TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

FACULTAD 4

Desarrollo de un componente para la reutilización de contenido educativo en Sistemas de Gestión de Contenidos Educativos implementados en Drupal 7.

Autor: Annerys Díaz Chacón.

Tutor: Ing. Javier Soler Martín.

Ing. Orlando F. Salvador Broche.

La Habana 2014

Año 56 de la Revolución

DECLARACIÓN DE AUTORÍA

Declaro que soy la autora de este trabajo y autorizo a la Facultad 4 de la Universidad de las Ciencias Informáticas; así como a dicho centro 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 año 2014.

_____

AUTOR
Annerys Díaz Chacón

_____ _____

TUTOR TUTOR

Ing. Javier Soler Martín Ing. Orlando F. Salvador Broche

RESUMEN

El desarrollo constante de las tecnologías de la información y las comunicaciones (TIC), fomentan la práctica de educación a distancia apoyada por actividades en línea, conocida como Aprendizaje Electrónico o e-learning, generando esquemas de producción basados en Herramientas Educativas; línea central de desarrollo del Centro de Tecnologías para la Formación (FORTES), perteneciente a la Universidad de las Ciencias Informáticas (UCI). Las mismas, juegan un rol específico dentro de un entorno e-learning, donde necesitan compartir la información que gestionan. Para lo cual los sistemas utilizan estándares y especificaciones, enmarcados en crear modelos de datos y protocolos para gestionar, describir o para compartir información referente a contenidos educativos usando metadatos. En este sentido, los Sistemas de Gestión de Contenido Educativo implementados en Drupal 7, utilizan modelos de datos semánticamente similares a los estándares, que son desaprovechados o no utilizados al máximo en la reutilización de contenido educativo. Por lo que se definió como objetivo de la investigación, desarrollar un componente para la transformación de contenido educativo estandarizado a modelos de datos específicos de Sistemas de Gestión de Contenido Educativo implementados en Drupal 7 y viceversa. Desarrollando el marco teórico conceptual, definiendo las características del sistema y generando los principales artefactos correspondientes al análisis, diseño e implementación. Aplicando las pruebas de caja blanca y caja negra al componente desarrollado con resultados satisfactorios, validando la funcionalidad y eficiencia del mismo.

Palabras claves: E-learning, Sistema de Gestión de Contenido Educativo, Drupal, SCORM.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El mundo actual se encuentra en contante cambio y desarrollo, proceso estrechamente vinculado al surgimiento continuo de nuevas formas de producción, relaciones sociales y tecnologías. Lo cual representa un enorme potencial de cambio en la acumulación de la información, la velocidad de transmisión, la superación de las limitaciones espaciales, la utilización simultánea de medios (imagen, sonido y texto), entre otros. De manera que *"las tecnologías de la información y las comunicaciones (TIC) pueden contribuir al acceso universal a la educación, la igualdad en la instrucción, el ejercicio de la enseñanza y el aprendizaje de calidad y el desarrollo profesional de los docentes, así como a la gestión, dirección y administración más eficientes del sistema educativo''* (1); fomentando así la práctica de educación a distancia apoyada por actividades en línea, conocido como Aprendizaje Electrónico o e-learning[1].

"El e-learning consiste en la educación y capacitación a través de Internet. Este tipo de enseñanza online permite la interacción del usuario con el material mediante la utilización de diversas herramientas informáticas". (2) Esto, combinado con el aprendizaje presencial, es denominado Aprendizaje Mixto o b-learning[2], y brinda una amplia gama de posibilidades, permitiendo un aprendizaje guiado y diferenciado, mediante el uso de tecnologías que enriquecen los conocimientos adquiridos en el aula. Generando esquemas de producción basados en Herramientas Educativas, línea central de desarrollo del Centro de Tecnologías para la Formación (FORTES), perteneciente a la Universidad de las Ciencias Informáticas (UCI). En el cual se concibe una estructura organizativa departamental para el desarrollo de componentes reutilizables, utilizados para ensamblar o extender las funcionalidades de aplicaciones de corte educativo.

Entre las Herramientas Educativas desarrolladas o personalizadas por el centro FORTES, se encuentran los Repositorios de Objetos de Aprendizaje, las Herramientas de Autor, los Sistemas de Gestión del Aprendizaje, los Sistemas de Gestión de

1 e-learning: Electronic Learning.
2 b-learning: Blended Learning.

Contenido Educativo, los Portafolios de Evidencias, las Bibliotecas Digitales, entre otras; las que juegan un rol específico dentro de un entorno e-learning, donde necesitan compartir la información que gestionan. Para lo cual, los sistemas deben seguir estándares y especificaciones, establecidos a nivel internacional para el desarrollo de aplicaciones de corte educativo. Estos se enmarcan en crear modelos de datos, procesos y protocolos para gestionar, describir y compartir información referente a contenidos educativos usando metadatos. Destacando entre los más usados y recomendados, OAI-PMH[3], SCORM[4], LOM[5], Dublincore[6], IMSCP[7], etc.

En este sentido, los Sistemas de Gestión de Contenido Educativo implementados en Drupal 7, utilizan modelos de datos particulares de su implementación, sin seguir estándares y especificaciones al modelar entidades en un entorno educativo. Aun así, estos manejan conceptos semánticamente similares a los estándares, que son desaprovechados o no utilizados al máximo en la reutilización de contenido educativo dentro de un entorno e-learning, característica indispensable de estos entornos.

Surgiendo así, a partir de la situación anteriormente descrita, el **problema a resolver**: ¿Cómo contribuir a la reutilización de contenido entre un Sistema de Gestión de Contenido Educativo implementado en Drupal 7 y el resto de las herramientas en un entorno e-learning?

Teniendo como **objetivo general**: Desarrollar un componente para la transformación de contenido estandarizado a modelos de datos específicos de Sistemas de Gestión de Contenido Educativo implementados en Drupal 7 y viceversa.

Definiendo como **objeto de estudio**: Los entornos e-learning.

Enmarcándose en el **campo de acción**: Componentes y estándares para la reutilización de contenido educativo entre herramientas de un entorno e-learning.

3 http://www.openarchives.org/pmh
4 http://www.scorm.com
5 IEEE 1484.12.1 – 2002
6 http://www.dublincore.org
7 http://www.imsglobal.org/content/packaging/

Indicando como **idea a defender**: Con el desarrollo de un componente para la transformación de contenido estandarizado a modelos de datos específicos de Sistemas de Gestión de Contenido Educativo implementados en Drupal 7 y viceversa, se contribuye a la reutilización de contenido entre un Sistema de Gestión de Contenido Educativo implementado en Drupal 7 y el resto de las herramientas en un entorno e-learning.

Definiendo los **objetivos específicos**:

1. Elaborar el marco teórico conceptual para la elaboración del componente.
2. Determinar las características del componente.
3. Desarrollar los principales artefactos de los flujos de Análisis y Diseño e Implementación del componente para la transformación de contenidos.
4. Realizar pruebas internas para garantizar la funcionalidad del componente.

Para el cumplimiento de los objetivos específicos se establecen como **tareas de investigación**:

1. Realizar una revisión bibliográfica actualizada para generar el marco teórico conceptual.
2. Analizar los principales conceptos y herramientas de un entorno e-learning.
3. Seleccionar las herramientas y tecnologías a utilizar en el desarrollo del componte.
4. Elaborar la propuesta de solución a la problemática planteada.
5. Definir los requerimientos funcionales y no funcionales del sistema.
6. Realizar el Análisis y Diseño del componente a desarrollar.
7. Implementar el componente para la transformación de contenidos.
8. Realizar las pruebas de caja blanca para validar el componente.
9. Realizar las pruebas de caja negra para validar el componente.

Para el desarrollo de la presente investigación se emplearon los siguientes métodos científico-teóricos:

- Histórico-lógico: Utilizado al tener en cuenta la evolución histórica del e-learning.

- Analítico-sintético: Utilizado en la extracción de los elementos más importantes relacionados con los entornos e-learning.
- Inductivo-deductivo: Una vez analizada y comprendida toda la información referente a los entornos e-learning, este método es utilizado como base fundamental para el trabajo con los Sistemas de Gestión de Contenido Educativo.

La presente investigación está organizada en cuatro capítulos:

Capítulo 1. Se aborda en detalles lo relacionado con la fundamentación teórica que sustenta el presente trabajo. Analizando los principales conceptos relacionados con el objeto de estudio, las herramientas, las tecnologías y la metodología utilizada.

Capítulo 2. Se elabora el modelo de dominio, la propuesta del sistema a implementar y los requisitos funcionales y no funcionales del mismo. Son determinadas las relaciones entre los actores y casos de uso, mediante el diagrama de casos de uso del sistema y la descripción de cada uno de los casos de uso.

Capítulo 3. Se realiza un análisis y diseño del sistema, generando así los artefactos diagramas de clases del análisis, diagramas de colaboración, diagramas de clases del diseño y modelo de datos, lo que permite transformar los requerimientos en un diseño de cómo va a ser implementado el sistema.

Capítulo 4. Se establecen los estándares de codificación a utilizar y se elaboran los artefactos diagrama de componentes y diagrama de despliegue. Se seleccionan los métodos de evaluación para el componente desarrollado y se muestran los resultados de aplicar los mismos.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

Este capítulo brinda el marco teórico conceptual que sustentará el desarrollo de los restantes capítulos. Se analizan los principales elementos de un entorno e-learning; como los objetos de aprendizaje, la interoperabilidad, las herramientas que lo conforman, y los estándares para la comunicación y reutilización de contenidos. Se define el ambiente de desarrollo, especificando la metodología, los lenguajes y las herramientas que se establecerán para ser empleadas en la construcción del componente.

1.2 Objeto de Aprendizaje

Los Objetos de Aprendizaje (OA) son utilizados en la instrucción, aprendizaje o enseñanza basada en computadora, y están fundamentados en la ciencia de la computación orientada a objetos. Según la IEEE, un OA *"es cualquier entidad digital o no digital que puede ser usada, reutilizada o referenciada durante un proceso de aprendizaje apoyado por la tecnología"*. (3) Wiley plantea que *"es cualquier recurso digital que pueda ser reutilizado como soporte para el aprendizaje"*. (4) Polsani lo define como "*una unidad didáctica de contenido, autocontenida e independiente, predispuesta para su reutilización en múltiples contextos instruccionales*". (5) Y según JORUM+ Project, es *"cualquier recurso que puede ser utilizado para facilitar la enseñanza y el aprendizaje y que ha sido descrito utilizando metadatos"*. (6)

Dada la amplia variedad de definiciones, se define para el presente trabajo, que un OA es *cualquier recurso compuesto de uno o varios elementos digitales, descrito con metadatos, que pueda ser utilizado y reutilizado dentro de un entorno e-learning*. Se define además que los mismos deben ser fáciles de localizar, utilizar, almacenar y compartir, por lo que según Rehak & Mason deben ser:

- **Reutilizables:** El recurso debe ser modular para servir como base o componente de otro recurso. También debe tener la tecnología, la estructura y los componentes necesarios para ser incluido en diversas aplicaciones.

- **Accesibles:** Pueden ser indexados para una localización y recuperación eficiente, utilizando estándares de metadatos.
- **Interoperables:** Pueden operar entre diferentes plataformas de hardware y software.
- **Portables:** Pueden moverse y albergarse en diferentes plataformas de manera transparente, sin cambio alguno en su estructura o contenido.
- **Durables:** Deben permanecer intactos a las actualizaciones (upgrades) de software y hardware. (7)

1.3 Interoperabilidad

La interoperabilidad según la IEEE, es la "habilidad de dos o más sistemas o componentes de intercambiar información y usar la información intercambiada". (8) Es según ISO, *"la capacidad de un sistema o componente de un sistema para proporcionar intercambio de información y procesamiento cooperativo entre aplicaciones"*. (9) Y es según la Comisión Europea, *"la habilidad de los sistemas TIC, y de los procesos de negocios que ellas soportan, de intercambiar datos y posibilitar compartir información y conocimiento".* (8)

Partiendo de las definiciones antes expuestas, de la bibliografía consultada y basado en el contexto del componente a desarrollar, se define la interoperabilidad como *la habilidad de los sistemas, de intercambiar información y conocimiento, haciendo uso de estándares que garanticen un mismo formato en la estructura de los datos*. Y esta puede ser organizada en los siguientes niveles:

- El **nivel técnico**, se encarga de proporcionar mecanismos comunes de transferencia de los datos y de invocación de funciones, transparentes al sustrato de redes y sistemas informáticos existentes.
- El **nivel semántico**, se encarga de garantizar el significado preciso de la información intercambiada y permite que pueda ser entendida por cualquier aplicación. En este sentido, el lenguaje XML es señalado en reiteradas ocasiones, como el propicio para desarrollar estas semánticas comunes.
- El **nivel organizativo**, el cual garantiza la coordinación y el alineamiento de los procedimientos administrativos que intervienen en la provisión de los servicios de gobierno electrónico. Para lo cual es imprescindible disponer del conjunto de estándares y especificaciones que brindan el marco que define y

comprende el conjunto mínimo de normas técnicas que deben adoptar los organismos. (8)

1.4 Plataformas e-learning

Las plataformas e-learning son sistemas que posibilitan la simulación de aulas virtuales, produciéndose en ellas la interacción entre tutores y alumnos, y entre los mismos alumnos. Permiten además la realización de evaluaciones, el intercambio de contenidos, la participación en foros, chats, entre otros. Estos escenarios deben considerar aspectos esenciales como el diseño de la institución, la pedagogía, la tecnología, la evaluación, el soporte, entre otros aspectos, siendo una combinación de recursos, interactividad, apoyo y actividades de aprendizaje estructuradas. (6)

"Para la reutilización e interoperabilidad de contenidos en diferentes plataformas, debe cumplirse una doble premisa: por un lado los cursos deben seguir un estándar y por otro lado las plataformas deben soportar dicho estándar, con lo que se facilita el uso de cursos realizados por la propia organización y por terceros". (10)

Estas plataformas están conformadas por cuatro características básicas:

1. La **interactividad**, basada en la concientización del protagonismo de los usuarios en su formación.
2. La **flexibilización**, permitiéndole adaptarse a la institución, a los planes de estudio, y a los contenidos y estilos pedagógicos que se llevan a cabo en una organización determinada.
3. La **escalabilidad**, capacidad de funcionar de igual forma sin depender del número de usuarios que accedan a la plataforma.
4. La **estandarización**, capacidad de utilizar cursos realizados por terceros, siempre que cumplan con los estándares, garantizando la durabilidad y realizando seguimiento al comportamiento de los estudiantes dentro de los cursos. (10)

1.4.1 Sistema de Gestión del Aprendizaje

Un Sistema de Gestión del Aprendizaje (LMS: Learning Management System), es una aplicación de software instalada en un servidor, que se emplea para administrar,

distribuir y controlar las actividades de formación e-Learning de una institución u organización. Estos sistemas cumplen funciones como la gestión de los usuarios, los recursos, y las actividades de formación. Permiten controlar y seguir el proceso de enseñanza y aprendizaje, mediante la generación de informes, los foros de discusión, las videoconferencias, entre otras modalidades, generando una evaluación del estudiante y promoviendo la colaboración entre estos. (6)

"Permite descargar documentos y contenidos almacenados en un repositorio que pueden convertirse en apuntes, documentos de consulta, es decir, informaciones que sirven al alumno en su proceso de formación. Se utiliza para diseñar y planificar el proceso educativo en un ambiente propicio de colaboración y autorregulación temporal didáctica, que genera en los estudiantes, a la vez, nuevos conocimientos sobre la materia en que versa el programa educativo, competencias tecnológicas, competencias comunicativas y la comprensión de un «entorno virtual de aprendizaje» en el que las relaciones alumno-profesor y alumno-alumno se reorganizan para «acompañarse virtualmente» evitando la deshumanización y «frialdad» de proceso educativo soportado en un software en línea".(11)

1.4.2 Sistema de Gestión de Contenido Educativo

Un Sistema de Gestión de Contenido Educativo (LCMS: Learning Content Management System), es la infraestructura sobre la cual el aprendizaje online es creado y desarrollado. Tiene su origen en los Sistemas de Gestión de Contenidos (CMS: Content Management System) y tiene como objetivo simplificar la creación y administración de contenidos en línea. Se utiliza para manejar contenidos que forman parte de un programa de educación, en forma de módulos que se pueden personalizar, manejar, y que se pueden usar en diferentes ocasiones o cursos. Utilizando generalmente lenguaje XML y siguiendo los estándares y especificaciones de la enseñanza digital como IMS, AICC y SCORM.(6)

1.4.3 Repositorio de Objetos de Aprendizaje

"Debido a que los OA tienen como una de sus características principales la reutilización, necesitan un lugar para almacenarlos y clasificarlos, para posteriormente darles mantenimiento, poder localizarlos y poder compartirlos con otras aplicaciones. Este almacén es lo que se conoce como Repositorio de Objetos de Aprendizaje".(7)

Un Repositorio de Objetos de Aprendizaje (ROA) es una combinación entre una biblioteca digital y un buscador, pero mucho más sofisticado que ambos, con criterios de búsqueda que consideran aspectos más allá del título, los autores y las palabras claves. Destacado por no albergar físicamente los OA que contienen, sino apuntar a ellos utilizando los metadatos asociados, lo que facilita la indexación de los mismos para ser buscados fácilmente en Internet.

Los ROA pueden clasificarse atendiendo a:

- **La forma de concentrar sus recursos en:**

    1. Los que contienen los objetos de aprendizaje y sus metadatos dentro de un mismo sistema e incluso dentro de un mismo servidor.
    2. Los que contienen sólo los metadatos y se accede al objeto a través de una referencia a su ubicación física que se encuentra en otro sistema o repositorio de objetos.

- **La forma en la que los catálogos de metadatos se organizan:**

    1. **Centralizados:** donde los metadatos de los OA están contenidos en un mismo servidor, aunque el objeto esté localizado en otro servidor.
    2. **Distribuidos:** operan a través de varios servidores, cada uno contiene diferentes grupos de metadatos y se comunican entre ellos para intercambiarlos. (6)

1.5 Estándares y especificaciones

“Un estándar es un conjunto de especificaciones técnicas documentadas que regulan la realización de un proceso o la fabricación de un producto. Si de lo que se trata es de normalizar la elaboración de un producto, el objetivo de la estandarización es fundamentalmente la interoperabilidad entre artículos construidos por diferentes fabricantes”.(12) Los estándares pueden ser de **facto** (de hecho o por convención), de **jure** (por ley o por regulación) y de **acuerdo**.

- Los **estándares de facto** se pueden subdividir en dos clases: propietario (son propiedad de la compañía que los inventó) y no propietario (desarrollados por grupos o comités que los han transferido al dominio público).
- Los **estándares de jure** son aquellos que han sido legislados por un organismo oficialmente reconocido.
- Los **estándares de acuerdo** son aquellos que son definidos por convenio, alianza o pacto entre proveedores, usuarios, manufactureros entre otros. De manera que, si una tecnología, formato o método ha sido ratificado por algún organismo oficial de estandarización, se trata de un estándar, en otro caso es una especificación. Aunque, una especificación puede considerarse un estándar de facto si su uso es extendido y entretanto se ratifica como estándar. (12)

Actualmente son varias las organizaciones que se dedican a la definición y establecimiento de estándares y especificaciones para el e-learning, debido al gran alcance que tienen los entornos e-learning en todo el mundo, y a la necesidad de que estos sean integrados e interoperables. Entre las más destacadas encontramos la IEEE (The Institute of Electrical and Electronics Engineers), ADL (Advanced Distributed Learning), OAI (Open Archives Iniciative), etc. (12)

1.5.1 Learning Object Metadata

Learning Object Metadata (LOM) *"es un modelo de datos, usualmente codificado en XML, usado para describir un objeto de aprendizaje y otros recursos digitales similares usados para el apoyo al aprendizaje. Su propósito es ayudar a la reutilización de objetos de aprendizaje y facilitar su interaccionalidad, usualmente en el contexto de sistemas de aprendizaje on-line: (online learning management systems (LMS))".* (13)

LOM es acreditado por el estándar 1484.12.1 de la (IEEE), como el estándar de metadatos para OA.(3) Especificando la semántica y la sintáctica de un conjunto mínimo de metadatos necesarios para, completa y adecuadamente, identificar, administrar, localizar y evaluar un OA. Está compuesto por 9 categorías: **general**, **ciclo de vida**, **metadatos**, **técnico**, **educacional**, **derechos**, **relación**, **anotación** y **clasificación**. Una vez generados los recursos educativos éstos se pueden dotar de información adicional, que luego permite realizar búsquedas sobre la base de diferentes criterios.(14)

1.5.2 DUBLIN CORE

"Dublin Core es un modelo de metadatos elaborado y auspiciado por la DCMI (Dublin Core Metadata Initiative), una organización dedicada a fomentar la adopción extensa de los estándares interoperables de los metadatos y a promover el desarrollo de los vocabularios especializados de metadatos para describir recursos".(15) Destacando entre las iniciativas para normalizar y estandarizar los metadatos sobre los recursos electrónicos. Creado inicialmente para catalogar y compartir información sobre libros y usado actualmente en muchas de las páginas web existentes en Internet.

