Universidad de las Ciencias Informáticas

Facultad 4

Componente sobre el marco de trabajo Xalix, para el trabajo colaborativo en la creación de recursos educativos

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

Autor: Roylan Tumbarell Quintana

Tutor: Ing. Osvaldo Ernesto Stable Vilchez

Co-tutor: Ing. Yenima Hernández Orozco

La Habana, Junio del 2016

DECLARACIÓN DE AUTORÍA

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

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Roylan Tumbarell Quintana

Ing. Osvaldo Ernesto Stable Vilches

Ing. Yenima Hernández Orozco

_____ _____

Resumen

La introducción de las Tecnologías de la Información y las Comunicaciones (TIC) en la educación trae consigo el uso de diferentes recursos educativos digitales que contribuyan a elevar el desempeño de los docentes y el poder receptivo de los estudiantes. Estos recursos son creados por los docentes mediante las Herramientas de Autor, para lo cual no se requiere de un amplio conocimiento de informática. La Universidad de las Ciencias Informáticas cuenta con una Herramienta de Autor denominada CRODA que contiene funcionalidades que permiten crear recursos educativos colaborativamente, pero como se encuentra en el proceso de integración en la arquitectura basada en componentes Xalix sus funcionalidades no se acoplan directamente, debido a que no se encuentran bajo el sistema de integración creado para la arquitectura. Para darle solución al problema se decide crear un componente sobre el marco de trabajo Xalix que esté orientado al trabajo colaborativo en la creación de recursos educativos. En pos de desarrollar el componente se estudiaron los sistemas LAMS, TeamBox y COM8s, los cuales están orientados a la realización de tareas colaborativamente, además se utilizaron las herramientas y lenguajes de programación que propone el equipo de desarrollo de la arquitectura Xalix. Como resultado se obtiene un componente integrado en el marco de trabajo Xalix, que posee funcionalidades orientadas al trabajo colaborativo en la creación de recursos.

Palabras clave

arquitectura, autor, componente, CRODA, educativo, herramienta, integración, recurso, Xalix

Índice

Índice de ilustraciones

Índice de tablas

Introducción

Nos encontramos en tiempos donde las personas realizan un extraordinario uso de los dispositivos digitales que permiten y ayudan al intercambio de información, comunicación y conocimiento, los cuales con el desarrollo de la ciencia y la tecnología han evolucionado en cuanto a disponibilidad, accesibilidad y utilidad.

Estos avances tecnológicos son una de las principales causas de los cambios que se han ocasionado en muchas de las esferas sociales como la cultura, el deporte, la medicina, la educación entre otras que poseen un vínculo con la tecnología. En la educación un cambio destacado es el diseño metodológico mediante el cual se desarrolla el proceso de enseñanza-aprendizaje. Proceso mediante el cual un docente o profesor transmite el conocimiento a los estudiantes. El conocimiento que se involucra puede ser creado o reproducido por el docente, mediante el uso de diferentes recursos, dígase libros, documentos, revistas, mapas, publicaciones entre otros.

La introducción de las TIC en la educación, como uno de estos recursos, hace que la utilización de las herramientas tradicionales se considere una opción casi obsoleta. Puesto que ofrecen la ventaja de que la información se encuentra con un mayor grado de organización y en formato digital, permitiendo una mejor reutilización y modificación.

Los recursos en formato digital, en la educación, también se denominan recursos educativos, pero estos se diferencian de los tradicionales en que se ajustan a la nueva tendencia por la cual está transitando la sociedad actual, la cual es la utilización de los dispositivos electrónicos digitales en la realización de tareas comunes y cotidianas.(1)

Algunos cambios que se evidencian en el diseño metodológico de la educación, son los distintos enfoques que contiene. Los cuales, van orientados a la concepción del aprendizaje constructivista, donde el estudiante pasa a ser el eje central del proceso enseñanza-aprendizaje, y el profesor solo un mediador entre el conocimiento y el estudiante. Estos enfoques permiten que surja un aumento en el desempeño del docente y el poder receptivo de los estudiantes, debido a que cada escolar debe crear el conocimiento mediante sus propias experiencias. El modelo Entorno de Aprendizaje Constructivista creado por Jonassen y Rorher-Murphy en 1999, es un ejemplo de los nuevos enfoques, este modelo apunta que, "… el conocimiento sea elaborado individualmente y socialmente por los estudiantes, a través de sus propias experiencias y representaciones del mundo y sobre la base de los conocimientos declarativos ya conocidos." (2).

Los nuevos recursos educativos, a pesar de ser tan completos en cuanto a contenido, su uso depende de otras herramientas que permitas su creación, almacenamiento y hasta el mismo uso como un material, ya que estos no son elementos tangibles que se pueden colocar sobre una mesa. Por ello, también surgen con el objetivo de realizar la gestión de los mismos, la Herramienta de Autor, que le permite a una persona sin conocimientos avanzados en informática, crear un recurso educativo de manera rápida y sencilla; los repositorios de Objetos de Aprendizaje, que son las aplicaciones encargadas del almacenamiento de los recursos; y las plataformas virtuales o plataformas de *e-learning*, donde el *e-learning* según el Centro de Formación Permanente (CFP) de la Universidad de Sevilla, es el "procesos de enseñanza-aprendizaje que se llevan a cabo a través de internet" (4) y las plataformas virtuales son " un espacio virtual de aprendizaje orientado a facilitar la experiencia de capacitación a distancia " (6), en el cual se pueden montar y hacer uso de los recursos educativos.

La Universidad de las Ciencias Informáticas (UCI), como una de las instituciones con mayor nivel de informatización del país, hace uso de *e-learning* y recursos educativos, con el objetivo de lograr un mayor nivel en el proceso de enseñanza-aprendizaje. Dentro de la universidad se encuentra el Centro de Tecnologías para la Formación (FORTES), el cual se encarga de brindar servicios y productos orientados a dar soporte a la formación mediante el uso de las TIC. En este centro se desarrollan varios sistemas de gestión de recursos educativos, entre estos se encuentra ZERA como plataforma virtual, RHODA para el almacenamiento y gestión del desarrollo de estos recursos y la Herramienta de Autor CRODA para la creación de los mismos.

