. Anotación

Nº	Nombre	Carácter	Tamaño	Orden	Espacio de valores	Tipo de datos
8	Anotación	(op.)	m.m.p. 30	No ordenado		
8.1	Entidad	(ob.)	1	No especificado	vCard tal y como se define en el IMC vCard 3.0 (RFC 2425, RFC 2426)	CharacterString 1000
8.2	Fecha	(ob.)	1	No especificado		Fecha
8.3	Descripción	(ob.)	1	No especificado		LangString 1000

Esta categoría proporciona comentarios sobre la utilización pedagógica de este objeto educativo, e información sobre quién creó el comentario y cuando fue creado. Esta categoría permite a los educadores compartir sus valoraciones sobre el objeto educativo, recomendaciones para su utilización, etc.

Entidad:

La entidad (persona u organización) que creó esta anotación.

Ejemplo etiqueta en xml:

<entity>BEGIN:VCARD VERSION 3.0 FN:GT8/SC36/CTN71 EMAIL;TYPE=INTERNET:gt8@aenor.es ORG:AENOR END:VCARD</entity>

<entity>BEGIN:VCARD VERSION 3.0 FN:GT9/SC36/CTN71 EMAIL;TYPE=INTERNET:gt9@aenor.es ORG:AENOR END:VCARD</entity>

Fecha:

La fecha en la que se creó esta anotación. Para más información sobre la estructura de los elementos tipo “Fecha”.

Ejemplo etiqueta en xml:

<date>

<dateTime uniqueElementName="dateTime">2006-09-13</dateTime>

<description>

<string language="es">Se ha usado en el campus virtual</string>

</description>

<dateTime uniqueElementName="dateTime">2006-10-12</dateTime>

<description>

<string language="es">Se uso en un grupo de integración</string>

44/62 – LOM-ES v.1.0 12/03/2008

</description>

</date>

Categoría 9. Clasificación

Nº	Nombre	Carácter	Tamaño	Orden	Espacio de valores	Tipo de datos
9	Clasificación	(re.)	m.m.p. 40	No ordenado		
9.1	Propósito	(ob.)	1	No especificado	disciplina, idea, prerrequisito, objetivo educativo, restricciones de accesibilidad, nivel educativo, nivel de habilidad, nivel de seguridad, competencia	Vocabulario
9.2	Ruta Taxonómica	(ob.)	m.m.p. 15	No ordenado		
9.2.1	Fuente	(ob.)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	LangString 1000
9.2.2	Taxón	(ob.)	m.m.p. 15	Ordenado		
9.2.2.1	Identificador	(ob.)	1	No especificado	Repertorio del ISO/IEC 10646-1:2000	CharacterString 100
9.2.2.2	Entrada	(ob.)	1	No especificado		LangString 500
9.3	Descripción	(op.)	1	No especificado		LangString 2000
9.4	Palabras clave	(re.)	m.m.p. 40	Ordenado		LangString 1000

Esta categoría describe dónde se sitúa este objeto digital dentro de un sistema de clasificación concreto. Para definir múltiples clasificaciones, deben utilizarse múltiples instancias de esta categoría.

Propósito:

El propósito que se persigue al clasificar este OA. Por otro lado y como se ha descrito en las categorías 1 y 5, la cobertura curricular aproximada estaría relacionada con el nivel de agregación y el tipo de objeto (elementos 1.8 y 5.2 respectivamente). De esta forma, para niveles de agregación 2 o superior es conveniente seguir unos criterios mínimos de clasificación con respecto a los Propósitos *Disciplina* y *Nivel Educativo*.

NOTA 1: se prescribe la clasificación con respecto a los Propósitos *Disciplina* y *Nivel Educativo* en objetos de nivel de agregación (1.8) 2 o superior.

Glosario de términos.

UML: Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modelling Language) es el lenguaje de modelado de sistemas de software más conocido en la actualidad.

SCORM (Sharable Content Object Reference Model): Modelo de referencia para el desarrollo e integración de contenidos educativos.

FTP (File Transfer Protocol): Es un protocolo de transferencia de archivos entre sistemas conectados a una red TCP basado en la arquitectura cliente-servidor.

XML: Un metalenguaje extensible de etiquetas desarrollado.

IMS: Es un consorcio internacional que ha propuesto un conjunto de especificaciones sobre distintos aspectos que intervienen en el modelado del aprendizaje en *e-learning.*

HTTP (HyperText Transfer Protocol): El protocolo de transferencia de hipertexto usado en cada transacción de la Web.

HTTPs (Protocolo seguro de transferencia de hipertexto): es un protocolo de red basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto.

SQL: Lenguaje de consulta estructurado.

Apache: Servidor HTTP de páginas Web desarrollado por la Apache Sotfware Fundation.

Aplicación: Es el programa que el usuario activa para trabajar en el ordenador. Existen muchos programas de ordenador que pueden clasificarse como aplicación. Generalmente se les conoce como Software.

Browser: Navegador. Herramienta para "Visitar" o visualizar el contenido de las páginas Web y sitios FTP en Internet.

Content Management Systems (CMS) o Sistemas de Gestión de Contenidos: Aplicaciones software que en la industria de las publicaciones online permiten la generación de los sitios web dinámicos.

Especificación: Es un documento técnico que describe los componentes (parte estática) y el comportamiento (parte dinámica) de un determinado sistema.

Estructura cliente / servidor: Consiste en que los clientes piden que una tarea sea realizada y el servidor realiza dicha tarea y regresa la información al cliente a través de la red.

e-Learning: Abarca al conjunto de las metodologías y estrategias de aprendizaje que emplean tecnología digital o informática para producir, transmitir, distribuir, y organizar conocimiento entre individuos, comunidades y organizaciones

Intranet: Red privada, desarrollada dentro de una compañía que utiliza el mismo software y provee de información similar que Internet, solo que es únicamente para el uso interno.

Metadatos: Conjunto de atributos o elementos necesarios para describir un recurso en cuestión.

Objetos de aprendizaje: cualquier recurso con una intención formativa, compuesto de uno o varios elementos digitales, descrito con metadatos, que pueda ser utilizado y reutilizado dentro de un entorno e-learning.

Open Source: *Código abierto.* Manera de nombrar también a las aplicaciones desarrolladas bajo el amparo de un producto con licencia GPL.

TICs: Tecnologías de la Informática y las Comunicaciones.