1.5.3 IMS Content Packaging

IMS Content Packaging (IMS-CP) es una especificación de gran importancia para los ROA, ya que les provee la función para formar paquetes de contenidos que estos pueden exportar a los LMS y permite que los LCMS puedan utilizar los OA. Brinda a los ROA la capacidad de importar y extraer el contenido de los paquetes que recibe. Está compuesto por el manifiesto y el contenido. El manifiesto, consta de secciones, que son los elementos XML que le describen a través de metadatos, especificando la organización del contenido, las referencias a los recursos que componen el contenido incluyendo sus metadatos y las referencias a archivos externos. Y el contenido es el recurso en sí, que junto al manifiesto se comprime, y es llamado Archivo de Intercambió de Paquete o PIF[8]. (14)

1.5.4 Sharable Content Object Reference Model

Sharable Content Object Reference Model (SCORM) es un conjunto de especificaciones para el desarrollo, empaquetamiento y distribución de material educativo en cualquier momento y en cualquier lugar. El mismo asegura que un material sea: **Reutilizable**, **Accesible**, **Interoperable** y **Durable**; imprescindibles para el empaquetamiento de los cursos on-line, ya que las plataformas de e-learning organizan sus bases de datos de diferentes maneras, lo que hace muy complejo el

8 PIF: Package Interchange File.

manejo de los cursos entre distintas plataformas.(14)

SCORM posee su documentación distribuida en cuatro libros, que son actualizados de forma independiente. El primer libro, **SCORM Overview** describe la historia y los objetivos de la Iniciativa ADL y de SCORM, incluyendo las especificaciones y los estándares que SCORM ha adoptado para su definición, al igual que su relación con los restantes libros. El segundo, **SCORM Content Aggregation Model** (**CAM**), describe los componentes utilizados en el aprendizaje, cómo empaquetar los componentes para el intercambio entre sistemas, cómo describir esos componentes para permitir la búsqueda y la recuperación, y cómo definir las reglas de secuencia de los componentes. Más específicamente, describe el modelo de datos concreto en XML, que permite serializar todos los datos del modelo conceptual definido en los restantes libros (metadatos y secuencia y navegación), a la vez que define un modelo para la organización de contenidos de forma jerárquica. El tercero, **SCORM Run-Time Environment** (**RTE**), describe el medio para interoperar contenidos de aprendizaje basados en Sharable Content Object (SCO) y los LMS (16). El cuarto y último libro, **SCORM Sequencing and Navigation** (**SN**) describe las reglas que un LMS debe seguir a fin de presentar un aprendizaje específico.(17)

Componentes del modelo:

- **Asset:** Son una representación electrónica de los medios de comunicación (texto, imágenes, sonido, objetos de evaluación, etc.), que pueden ser proporcionados por un cliente Web y presentados a un alumno.
- **Sharable Content Object (SCO):** Un SCO es una colección de uno o más Asset representado en un único recurso de aprendizaje. Utilizado para la comunicación de servicios en tiempo de ejecución con un LMS, mediante el estándar IEEE ECMAScript Application Programming Interface.
- **Activities:** Las actividades (Figura 1) representan conceptualmente acciones dentro de la instrucción. Las que pueden referenciar un recurso de aprendizaje (SCO o Asset), estar representadas en una organización de contenidos y poseer otras actividades (subactividades), no poseyendo un límite de anidamiento definido. Pudiendo asociar sus niveles jerárquicos a una taxonomía (capítulo, módulo, etc.)

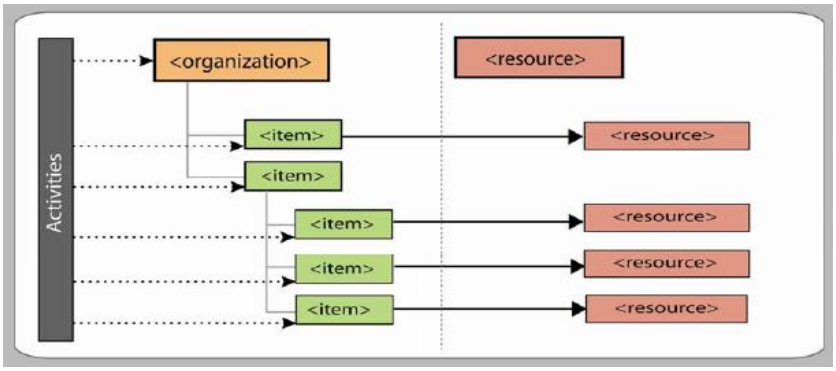


Figura 1. Actividades de SCORM

- **Content Organization:** Una organización de contenidos es una representación o mapa, descrito por metadatos y permite definir el uso previsto de los contenidos a través de unidades estructuradas de enseñanza (actividades), mostrando como estas se relacionan entre sí (Figura 2).

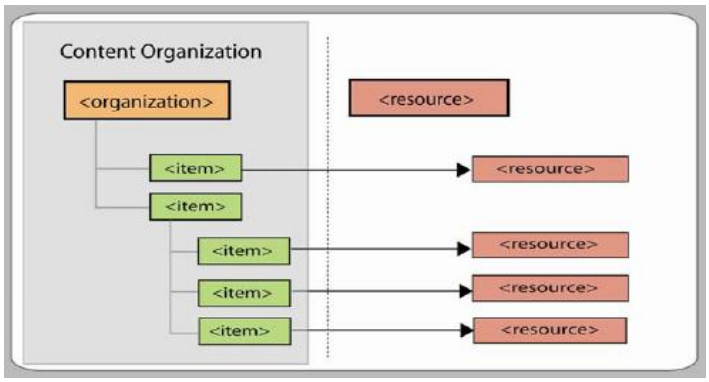


Figura 2. Organización de contenidos de SCORM

- **Content Aggregation:** La agregación de contenidos (Figura 3), puede ser utilizado como una acción y como una manera de describir una entidad conceptual. Se utiliza para describir una entidad creada como parte del proceso o acción, para describir el contenido de un paquete y la estructura de los mismos, con el fin de hacer transferencias entre sistemas o almacenarlos en un repositorio. (14)

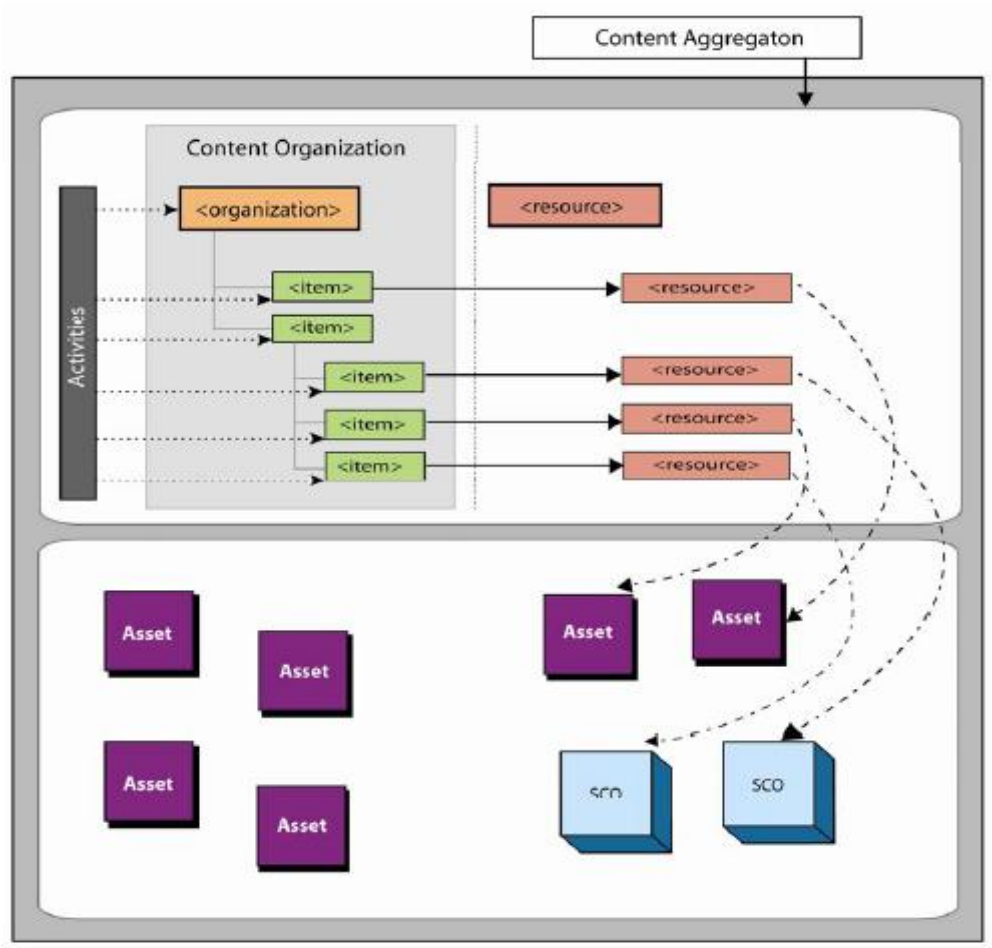


Figura 3. Agregación de contenidos de SCORM

Componentes del paquete:

SCORM utiliza la especificación IMS-CP (Figura 4), para describir estructuras de datos utilizadas en la interoperabilidad de contenido. Teniendo como objetivo definir un conjunto estandarizado de estructuras entre sistemas que desean importar, exportar, agregar y desagregar los paquetes de contenidos, compuestos por dos componentes principales:

- **Imsmanifest.xml:** Documento XML especial que describe la estructura de los contenidos y asocia los recursos al paquete llamado archivo del manifiesto.
- **El contenido:** Archivos físicos que componen el paquete de contenidos. (14)

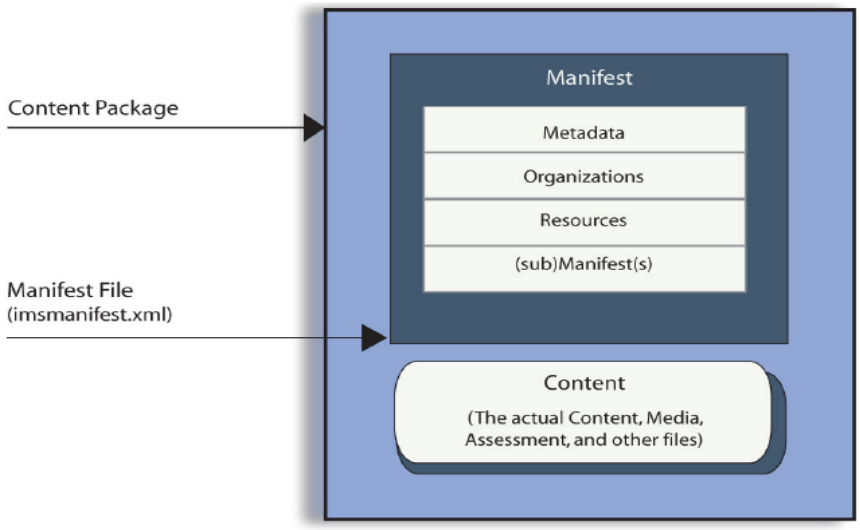


Figura 4. Diagrama conceptual del paquete de contenidos

Componentes del manifiesto:

El archivo del manifiesto (Figura 5), representa la información necesitada para describir el contenido del paquete, compuesto de cuatro secciones principales:

- **Metadatos:** Los datos que describen el paquete de contenido como un todo.
- **Organizaciones:** Contienen la estructura de los contenidos o de la organización de los recursos de aprendizaje que constituyen una unidad o unidades de instrucción independiente.
- **Recursos:** Define los recursos de aprendizaje agrupados en el paquete de contenido.
- **(Sub) manifiestos:** Describen cualquier unidad lógica de instrucción anidada (que pueden ser tratados como unidades independientes). (14)

Figura 5. Componentes del manifiesto

1.5.5 Open Archives Initiative – Protocol for Metadata Harvesting

La Open Archives Initiative (OAI) se creó con la misión de desarrollar y promover estándares de interoperabilidad para facilitar la difusión eficiente de contenidos en Internet; principalmente publicaciones científicas, y posteriormente para permitir el intercambio de múltiples formatos bibliográficos entre distintas máquinas utilizando un protocolo común para la transmisión de metadatos sobre cualquier material almacenado en soporte electrónico, los cuales se deben codificar en Dublin Core. OAI no define un ningún esquema para la gestión de derechos, sino que los temas relacionados con restricciones en el acceso y gestión de la propiedad intelectual son responsabilidad de los proveedores de datos. (18)

Open Archives Initiative - Protocol for Metadata Harvesting (OAI-PMH), *"utiliza transacciones HTTP para emitir preguntas y obtener respuestas entre un servidor o archivo y un cliente o servicio recolector de metadatos. El segundo puede pedir al primero que le envíe metadatos según determinados criterios como por ejemplo la fecha de creación de los datos. En respuesta el primero devuelve un conjunto de registros en formato XML, incluyendo identificadores (URL por ejemplo) de los objetos descritos en cada registro".*(18)

1.6 Metodología de Desarrollo de Software

Las Metodologías de Desarrollo de Software son procesos detallados, que se basan en la combinación de distintos modelos de procesos que pueden ser genéricos, en cascada, evolutivos, incrementales, en espiral, entre otros. Estas definen los artefactos, roles y actividades, al igual que las prácticas y técnicas recomendadas, las guías de adaptación de la metodología al proyecto, las guías para uso de herramientas de apoyo, entre otras.(19)

1.6.1 Metodologías Tradicionales y Ágiles

Las metodologías pueden clasificarse en Tradicionales o Ágiles. Estás son seleccionadas con el objetivo aumentar la calidad del producto, con los recursos y tiempos establecidos. Las **Metodologías Ágiles**, son aquellas enfocadas principalmente en los individuos y las interrelaciones entre ellos, como pilar para lograr un producto que satisfaga las necesidades del cliente, en un periodo relativamente

corto. Enfocando los esfuerzos en la creación de un producto de software funcional, sin prestar demasiada atención a la documentación del proceso. Se basan en dar mayor importancia a la colaboración con el cliente, que al cumplimiento de los contratos y negociación de los mismos. Plantean que la capacidad de respuesta ante un cambio es más importante que el seguimiento estricto del plan. Están orientadas a la generación de código con ciclos muy cortos de desarrollo y se dirigen a equipos de desarrollo pequeños. (20)

Las **Metodologías Tradicionales**, considerando su filosofía de desarrollo, son aquellas con mayor énfasis en la planificación y control del proyecto. Enfocadas en la especificación precisa de requisitos y el modelado. Estas, eran inicialmente artesanales en su totalidad, pero para agilizar el proceso y lograr las metas deseadas, se adaptaron metodologías existentes en otras áreas, logrando un desarrollo de software dividido en etapas de manera secuencial, guiadas por una fuerte planificación durante todo el proceso de desarrollo; donde se realiza una intensa etapa de análisis y diseño antes de la construcción del sistema.(20)

Con el fin de analizar las mismas y poder definir cuál es la metodología candidata para el desarrollo del componente, se analizarán a continuación el Proceso Unificado de Desarrollo y la Programación Extrema.

1.6.2 Proceso Unificado de Desarrollo

El Proceso Unificado de Desarrollo (RUP: Rational Unified Process) sigue un conjunto de procedimientos necesarios para transformar los requisitos del usuario en un sistema de software. Definiendo en sí, los Trabajadores (quién), las Actividades (cómo), los Artefactos (qué) y los Flujo de actividades (Cuándo). Los **Trabajadores** (roles), están compuestos por individuos, y/o sistemas automatizados, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos. Las **Actividades** son tareas que tiene un propósito claro, realizadas por trabajadores en las que se manipulan elementos. Los **Artefactos**, son los que producen, modifican y usan productos tangibles del proyecto. Estos pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables. Y los **Flujos de actividades**, son secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable. (21)

RUP agrupa las actividades en nueve flujos de trabajo principales, entre los cuales, los seis primeros son conocidos como flujos de ingeniería y los tres últimos como de apoyo. En la Figura 6 se representa el proceso en el que se grafican los flujos de trabajo y las fases, mostrando la dinámica expresada en iteraciones y puntos de control. (21)

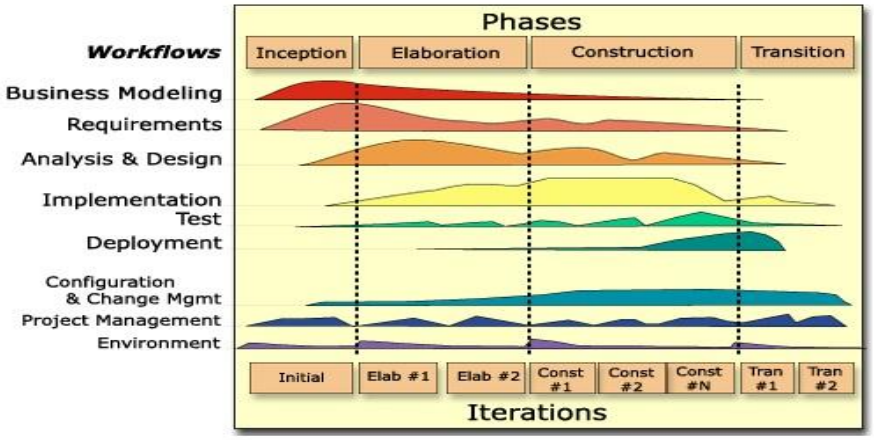


Figura 6. Proceso Unificado de Desarrollo

Flujos de trabajo:

- **Modelamiento del negocio:** Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
- **Requerimientos:** Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
- **Análisis y Diseño:** Describe cómo el sistema será realizado a partir de las funcionalidades previstas y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe implementar.
- **Implementación:** Define cómo se organizan las clases y objetos en los componentes, los nodos que se usarán, y su ubicación y estructura en las capas de la aplicación.
- **Prueba (Testeo):** Busca los defectos a los largo del ciclo de vida del proyecto.
- **Instalación:** Produce el release del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.
- **Administración del proyecto:** Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
- **Administración de configuración y cambios:** Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en

cuanto a: utilización/actualización concurrente de elementos, control de versiones, etc.

- **Ambiente:** Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización. (21)

Fases:

- **Conceptualización (Concepción o Inicio):** Se describe el negocio y se delimita el proyecto describiendo su alcance con la identificación de los casos de uso del sistema.
- **Elaboración:** Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo al alcance definido.
- **Construcción:** Se obtiene un producto listo para su utilización, documentado y con su respectivo manual de usuario. Se obtiene uno o varios release del producto que han pasado las pruebas y se ponen a consideración de un subconjunto de usuarios.
- **Transición:** El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores. (21)

Características definitorias de RUP:

Lo que realmente define y hace único a RUP, es tener un ciclo de vida dirigido por casos de uso, ser centrado en la arquitectura y ser iterativo e incremental:

- **Dirigido por casos de uso:** En el modelamiento del negocio se captan las necesidades y deseos de los usuarios futuros y se representan a través de los requerimientos, los cuales son reflejados en los casos de uso. Estos guían el proceso de desarrollo, ya que los modelos realizados en cada uno de los flujos de trabajo representan la realización de los casos de uso.
- **Centrado en la arquitectura:** La arquitectura muestra la visión común del sistema completo, en la que el equipo de proyecto y los usuarios deben estar

de acuerdo. Describe los elementos del modelo que son más importantes para la construcción; los cimientos del sistema que son necesarios para comprenderlo, desarrollarlo y producirlo económicamente. RUP se desarrolla mediante iteraciones, comenzando por los casos de uso arquitectónicamente significativos. El modelo de arquitectura se representa a través de vistas en las que se incluyen los diagramas UML.

- **Iterativo e incremental:** RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Aunque cada iteración tiene que proporcionar un incremento en el proceso de desarrollo, todas deben aportar al principal resultado de la fase en la que se desarrolla, por lo que los puntos de control evalúan:

    - Conceptualización (Objetivos)
    - Elaboración (Arquitectura)
    - Construcción (Funcionalidad Operativa)
    - Transición (Release del sistema) (21)

Modelo 4+1 Vistas:

Mediante cinco vistas concurrentes propuestas en el Modelo 4+1 Vistas, RUP describe la Arquitectura de Software. Permitiendo tanto a clientes como a especialistas una aproximación a los datos a través de una vista lógica. Los ingenieros de sistema pueden inicialmente abordar la vista física y luego ir al proceso. Los administradores de proyectos y los encargados de la configuración de software pueden enfocarse en la vista de desarrollo. Asimismo estas están guiadas por la vista de casos de uso, que describen las funcionalidades del sistema que más inciden sobre la arquitectura.

- **Vista de Casos de Uso:** Representa un subconjunto del artefacto Modelo de Casos de Uso y lista los casos de uso o escenarios más significativos, con las funciones centrales del sistema. En caso de ser el sistema extenso se deben organizar en paquetes, lo cual facilita su comprensión.
- **Vista Lógica:** Representa un subconjunto del artefacto Modelo de Diseño, representando los elementos de diseño más significativos para la arquitectura del sistema. Describe las clases más importantes, su organización en paquetes y subsistemas, y estos a su vez en capas.

- **Vista de Implementación:** Describe la arquitectura general del Modelo de Implementación, en correspondencia con la Vista Lógica, representando la descomposición del software en capas y paquetes importantes para la arquitectura.
- **Vista de Despliegue:** Suministra la base para comprender la distribución física del sistema a través de nodos, modelando la configuración en función del sistema (software/hardware) y las relaciones entre componentes. Es más usado en sistemas distribuidos.
- **Vista Concurrente o de Procesos:** Describe el diseño de concurrencia y aspectos de sincronización. Especifica las líneas de mando que ejecutan cada operación en cada una de las clases de la Vista Lógica. El programa es dividido en varios programas o librerías de subsistemas, ya que la vista es realizada en varios niveles de abstracción, dividiendo el software en conjuntos independientes de tareas. (21)

1.6.3 Programación Extrema

La Programación Extrema (XP: Extreme Programming), es una metodología ligera de desarrollo de software basada en la simplicidad, la comunicación, y la reutilización de código desarrollado. La misma se enfoca directamente al objetivo, usando las relaciones interpersonales y la rapidez de reacción. Basa su desarrollo en la realización de pruebas unitarias, partiendo en sí de las pruebas hechas a procesos de mayor importancia, de la refactorización y la programación en parejas. El ciclo de vida de XP está compuesto por seis faces: **explotación**, **planificación de la entrega**, **iteraciones**, **producción, mantenimiento** y **muerte del proyecto**. (22)

La Programación Extrema propone (20):

- Iniciar desde la base o lo pequeño e ir agregando funcionalidades con la retroalimentación continua.
- Integrar al cliente final como parte del equipo de trabajo.
- Prestar especial atención al control de los cambios.
- No introducir las funcionalidades hasta que sean necesarias.