FORTES posee también un marco de trabajo o arquitectura de desarrollo creada en el mismo centro, la cual se denomina Xalix. La principal característica de esta arquitectura es que basa su funcionamiento en componentes, debido a que extiende del concepto de *bundle* de Symfony 2, donde se toman los mismos como *plugin*, y crea un sistema de gestión que incluye la instalación y desinstalación, con el objetivo de obtener una mayor escalabilidad y flexibilidad en los productos desarrollados sobre este marco de trabajo.

CRODA está desarrollado bajo las especificaciones IMS-LD[1], cuenta con tres versiones y en cada una de ellas se pueden encontrar funcionalidades básicas que permiten la creación de recursos educativos de manera simple y rápida, cumpliendo con los estándares y normas que les permite ser reutilizables. Estas

[1] IMS-LD es un lenguaje de modelado educativo que tiene como objetivo definir formalmente una estructura semántica para anotar los procesos de enseñanza y aprendizaje, y así convertirlos en entidades reutilizables entre diferentes cursos y aplicaciones.

versiones poseen arquitecturas acordes con el desarrollo, evolución y necesidades del centro de producción.

Todos los productos pertenecientes al centro, junto a la nueva arquitectura basada en componente, se encuentran en constante evolución, permitiendo estar a la altura de las necesidades de desarrollo del presente avance tecnológico que se evidencia en la rama del desarrollo de software. El principal cambio que se lleva a cabo en el centro es acoplar todos los sistemas que pertenecen al mismo con el nuevo marco de trabajo Xalix, pero existen diferencias en cuanto a compatibilidad, dadas las herramientas utilizadas para el desarrollo de estos.

La herramienta de autor web CRODA 3.0, está desarrollada sobre Symfony 2, jQuery, Postgre y Bootstrap, cumpliendo de esta forma con lo necesario para formar parte de la estrategia marcaria y la cartera de productos de la Universidad de las Ciencias Informáticas. Esta última versión, posee un conjunto de funcionalidades que permiten la creación de los recursos educativos de manera colaborativa las cuales permiten que los usuarios formen grupos de trabajo en el cual pueden desarrollar recursos educativos en conjunto y de una forma organizada, obteniendo recursos mejor elaborados y con mayor calidad.

Por ello, permitir realizar recursos educativos de manera colaborativa es una de las principales funcionalidades que se desea incluir a la nueva arquitectura del centro FORTES, dado que los grupos de trabajo colaborativo, tienen como objetivo que se mantenga una constante comunicación entre los miembros, y dejar que estos aporten ideas innovadoras y creativas basadas en sus conocimientos previos y sus habilidades cognitivas. Pero como inconveniente esta funcionalidad no logra acoplarse directamente en la arquitectura basada en componente, debido a que no se ajusta de forma independiente a la arquitectura, ya que su creación se realizó en un sistema diferente y no se encuentra bajo el sistema de integración creado para dicha arquitectura.

Por lo anteriormente expuesto se toma como **problema de investigación:** ¿Cómo contribuir a la participación conjunta entre los usuarios para la creación de recursos educativos sobre el marco de trabajo Xalix?

Se define como **objeto de estudio:** proceso de creación de recursos educativos mediante el trabajo colaborativo entre usuarios.

El **campo de acción** está enfocado al componente sobre el marco de trabajo Xalix, para el trabajo colaborativo en la creación de recursos educativos.

La presente investigación tiene como **objetivo general:** desarrollar un componente sobre el marco de trabajo Xalix, para el trabajo colaborativo en la creación de recursos educativos.

El objetivo general se desglosa en los siguientes **objetivos específicos**:

- Elaborar el estado del arte orientado a la creación de recursos educativos colaborativamente.
- Realizar el análisis y diseño del componente para trabajo colaborativo en la creación de recursos educativos.
- Implementar el componente para el trabajo colaborativo en la creación de recursos educativos sobre el marco de trabajo Xalix.
- Validad mediante pruebas el componente para el trabajo colaborativo en la creación de recursos educativos.

Como punto de partida para el desarrollo de la investigación se plantean las siguientes **preguntas científicas**:

- ¿Qué tecnologías y herramientas son necesarias para el desarrollo de recursos educativos de forma colaborativa?
- ¿Cómo especificar las funcionalidades que debe poseer un componente para el trabajo colaborativo en la creación de recursos educativos, que se ajusten a la arquitectura basada en componentes Xalix?
- ¿Cómo incorporar las funcionalidades requeridas para el trabajo colaborativo en la creación de recursos educativos en la arquitectura basada en componentes Xalix?
- ¿Es seguro el uso de la propuesta de solución desarrollada?

En el desarrollo de la investigación se hizo uso de los siguientes **métodos investigativos**:

Métodos teóricos:

- **Histórico-lógico:** permitió realizar un análisis de la evolución de los sistemas similares en cuanto a su estructura y funcionamiento, obteniéndose una síntesis de su evolución a lo largo del tiempo.

- **Analítico-sintético:** permitió el estudio de los principales conceptos y definiciones asociados al problema a resolver dando lugar a la extracción de los elementos importantes a tener en cuenta en el desarrollo de la solución propuesta.
- **Inductivo-deductivo:** posibilitó el estudio de los principales motivos del surgimiento y los estándares usados en los sistemas que tributan al trabajo colaborativo, obteniéndose conclusiones que marcaron un principio de desarrollo y la identificación de alternativas factibles a incorporar en la propuesta de solución.