Esta metodología presenta inconvenientes como la dificultad para predecir los costos y el tiempo de desarrollo. Igualmente, la constante refactorización es una desventaja en sí, ya que los diagramas UML tienden a estar desactualizados. (20)

1.6.4 Selección de la metodología

Teniendo en cuenta que se quiere desarrollar un componente para lograr la reutilización de contenido educativo, es óptimo utilizar las metodologías tradicionales, ya que las mismas se centran en llevar una documentación exhaustiva de todo el proyecto, focalizándose en torno a la documentación, la planificación y los procesos. Seleccionando particularmente RUP, ya que es muy funcional en proyectos de innovación y sigue pasos intuitivos, necesarios a la hora de desarrollar el software, haciendo un seguimiento detallado en cada una de estas fases. Por otro lado existe gran experiencia en el uso de dicha metodología de desarrollo por parte de los equipos de desarrollo del centro FORTES.

1.7 Lenguaje Unificado de Modelado

El Lenguaje Unificado de Modelado (UML: Unified Modeling Language), es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimientos sobre los sistemas que se deben construir. Es usado para entender, diseñar, hojear, configurar, mantener y controlar la información sobre los sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. Incluye conceptos semánticos, notación, y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Es utilizado en herramientas interactivas de modelado visual que tengan generadores de código, así como generadores de informes. Sus especificaciones no definen un proceso estándar, pero es muy útil en procesos de desarrollo iterativo, principalmente en desarrollo orientados a objetos. (23)

UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema, modelado como una colección de objetos discretos que interactúan para realizar un trabajo que finalmente beneficia a un usuario externo. Contiene construcciones organizativas que agrupan los modelos en paquetes, lo que permite dividir los sistemas en piezas de trabajo, para entender y controlar sus dependencias,

y para gestionar las distintas versiones del modelo en un entorno de desarrollo complejo. No es un lenguaje de programación, sino que sus herramientas generan código fuente para una gran variedad de lenguajes de programación, y permiten construir modelos mediante la ingeniería inversa, a partir de programas existentes.(23) Para la construcción del componente de interoperabilidad se usará UML 2.

1.8 Entorno Integrado de Desarrollo

Un Entorno Integrado de Desarrollo (IDE: Integrated Development Environment), es un sistema que integra sólidamente la edición orientada al lenguaje, la compilación o interpretación, la depuración, las medidas de rendimiento, la incorporación de las fuentes a un sistema de control de fuentes, etc., con el fin de facilitar el trabajo del desarrollador de software. (24)

1.8.1 Netbeans

El Entorno Integrado de Desarrollo NetBeans, es considerado la manera más rápida y fácil para desarrollar aplicaciones Java de escritorio, móviles y aplicaciones web, así como aplicaciones con HTML, JavaScript y CSS. Proporciona un gran conjunto de herramientas para PHP y C / C++. Es gratuito y de código abierto y tiene una gran comunidad de usuarios y desarrolladores de todo el mundo. Es el mejor soporte para las últimas versiones de Java. Es el primer IDE libre que proporciona soporte para JDK 8, JDK 7, Java EE 7 incluyendo sus mejoras de HTML5 y Java FX 2. (25)

Brinda una fácil y eficiente gestión de proyecto que permite mantener una visión clara de los proyectos grandes, con miles de carpetas y archivos, y millones de líneas de código. NetBeans IDE 7.4 amplía el soporte avanzado de desarrollo para HTML5, que introdujo la versión 7.3 para las aplicaciones Java EE y PHP, al tiempo que ofrece un nuevo soporte para el desarrollo web móvil en las plataformas Android y iOS. (25) Decidiéndose usar la versión 7.4, ya que presenta una gran integración con PHP, y cuenta con módulos para el trabajo con Symfony y Drupal, necesarios para el desarrollo del componente.

1.9 Herramientas CASE

Las Herramientas CASE (Computer Aided Software Engineering), tienen la finalidad de automatizar o apoyar una o más fases del ciclo de vida del desarrollo de sistemas; siendo el primer paso para aplicaciones de bases de datos cuando se hace la planificación, diseño y modelado de las mismas, permitiendo llevar a cabo el resto de tareas del modo más eficiente y efectivo posible. (26)

1.9.1 Visual Paradigm

"Visual Paradigm for UML es una herramienta CASE que soporta el modelado mediante UML y proporciona asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software". (27) Tiene ventajas muy significativas como el dibujo, facilitadas mediante herramientas de modelado UML. Permite la estandarización de la documentación, la corrección sintáctica, la coherencia, la integración con otras aplicaciones, el trabajo multiusuario, la reutilización, la generación de código y la generación de informes.

La versión que se usará en el desarrollo del componente será, Visual Paradigm for UML 8.0 (VP-UML 8.0). La misma presenta mejoras respecto a las versiones anteriores, ya que escribe flujo de eventos con sentencias de control, exporta MS Excel en formato normalizado, genera informes de los proyectos dependientes, genera sentencias ALTER para las columnas que aceptan valores NULL, genera OrderByIndex para el código ORM, personaliza getter y setter ORM, entre otras mejoras. Presenta nuevas funcionalidades como confirmar cambios parciales del servidor, escribir reglas de negocio, dibujar el diagrama de datos, proteger los diagramas por contraseña, crear conexiones con puntos de inflexión, examinar los diagramas de proyectos dependientes, entre otras. (27)

1.10 Lenguajes de programación

"Los lenguajes de programación son lenguajes creados por el ser humano para poder comunicarse con las computadoras. Estos son un conjunto de símbolos y palabras que permiten al usuario de una computadora darle instrucciones y órdenes para que esta las pueda realizar". (28)

1.10.1 Java Script

"JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios". (29) Es usado masivamente por la mayoría de los navegadores, y los más modernos disponibles actualmente incluyen soporte de JavaScript hasta la versión correspondiente a la tercera edición del estándar ECMA-262. Su popularidad disminuyó ante la aparición de Flash, ya que este permitía realizar algunas acciones imposibles de llevar a cabo mediante JavaScript, pero esto fue revertido ante el surgimiento de las aplicaciones AJAX programadas con JavaScript y más adelante con el surgimiento de HTML 5.

1.10.2 PHP

"PHP (Hypertext Preprocessor) es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor". (30) Es un lenguaje de programación diseñado originalmente para el desarrollo web de contenido dinámico. Es uno de los primeros lenguajes que permitió la incorporación directa del HTML, interpretado el código en un servidor web con un módulo de procesamiento PHP para generar la página web resultante. Su evolución es tal, que actualmente posee una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Es usado en la mayoría de los navegadores web, sistemas operativos y plataformas.

El 13 de julio de 2004, fue lanzado PHP 5, utilizando el motor Zend Engine 2, del cual incluye todas sus ventajas, entre las que se encuentran:

- Mejor soporte para la programación orientada a objetos.
- Mejoras de rendimiento.
- Mejor soporte para MySQL con extensión completamente reescrita.
- Mejor soporte a XML (XPath, DOM, etc.).
- Soporte nativo para SQLite.
- Soporte integrado para SOAP.
- Iteradores de datos.
- Manejo de excepciones.

- Mejoras con la implementación de Oracle. (31)

1.10.3 XML

XML (Extensible Markup Language) es un metalenguaje[9] extensible de etiquetas que fue desarrollado por el Word Wide Web. Es una adaptación del SGML[10], un lenguaje que permite la organización y el etiquetado de documentos. Es decir, XML no es un lenguaje en sí mismo, sino un sistema que permite definir lenguajes de acuerdo a las necesidades. El XHTML[11], el MathML[12] y el SVG[13] son algunos de los lenguajes que el XML tiene la capacidad de definir. (32)

Algunos de los campos de aplicación del XML son, las bases de datos, los documentos de texto, las hojas de cálculo, las páginas web, entre otros. Entre sus ventajas se puede destacar que es extensible, su analizador es estándar, y facilita el análisis y el procesamiento de los documentos XML desarrollados por terceros. Permite relacionar aplicaciones de diferentes lenguajes y plataformas y facilitar la organización de los recursos y la configuración. (13) Es utilizado por los servicios web para realizar una serie de operaciones, a través de métodos concretos que aprovechan el metalenguaje para sus comunicaciones; gracias a lo cual cualquier plataforma puede hacer uso de sus ventajas. (33)

Está compuesto por un conjunto de componentes y lenguajes que juegan roles fundamentales como son (34):

- **XSLT (hojas de estilo):** Lenguaje que permite especificar como un documento XML que pertenece a un esquema concreto es transformado a otro formato, como HTML, texto plano, etc.
- **XPATH y XQUERY:** Lenguaje que permite consultar un documento XML y obtener partes concretas del mismo.

9 Metalenguaje: Un lenguaje que se utiliza para decir algo acerca de otro lenguaje.
10 SGML: Standard Generalized Markup Language.
11 XHTML: eXtensible HyperText Markup Language.
12 MathML: Mathematical Markup Language.
13 SVG: Scalable Vector Graphics.

- **DOCTYPE y XML SCHEMA:** Componentes que permiten especificar el esquema que debe regir un documento XML en concreto, siendo un componente fundamental de la tecnología ya que permite determinar cuándo un documento XML es válido.

1.11 Sistema de Control de Versiones

Un Sistema de Control de Versiones (VCS: Version Control System), registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, permitiendo recuperar versiones específicas del código fuente o cualquier tipo archivo que se encuentre en un ordenador. De igual manera, un VCS revierte archivos a un estado anterior, revierte un proyecto entero a un estado anterior, compara cambios a lo largo del tiempo y registra las acciones realizadas por los usuarios en el sistema; proceso realizado a muy bajo coste, basados en procesos locales, centralizados y distribuidos. (35)

Sistemas de control de versiones locales (LVCS: Local Version Control Systems). Es un método usado para copiar archivos a otros directorios (indicando la fecha, la hora, etc.), de manera muy simple pero propensa a errores como guardar el archivo equivocado, sobrescribir archivos, entre otros. Por lo que se usan VCS locales con una base de datos que registra los cambios realizados sobre los archivos. Lo cual es implementado en herramientas como RCS, aún usada en muchos ordenadores e incluidas en el sistema operativo Mac OS X, para instalar las herramientas de desarrollo. Guardando básicamente un conjunto de parches (diferencias entre archivos) de una versión a otra en un formato especial en disco. (35)

Sistemas de control de versiones centralizados (CVCS: Centralized Version Control Systems). Es un método usado para la colaboración de los desarrolladores entre sistemas. Constando con un único servidor que contiene todos los archivos versionados, y varios clientes que descargan los archivos desde ese lugar central. Aplicado por sistemas como **CVS**, **Subversion**, y **Perforce**, siendo durante muchos años, el estándar adoptado para realizar control de versiones. Sin embargo, esta configuración también tiene serias desventajas, ya que posee un punto único de fallo, el servidor centralizado; el cual puede salir de servicio (haciendo imposible colaborar o guardar cambios versionados) o ser corrompido el disco duro en el que se encuentra la base de datos central, perdiendo toda la historia del proyecto, si no se han llevado

copias de seguridad adecuadamente. (35)

Sistemas de control de versiones distribuidos (DVCS: Distributed Version Control Systems). *"En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no sólo descargan la última instantánea de los archivos: replican completamente el repositorio. Así, si un servidor muere, y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios de los clientes puede copiarse en el servidor para restaurarlo. Cada vez que se descarga una instantánea, en realidad se hace una copia de seguridad completa de todos los datos"* (véase Figura 7). (35) Permitiendo el trabajo con varios repositorios, colaborando distintos grupos simultáneamente dentro de un mismo proyecto, estableciendo un modelo jerárquico, mediante varios flujos de trabajo.

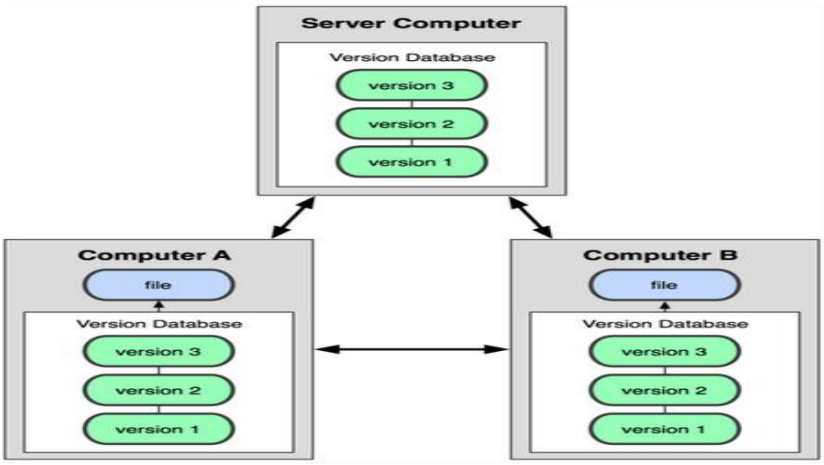


Figura 7. Diagrama de control de versiones distribuido

1.11.1 Git

El DVCS Git surgió en el 2005, impulsado por la comunidad de desarrollo de Linux, con el fin de desarrollar su propia herramienta, basada en lecciones aprendidas durante el uso de BitKeeper[14]. Generando así una herramienta extremadamente rápida, eficiente para proyectos grandes, y con un sistema de ramificación (branching) para desarrollos no lineales, destacada por:

14 BitKeeper. www.bitkeeper.com/

- Almacenar sus datos como mini sistemas de archivos, al confirmar una modificación o cambio de estado. Diferenciándose de otros VCS (CVS, Subversion, Perforce, Bazaar, etc.), al realizar enlaces a archivos almacenados, si estos no han sido modificados, en lugar de almacenar la información como una lista de cambios en los archivos.
- Solo necesitan archivos y recursos locales para operar. Permitiendo navegar por la historia del proyecto, leer en la base de datos local y revisar cambios entre versiones, sin necesidad de acceder a un servidor remoto; lo cual es imposible usando Perforce, Subversion o CVS, que permiten editar archivos, pero no confirmar los cambios en la base de datos.
- Muestra un alto nivel de integridad, mediante la suma de comprobación (checksum) antes de almacenar contenidos, preservando la información durante su transmisión e impidiendo la corrupción de archivos.
- Es usada por importantes proyectos como Google, Facebook, Microsoft, Twitter, PostgreSQL, Android, QT, GNOME, Eclipse, GitHub, entre otros. (35)

Estado de los archivos en Git:

- **Confirmado (committed):** Los datos están almacenados de manera segura en la base de datos local.
- **Modificado (modified):** El archivo ha sido modificado pero no confirmado en la base de datos.
- **Preparado (staged):** El archivo modificado ha sido marcado para ser enviado en la próxima confirmación. (35)

Secciones principales de un proyecto de Git:

- **El directorio de Git (Git directory):** Es donde se almacenan los metadatos y la base de datos de objetos para un proyecto. Es la parte más importante de Git, y es lo que se copia al clonar un repositorio desde otro ordenador.
- **El directorio de trabajo (working directory):** Es una copia de una versión del proyecto, colocada en un disco para ser usado o modificado.
- **El área de preparación (staging area):** Es un archivo, generalmente contenido en el directorio, que almacena información acerca de la próxima confirmación, denominado índice. (35)

1.12 Sistema de Gestión de Contenidos

Un Sistema de Gestión de Contenidos (CMS: Content Management System), permite la gestión de contenidos en un medio digital, más específicamente, los contenidos de una página web. De manera que es una herramienta que permite crear, clasificar y publicar cualquier tipo de información en una página web. (36)

1.12.1 Drupal

Drupal es un CMS de código abierto con licencia GNU/GPL. Está escrito en PHP y es desarrollado y mantenido por una activa comunidad de usuarios. Destaca por la calidad de su código y de las páginas generadas, el respeto de los estándares de la web, y un énfasis especial en la usabilidad y consistencia de todo el sistema. Brindando además, una gestión de usuarios basado en la autenticación y permisos a roles; y a la hora de gestionar contenidos, brinda utilidades como el control de versiones, los enlaces permanentes, el manejo de contenido como objetos o nodos, las plantillas y la sindicación de contenidos. En el caso de las plataformas, permite que estás tengan independencia de la base de datos, que sean multiplataforma y permite múltiples idiomas y localización. Permite la administración vía web, hacer análisis, seguimiento, obtener estadísticas, y generar registros o informes. (37)

Drupal, posee un diseño modular que puede ser fácilmente ampliado y adaptado para proyectos de diferentes envergaduras, lo cual es especialmente idóneo para construir y gestionar comunidades en Internet. Estos módulos, están compuestos por colecciones de funciones que se enlazan dentro de Drupal y ofrecen funcionalidades no contenidas en los módulos del núcleo, los cuales pueden ser descargados y son llamados módulos contribuidos, o pueden ser módulos específicos, que son desarrollados para cubrir funcionalidades no cubiertas por los antes expuestos. (37)

Su sitio principal de desarrollo y coordinación es drupal.org, el cual posee una amplia selección de módulos contribuidos, creados por varios miles de usuarios de todo el mundo y guardados en repositorios Git, donde continúan desarrollándose. Hasta la fecha se han liberado las versiones de Drupal 1, 2, 3, 4, 5, 6 y 7 como versiones estables y Drupal 8 como versión de prueba. Para el desarrollo del componente se usará la 7, ya que es la que tiene actualmente mantenimiento activo por la comunidad de Drupal, es la versión estable recomendada por la comunidad de desarrollo y posee

una arquitectura superior a las anteriores. (37)
Dentro del conjunto de módulos del núcleo y contribuidos de Drupal, se enmarcan una serie de módulos que comúnmente son utilizados en el campo de acción de la presente investigación, los cuales serán expuestos a continuación por su importancia en el desarrollos del componente a desarrollar.

Views:

El módulo contribuido Views[15] (también llamado Vistas), es uno de los componentes más útiles y potentes para construir páginas dinámicas. Mediante el mismo, se pueden generar una serie de funcionalidades como (38):

- Mostrar todos los nodos de un tipo de contenido determinado, y añadir filtros de búsqueda para localizar un contenido específico.
- Mostrar en un bloque un listado de los últimos usuarios registrados en el sitio.
- Mostrar una tabla con datos ordenados a partir de un conjunto de nodos de un tipo de contenido determinado.
- Mostar un bloque con los contenidos más visitados.
- Mostrar un grid o rejilla con una colección de imágenes.

Este módulo permite agregar nuevos campos, filtrarlos según el grupo al que pertenecen, configurarlos, reordenarlos y sobrescribirlos. Puede aplicar criterios de ordenación usando cualquier campo de entre los disponibles, independientemente de que se haya añadido a la vista o no. Define condiciones para que se muestren elementos determinados mediante filtros, que pueden estar ocultos o expuestos al usuario. Utiliza filtros contextuales para pasar argumentos a las vistas a través de la URL. Establece relaciones para ampliar las consultas realizadas, permitiendo el acceso a los campos de los elementos seleccionados. Brinda una serie de opciones avanzadas, como el uso de AJAX, la agregación, la configuración de consultas, el manejo del idioma en un campo determinado, entre otras. Brindando además el manejo de permisos y de módulos adicionales como Views Summarize, Views Bulk

15 Views: https://drupal.org/project/views

Operations y Views Accordion. Incorporando importantes cambios en la versión 3.x del módulo Views de Drupal 7 con respecto a Drupal 6 y la versión 2.x, reflejadas principalmente en la interfaz de las vistas, con respecto a los campos disponibles y sus opciones de configuración. (38)

Book:

"Un book (libro) en Drupal es un conjunto de páginas vinculadas y organizadas en una estructura jerárquica, como podrían ser capítulos, secciones y subsecciones". (39) Son usados para representar contenidos que requieran de alguna organización jerárquica; mediante la creación de nuevos tipos de contenidos específicos organizados en estructuras jerárquicas, formando así un libro. Puede ser modificado mediante otros módulos como Book Delete, Book Block y Book Access, variando sus funcionalidades o añadiendo otras, ajustándolo según las necesidades del sistema.

Features:

"El módulo Features[16] *permite agrupar funcionalidades y reutilizarlas. Básicamente las features son grupos de funcionalidades y objetos que pueden instalarse, activarse y desactivarse en un único paso".* (40) Permite crear una feature que contenga un tipo de contenido (con sus campos adicionales) y una vista relacionada con el tipo de contenido, creados automáticamente con todos sus elementos al instalar o desactivar el módulo en un sitio. Utilizadas para exportar/importar estructuras de objetos, pero no de los contenidos, por lo que no pueden ser incluidas en los nodos, usuarios, etc. Especialmente útil para desarrollar portales diferentes, pero que comparten grupos de funcionalidades, como es el caso de un blog, un foro, una tienda virtual, entre otros. Añaden grupos funcionales de manera rápida y sencilla, sin las cuales habría que instalar y configurar todos los módulos desde cero.

Sólo puede añadirse a una feature los elementos (vistas, reglar, tipos de contenidos, taxonomías, etc.) de los módulos compatibles. De igual manera, estas podrán ser

16 Features: https://drupal.org/project/features

añadidas a módulos como Drupal Reset, que elimina todas las tablas y archivos del sitio, devolviéndolo al proceso de instalación, lo que es especialmente útil durante el desarrollo de features y perfiles de instalación. Incluyendo además comandos de Drush para realizar operaciones similares a las realizadas a través de la interfaz de usuario. Siendo además incluidas en módulos como Featured news feature, Events calendar feature y Enterprise blog. (40)

Biblio:

El módulo contribuido Biblio[17] provee a una instalación de Drupal la capacidad de gestionar información bibliográfica. Permite importar como contenidos de Drupal formatos bibliográficos ampliamente difundidos, como: PubMed, BibTex, MARC, EndNote tagged and XML, a la vez que permite exportar contenidos de Drupal a los formatos: BibTex y EndNote tagged and XML. Está relacionado con un conjunto de módulos que extienden su funcionalidad, como Views OAI-PMH[18] que brinda la capacidad de exportar catálogos siguiendo el protocolo OAI-PMH y el formato de metadatos Dublin-Core. Teniendo como ventaja, que es una extensión del módulo Views, lo que permite una alta personalización del mapeo entre los campos de los contenidos exportados y el modelo de datos de Dublin-Core. Este módulo es un componente fundamental para la elaboración de bibliotecas con Drupal.

1.13 Sistema Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (DBMS: Database Management System), es un software que permite procesar, describir, administrar y recuperar los datos almacenados en una base de datos. Proporciona un conjunto coordinado de programas, procedimientos y lenguajes que permiten realizar tareas con los datos, garantizando además la seguridad e integridad, en lo cual reside su éxito. (41)

17 Biblio: https://drupal.org/project/biblio

18 Views OAI-PMH: https://drupal.org/project/views_oai_pmh

1.13.1 PostgreSQL

PostgreSQL es un DBMS objeto-relacional distribuido bajo licencia BSD[19]. Utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos, garantizando ante el fallo de un proceso no afectar al resto, por lo que el sistema continuará funcionando y podrá garantizar la estabilidad del sistema. Es considerada una de las bases de datos más potentes y robustas del mercado, principalmente por las características técnicas de la serie de producción (9.3), la cual se usará en el desarrollo del componente. Funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema. Entre sus características principales se encuentran:

- Es una base de datos 100% ACID (Atomicity, Consistency, Isolation, Durability).
- Soporta distintos tipos de datos y permite la creación de tipos propios.
- Brinda soporte a la codificación Unicode.
- Utiliza juegos de caracteres internacionales.
- Presenta Multi-Version Concurrency Control (MVCC).
- Posee múltiples métodos de autentificación.
- Posee acceso encriptado vía SSL.
- Se encuentra disponible para Linux y UNIX en todas sus versiones y para Windows de 32/64bit. (42)

1.14 Servidor Web

“Un servidor web es un programa que se ejecuta continuamente en un computador, manteniéndose a la espera de peticiones de ejecución que le hará un cliente o un usuario de Internet. El servidor web se encarga de contestar a estas peticiones de forma adecuada, entregando como resultado una página web o información de todo tipo de acuerdo a los comandos solicitados”. (43)

19 BSD (Berkeley Software Distribution): Es una licencia de software libre permisiva, ya que permite el uso de código fuente en software no libre.

1.14.1 Apache

El servidor web Apache, es un servidor HTTP de código abierto, para plataformas Unix, Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Es desarrollado por el proyecto HTTP Server (httpd) de la Apache Software Foundation[20]. El cual se usará en su versión 2 para el desarrollo del componente de interoperabilidad. Entre sus ventajas, se destaca que es modular, de código abierto, multi-plataforma, extensible y debido a la fácil adquisición de ayuda y soporte, tiene una gran popularidad. Es usado principalmente para enviar páginas web, ya sean estáticas o dinámicas. Es el componente de servidor web en la plataforma de aplicaciones LAMP, junto a MySQL y los lenguajes de programación PHP, Perl y Python. Es redistribuido como parte de varios paquetes propietarios de software, incluyendo la base de datos Oracle y el IBM WebSphere application server. Es integrado por Mac OS X como parte de su propio servidor web y como soporte de su servidor de aplicaciones WebObjects. Es incluido con Novell NetWare 6.5, donde es el servidor web por defecto, y en muchas distribuciones Linux. (44)

1.15 Conclusiones del capítulo

En la realización de este capítulo, se ha plasmado el marco teórico metodológico. Definiendo para ello conceptos de vital importancia como objeto de aprendizaje e interoperabilidad, analizando las principales herramientas que componen un entorno e-learning y determinando el uso de los estándares LOM, IMS-CP y SCORM 2004. Seleccionando como herramientas y tecnologías para el desarrollo del componente de interoperabilidad, la Metodología de Desarrollo de Software RUP, el Lenguaje de Modelado UML 2, el Entorno Integrado de Desarrollo Netbeans 7.4, la Herramienta Case Visual Paradigm for UML 8.0, el Sistema de Control de Versiones Git 1.9, el Sistema Gestor de Contenidos Drupal 7, el Sistema Gestor de Base de Datos Postgresql 9.3 y el Servidor Web Apache 2.

20 Apache Software Foundation: www.apache.org/foundation/

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

En el presente capítulo se describen las características del sistema a desarrollar, para lo cual se realizan artefactos como el modelo de dominio y se plantea la propuesta del sistema. Igualmente se definen los requisitos funcionales y no funcionales, lo que permite tener una visión más amplia de las necesidades que el usuario o cliente final desean y necesitan. Generándose además los diagramas de casos de uso del sistema y la descripción de los casos de uso.

2.2 Modelo de dominio

El modelo de dominio (Figura 8) permite describir, mediante la representación en un marco conceptual, la relación entre los principales conceptos que conforman el dominio del componente a desarrollar (21), expresados a continuación:

- **Portal:** Sistema de Gestión de Contenido Educativo implementado en Drupal 7.
- **Usuario:** Persona que realiza acciones sobre el sistema.
- **Libro:** Conjunto de páginas vinculadas y organizadas para representar contenidos que requieran de alguna organización jerárquica; mediante la creación de nuevos tipos de contenidos específicos.
- **Contenidos:** Conjunto predefinido de tipos de datos (campos) que se relacionan entre sí.
- **Taxonomía:** Conjunto de términos utilizados para clasificar contenidos.
- **Comentario:** Usado por un usuario para comentar un contenido.
- **SCORM:** Conjunto de especificaciones para desarrollar, empaquetar y distribuir material educativo.
- **Asset:** Representación electrónica de un medio de comunicación.
- **Sco:** Colección de Asset en un recurso de aprendizaje.
- **Metadatos:** Descripción de contenidos.

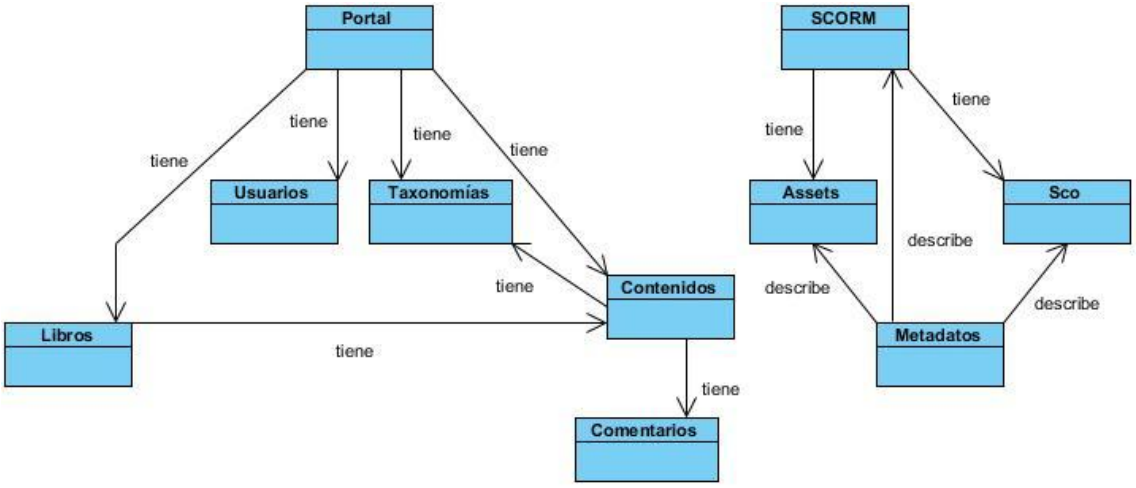


Figura 8. Modelo de dominio

2.3 Propuesta del sistema

Dada la necesidad de reutilizar contenido educativo entre herramientas de un entorno e-learning y un Sistema de Gestión de Contenido Educativo (LCMS) implementado en Drupal 7; se desarrollará un componente para importar/exportar contenido educativo estandarizado, mediante la transformación entre un modelo de datos SCORM 2004 y un modelo de datos de Drupal 7. Para lo cual, se realizarán los módulos SCORM Import, SCORM Export y SCORM Features.

El módulo **SCORM Import**, permitirá reutilizar contenidos educativos estandarizados desarrollados por herramientas de un entorno e-learning en un LCMS implementado en Drupal 7, obteniendo un libro de Drupal a partir de un SCORM 2004. Este módulo generará los vocabularios correspondientes al modelo taxonómico de Drupal, partiendo de las rutas taxonómicas encontradas en los metadatos del manifiesto del PIF. Igualmente generará contenidos de Drupal por cada SCO o Asset del SCORM 2004, a partir de los metadatos y recursos asociados a él y agregando los comentarios obtenidos de los metadatos anotaciones. Generando por cada ítem de una organización presente en el SCORM 2004, una entrada en el índice de un libro de Drupal, manteniendo los niveles de agregación presentes en el manifiesto. Utilizando para ello las reglas de mapeo (Figura 9) y el tipo de contenido de Drupal (SCORM content) implementado en el módulo SCORM Features.

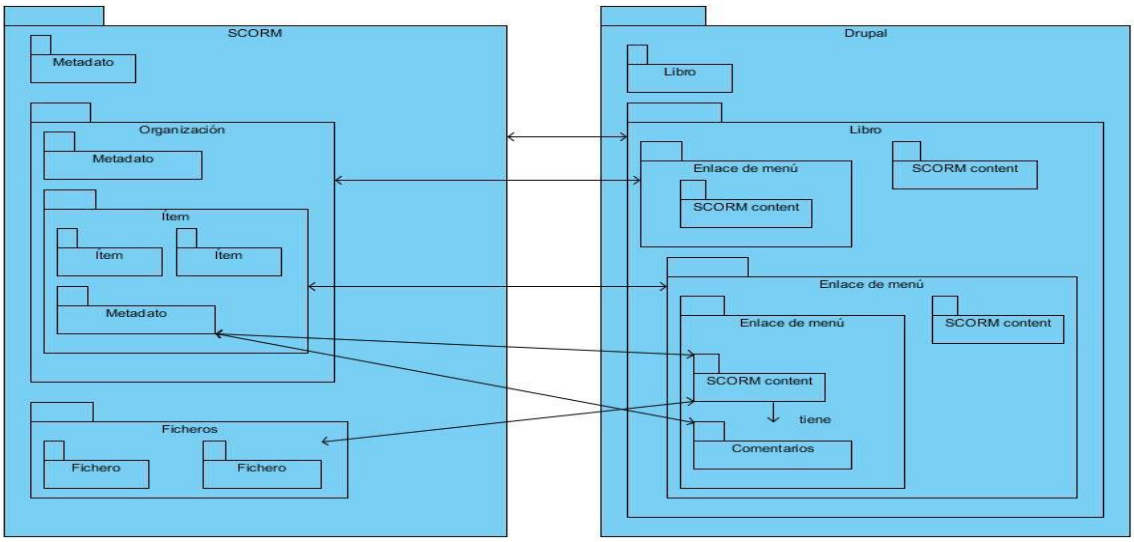


Figura 9. Mapeo del módulo Importar SCORM

El módulo **SCORM Export**, permitirá reutilizar en herramientas de un entorno e-learning los contenidos educativos desarrollados por un LCMS implementado en Drupal 7. Obteniendo por cada libro de Drupal un compactado de SCORM 2004 (en formato zip); generando una organización de SCORM a partir del índice extraído de un libro de Drupal. Igualmente generará un Asset de SCORM 2004 por cada contenido asociado a un libro. Y generará los metadatos correspondientes a partir de las propiedades de los contenidos, sus comentarios y las taxonomías asociadas a un libro; asociando como recurso Asset los campos archivos que posean sus contenidos. Siguiendo una serie de reglas para transformar el modelo de datos específico de Drupal 7 en un modelo de datos SCORM 2004 (Figura 10).

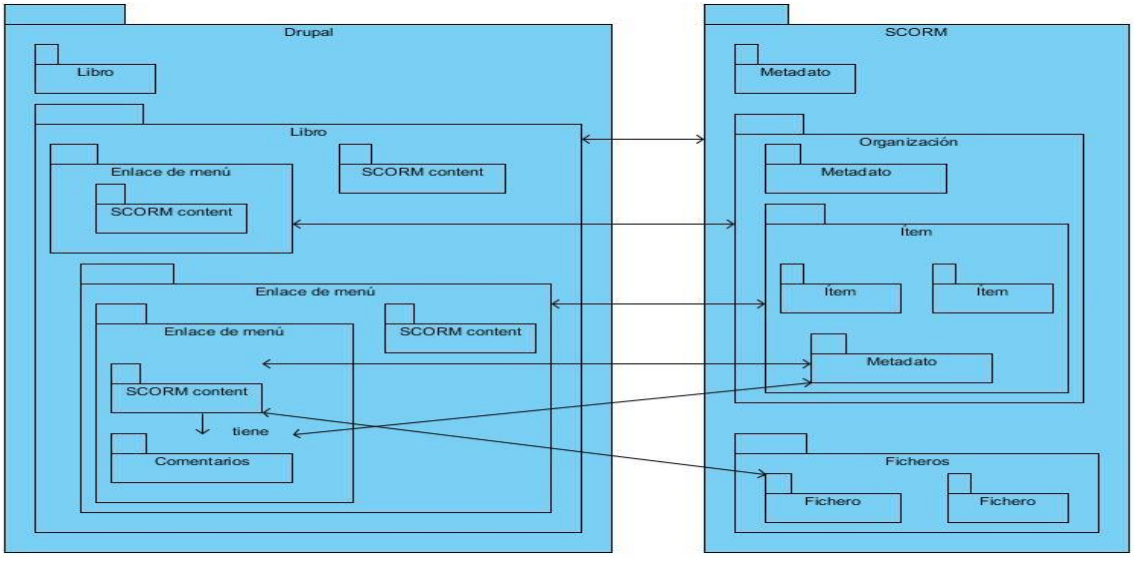


Figura 10. Mapeo del módulo Exportar libro de Drupal

El módulo **SCORM Features**, será generado con el propósito de crear todas las dependencias necesarias (Tipos de contenido, Taxonomías y Roles) por los restantes módulos en el momento de instalación de los mismos.

2.4 Requerimientos

Los requerimientos son reglas, condiciones o capacidades que necesita un usuario para resolver un problema o lograr un objetivo, y estos pueden ser requerimientos funcionales o no funcionales. Los cuales deben especificarse por escrito, ser posibles de probar o verificar, ser escritos como una característica del sistema a entregar y ser lo más abstracto y conciso posible. (21)

2.4.1 Requerimientos funcionales

Los requerimientos funcionales (RF) son las condiciones o capacidades que el sistema debe cumplir lo suficientemente bien para satisfacer las necesidades de los clientes y usuarios finales. Estos tienen su origen en la realización de los casos de uso del negocio, que brindan las actividades que serán objeto de automatización, y aunque no son exactamente los RF, son el punto de partida para identificar qué debe hacer el sistema. (21) A continuación se listan los RF del componente a desarrollar.

RF 1: Importar un SCORM 2004 a Drupal.

1.1: Validar el PIF[21] del SCORM.

1.2: Generar taxonomías de Drupal a partir de las rutas taxonómicas de los metadatos del PIF.

1.3: Generar contenidos de Drupal a partir de los metadatos del PIF.

1.4: Generar comentarios en Drupal a partir de los metadatos del PIF.

1.5: Generar libro en Drupal a partir de cada una de las organizaciones presentes en el manifiesto del PIF.

21 PIF: Package Interchange File.

RF 2: Exportar un libro de Drupal a un SCORM 2004.

1.1: Exportar el índice de un libro como una organización de SCORM 2004.

1.2: Exportar los contenidos asociados a un libro

2.4.2 Requerimientos no funcionales

Los requerimientos no funcionales (RNF) son las cualidades o propiedades que el producto debe tener, y que lo hacen atractivo, usable, rápido y confiable. Están estrechamente vinculados al éxito del producto y generalmente responden a RF. Estos forman una parte significativa de la especificación, ya que permite a clientes y usuarios valorar cuan usable, seguro, conveniente y agradable es el producto, marcando la diferencia en la aceptación del mismo. (21) Para el desarrollo del componente se especificaron los siguientes RNF:

RNF: Software

Solo se usarán librerías y componentes desarrollados por terceros, cuando estén liberados bajo licencia de software libre.

RNF: Seguridad

- **Integridad:** El sistema debe mantener la integridad del modelo de datos en caso de que el proceso de exportación/importación no se concluya por fallas internas o externas.
- **Disponibilidad:** El componente debe estar disponible para ser accedido por los usuarios autorizados en el momento requerido.

2.5 Modelo de casos de uso

El modelo de casos de uso está compuesto por los actores, los casos de uso y sus relaciones. El mismo permite que los desarrolladores y los clientes tengan una misma visión sobre los requisitos de la aplicación. (21)

2.5.1 Actores del sistema

Un actor del sistema es una persona, un dispositivo de hardware u otro sistema externo a este, que interactúa con este y se beneficia de esa interacción. (21) A continuación se muestran los actores del componente a desarrollar en la Tabla 1.

Actores del sistema	Descripciones
Usuario	Persona que interactúa con la aplicación con el objetivo de ver todos los contenidos educativos expuestos y exportarlos a un formato SCORM 2004.
Autor de contenidos	Persona que interactúa con la aplicación y posee los privilegios para importar contenidos desde un formato SCORM 2004.
CRON	El CRON del sistema operativo. Componente del sistema operativo que invoca periódicamente tareas que le son programadas.

Tabla 1. Actores del sistema

2.5.2 Casos de uso del sistema

Los casos de uso (CU) describen el comportamiento del sistema desde el punto de vista del usuario y permiten establecer acuerdos entre clientes y desarrolladores sobre las condiciones del sistema, brindando una secuencia de actividades a automatizar que puede implicar pasos dentro del mismo. (21)

2.5.3 Diagrama de casos de uso del sistema

Un diagrama de casos de uso del sistema, representa gráficamente los procesos y su interacción con los actores (Figura 11). Es un fragmento de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. (21)

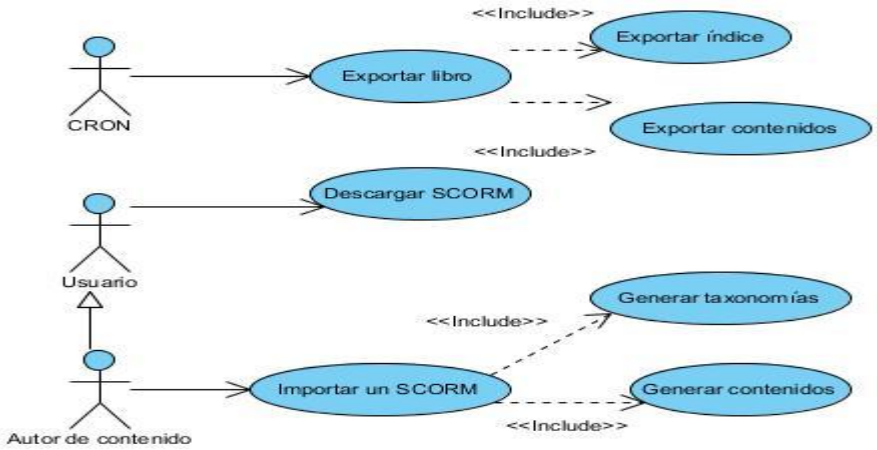


Figura 11. Diagrama de casos de uso del sistema

2.5.4 Descripciones de casos de uso

La descripción textual de un CU permite ganar claridad en el comportamiento o interacción entre CU y actores. (21) A continuación se describen los CU arquitectónicamente significativos (Tabla 2 y 3), y las restantes descripciones se encuentran en el Anexo 1. Descripción de casos de uso.

Caso de uso Exportar libro	
Objetivos	Exportar un libro de Drupal como un paquete SCORM 2004.
Actores	CRON
Resumen	Se invoca la funcionalidad de exportar un libro de Drupal y el sistema genera un paquete SCORM 2004 compactado en formato zip.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	Deben existir libros de Drupal disponibles para ser exportados.
Postcondiciones	Se genera un máximo de cinco de paquetes SCORM 2004, del total de libros disponibles para ser exportados.
Flujo de eventos	
Flujo básico exportar libro.	
Actor	**Sistema**
1. Invoca la funcionalidad de exportar libro de Drupal a SCORM 2004.	2. Crea un listado de los libros marcados para ser exportados, cuya última versión no se encuentra convertida a SCORM

		2004, ordenados ascendentemente por la fecha de modificación. 3. Selecciona un máximo de cinco libros del listado. 4. Para cada uno de los libros seleccionados: 4.1. Genera una organización de SCORM 2004. Ver el CU Exportar índice. 4.2. Genera un SCO de SCORM 2004 por cada contenido asociado al libro de Drupal. Ver el CU Exportar contenido. 5. Termina el caso de uso.
Relaciones	CU Incluidos	- CU Exportar índice - CU Exportar contenidos
	CU Extendidos	
Requisitos no funcionales	- Integridad - Disponibilidad	
Asuntos pendientes		

Tabla 2. Descripción del caso de uso Exportar libro

Caso de uso Importar un SCORM	
Objetivos	Generar un libro de Drupal a partir de un objeto de aprendizaje en formato SCORM 2004.
Actores	Autor de contenido.
Resumen	Se selecciona la opción de importar un OA y el sistema genera un libro de Drupal.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	
Postcondiciones	Generar un libro de Drupal.
Flujo de eventos	
Flujo básico importar un SCORM.	
Actor	**Sistema**
1. Selecciona la opción de importar un OA a un libro de Drupal.	2. Muestra el formulario para importar el OA.

3. Busca y selecciona el OA a importar. 4. Invoca la funcionalidad de importar el OA a Drupal.	5. Valida el PIF del SCORM 2004 que se desea importar. 6. Genera taxonomías. Ver CU Generar taxonomías. 7. Genera contenidos de Drupal. Ver CU Generar contenidos. 8. Genera un libro de Drupal a partir del SCORM 2004, creando por cada ítem y cada organización una entrada en el índice del libro. 9. Muestra un mensaje de información: "El libro ha sido incorporado correctamente" 9. Termina el CU.
Flujos alternos	
5.a Datos incorrectos (PIF del SCORM)	
Actor	**Sistema**
	1. Muestra un mensaje de información: "El OA no tiene formato SCORM 2004". 2. Regresa al paso 2 del flujo básico.