Método empírico:

- **Observación:** mediante este método se analizaron los principales comportamientos surgidos a partir de los sistemas de trabajo colaborativo en la web, además se utilizó en la evaluación de los resultados de las pruebas realizadas a la solución propuesta.

El documento constará de tres capítulos, en los que se describe detalladamente el proceso de desarrollo de la tesis. A continuación, se muestra un resumen de los capítulos:

Capítulo 1. Fundamentación teórica: En este capítulo se hará una descripción de todos los referentes teóricos, conceptos y fundamentos generales que son de interés y sirven de apoyo para el desarrollo de la solución propuesta. Se realiza un análisis de soluciones similares, lo que ofrecerá como resultado una explicación de sus principales características y funcionalidades. Se describirán las diferentes herramientas y lenguajes de programación necesarias para el proceso de creación del componente de distribución de responsabilidades sobre el marco de trabajo Xalix. También se realizará una descripción de la metodología a usar en el proceso de ingeniería de software asociado a la investigación.

Capítulo 2. Propuesta de solución: Se describe el proceso de desarrollo de la propuesta de solución especificando sus características principales. Se expone el modelo conceptual y los requisitos funcionales y no funcionales de la aplicación a implementar. Y se exponen los artefactos que se generan a partir de la metodología de desarrollo de software utilizada.

Capítulo 3. Implementación y prueba: En este capítulo se presentarán las diferentes clases a utilizar en la implementación de la solución propuesta. Además de las diferentes pruebas a las cuales será sometido el componente para la colaboración en la creación de recursos educativos.

Capítulo 1: Fundamentación teórica

En el presente capítulo se analizan los conceptos necesarios para la realización de la solución propuesta. Se realiza un estudio de soluciones similares con el objetivo de establecer pautas y puntos de partidas en el desarrollo de la investigación, además se creará una concepción de las principales funcionalidades que formarán parte del componente para el trabajo colaborativo en la creación de recursos educativos. También se realiza un resumen de las herramientas y tecnologías que se usarán en el desarrollo de la solución propuesta.

Conceptos asociados

Componente

Es un elemento de un sistema de software que ofrece un conjunto de servicios o funcionalidades a través de interfaces definidas. Según la OMG[2] (Object Management Group) un componente es "una unidad modular con interfaces bien definidas que es reemplazable dentro del contexto"(8).

En el marco de trabajo donde se desarrollará la aplicación un componente es tratado como un plugin, el cual puede ser instalado o desinstalado independientemente de los demás componentes del sistema, sin afectar la integridad de los mismos.

Trabajo colaborativo

Según el catedrático de la Facultad de Derecho de la Universidad de Harvard Yochai Benkler el trabajo colaborativo es "el sistema de producción, distribución y consumo de bienes de información que se caracteriza por acciones individuales descentralizadas, ejecutadas a través de medios ampliamente distribuidos y ajenos al mercado y a sus estrategias"(3). El trabajo colaborativo está basado en los principios del bien común y del altruismo, que guían un proyecto y comparten quienes, generalmente voluntariamente, forman parte en él, que son expertos o al menos conocedores de la información que ponen a disposición de quien la quiera usar.

[2] OMG es un consorcio de estándares de tecnología sin fines de lucro fundado en 1996.

Recursos educativos

Los recursos educativos son cualquier material que se ha elaborado con la intención de facilitar al docente su función y a su vez la del alumno (11). Con la aparición de las TIC en la educación los recursos educativos obtienen otras denominaciones como por ejemplo recursos educativos abiertos, los cuales se ajustan al uso de estas herramientas tecnológicas en el proceso de enseñanza-aprendizaje.

Los recursos educativos abiertos, según la UNESCO son "materiales de enseñanza, aprendizaje o investigación que se encuentran en el dominio público o que han sido publicados con una licencia de propiedad intelectual que permite su utilización, adaptación y distribución gratuitas"(12).

Tecnologías de la Información y las Comunicaciones

Las TIC son el principal mediador entre los recursos educativos y el objetivo fundamental de los mismos. Se imaginan como el universo de dos conjuntos, representados por las tradicionales Tecnologías de la Comunicación, constituidas principalmente por la radio, la televisión y la telefonía convencional y por las Tecnologías de la información, caracterizadas por la digitalización de las tecnologías de registros de contenidos.(14)

Las TIC son aquellas herramientas informáticas que procesan, almacenan, recuperan y presentan información. Para cualquier uso que se haga de las TIC en la educación, se catalogan como medios y no fines, es decir, son herramientas y materiales que facilitan el aprendizaje y el desarrollo de habilidades. Mediante estas herramientas se puede acceder a los diferentes entornos virtuales, en los cuales se encuentran los recursos educativos.

E-learning

El *e-learning* es también conocido como *Learning Management System* (LMS) o entorno virtual. Es un término que varios autores e instituciones definen según un criterio que se basa en sus propias necesidades. Dado que es tan amplia la colección de definiciones existentes se tomará como referencia la emitida por los autores Anju Relan y Benjan B. Gillani, donde expresan que es "la aplicación de un repertorio de estrategias instruccionales orientadas cognitivamente, y llevadas a cabo en un ambiente de aprendizaje constructivista y colaborativo, utilizando los atributos y recursos de Internet" (5). Estos autores especifican que los *e-learning* son un conjunto de "*estrategias instruccionales orientadas cognitivamente*", lo cual quiere decir que son un grupo de contenidos educativos con una organización secuencial, que

poseen un objetivo educativo bien definido y que se orientan específicamente a un grupo de estudiantes. Otra definición sobre los LMS es la que proyecta el e-ABC donde presentan que es “un espacio virtual de aprendizaje orientado a facilitar la experiencia de capacitación a distancia, tanto para empresas como para instituciones educativas” (7). Este concepto es más abarcador, dado que explica que en cualquier tipo de institución se encuentran actores que se involucran en el proceso de aprendizaje mediante internet, el cual permite acceder a los distintos espacios virtuales; entiéndase por espacio virtual, entornos o plataformas virtuales.