Relaciones	CU Incluidos	- CU Generar taxonomías - CU Generar contenidos
	CU Extendidos	
Requisitos no funcionales	- Integridad - Disponibilidad	
Asuntos pendientes		

Tabla 3. Descripción del caso de uso Importar un SCORM

2.6 Conclusiones del capítulo

En este capítulo han sido definidas las características que debe tener el sistema, partiendo de los requerimientos que debe poseer el mismo y de la propuesta de solución planteada. Modelando así elementos fundamentales de la solución mediante actores y casos de uso, representados en el diagrama de caso de uso del sistema y detallados en la descripción de cada uno de los casos de uso.

CAPÍTULO 3. ANÁLISIS Y DISEÑO DEL SISTEMA

3.1 Introducción

Este capítulo tiene como propósito transformar los requerimientos en un diseño de cómo va a ser implementado el sistema, mediante la evolución hacia una arquitectura del software robusta. Para lo cual se genera el modelo de análisis y el modelo de diseño, los cuales son definidos por los diagramas de clases del análisis, los diagramas de colaboración, los diagramas de clases del diseño y el modelo de datos.

3.2 Modelo de análisis

El modelo de análisis tiene como finalidad analizar los requisitos del sistema, para refinarlos y estructurarlos, de manera que se obtenga una comprensión y descripción para estructurar el sistema y la arquitectura del mismo. No tiene en cuenta el lenguaje de programación, ni la plataforma en la que se ejecutará la aplicación, ni los componentes prefabricados o reusables de otras aplicaciones, entre otras características que afectan al sistema, ya que el objetivo del análisis es comprender los requisitos del software y no centrarse en la implementación de la solución. (21)

3.2.1 Diagramas de clases del análisis

Los diagramas de clases del análisis, son artefactos construidos en el modelo de análisis, partiendo de identificar las clases que describen la realización de los casos de uso, sus atributos y las relaciones entre ellas. (21) En este acápite se representan los diagramas de casos de uso arquitectónicamente significativos (Figura 12 y 13), y el resto de los diagramas se encuentran en el Anexo 2. Diagramas de clases del análisis.

Caso de uso Importar un SCORM

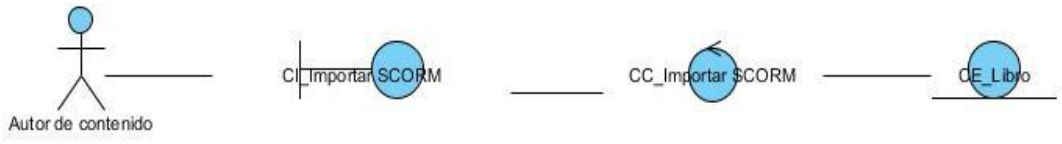


Figura 12. Diagrama de clase del análisis del caso de uso Importar un SCORM

Figura 13. Diagrama de clase del análisis del caso de uso Exportar libro

3.2.2 Diagramas de colaboración

Un diagrama de colaboración muestra la interacción de forma geométrica entre objetos que efectúan operaciones. Incluye a su vez, mensajes en forma de flechas, asociadas a líneas de relación entre roles. (21) En este acápite se representaran los diagramas de colaboración para los caos de uso arquitectónicamente significativos (Figura 14 y 15), y el resto de los mismos se encuentran en el Anexo 3. Diagramas de colaboración.

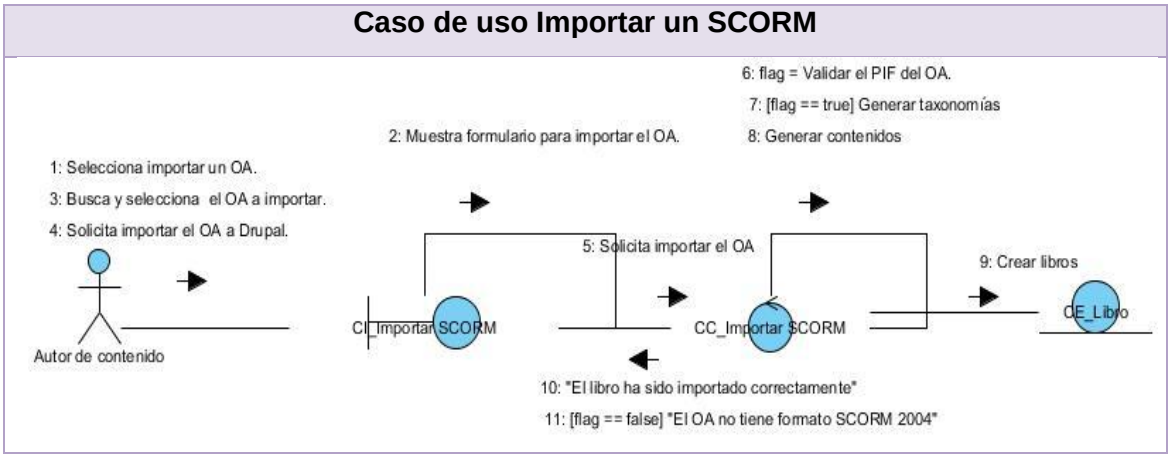


Figura 14. Diagrama de colaboración del caso de uso Importar un SCORM

Figura 15. Diagrama de colaboración del caso de uso Exportar libro

3.3 Modelo de diseño

El modelo de diseño, es un modelo de objetos que describe la realización física de los casos de uso, centrándose en como los RF y RNF, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Sirve de abstracción a la implementación del sistema y es utilizado como una entrada fundamental a las actividades de implementación. (21)

3.3.1 Patrones de diseño para asignar responsabilidades (GRASP)

Los patrones de diseño son un grupo de principios generales basados en la experiencia que guían la creación del software, con el fin de asignar responsabilidades. Están constituidos por la descripción de las clases, de manera que permita adaptar un problema de diseño general a un contexto particular, promoviendo la reutilización y la agilización del proceso de desarrollo de software. A continuación se describen un conjunto de patrones que guiaron el diseño de solución de la presente investigación:

- **Experto:** Se encarga de asignar una responsabilidad al experto en la información. Lo que permite conservar el encapsulamiento y provee un bajo nivel de acoplamiento.
- **Creador:** Se encarga de definir quién es responsable de crear una nueva instancia de alguna clase.
- **Controlador:** Se encarga de definir quién atiende un evento en el sistema, sirviendo de intermediario entre la interfaz de usuario y la lógica de la aplicación.
- **Bajo acoplamiento:** Se encarga de definir como dar el soporte para que exista una mínima dependencia y un aumento de la reutilización. Esto evita que una clase dependa de muchas otras, haciéndose difícil el mantenimiento y la reutilización. (19)

3.3.2 Diagramas de clases del diseño

Un diagrama de clases del diseño es la representación del sistema, mediante las relaciones entre las clases cliente, servidoras, entidades y los formularios. Las clases servidoras construyen las clases clientes, que están compuestas por formularios que realizan envíos a la servidora, la cual accede a las entidades. (21) A continuación se presentan los diagramas de clases del diseño de los casos de uso arquitectónicamente significativos (Figura 16 y 17), y los restantes se encuentran en Anexo 4. Diagramas de clases del diseño.

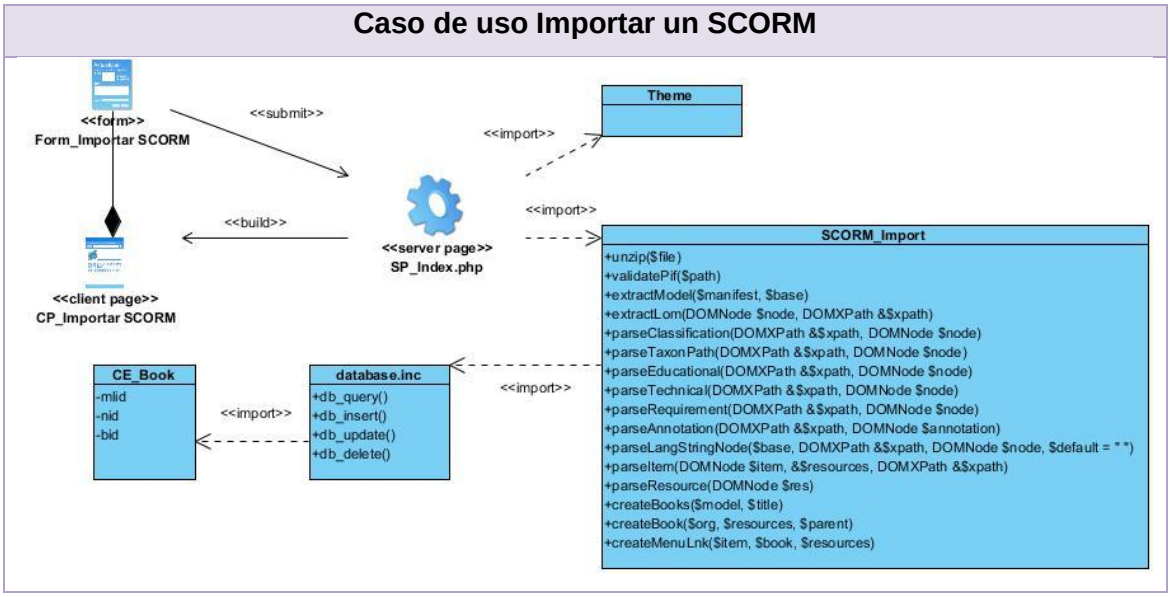


Figura 16. Diagrama de clase del diseño del caso de uso Importar un SCORM

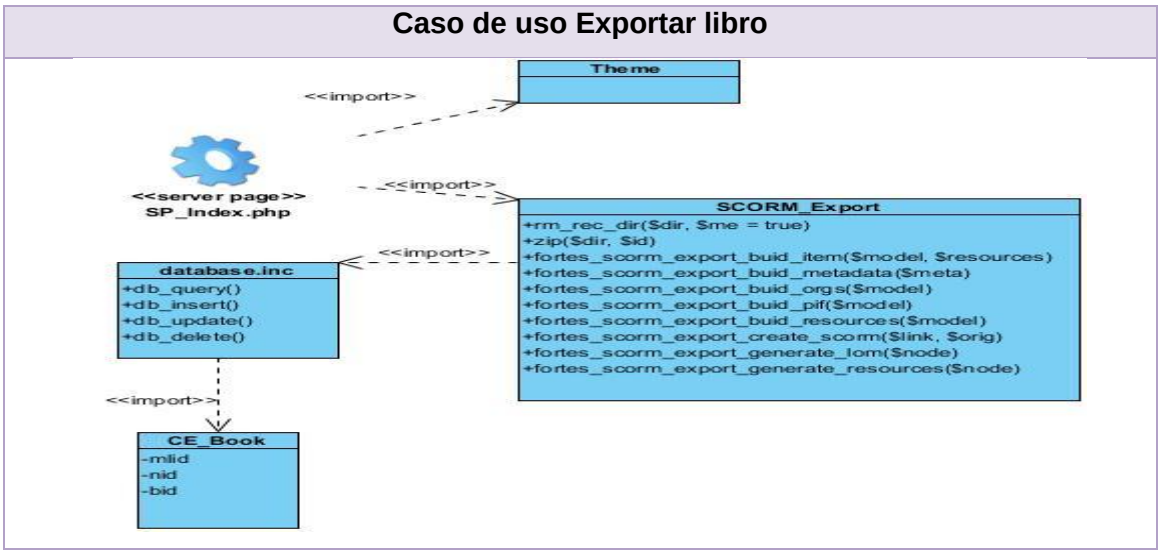


Figura 17. Diagrama de clase del diseño del caso de uso Exportar libro

3.3.3 Modelo de datos

Un modelo de datos, describe las representaciones lógicas y físicas de datos persistentes utilizados por la aplicación, e incluye elementos de modelo para procedimientos almacenados, disparadores, restricciones, etc., que definen la interacción de los componentes en un DBMS. (21) A continuación se representa el modelo de datos del componente a desarrollar en la Figura 18.

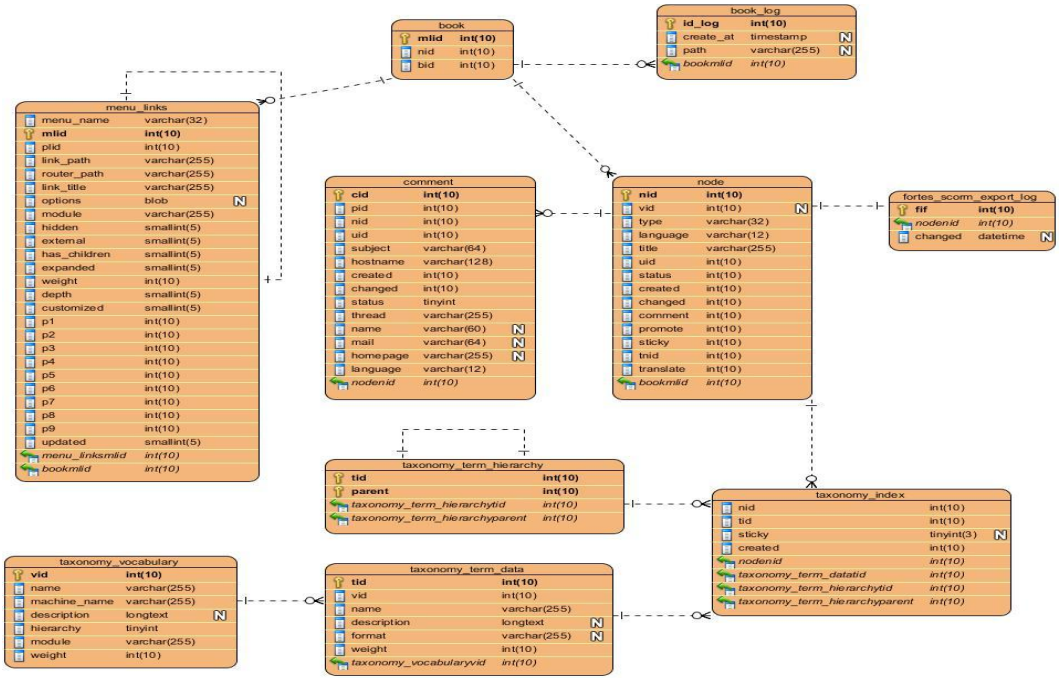


Figura 18. Modelo de datos

3.4 Conclusiones del capítulo

En este capítulo fueron creados significativos artefactos de ingeniería de software, como los diagramas de clases del análisis, los diagramas de colaboración, los diagramas de clases del diseño y el modelo de datos, referentes al análisis y diseño del componente a desarrollar. Los que permitieron un mayor entendimiento del mismo y preparó una arquitectura sólida para la implementación de la solución propuesta.

CAPÍTULO 4. IMPLEMENTACIÓN Y PRUEBA

4.1 Introducción

Este capítulo, abarca los elementos fundamentales de la fase de construcción, enfocándose en la implementación y las pruebas del componente de interoperabilidad. Describiendo las características fundamentales del mismo mediante los estándares de codificación, el modelo de implementación y la obtención del diagrama de despliegue y el diagrama de componente. Validando mediante pruebas aplicadas al producto final, el correcto funcionamiento y rendimiento requerido.

4.2 Estándares de codificación

Los estándares de codificación son usados en el desarrollo de sistemas, con el fin de lograr programas más comprensibles y de mayor calidad; aportando una mayor eficiencia al incluir un conjunto de reglas o normas al proceso de codificación. Para lo cual se utilizaran en la implementación del componente, los expuestos en el sitio oficial de Drupal (Drupal.org). (37)

Operadores:

- Todos los operadores binarios, como +, -, =, =, ==, >, etc., deben tener un espacio antes y después del operador, para facilitar la lectura.
- Los operadores unarios (operadores que operan en un solo valor), como ++, no deben tener un espacio entre el operador y la variable o el número que están operando. (37)

Estructuras de control (if, for y switch)

- Utilizar siempre elseif en lugar de "else if".
- Las sentencias de control deben tener un espacio entre la palabra clave de control y paréntesis de apertura, para distinguirlas de las funciones. Utilizando siempre las llaves aunque sean técnicamente opcionales, para aumentar la legibilidad y reducir la probabilidad de errores lógicos al añadir nuevas líneas. Estableciendo la llave de apertura, en la línea de declaración de apertura precedida de un espacio y la llave de cierre, en una línea independiente con la

sangría al nivel de la declaración de apertura, como se muestra a continuación:

```
switch (condition) {
  case 1:
    action1;
    break;

  case 2:
    action2;
    break;

  default:
    defaultaction;
}
```

```
do {
  actions;
} while ($condition);
```

- La longitud de las líneas de código no deben ser superior a 80 caracteres. Sin embargo, los nombres de funciones, las definiciones de función/clase, las declaraciones de variables, entre otras, se les permite superar los 80 caracteres. Mientras que las condiciones de la estructura de control pueden superar los 80 caracteres, si es que son fáciles de leer y entender. (37)

Etiquetas de PHP

- Utilizar siempre <?php ?> para delimitar el código PHP en lugar de <? ?>, ya que es necesario para la implementación en Drupal y es también la forma más portátil de incluir código PHP en diferentes sistemas operativos y configuraciones.
- Se omite a partir de Drupal 4.7 el uso de ?> al final de los archivos de código, eliminando la posibilidad de espacios en blanco no deseados al final de los archivos, causando problemas como la validación de XHTML/XML, entre otros. (37)

Nomenclaturas

- Las funciones y variables deben nombrarse con minúsculas, separando las palabras con un guión bajo. Nombrando en el caso de las funciones según su agrupación/módulo como un prefijo para evitar conflictos de nombres entre los

módulos.

- Las constantes deben ser siempre totalmente en mayúsculas, con caracteres de subrayado para separar las palabras, incluyendo las constantes de PHP predefinidas como TRUE, FALSE y NULL.
- Para definir variables globales, su nombre debe comenzar con un guión bajo seguido por el nombre del módulo/tema y otro subrayado.
- Todos los archivos de documentación deben tener el nombre del archivo de extensión ".txt" para facilitar la visualización de ellos en los sistemas y los nombres de archivo deben estar en mayúscula con la extensión en minúscula. (37)

Concatenación de String

- Utilizar un espacio entre el punto y las partes concatenadas para mejorar la legibilidad.

```php
<?php
  $string = 'Foo' . $bar;
  $string = $bar . 'foo';
  $string = bar() . 'foo';
  $string = 'foo' . 'bar';
?>
```

- Al concatenar variables simples, utilizar comillas y agregar la variable dentro; de lo contrario, utilizar comillas simples.

```php
<?php
  $string = "Foo $bar";
?>
```

- Utilizar un espacio a cada lado como el operador de asignación, en el caso de utilizar el operador de concatenación (' .= '). (37)

```php
<?php
$string .= 'Foo';
$string .= $bar;
$string .= baz();
?>
```

Llamadas a funciones

- Para hacer llamadas a funciones, no dejar espacio entre el nombre de la función, el paréntesis de apertura y el primer parámetro, dejar entre las comas y cada parámetro, y no dejar entre el último parámetro, el paréntesis de cierre y

el punto y coma, como se muestra a continuación.

```
$var = foo($bar, $baz, $quux);
```

- Dejar un espacio a ambos lados de un signo igual, usado para asignar el valor de retorno de una función a una variable y en el caso de un bloque de asignaciones relacionadas, se puede insertar espacios para promover la legibilidad. (37)

```
$short        = foo($bar);
$long_variable = foo($baz);
```

4.3 Modelo de implementación

El modelo de implementación describe la organización de los componentes según los mecanismos de estructuración disponibles en el entorno de implementación y los lenguajes de programación utilizados. Detallando como se implementan en términos de componentes y sus dependencias, los elementos obtenidos del modelo de diseño. Siendo así, la entrada principal a las pruebas de software que siguen a la etapa de implementación. (21)

4.3.1 Diagrama de componentes

"Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo de diseño". (21) Por lo que el diagrama de componentes, se encarga de mostrar los componentes del software y las relaciones lógicas entre ellos en un sistema, además de estructurar el modelo de implementación en términos de subsistemas de implementación. Modelando así la vista estática del sistema, representada a continuación en la Figura 19.