Los sistemas *e-learning* utilizan herramientas de tratamiento multimedia para establecer una comunicación no formal entre los actores involucrados, quienes pueden ser varias personas que desean compartir información, dos o más empresas o simplemente un solo usuario que lo utiliza para nutrirse de conocimiento sobre determinado contenido. En estos sistemas la información ya ha sido predefinida y compartida por otros, por ello se han convertido en herramientas básicas para el desarrollo del proceso enseñanza-aprendizaje en los centros educativos, brindando la posibilidad de fomentar la interacción entre profesores, estudiantes y contenido, además de permitir la realización de evaluaciones, el intercambio de archivos, la participación en foros, chats y una amplia gama de funcionalidades.

Herramientas de Autor

Las herramientas de autor son aplicaciones informáticas que tienen como principales funcionalidades la creación, publicación y gestión de los materiales educativos en formato digital mediante en tratamiento de multimedia. Según el proyecto Babeltic las herramientas de autor son “aplicaciones informáticas que facilitan la creación, publicación y gestión de los materiales educativos en formato digital a utilizar en la educación a distancia mediada por las TIC”.(9)

Las herramientas de autor proveen módulos desde los cuáles se pueden organizar actividades o interconectar pequeños componentes para adecuar el contenido a los objetivos, los conocimientos y habilidades que se busque desarrollar. Brindan la posibilidad de diseñar en módulos, sin necesidad de conocimientos de programación y a partir de plantillas.

Generalmente son herramientas de carácter multimedia que permiten combinar documentos digitales, imágenes, sonidos, videos y actividades interactivas para crear recursos educativos que pueden insertarse en entornos virtuales de aprendizaje o plataformas educativas. Algunos de estos programas utilizan

metadatos y permiten empaquetar el contenido según estándares como SCORM o IMS para asegurar su compatibilidad con distintos tipos de entornos virtuales.(19)

SCORM

SCORM es una colección de especificaciones y estándares creados para regular la creación de contenidos educativos y plataformas *e-learning*, fue creada por *Advanced Distributed Learning* (ADL), la cual es una organización surgida en 1999 como resultado de la colaboración entre la Industria Aeronáutica de los Estados Unidos (AICC) y el departamento de Defensa de los Estados Unidos; esta colección de estándares surge producto a que los contenidos educativos se creaban con especificaciones propietarias, por lo que no era posible establecer un intercambio de los mismos.(21)

Los principales requerimientos que SCORM busca satisfacer son:

Usabilidad: es la capacidad de acceder a un componente de enseñanza desde un sitio distante a través de las tecnologías web, así como distribuirlos a otros sitios.

Adaptabilidad: es la capacidad de personalizar la información en función de las necesidades de las personas y organizaciones.

Durabilidad: es la capacidad de resistir a la evolución de las tecnologías sin necesitar una reconcepción, una reconfiguración o una reescritura de código.

Interoperabilidad: es la capacidad de utilizarse en otro emplazamiento y con otro conjunto de herramientas o sobre una plataforma de componentes de enseñanza desarrolladas dentro de un sitio, con un cierto conjunto de herramientas o sobre una cierta plataforma.

Reusabilidad: es la flexibilidad que permite integrar componentes de enseñanza dentro de múltiples contextos y aplicaciones.

IMS

Son métricas que se ocupan de describir y codificar el diseño pedagógico, es decir las metodologías educativas implícitas en un proceso de enseñanza, de forma que sean procesables por un LMS. Existen varias especificaciones IMS entre las que se encuentran IMS-CP (del inglés, *IMS Content Packaging*), el cual define contenidos de aprendizaje y actividades de evaluación que se pueden ser distribuidos a través de sistemas de gestión de contenidos; IMS-LTI (del inglés, *IMS Learning Tool Interoperability*), que es un estándar técnico de integración de aplicaciones para aprendizaje, IMS-QTI (del inglés, *IMS Question & Test*

Interoperability), que define un formato para la representación de contenidos y resultados de evaluaciones educativas e IMS-LD (del inglés, *IMS Learning Design*) el cual define una estructura para representar y codificar diseños de aprendizaje.(23)

La especificación IMS-LD es un lenguaje de modelado educativo que tiene como objetivo definir formalmente una estructura semántica para anotar los procesos de enseñanza y aprendizaje, de forma que sean procesables en un LMS y se conviertan en entidades reutilizables entre diferentes aplicaciones. Para ello describe procesos que agrupan las actividades que realizarán alumnos y profesores, indicando en qué momento, con qué recursos didácticos y servicios, y bajo qué condiciones lo harán.

Daniel Burgos especifica que un IMS-LD es "una especificación para definir Unidades de Aprendizaje (UoL), que definen quién, cuándo, dónde, cómo y para qué utiliza una serie de recursos y modela procesos de aprendizaje y de comunicación interactiva" (25). Según Daniel Burgos las UoL son "cualquier recurso o grupo de recursos organizados metodológicamente para definir una experiencia o proceso de aprendizaje" (27). Un resumen de las dos definiciones es que los estándares IMS-LD definen como se crearán las plataformas educativas y como se deben organizar los contenidos dentro de estas.

Para disminuir la complejidad de su implementación se divide en tres niveles denominados A, B y C, donde el primero contiene el vocabulario básico que soporta la diversidad pedagógica, el nivel B incluye propiedades y condiciones para diseñar ambientes de aprendizaje personalizados y modelos colaborativos de aprendizaje, y el nivel C agrega la posibilidad de realizar notificaciones.

Descripción de las soluciones similares

Las soluciones similares son las aplicaciones que comparten características comunes con la propuesta de solución planteada. El estudio de estas soluciones similares brinda la posibilidad de concebir un conjunto de funcionalidades y características que apunten a una posible descripción del componente para el trabajo colaborativo en la creación de recursos educativos.

Las soluciones similares que se identificaron y se escogieron para realizar la investigación son LAMS, TeamBox y Com8s, las cuales se describen a continuación.

LAMS

LAMS (por sus siglas en inglés, *Learning Activity Management System*), es un entorno virtual de enseñanza y aprendizaje, de orientación constructivista para la creación y gestión de actividades educativas. Desde

que se lanzó como software libre en 2004, LAMS se ha centrado en los enfoques constructivistas, enfatizando el contexto, las actividades y el aprendizaje en grupo, que ha llevado a la creación de excelentes secuencias de aprendizaje que son compartidas en la Comunidad LAMS a través de su repositorio. LAMS se puede usar en conjunto con otros LMS como Moodle, Sakai, LRN, WebCT o BlackBoard.

En LAMS los usuarios pueden usar un entorno visual para crear secuencias de aprendizaje y diseños de actividades educativas de manera sencilla e intuitiva. Las mismas pueden ser guardadas, reusadas o compartidas con otros usuarios.

Crear secuencias de actividades educativas es sin duda una de las ventajas que LAMS tiene sobre otros entornos de aprendizaje, debido a que estas secuencias educativas son utilizadas por estudiantes de manera colaborativa, contribuyendo así a la motivación por el aprendizaje y la creatividad.

Algunas de los beneficios que ofrece LAMS son:

- Posee una interfaz visual intuitiva.
- Permite actividades de colaboración en la interacción con el sistema.
- Funciona en la mayoría de los exploradores web y su servidor es multiplataforma.
- Es un software que se distribuye gratuitamente bajo la licencia de la GPL v2.
- Incluye soporte para especificaciones educativas como *IMS Content Packaging*, *IMS Metadata* y *IMS Learning Design*.

Dentro de las desventajas se encuentra que:

- Debe ser montado sobre un servidor.
- Posee dependencias con Java JDK
- Requiere de un gestor de bases de datos para poder funcionar.

TeamBox

Es una herramienta para gestionar proyectos mediante la web, la cual cuenta con una versión gratuita que permite la gestión de hasta 3 proyectos simultáneamente. Dentro de sus características, se encuentra que, permite la invitación de colaboradores a los proyectos para que puedan participar en él realizando tareas. Dentro de estos proyectos los colaboradores pueden compartir información directamente, se pueden compartir archivos mediante un espacio que se le asigna a cada proyecto cuando es creado y también se puede ver el tiempo empleado para realizar cada atarea.

Com8s

Com8s es una herramienta de colaboración surgida en Brasil, disponible en la web. Permite al usuario almacenar, organizar y compartir los contenidos y actividades con grupos de estudio, de trabajo o de personas.

Se trata de una red de contactos, pero con numerosos aplicativos integrados, que nos permiten establecer diferentes espacios colaborativos con diferentes permisos, por ejemplo, un espacio por curso, o determinados accesos para determinados subgrupos dentro de un mismo espacio.

Su objetivo principal es facilitar la vida del usuario, a través de varias características que le permiten trabajar en colaboración, interactuar y comunicarse en tiempo real, optimizando el tiempo de trabajo. Sin embargo, la Com8s puede considerarse como una nueva era de redes de colaboración, con los objetivos de mejorar la relación, la comunicación y el trabajo de individuos o grupos, integrando en un solo lugar, varias aplicaciones web.

Marco de trabajo Xalix

El marco de trabajo Xalix surge en el centro FORTES perteneciente a la facultad 4 de la Universidad de las Ciencias Informáticas, con el objetivo de obtener una homogenización en la arquitectura de los distintos sistemas que se desarrollan en dicho centro.

Xalix es una arquitectura de desarrollo basada en componentes que extiende del concepto de bundle de Symfony, tomándolos como plugins y creando un sistema de gestión de los mismos que incluye la instalación y desinstalación de los mismos. Esta característica otorga independencia a los componentes desarrollados, lo que permite que el mantenimiento o la transformación de uno de ellos no influya en la integridad de los demás componentes.

Otras características que diferencian a Xalix de Symfony son:

- El appKernel, ya no almacena la referencia a los bundles de terceros debido a que esta tarea se realiza mediante el fichero *bundles.ini* ubicado en el directorio *config* de *app*. Para hacer la referencia a los bundles que se instalarán, dentro del fichero *bundles.ini* se debe colocar la siguiente estructura de código: Symfony\Bundle\FrameworkBundle\FrameworkBundle = true Symfony\Bundle\SecurityBundle\SecurityBundle = true, donde el valor que se encuentra después del signo = (*true* o *false*), define si se instala o no el componente.

- El fichero *config.yml* es eliminado del directorio *app/config* y la configuración es controlada por cada plugin.

Para la instalación del marco de trabajo se debe seguir los siguientes pasos:

- Descargar la aplicación desde la dirección https://repo-fortes.uci.cu/mooc/trunk/zera2/ mediante el uso de un cliente Subversion el cual puede ser Tortoise si el sistema operativo es Windows o Kdesvn si es alguna de las distribuciones de GNU/Linux. El cliente Subversion no debe ser específicamente los mencionados.
- Actualizar las dependencias del sistema mediante la ejecución del comando "*php composer.phar update*" en la consola.
- Ejecutar el comando "*xalix:install*".
- Eliminar la *cache*.

Para el desarrollo de sus productos Xalix incluye la siguiente base de tecnologías:

- Lenguaje de programación para el servidor: PHP 5.4.x.
- Lenguaje para el cliente: HTML5.
- Gestor de base de datos: PostgreSQL 9.4.x.
- Librería de CSS: Bootstrap 3.0.0.
- Librería de JavaScript: jQuery v1.10.2 con jQuery UI 1.10.3.

El marco de trabajo Xalix es un completo framework de trabajo que permite hacer uso de los distintos componentes que contiene y adopta una arquitectura basada componentes donde cada elemento puede desacoplarse y evolucionar de manera independiente, lo que conforma un producto genérico una vez ensamblado y posee un mecanismo de integración para los componentes creados.