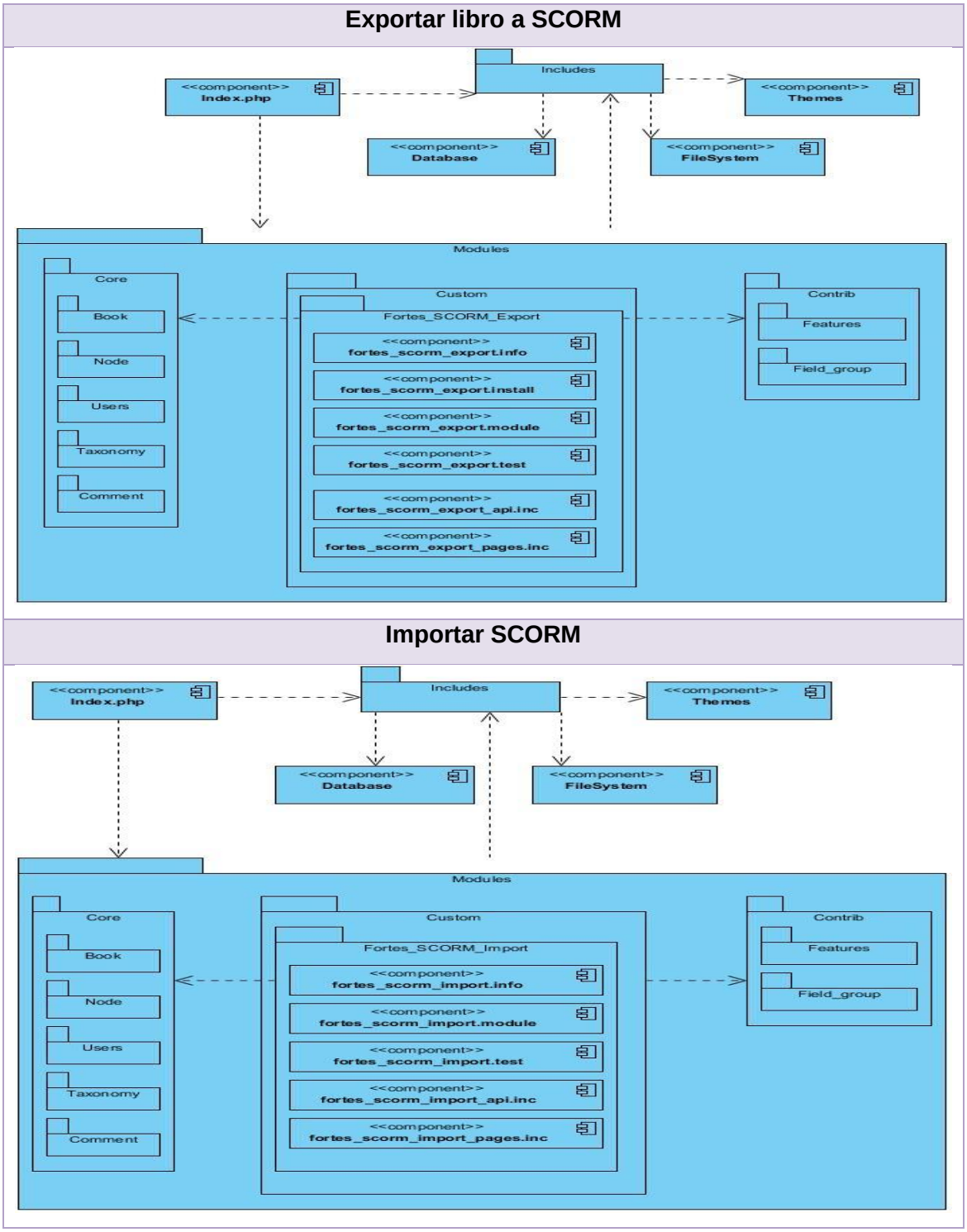


Figura 19. Diagrama de componentes

4.3.2 Diagrama de despliegue

"El diagrama de despliegue es un modelo de objeto que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo". (21) Expuesto mediante una colección de nodos (Figura 20); donde cada

nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo de hardware similar.

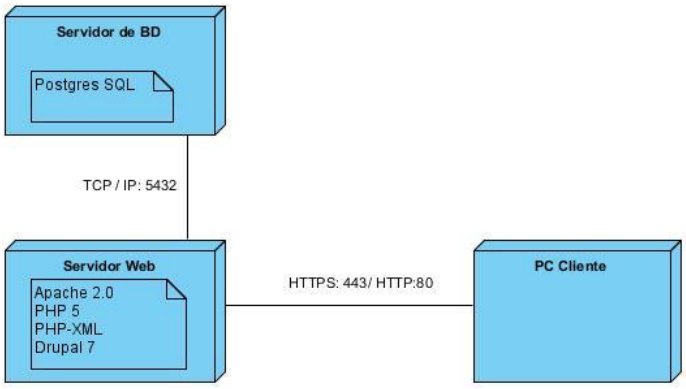


Figura 20. Diagrama de despliegue

4.4 Pruebas

"Las pruebas del software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación". (19) Proporcionando técnicas de evaluación dinámica, con distintos criterios para generar casos de prueba que provoquen fallos en los programas, agrupadas en:

- **Técnicas de caja blanca**: Se basan en un minucioso examen de los detalles procedimentales del código a evaluar, por lo que es necesario conocer la lógica del programa. (Figura 21)

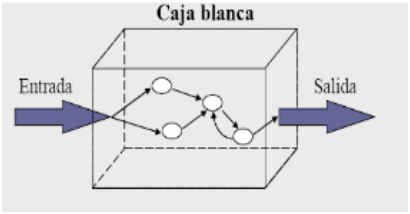


Figura 21. Representación de pruebas de Caja Blanca

- **Técnicas de caja negra**: Realizan pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas del mismo. No es necesario conocer la lógica interna del programa, únicamente la funcionalidad que debe realizar. (Figura 22)

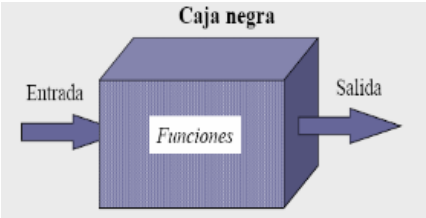


Figura 22. Representación de pruebas de Caja Negra

De manera que, la aplicación de las pruebas de caja blanca, deben probar todos los posibles caminos por los que puede pasar el flujo de control de un programa para disponer de un código perfectamente correcto. Sin embargo, para programas de cierta envergadura, el número de casos de prueba que habría que generar sería excesivo, ya que el número de caminos incrementa exponencialmente a medida que el número de sentencias condicionales y bucles aumenta; por lo que se debe elegir y ejercitar ciertos caminos representativos de un programa. De igual forma, en la aplicación de las pruebas de caja negra, tampoco es factible probar todas las entradas a un programa, por lo que se seleccionan un conjunto representativo de entradas y se generan los correspondientes casos de prueba, con el fin de provocar fallos en los programas. Siendo así, técnicas complementarias que han de aplicarse al realizar una prueba dinámica, ya que pueden ayudar a identificar distintos tipos de faltas en un programa. (19)

4.4.1 Pruebas de caja blanca

"La prueba de caja blanca se basa en el diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivarlos. Mediante la prueba de la caja blanca el ingeniero del software puede obtener casos de prueba que:

- *Garanticen que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo, programa o método.*
- *Ejerciten todas las decisiones lógicas en las vertientes verdadera y falsa.*
- *Ejecuten todos los bucles en sus límites operacionales.*
- *Ejerciten las estructuras internas de datos para asegurar su validez.*

Es por ello que se considera a la prueba de Caja Blanca como uno de los tipos de pruebas más importantes que se le aplican al software, logrando como resultado que disminuya en un gran porciento el número de errores existentes en los sistemas y por

ende una mayor calidad y confiabilidad". (19)

4.4.1.1 Pruebas unitarias. Módulo Testing de Drupal

En la realización de las pruebas de caja blanca, aplicadas al componente de interoperabilidad, se establecieron las pruebas a **nivel de unidad**, visto desde el contexto de la ingeniería de software. Empleando de forma recurrente las técnicas de prueba que recorren caminos específicos en una estructura de control del componente. Tomando como guía la descripción del diseño a nivel de componentes y concentrando el esfuerzo en la lógica del procesamiento interno. Seleccionando las rutas de ejecución, mediante el diseño de los casos de prueba, con el fin de descubrir errores de cálculos incorrectos, comparaciones erróneas o flujos de control inapropiados. (19)

Proceso realizado usando el módulo **Testing** del núcleo de Drupal 7.x, basado originalmente en la biblioteca SimpleTest PHP[22]. El cual, una vez ejecutadas las pruebas, muestra si estas pasaron, fallaron o tuvieron alguna excepción, permitiendo así refinar el código y seguir el proceso logrando un refinamiento del sistema (Anexo 5).

Iteraciones:

Después de ser configuradas las pruebas, se realizaron tres iteraciones (Figura 23). Obteniéndose en la primera iteración resultados adversos, detectando cuatro métodos que no producían las salidas esperadas (Tabla 4), los cuales fueron corregidos, y se pasó a la siguiente etapa de prueba manteniendo los casos de prueba con el fin de garantizar la corrección de los errores detectados.

22 SimpleTest PHP: http://www.lastcraft.com/simple_test.php

Métodos	Cantidad de juegos de datos	Salidas correctas	Salidas incorrectas
parseAnnotation()	3	2	1
parseLangStringNode()	3	1	2
parseTechnical()	3	3	0
parseRequirement()	3	0	3
parseResource()	3	1	2

Tabla 4. Primera iteración

Luego de las correcciones anteriores, se realizó una nueva iteración de las pruebas unitarias, arrojando cuatro métodos con resultados incorrectos (Tabla 5), procediendo así a un nuevo ciclo de pruebas sin modificar los casos de prueba.

Métodos	Cantidad de juegos de datos	Salidas correctas	Salidas incorrectas
parseAnnotation()	3	2	1
parseLangStringNode()	3	2	1
parseTechnical()	3	3	0
parseRequirement()	3	2	1
parseResource()	3	2	1

Tabla 5. Segunda iteración

Culminando tres ciclos de pruebas con todas las funcionalidades listas para realizar las tareas para las que fueron concebidas e implementadas (Tabla 6).

Métodos	Cantidad de juegos de datos	Salidas correctas	Salidas incorrectas
parseAnnotation()	3	3	0
parseLangStringNode()	3	3	0
parseTechnical()	3	3	0
parseRequirement()	3	3	0
parseResource()	3	3	0

Tabla 6. Tercera iteración

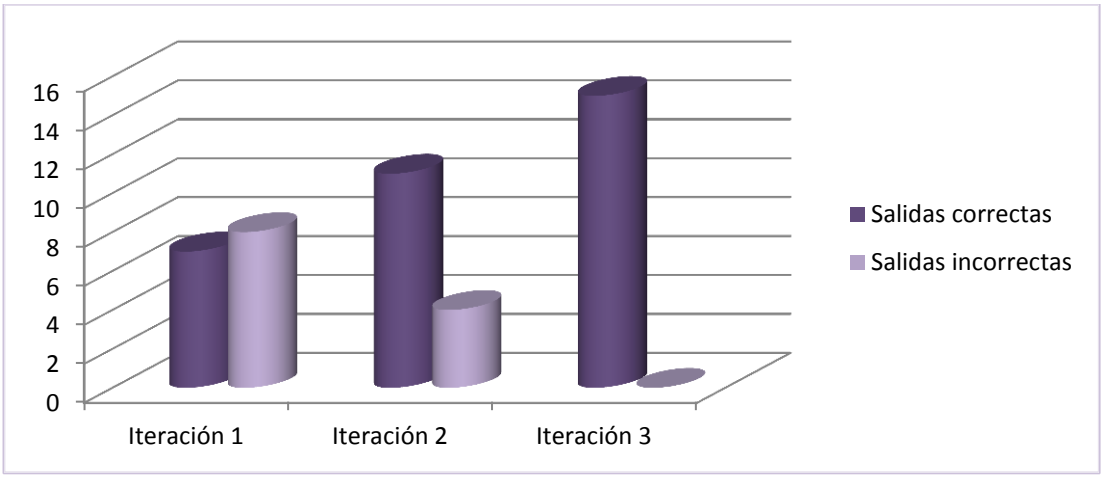


Figura 23. Resultados por iteración

4.4.2 Pruebas de caja negra

Las pruebas de Caja Negra se centran principalmente en los requerimientos funcionales del software, permitiendo obtener un conjunto de condiciones de entrada que ejerciten los mismos. Siendo un complemento que intenta descubrir diferentes tipos de errores a los encontrados en los métodos de la caja blanca, como:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación. (19)

Preparando casos de pruebas con un conjunto de datos que ayuden a la ejecución de los mismos y que permitan que el sistema se ejecute en todas sus variantes, para hallar errores o validar funcionalidades, utilizando varias técnicas entre las que se encuentran:

- **Partición de Equivalencia:** Divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
- **Análisis de Valores Límites:** Prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- **Grafos de Causa-Efecto:** Permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones. (19)

4.4.2.1 Partición de equivalencia

La técnica partición de equivalencia, es dentro del método de Caja Negra una de las más efectivas. Permite examinar los valores válidos e inválidos de las entradas existentes en el software, descubriendo de forma inmediata errores que para su detección requerirían la ejecución de muchos casos de prueba. Se basa en la división del dominio de entrada de un programa en un número finito de clases de equivalencia. Asume que una prueba realizada con un valor representativo de una clase es equivalente a una prueba realizada con cualquier otro valor de dicha clase. Por lo cual, si el caso de prueba correspondiente a una clase de equivalencia detecta un error, el resto de los casos de prueba de dicha clase de equivalencia deben detectar el mismo error y viceversa. Utilizando para el diseño de los casos de prueba, la identificación de las clases de equivalencia y los escenarios de prueba. (45)

Clases de equivalencia:

"Una clase de equivalencia representa un conjunto de estados válidos y no válidos para las condiciones de entrada de un programa". (45) Identificadas al examinar las condiciones de entrada y dividiéndola en dos o más grupos. Definidas en clases de equivalencia válidas, que representan entradas válidas al programa, y clases de equivalencia no válidas, que representan valores de entrada erróneos. (Tabla 7)

Variables	Clases de equivalencia válidas	Clases de equivalencia no válidas
Nombre del libro	Cualquier valor alfanumérico hasta 255 caracteres.	Campo vacío o nulo.
PIF	Fichero que contiene un PIF con formato SCORM 2004.	Campo vacío y fichero que no contiene un PIF con formato SCORM 2004.
Ruta taxonómica	0 < Clasificaciones <= 40	40 < Clasificaciones
Anotaciones	0 < Anotaciones <= 30	30 < Anotaciones

Tabla 7. Partición de equivalencia del caso de uso Importar un SCORM

Para lo cual se siguen pautas como:

- Si una condición de entrada especifica un rango de valores, identificar una clase de equivalencia válida y dos clases no válidas.
- Si una condición de entrada especifica un valor o número de valores, identificar una clase válida y dos clases no válidas.
- Si una condición de entrada especifica un conjunto de valores de entrada, identificar una clase de equivalencia válida y una no válida.
- Si una condición de entrada especifica una situación que debe ocurrir, esto es, es lógica, identificar una clase válida y una no válida.
- En general, si hay alguna razón para creer que los elementos de una clase de equivalencia no se tratan de igual modo por el programa, dividir la clase de equivalencia entre clases de equivalencia más pequeñas para cada tipo de elementos. (45)

Descripción de los casos de prueba:

La descripción de los casos de prueba constituye una guía precisa, de la cual depende el éxito de la prueba. Está conformada por posibles entradas, condiciones de ejecución y las posibles salidas que puedan generar para lograr un resultado esperado. (45) Para lo cual se realizó un estudio detallado de las descripciones textuales de los casos de uso, con el fin de identificar los escenarios de pruebas (Tabla 8) y cuáles podrían ser sus flujos básicos y alternos, así como las variables de comportamiento (Tabla 7), con el fin de introducir valores reales durante la ejecución de las pruebas, procediendo a confeccionar la Matriz de Datos. (Tabla 9)

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1 Importar un	EC 1.1 Selecciona importar un SCORM.	El usuario insertar un SCORM para ser importado.
	EC 1.1 Datos obligatorios	Existen datos obligatorios sin introducir.
	EC 1.2 Datos incorrectos	Los datos entrados por el usuario son incorrectos.

SCORM	EC 1.3 Generar taxonomías	Se generan vocabularios en el modelo taxonómico de Drupal.
	EC 1.4 Generar contenidos	Se generan contenidos y comentarios en Drupal.

Tabla 8. Escenarios para el caso de uso Importar un SCORM

Escenario	Nombre del libro	PIF	Ruta taxonómica	Anotaciones	Respuesta del sistema	Flujo central
EC 1.1 Selecciona importar un SCORM.	V	V	V	V	El sistema valida el PIF del SCORM y genera un libro de Drupal por cada organización presente en el SCORM.	Administración de contenido /Importar SCORM
EC 1.1 Datos obligatorios	I	NA	NA	NA	El sistema muestra el formulario señalando el error de campo obligatorio	Administración de contenido /Importar SCORM
	I	V	NA	NA		
	V	I	NA	NA		
	NA	I	NA	NA		
EC 1.2 Datos incorrectos	I	NA	NA	NA	El sistema muestra el formulario señalando el error de datos incorrectos	Administración de contenido /Importar SCORM
	I	V	NA	NA		
	V	I	NA	NA		
	NA	I	NA	NA		
EC 1.3 Generar taxonomías	V	V	V	NA	El sistema genera vocabularios en el modelo taxonómico de Drupal.	Administración de contenido/ Importar SCORM
EC 1.4 Generar contenidos	V	V	NA	V	El sistema genera contenidos y comentarios.	Administración de contenido/ Importar SCORM

Tabla 9. Matriz de datos

Resultado de las pruebas de caja negra:

En la evaluación del componente de interoperabilidad, se desarrollaron tres iteraciones basadas en los métodos y técnicas descritos anteriormente. Proceso que arrojó resultados visibles, mostrando la calidad a un alto grado de detalle, mediante la representación del impacto de las **no conformidades** (NC) (Anexo 5), encontradas en las iteraciones (Tabla 10). Comprobando, mediante pruebas iterativas e incrementales, la corrección de los errores detectados (Figura 24), contribuyendo así a la obtención de la calidad y funcionalidad del sistema.

Clasificación de las NC	1ra Iteración	2da Iteración	3ra Iteración
Alta	2	1	0
Media	3	2	0
Baja	2	1	0

Tabla 10. Cantidad de no conformidades detectadas en las iteraciones

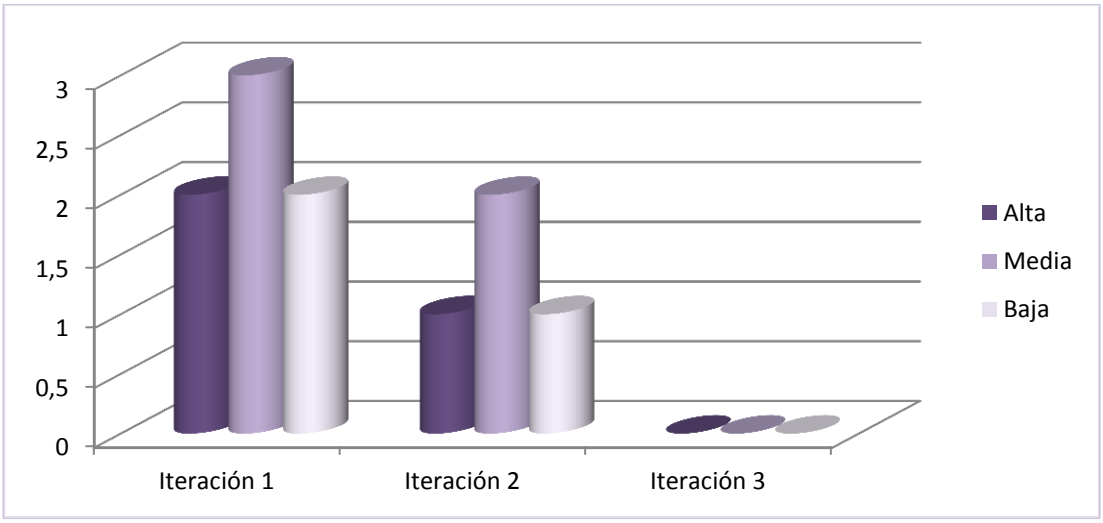


Figura 24. Cantidad de no conformidades por iteración

4.5 Conclusiones del capítulo

La realización de este capítulo permitió definir de manera precisa los estándares de codificación a tener en cuenta en la implementación del componente, necesarios en el trabajo con Drupal y posteriores desarrollos. Además se realizaron las pruebas de caja blanca y caja negra, detectando errores en el componente que fueron mitigados para obtener una solución con la calidad requerida.

CONCLUSIONES

Los objetivos previstos para la presente investigación fueron cumplidos tras el desarrollo de la investigación, arribando a las siguientes conclusiones:

1. Se elaboró el marco teórico conceptual de la investigación.
2. Se determinaron las principales características del componente que da solución a la problemática planteada.
3. Se desarrollaron los principales artefactos de los flujos de Análisis y Diseño e Implementación del componente para la transformación de contenidos.
4. Se realizaron las pruebas internas para garantizar la funcionalidad del componente, concluyendo con resultados satisfactorios.

RECOMENDACIONES

Una vez que los objetivos de la investigación fueron cumplidos obteniendo un componente capaz de solucionar la problemática planteada, se recomienda lo siguiente:

1. Implementar estándares que permitan incorporar los PIF generados por el módulo para exportar libro de forma automática a Repositorios de Objetos de Aprendizaje.
2. Implementar mapeos del modelo de datos de Drupal 7 y otras secciones de SCORM 2004 como es la secuencia y navegación (SN) y el entorno de ejecución (RTE)
3. Desarrollar un módulo que permita configurar a los administradores mapeos personalizados para distintos tipos de contenidos de Drupal 7 con los metadatos de LOM.
4. Realizar una versión del componente para la versión 8 de Drupal.

BIBLIOGRAFÍA

1. Unesco. [En línea] [Citado el: 29 de 11 de 2013.] http://www.unesco.org.

2. e-ABCLearning. [En línea] [Citado el: 10 de 12 de 2014.] http://www.e-abclearning.com/definicione-learning.

3. **Learning Technology Standards Committee.** IEEE LOM Learning Object Metadata. [En línea] [Citado el: 19 de 1 de 2014.] http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf.

4. **Wiley, David.** The Instructional Use of Learning Objects. [En línea] 2000. [Citado el: 15 de 1 de 2014.] http://reusability.org/read/.

5. **Polsani, P. R.** *Use and abuse of reusable learning objects.* s.l. : Journal of Digital Information, 2003.

6. **Guzmán, Clara López.** *Los Repositorios de Objetos de Aprendizaje.* Salamanca, España : s.n., 2005.

7. Que son los Objetos de Aprendizaje (OA). *Tecnología Educativa.* [En línea] [Citado el: 29 de 11 de 2013.] http://a01004363.blogspot.com/2010/04/que-son-los-objetos-de-aprendizaje-oa.html.

8. **Indarte, Selene.** Interoperabilidad. *Manual de salud electrónica para directivos de servicios y sistemas de salud.* [En línea] [Citado el: 23 de 11 de 2013.] http://www.seis.es/documentos/informes/secciones/adjunto1/15_Interoperabilidad.pdf.

9. **WORLD METEOROLOGICAL ORGANIZATION.** ISO 191xx series of geographic information standards. [En línea] [Citado el: 16 de 3 de 2014.] www.wmo.int/pages/prog/www/.../ISO_191xx.doc.

10. **Boneu, Josep M.** Plataformas abiertas de e-learning para el soporte de contenidos educativos abiertos. [En línea] [Citado el: 9 de 11 de 2013.] www.uoc.edu/rusc/4/1/dt/esp/boneu.pdf.

11. **Farley, Luis y Ortiz F.** Qué son lo LMS. *Campus Virtual: La educación mas allá de los LMS. Revista de Universidad y Sociedad del Conocimiento.* [En línea] [Citado el: 20 de 11 de 2013.] http://www.uoc.edu.

12. Estándares de e-learning. Capítulo 16. *Libro de Buenas Prácticas de e-learning.* [En línea] [Citado el: 7 de 11 de 2013.] http://www.buenaspracticas-elearning.com.

13. Metadatos-Recuperación de Documentos XML/RDF. [En línea] [Citado el: 17 de 3 de 2014.] http://www.metadatos-xmlrdf.com.

14. **Advanced Distributed Learning.** *SCORM 2004 4th edition. Content Aggregation Model. Versión 1.1.* 2009.

15. Dublin Core. [En línea] [Citado el: 15 de 1 de 2014.] http://www.dublincore.org/.

16. **Advanced Distributed Learning.** *SCORM 2004 4th edition. Run-Time Environment. Versión 1.1.* 2009.

17. —. *SCORM 2004 4th edition. Sequencing and Navigation. Versión 1.1.* 2009.

18. *OAI-PMH: Protocolo para la transmisión de contenidos en Internet.* **Barrueco, José Manuel y Coll, Imma Subirats.**

19. **Pressman, Roger S.** *Ingeniería de Software un enfoque práctico 5ta edición.* Quinta Edición.

20. **Figueroa, Roberth G., Solís, Camilo J. y Armando.** *METODOLOGÍAS TRADICIONALES VS. METODOLOGÍAS ÁGILES.* Universidad Técnica Particular de Loja, Escuela. s.l. : Universidad Técnica Particular de Loja, Escuela de Ciencias de Computación.

21. **Jacobson, Ivar, Booch, Grady and Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software .* s.l. : Pearson Educacion , 2000.

22. **Escribano, G.F.** Departamentos de Sistemas Informáticos. [En línea] [Citado el: 30 de 1 de 2014.] http://www.dsi.uclm.es/asignaturas/42551/trabajosAnteriores/Presentacion-XP.pdf..

23. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *Lenguaje Unificado de Modelado.*

24. **Barahona, Jesús González y Pascual, Joaquín Seoane.** Introducción al software libre. [En línea] Grupo de Sistemas y Comunicaciones, ESCET, Universidad Rey Juan Carlos de Madrid. [Citado el: 25 de 11 de 2013.] http://curso-sobre.berlios.de/introsobre/2.0.1/sobre.html/sec-ide.html.

25. NetBeans IDE. [En línea] [Citado el: 16 de 2 de 2014.] https://netbeans.org.

26. **Universidad Politécnica de Valencia.** Introducción a Herramientas CASE y System Architect. *Metodología y Tecnología de la Programación.* [En línea] [Citado el: 10 de 3 de 2014.] http://users.dsic.upv.es/asignaturas/eui/mtp/doc-practicas/intro_case_SA.pdf.

27. Guión Visual Paradigm for UML. [En línea] [Citado el: 16 de 12 de 2013.] http://www.ie.inf.uc3m.es.

28. La Revista Informática.com. [En línea] [Citado el: 24 de 1 de 2014.] http://www.larevistainformatica.com.

29. Introducción a JavaScript. *Libros Web.* [En línea] [Citado el: 18 de 1 de 2014.] http://librosweb.es.

30. PHP Ya. [En línea] [Citado el: 27 de 1 de 2014.] http://www.phpya.com.ar/.

31. Historia de PHP y Proyectos Relacionados. [En línea] [Citado el: 19 de 1 de 2014.] http://es2.php.net/history.

32. Definición de. [En línea] [Citado el: 16 de 1 de 2014.] http://definicion.de/xml/.

33. **J. E. Villate.** *Introducción al XML.* 2001.

34. XML.com. [En línea] http://www.xml.com.

35. Git - fast version control. [En línea] Software Freedom Conservancy. [Citado el: 16 de 2 de 2014.] http://git-scm.com/.

36. Desarrolloweb.com. *Qué es un CMS.* [En línea] [Citado el: 29 de 11 de 2013.] http://www.desarrolloweb.com/articulos/que-es-un-cms.html.

37. Drupal. [En línea] [Citado el: 3 de 2 de 2014.] http://drupal.org.es/drupal.

38. **Rodríguez, Fran Gil.** *Experto en Drupal 7. Nivel Intermedio.* s.l. : Forcontu S.L.

39. —. *Experto en Drupal 7. Nivel Inicial.* s.l. : Forcontu S.L.

40. —. *Experto en Drupal 7. Nivel Avanzado.* s.l. : Forcontu S.L.

41. BaseDatosOfimaticas. [En línea] [Citado el: 8 de 11 de 2013.] http://basedatosofimaticas.wikispaces.com.

42. PostgreSQL-es. [En línea] [Citado el: 23 de 1 de 2014.] http://www.postgresql.org.es.

43. MIS RESPUESTAS.COM. [En línea] [Citado el: 18 de 12 de 2013.] http://www.misrespuestas.com.

44. **McCool, Rob.** Apache. [En línea] Apache Software Foundation. http://httpd.apache.org.

45. **Juristo, Natalia, Moreno, Ana M. y Vegas, Sira.** *TÉCNICAS DE EVALUACIÓN DE SOFTWARE.* 2006.

46. **Wesley, Addison.** *El proceso unificado de desarrollo de software.* 1999.

47. *Reusable Learning Objects (RLOs) for Computer Science Students.* **Balci y Inceoglu.** 2006.

48. **Forouzan, Behrouz A.** *Transmisión de datos y redes de comunicación.* Madrid : s.n., 2002.

49. **Sommerville, Ian.** *Ingeniería de Software.* Madrid : Pearson Educación, S.A., 2005.

50. **K.VanDyk, John.** *Pro Drupal Development, Second Edition.* 2008.

51. **Tomlinson, John K. VanDyk Todd.** *Pro Drupal 7 Deveploment Drupal 7 Development.* 2010.

52. Hacer Reforma, el nuevo pacto educativo. [aut. libro] Juan Carlos Tedesco. *Educación, competitividad y ciudadanía en la sociedad moderna.* Madrid : Grupo Anaya.

53. **Cambra, Pedro, y otros, y otros.** *Experto en Drupal 7. Nivel Experto Vol. I.* s.l. : Forcontu S.L.

54. Un paso electrónico:Infraestructuras y servicios comunies. *Interoperabililidad e intercambio de datos.* s.l. : creative commons.

55. **Tomlinson, Todd y k.VanDyk, John.** *Pro Drupal 7 Develpment: Third Edition.* 2010.

56. **Laverde, Andrés Chiappe.** OBJETOS DE APRENDIZAJE: CONCEPTUALIZACIÓN Y PRODUCCIÓN. [En línea] [Citado el: 15 de 1 de 2014.] www.cudi.edu.mx/diplomadoOA.

57. Blog - Biblioteca Universidad Arturo Prat. [En línea] [Citado el: 6 de 11 de 2013.] http://bibliopress.wordpress.com.

58. **Castro, Eduardo Peñalosa y Durán, Patricia Landa.** Objeto de aprendizaje: Una propuesta de conceptualización, texonomía y metodología. [En línea] [Citado el: 25 de 10 de 2013.] www.revistas.unam.mx/index.php/repi/article/.

59. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** METODOLOGIAS PARA DESARROLLO DE SOFTWARE. *Procesos de Software.* [En línea] [Citado el: 3 de 2 de 2014.] http://procesosdesoftware.wikispaces.com.

60. LMS y LCMS: Funcionalidades y beneficios. [En línea] [Citado el: 15 de 2 de 2014.] http://www.centrocp.com.

61. Marco de Interoperabilidad. [En línea] [Citado el: 24 de 11 de 2013.] http://marcodeinteroperabilidad.blogspot.com/.

62. METAUNIVERSIDAD. [En línea] [Citado el: 23 de 2 de 2014.] http://metauniversidad.com.

63. newWweb. [En línea] [Citado el: 27 de 11 de 2013.] http://newwweb.com.mx.

64. Repositorios de Objetos de Aprendizaje. [En línea] [Citado el: 18 de 11 de 2013.] http://cvonline.uaeh.edu.mx.

65. **Pressman, Roger S.** *Ingeniería de Software un enfoque práctico 6ta edición.*

ANEXOS

Anexo 1. Descripción de los casos de uso

Caso de uso Exportar índice		
Objetivos	Generar una organización de SCORM 2004.	
Actores	CRON	
Resumen	Se invoca la funcionalidad de exportar un lib 2004ro de Drupal y el sistema genera una organización de SCORM.	
Complejidad	Alta	
Prioridad	Alta	
Precondiciones	Debe existir un libro de Drupal disponible para ser exportado.	
Postcondiciones	Se genera una organización de SCORM.	
Flujo de eventos		
Flujo básico exportar índice.		
Actor		**Sistema**
1. Invoca la funcionalidad de exportar libro de Drupal a SCORM 2004.		2. Extrae el índice del libro de Drupal. 3. Genera una organización de SCORM 2004 a partir del índice extraído. 4. Termina el caso de uso.
Relaciones	CU Incluidos	
	CU Extendidos	
Requisitos no funcionales	- Integridad	
Asuntos pendientes		

<table>
<tr><th colspan="4">Caso de uso Exportar contenido</th></tr>
<tr><td>Objetivos</td><td colspan="3">Generar el SCO de cada contenido asociado a un libro de Drupal.</td></tr>
<tr><td>Actores</td><td colspan="3">CRON</td></tr>
<tr><td>Resumen</td><td colspan="3">Se invoca la funcionalidad de exportar un libro de Drupal y el sistema genera un SCO de SCORM 2004 por cada contenido asociado a un libro de Drupal.</td></tr>
<tr><td>Complejidad</td><td colspan="3">Alta</td></tr>
<tr><td>Prioridad</td><td colspan="3">Alta</td></tr>
<tr><td>Precondiciones</td><td colspan="3">Debe existir un libro de Drupal a ser exportado.</td></tr>
<tr><td>Postcondiciones</td><td colspan="3">Se genera un SCO de SCORM 2004 por cada contenido asociado a un libro de Drupal.</td></tr>
<tr><td colspan="4">Flujo de eventos</td></tr>
<tr><td colspan="4">Flujo básico exportar contenidos.</td></tr>
<tr><td colspan="2">Actor</td><td colspan="2">Sistema</td></tr>
<tr><td colspan="2">1. Invoca la funcionalidad de exportar libro de Drupal a SCORM 2004.</td><td colspan="2">2. Por cada contenido asociado a un libro de Drupal genera un SCO del SCORM 2004:
2.1. Genera los metadatos correspondientes a partir de las propiedades de los contenidos, sus comentarios y las taxonomías asociadas.
2.2. Asocia como recurso del SCO los campos archivos que posea el contenido.
3. Asocia los SCO generados al ítem correspondiente a una organización de Drupal.
4. Termina el caso de uso.</td></tr>
<tr><td rowspan="2">Relaciones</td><td>CU Incluidos</td><td colspan="2"></td></tr>
<tr><td>CU Extendidos</td><td colspan="2"></td></tr>
<tr><td>Requisitos no funcionales</td><td colspan="3">- Integridad</td></tr>
<tr><td>Asuntos pendientes</td><td colspan="3"></td></tr>
</table>

<table>
<tr><th colspan="4">Caso de uso Descargar SCORM</th></tr>
<tr><td>Objetivos</td><td colspan="3">Descargar un libro de Drupal como un SCORM 2004.</td></tr>
<tr><td>Actores</td><td colspan="3">Usuario</td></tr>
<tr><td>Resumen</td><td colspan="3">Se solicita descargar un libro de Drupal como un SCORM 2004 y el sistema devuelve el flujo de descarga.</td></tr>
<tr><td>Complejidad</td><td colspan="3">Baja</td></tr>
<tr><td>Prioridad</td><td colspan="3">Media</td></tr>
<tr><td>Precondiciones</td><td colspan="3">El sistema debe tener una versión de SCORM 2004 generada del libro que se desea exportar.</td></tr>
<tr><td>Postcondiciones</td><td colspan="3">Descarga del paquete SCORM 2004.</td></tr>
<tr><td colspan="4">Flujo de eventos</td></tr>
<tr><td colspan="4">Flujo básico descargar SCORM.</td></tr>
<tr><td colspan="2">Actor</td><td colspan="2">Sistema</td></tr>
<tr><td colspan="2">1. Solicita descargar el SCORM 2004 de un libro que posee un vínculo de descarga.</td><td colspan="2">2. Devuelve al usuario el flujo de descarga del libro en un paquete zip correspondiente al SCORM 2004.
3. Termina el caso de uso.</td></tr>
<tr><td rowspan="2">Relaciones</td><td>CU Incluidos</td><td colspan="2"></td></tr>
<tr><td>CU Extendidos</td><td colspan="2"></td></tr>
<tr><td>Requisitos no funcionales</td><td colspan="3">- Disponibilidad</td></tr>
<tr><td>Asuntos pendientes</td><td colspan="3"></td></tr>
</table>

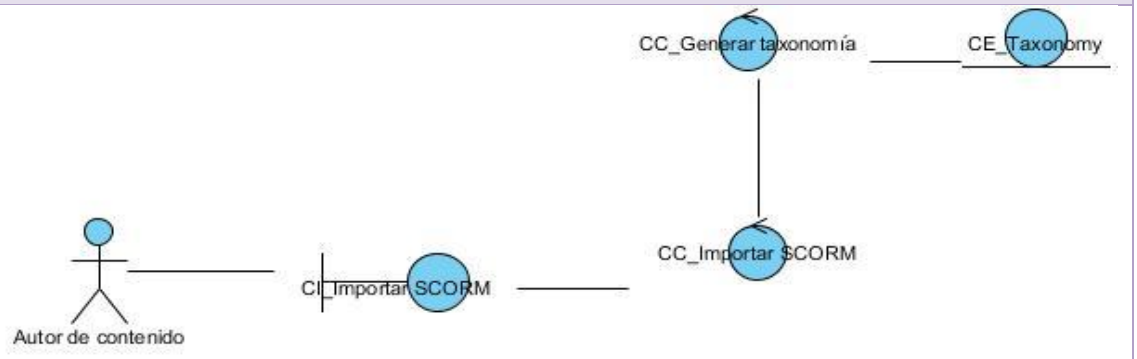
Caso de uso Generar taxonomías		
Objetivos	Generar taxonomías de Drupal.	
Actores	Autor de contenido.	
Resumen	Se selecciona la opción de importar un OA a un libro de Drupal y el sistema genera taxonomías a partir de los metadatos del manifiesto del PIF.	
Complejidad	Alta	
Prioridad	Alta	
Precondiciones		
Postcondiciones	Genera vocabularios en el modelo taxonómico de Drupal.	
Flujo de eventos		
Flujo básico generar taxonomías.		
Actor		**Sistema**
1. Invoca la funcionalidad de importar el OA a Drupal.		2. Busca todas las rutas taxonómicas existentes en los metadatos del manifiesto del PIF. 3. Genera los vocabularios correspondientes en el modelo taxonómico de Drupal a partir de las rutas taxonómicas encontradas. 4. Termina el caso de uso.
Relaciones	CU Incluidos	
	CU Extendidos	
Requisitos no funcionales	- Integridad	
Asuntos pendientes		

Caso de uso CU Generar contenidos		
Objetivos	Generar contenidos de Drupal.	
Actores	Autor de contenido.	
Resumen	Se selecciona la opción de importar un OA a un libro de Drupal y el sistema genera contenidos de Drupal por cada SCO del SCORM 2004.	
Complejidad	Alta	
Prioridad	Alta	
Precondiciones		
Postcondiciones	Genera contenidos de Drupal. Asocia comentarios a los contenidos generados.	
Flujo de eventos		
Flujo básico generar contenidos.		
Actor		**Sistema**
1. Invoca la funcionalidad de importar el OA a Drupal.		2. Por cada SCO del SCORM 2004 genera un contenido de Drupal a partir de los metadatos y recursos asociados a él. 3. Genera comentarios a partir de los metadatos anotaciones. 4. Asocia los comentarios al contenido creado en el orden correspondiente. 5. Termina el caso de uso.
Relaciones	CU Incluidos	
	CU Extendidos	
Requisitos no funcionales	- Integridad	
Asuntos pendientes		

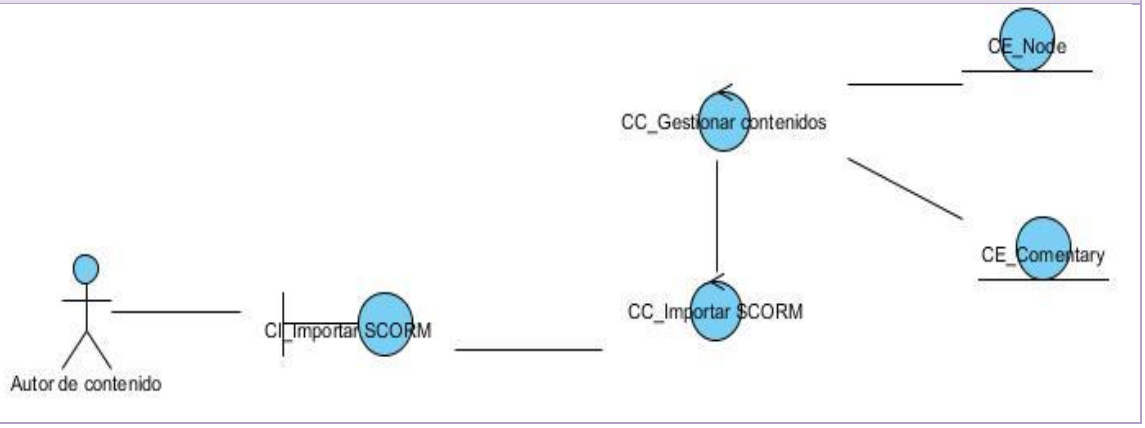
Anexo 2. Diagramas de clases del análisis