Metodología de desarrollo de software

La metodología de desarrollo de software es un conjunto de procedimientos y técnicas que se realizan sobre determinadas tareas definidas, además de las distintas fases y etapas por las cuales transita todo proceso de desarrollo de un producto informático. También detalla la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla.

Las metodologías de desarrollo de software están agrupadas en dos grupos, las tradicionales o pesadas que se destinan a controlar el proceso de desarrollo de software extensos, que conlleven un gran cumulo

de documentación y tiempo; y las ágiles que se orientan a los procesos de desarrollo de software que no requieran de mucho tiempo, recurso o integrantes en el equipo de desarrollo.

Metodologías ágiles

Según la especialista en gestión proyectos de ciencia y tecnología de la Universidad Francisco de Paula Stander y el master en ciencias Mauricio Rojas Contreras las metodologías ágiles "están orientadas para proyectos que necesitan de una solución a la medida, con una elevada simplificación sin dejar de lado el aseguramiento en la calidad del producto"(29). Debido a que el desarrollo del componente para la colaboración en la creación de recursos educativos debe ser un proceso sencillo pero que brinde un producto con calidad, la metodología a utilizar será ágil.

Dentro de las metodologías agiles se encuentran Programación Extrema (XP, por sus siglas en inglés), SCRUM, *Lean Software Development* (LSD), *Crystal Clear*, *Agile Unified Process* (AUP), entre otras. Para el desarrollo de la solución propuesta se hará uso de la variación de la metodología ágil AUP denominada AUP-UCI, creada en la Universidad de las Ciencias Informáticas.

Metodología AUP y AUP-UCI

La metodología de desarrollo de software AUP es una versión simplificada del *Rational Unified Process* (RUP) que es una metodología de desarrollo tradicional. AUP combina el proceso unificado tradicional con las técnicas ágiles. Este se centra en el desarrollo de productos mediante iteraciones, en las cuales se da cumplimiento a cierta cantidad de casos de uso, permitiendo la detección temprana de riesgos y un adecuado control de cambios. En esta metodología se definen cuatro fases para el desarrollo de un producto, estas fases son: Inicio, Elaboración, Construcción y Transición.(31)

La metodología AUP-UCI surge con el objetivo de homogeneizar el desarrollo productivo de la Universidad de las Ciencias Informáticas. Esta metodología define tres fases que están orientadas al ciclo de vida del desarrollo de software en la institución, la primera fase es Inicio al igual que AUP, pero con un enfoque diferente debido a que en esta fase se realizan las actividades relacionadas con la planeación del proyecto, se define el alcance del mismo y se decide si se ejecuta o no el proyecto; la segunda fase es Ejecución, en la cual se agrupan las tres fases restantes de AUP, es aquí donde se ejecutan las actividades requeridas para el desarrollo del software como el modelado del negocio, la obtención de los requisitos, la elaboración de la arquitectura y el diseño, la implementación y la liberación del producto; y la fase de Cierre que es donde se analizan los resultados del proyecto, su ejecución y las actividades formales de cierre del proyecto.

Las herramientas y tecnologías son un conjunto de elementos necesarios para el desarrollo de un producto informático. Para el desarrollo de los productos sobre el marco de trabajo Xalix, definen un

Herramienta de modelado

Las herramientas de modelado UML (*Unified Modeling Language*), son aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software, reduciendo el costo de las mismas en términos de tiempo, mediante la posibilidad de realizar un diseño del proyecto, cálculo de costos, implementación de parte del código a través del uso de un diseño dado, compilación automática, documentación o detección de errores.

Visual Paradigm 8.0

Visual Paradigm es la herramienta de modelado que se recomienda usar en los proyectos que tengan como objetivo un proceso de ingeniería de alta calidad, dado todas las ventajas que proporciona.

Entre las principales características que destacan en el Visual Paradigm están la posibilidad que le brinda al equipo de desarrollo de un proyecto, de generar el mapeo de objetos relacionales para diferentes lenguajes de programación como Java, .Net y PHP. También permite el trabajo con UML 2.1, mediante el cual se puede generar un entorno de modelado visual en el que se reúnen las necesidades de software, tecnología y comunicación. Otra característica, que es a su vez una ventaja, es que posee integración con el entorno de desarrollo integrado NetBean.

Lenguajes de programación

Los lenguajes de programación son un conjunto de símbolos que se rigen por determinadas reglas, que tienen como objetivo crear ordenes e instrucciones, que utiliza un programador para comunicarse con un ordenador y mediante este indicarle al mismo como y que acción tiene que realizar.

Según Sara Álvarez el lenguaje de programación “es un conjunto de reglas semánticas, así como sintácticas que los programadores usan para la codificación de instrucciones de un programa o algoritmo de programación”. En la mayoría de estos lenguajes de programación las instrucciones que se escriben siguen una estructura lógica, y contienen palabras y frases generalmente del inglés, que facilitan la compresión e incluso se pueden situar en determinado orden y formar una sentencia con toda la concordancia que conlleva una instrucción formal en la comunicación persona-persona.

Para el desarrollo del componente para el trabajo colaborativo en la creación e recursos educativos se utilizarán los lenguajes de programación definido por el equipo de desarrollo de la arquitectura Xalix, estos son PHP 5, HTML 5, Java Script, CSS y XML los cuales se describen a continuación.

PHP 5.4

PHP es un acrónimo recursivo que significa PHP *HyperText Prepocessor*. Es un lenguaje de código abierto usado del lado del servidor que se puede incrustar directamente dentro del código HTML en lugar de llamar a un archivo externo que procese los datos. Lo que distingue a PHP de los lenguajes del lado del cliente es que este se ejecuta en el servidor generando HTML y enviándolo al cliente, de modo que el cliente recibe el resultado de ejecutar el script[3], pero desconoce el código asociado.

Entre las principales ventajas de PHP5 se encuentra la versatilidad, la cual le permite ser usado en disimiles tipos de aplicaciones y funcionalidades web. Es un lenguaje fácil de aprender debido a que sus sintaxis son fáciles y también es de código abierto lo cual hace que la cooperación entre los usuarios lo lleven a ser un lenguaje evolucionado. También permite técnicas de programación orientada a objetos, es multiplataforma y posee conectividad con MySQL que es un sistema gestor de bases de datos creado bajo la licencia *General Public License.*

HTML 5.0

HTML 5 es un lenguaje de marcado para la creación de páginas web. Es el lenguaje de programación básico dentro de la W3(*World Wide Web*), que especifica dos variantes de sintaxis para HTML, la clásica HTML (text/html), conocida como HTML5, y una variante XHTML conocida como XHTML5. Contiene un conjunto amplio de tecnologías que permiten mayor alcance y diversidad en los sitios y las aplicaciones web. Estas tecnologías están divididas en grupos que están clasificados según su función, estos grupos son:

Semántica: permite describir con mayor recisión cuál es su contenido.

Conectividad: permite comunicarse con el servidor de formas nuevas e innovadoras.

Sin conexión y almacenamiento: permite a las páginas web almacenar datos localmente del lado del cliente y operar sin conexión de manera eficiente.

[3] Un script es un programa simple que por lo regular se almacena en un archivo de texto plano.

Multimedia: otorga un excelente soporte para utilizar contenido multimedia como lo son audio y video.

Gráfico y efecto 2D/3D: proporciona una amplia gama de nuevas características que se ocupan de los gráficos en la web como lo son canvas 2D[4], WebGL[5], SVG[6], etc.

Rendimiento e Integración: proporciona una mayor optimización de la velocidad y un mejor uso del hardware.

Acceso al dispositivo: proporciona APIs[7] para el uso de varios componentes internos de entrada y salida de nuestro dispositivo.

CSS3: nos ofrece una nueva variedad de opciones para hacer diseños sofisticados.

HTML5 brinda facilidades en la edición de aplicaciones web, proporcionándole suficiente información al navegador para que este sepa como mostrar una determinada página web.

CSS 3

CSS, (de sus siglas en inglés, *Cascading Style Sheets*), es el lenguaje donde se estructura toda la información que se envía al ordenador para que este muestre una página web bien estructurada y con los formatos de multimedia (negrita, colores, efectos, tipos de letra, etc.) que la hacen tan llamativa. Aplica reglas de estilo a los elementos HTML quedando de esta manera separada de la estructura HTML.

XML

XML (de sus siglas en inglés, *Extensible Markup Language*), es un lenguaje desarrollado por el *World Wide Web Consortium* (W3C), utilizado para almacenar datos en forma legible. Es un lenguaje de etiquetas, es decir, cada paquete de información está delimitado por etiquetas como se hace también en el lenguaje HTML, pero XML separa el contenido de la presentación. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones deben integrar información o comunicarse entre sí de una manera segura y fiable

---

[4] Canvas 2D permite el tratamiento multimedia en segunda dimensión.
[5] WebGL son estándares que permiten mostrar gráfico en 3D acelerados por hardware sin la necesidad de plugins.
[6] Scalable Vector Graphics (SVG), se traduce en gráficos basados en vectores escalables. Define un formato gráfico basado en XML para crear archivos vectoriales en 2D.
[7] Las APIs son un conjunto de códigos y especificaciones que las aplicaciones pueden seguir para comunicarse entre ellas.

Entre las ventajas de XML podemos encontrar que es extensible o sea después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación alguna; otra ventaja es que el analizador del código es un componente estándar, no es necesario crear un analizador específico para cada versión de lenguaje XML, esto posibilita el empleo de cualquiera de los analizadores disponibles, así de esta manera se evitan bugs[8].

XML permite jerarquizar, estructurar la información y describir los contenidos dentro del propio documento, así como la reutilización de partes del mismo. La información estructurada presenta varios contenidos (texto, imágenes, audio, etc.) y formas (hojas de cálculo, tablas de datos, libretas de direcciones, parámetros de configuración, dibujos técnicos, etc.).

JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente del lado del cliente, denominado *Navigator JavaScript*, donde la principal característica es que no requiere compilación ya que el navegador se encarga de compilarlo. Este permite crear efectos atractivos y dinámicos en las páginas web, que se ejecutan con rapidez debido a que se aloja y ejecuta en el ordenador del usuario.

Tradicionalmente se usa en páginas web HTML para realizar operaciones en el marco de la aplicación cliente, sin acceso a funciones del servidor. Actualmente se utiliza para enviar y recibir información del servidor junto con ayuda de otras tecnologías como AJAX[9]. JavaScript se interpreta en el lado del usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.

Biblioteca JavaScript

Una librería proporciona una serie de funciones y métodos concretos para simplificar tareas complejas, se pueden utilizar desde el código, respetando el API que proporcionan, pero sin necesidad de adaptar o modificar la estructura de la aplicación.

jQuery 2.0

[8] Los bugs son errores o fallos en el software que desencadena resultados no esperados.
[9] *Asynchronous JavaScript and XML* - JavaScript y XML Asíncrono.

jQuery 2.0 es una biblioteca de JavaScript que permite simplificar interacción con los documentos HTML, manipular el árbol DOM[10], manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

Framework de desarrollo

Un *framework* brinda una plataforma completa en la cual se implementa el código que define una aplicación en específico. Esta plataforma ya tiene predefinida su arquitectura, por lo cual la estructura del código que se implementa en la plataforma debe cumplir con las restricciones que la misma posee. Según Mario A. Sánchez un *framework* es "un conjunto de bibliotecas, utilizadas para implementar la estructura estándar de una aplicación.", las cuales tiene como objetivo ahorrar tiempo de implementación mediante la reutilización de código en el desarrollo de una aplicación.