Caso de uso Importar un SCORM

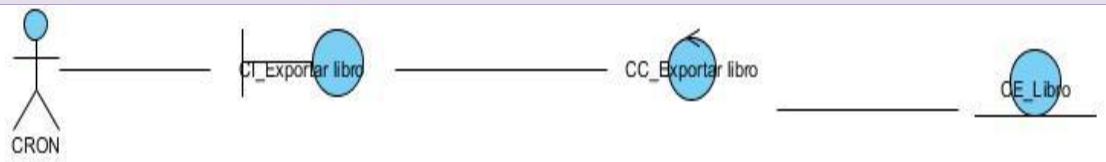
Caso de uso Generar taxonomías



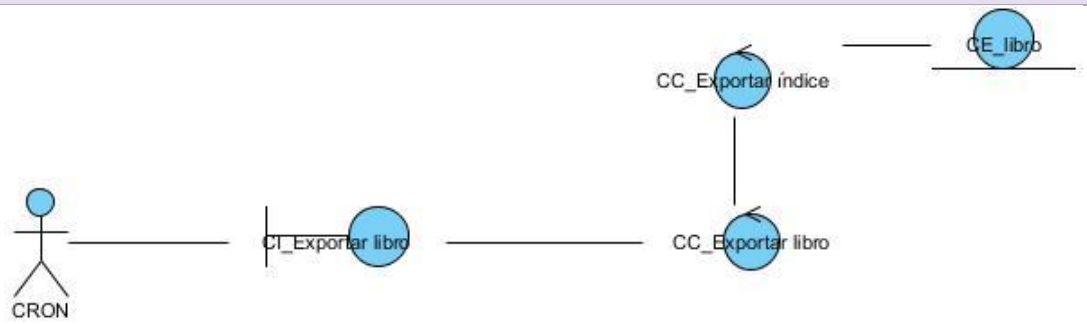
Caso de uso Generar contenidos



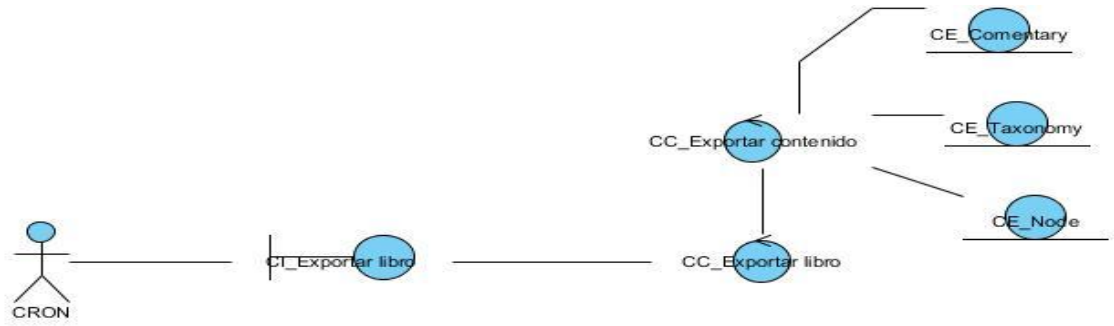
Caso de uso Exportar libro



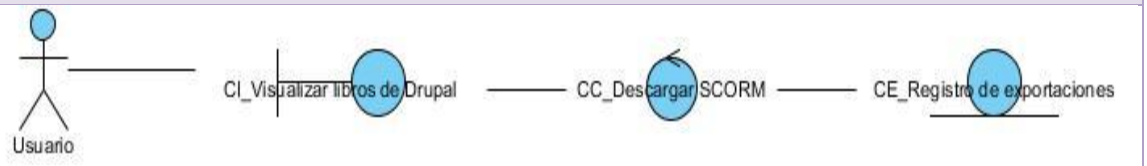
Caso de uso Exportar índice



Caso de uso Exportar contenidos

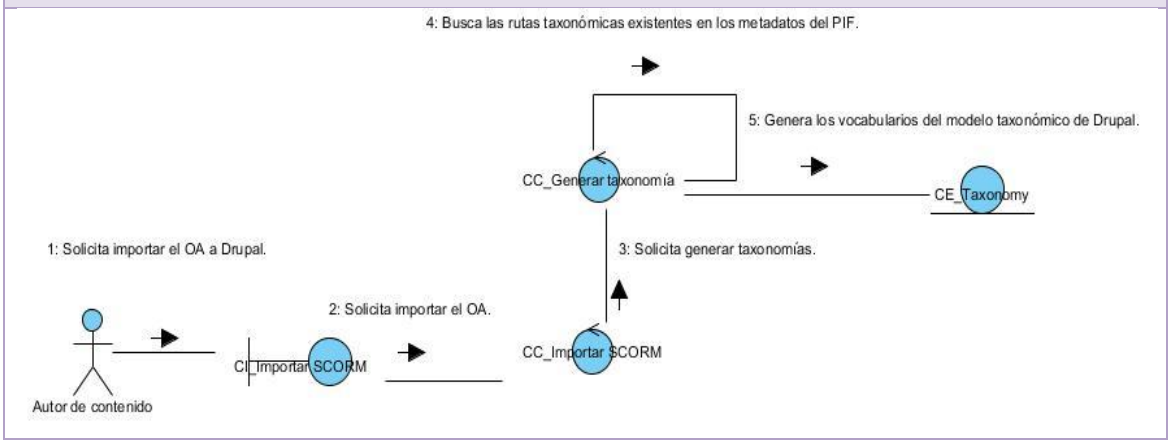


Caso de uso Descargar SCORM

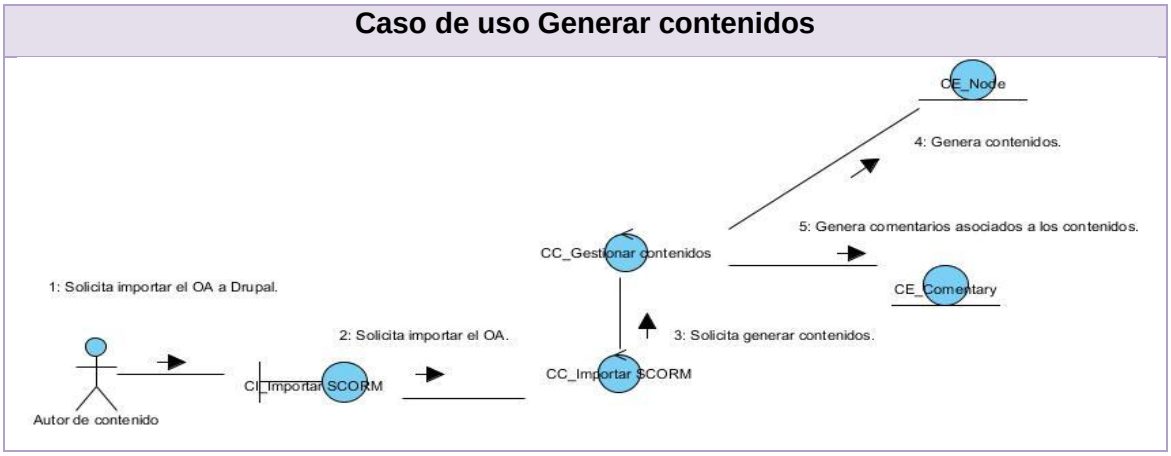


Anexo 3. Diagramas de colaboración

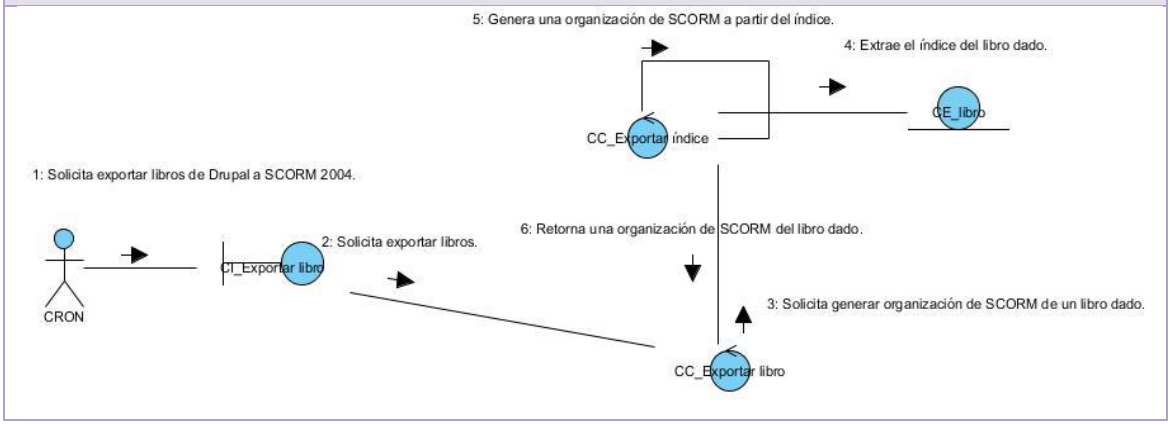
Caso de uso Generar taxonomías



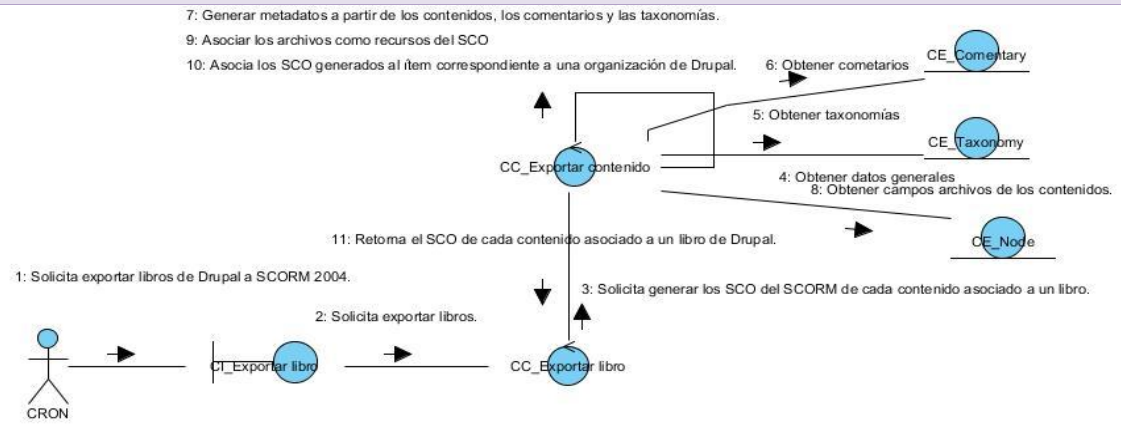
Caso de uso Generar contenidos



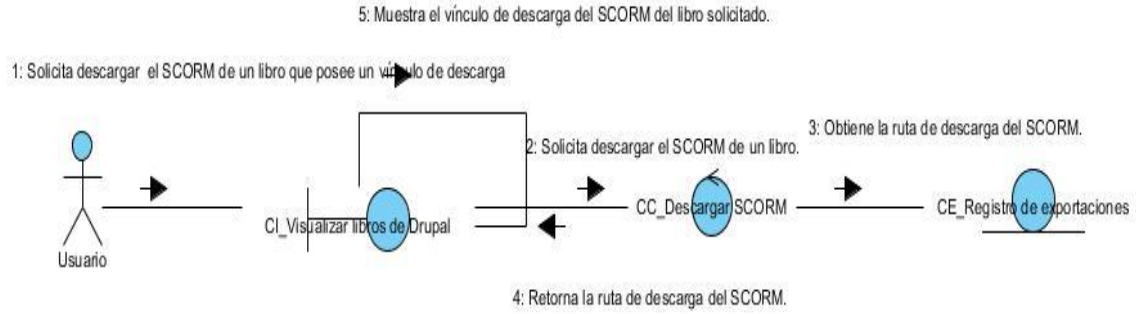
Caso de uso Exportar índice



Caso de uso Exportar contenidos

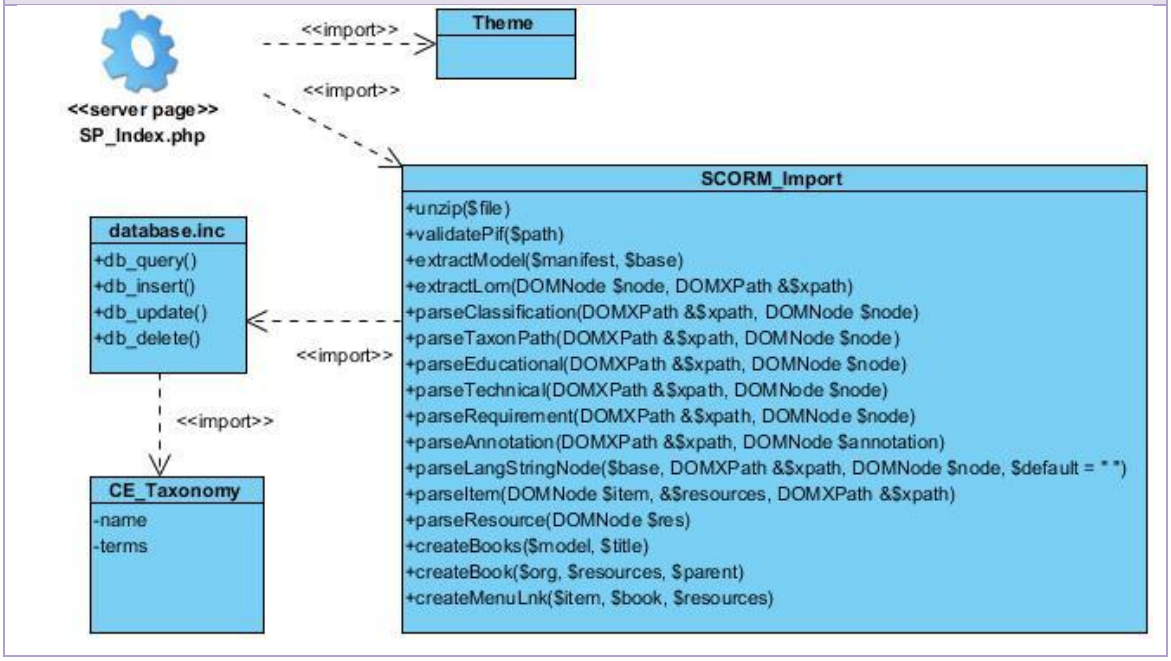


Caso de uso Descargar SCORM

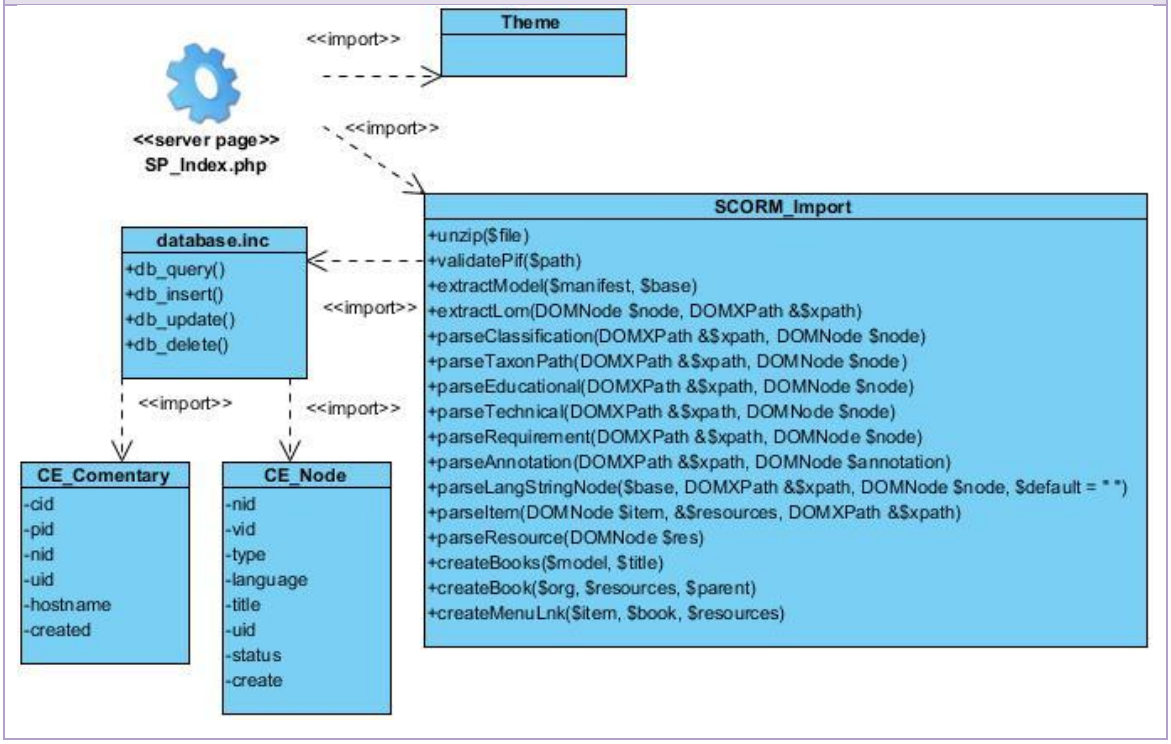


Anexo 4. Diagramas de clases del diseño.

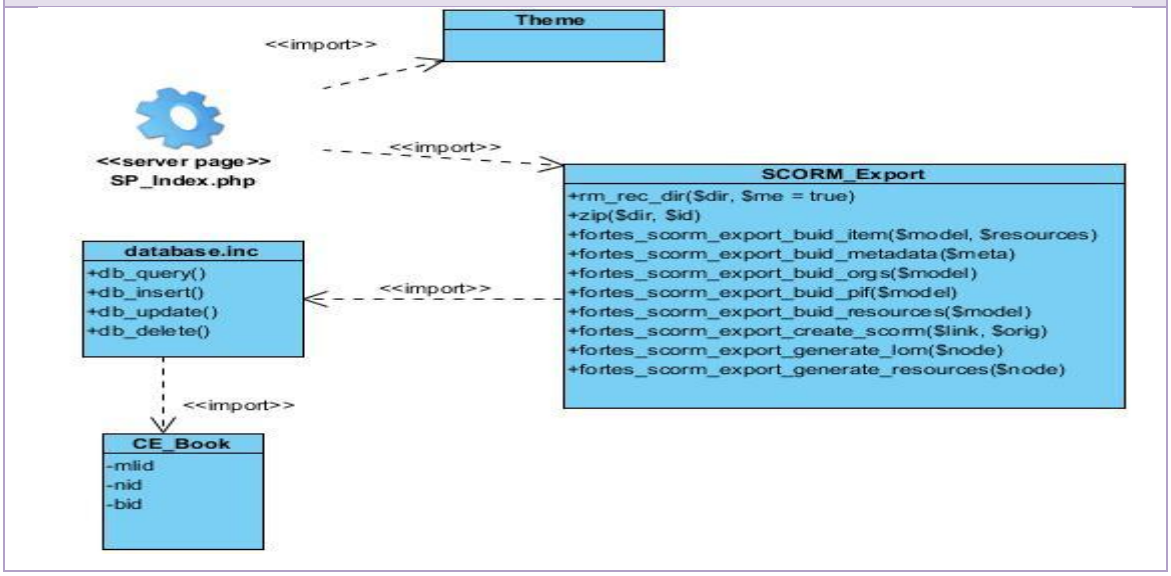
Caso de uso Generar taxonomías



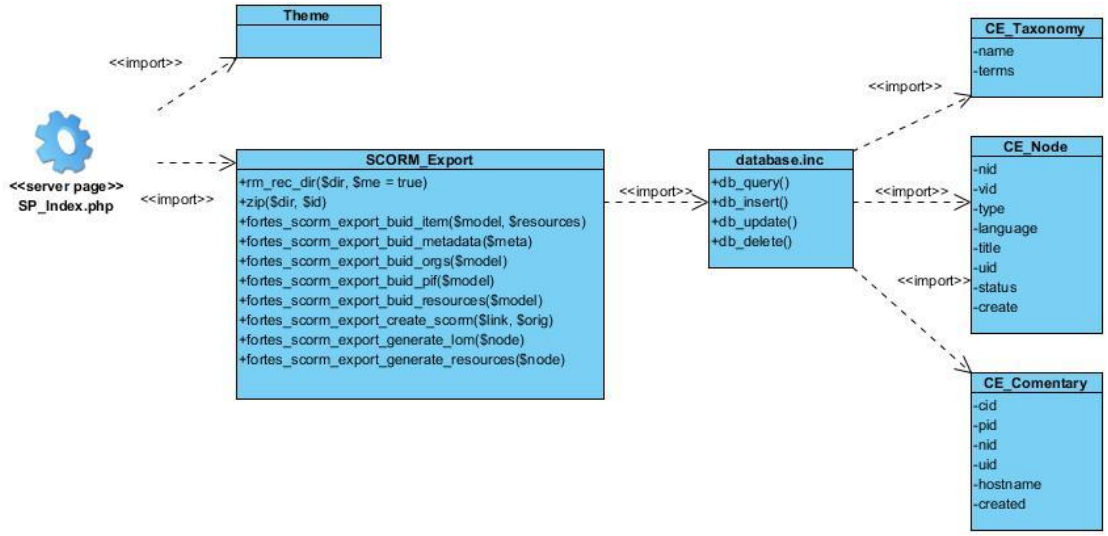
Caso de uso Generar contenidos



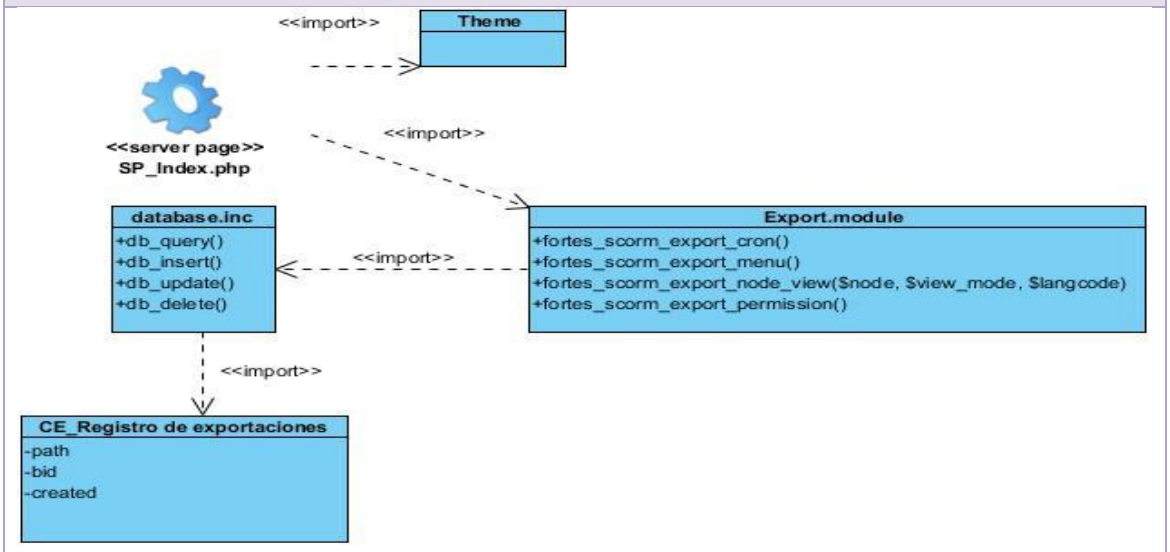
Caso de uso Exportar índice



Caso de uso Exportar contenidos



Caso de uso Descargar SCORM



Anexo 5. Procesos de pruebas

Resultados de la ejecución final de las pruebas unitarias:

MESSAGE	GROUP	FILENAME	LINE	FUNCTION	STATUS
Exito: el timestamp del primer comentario es **1325404800**	Other	fortes_scorm_import_unit.test	135	FortesScormImportUnitTestCase->doAnnotation()	✓
Exito: la entidad del segundo comentario es **pedro alcala**	Other	fortes_scorm_import_unit.test	137	FortesScormImportUnitTestCase->doAnnotation()	✓
Correcto: el title del oa es **testing oa**	Other	fortes_scorm_import_unit.test	113	FortesScormImportUnitTestCase->doLangString()	✓
Correcto: el title del oa es **DESCRIPTION**	Other	fortes_scorm_import_unit.test	119	FortesScormImportUnitTestCase->doLangString()	✓
Correcto: el title del oa es **test case 2 title**	Other	fortes_scorm_import_unit.test	113	FortesScormImportUnitTestCase->doLangString()	✓
Correcto: el title del oa es **test case 2 description**	Other	fortes_scorm_import_unit.test	119	FortesScormImportUnitTestCase->doLangString()	✓
Correcto: el title del oa es **test case 3 title**	Other	fortes_scorm_import_unit.test	113	FortesScormImportUnitTestCase->doLangString()	✓
Correcto: el title del oa es **test case 3 description**	Other	fortes_scorm_import_unit.test	119	FortesScormImportUnitTestCase->doLangString()	✓
Correcto: estructura del arreglo devuelto para **Technical**	Other	fortes_scorm_import_unit.test	102	FortesScormImportUnitTestCase->doTechnical()	✓
Correcto: estructura del arreglo devuelto para **Technical**	Other	fortes_scorm_import_unit.test	102	FortesScormImportUnitTestCase->doTechnical()	✓
Correcto: estructura del arreglo devuelto para **Technical**	Other	fortes_scorm_import_unit.test	102	FortesScormImportUnitTestCase->doTechnical()	✓
Correcto: estructura de requerimiento: **browser: opera Min Ver: 1.2, Max ver: 3.5 Or operating system: any Min Ver: 1.5, Max ver: 2.6**	Other	fortes_scorm_import_unit.test	90	FortesScormImportUnitTestCase->doRequirement()	✓

Criterio para clasificar no conformidades:

Importancia de la NC	Descripción	Observación
3	Alta	• Errores que impidan llegar al final del flujo básico. (liberación) • Errores en la interpretación del proceso de negocio que impediría el correcto funcionamiento de la aplicación para la consecución del fin. (aceptación del cliente)
2	Media	• Errores que impidan llegar al final en un flujo alterno
1	Baja	• Errores de validación y ortografía