Symfony 2.7.9

"Symfony es un proyecto PHP de software libre que permite crear aplicaciones y sitios web rápidos y seguros de forma profesional.". Symfony es un *framework* que agrupa varios componentes creados por el proyecto Symfony, este *framework* junto con todas las librerías y componentes que incluye se publican bajo la licencia MIT[11].

Entre las características de Symfony 2, se encuentra que se basa en la arquitectura Modelo Vista Controlador, está desarrollado en PHP 5.3 y es compatible con la mayoría de los gestores de bases de datos como MySQL, PostgreSQL, Oracle y Microsoft SQL Server, lo cual brinda posibilidades de que el trabajo con los datos sea un proceso sencillo, e incluso debido al uso de Doctrine para la interacción con la base de datos, permite que se pueda hacer un cambio de sistema gestor de bases de datos en cualquier momento del desarrollo de la aplicación.

Bootstrap 3.0

Bootstrap es un *framework* CSS de código abierto para diseño de sitios y aplicaciones web. Fue creado por el grupo de desarrolladores de twitter en el 2011. Este *framework* contiene plantillas de diseño con tipografía,

[10] El árbol DOM define la estructura lógica y el modo en que se manipulan y se accede a los elementos de un documento.
[11] Licencia de software libre que se generó en el Instituto Tecnológico de Massachusetts (MIT, *Massachusetts Institute of Technology*).

formularios, botones, menús de navegación y otros elementos de diseño basado en HTML y CSS, además cuenta con extensiones de JavaScript.

Entre las principales características de Bootstrap se encuentra que permite el desarrollo web responsive, lo que significa que las aplicaciones web obtenidas son compatibles con todo tipo de dispositivo electrónico, dígase móvil, *tablet* hasta computadoras de alta resolución. También brinda la posibilidad de que, al diseñar las interfaces, se divida el espacio de trabajo un una cuadricula denominada GRID, que contiene 12 columnas e infinita cantidad de filas, lo cual proporciona variantes en la organización de los distintos elementos que se ubicarán en la interfaz.

Bootstrap en su versión 3 es compatible con varios de los principales navegadores web existentes, entre los que se encuentran Google Chrome, Safari, Mozilla Firefox, Internet Explorer y Opera. También tiene integración con HTML5 y CCS3.

Entorno de Desarrollo Integrado

Un Entorno de Desarrollo Integrado IDE (por sus siglas en inglés, *Integrated Development Environment*), es un programa informático compuesto por un conjunto de herramientas de programación. Estos generalmente se conforman de un editor de código fuente, un compilador, herramientas de construcción automáticas y un depurador.

NetBeans 8.1

NetBeans es un IDE de código abierto, el cual desde 2010, es catalogado por la Oracle como el IDE oficial para la Plataforma Java ("*the official IDE for the Java Platform*").

NetBean está desarrollado sobre Java, que es un lenguaje de programación de libre uso para propósito comercial y no comercial. Una característica de este IDE es que sus productos son multiplataforma, o sea, no importa el sistema operativo en el cual se ejecute, este funciona correctamente.

NetBean es compatible con varios lenguajes de programación, entre los que se encuentran Java, C, C#, C++, PHP, JavaScript, HTML, CSS entre otros.

Sistema gestor de bases de datos

Un sistema gestor de bases de datos (SGBD) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en un medio de almacenamiento, además de

proporcionar herramientas para añadir, borrar, modificar y analizar los datos. Estos sistemas también proporcionan métodos para mantener la integridad de los datos, para administrar el acceso de usuarios a los datos y para recuperar la información si el sistema se corrompe, además permiten presentar la información de la base de datos en variados formatos dígase imagen, texto, números entre otros.

PostgreSQL 9.4

PostgreSQL es un SGBD relacional orientado a objetos y de uso libre, publicado bajo la licencia PostgreSQL. Algunas de las características que lo destacan por encima de los demás SGBD son la alta concurrencia, lo cual permite que mientras un proceso escribe en una tabla, otros puedan acceder a la misma tabla sin necesidad de bloqueos; otra característica es la amplia variedad de tipos nativos entre los que incluye números de precisión arbitraria, texto de largo ilimitado, figuras geométricas, direcciones IP (IPv4, IPv6), direcciones MAC, arrays, entre otras.

Algunas de las ventajas que destacan en PostgreSQL son la seguridad en términos generales, integridad en la base de datos, el uso de disparadores (*triggers*), los cuales responden condiciones ya predefinidas en el mismo sistema. Otra ventaja es la conexión con otras bases de datos y la posibilidad del uso de autorizaciones para poder acceder o realizar alguna acción sobre los datos almacenados brindando mayor seguridad.

Servidor web

Un servidor web es una o un conjunto de aplicaciones que se mantienen ejecutadas en un ordenador, las cuales se encargan de brindar una determina respuesta a cada solicitud que se realiza desde un ordenador cliente.

Apache 2.4.7

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras que implementa el protocolo HTTP. Entre las características que presenta Apache se encuentran que posee bases de datos de autenticación y negociado de contenido.

Apache es un servidor web clave en el desarrollo de la *World Wide Web* y en el 2005 alrededor del 70% de los sitios web a nivel mundial usaban el servidor Apache en su versión 2.2. Este servidor web cuenta con

varias ventajas entre las que se encuentra la modularidad, debido a que cuenta con un core[12] y un conjunto de módulos que aportan las funcionalidades básicas de un servidor web; es un servidor de código abierto, o sea es de libre uso y es multiplataforma debido a que es compatible con varios sistemas operativos.

Conclusiones parciales

La realización del estado del arte orientado a la creación de recursos educativos colaborativamente, esclareció la necesidad de llevar a cabo el desarrollo de un componente que permita la participación conjunta de los usuarios en la creación de recursos educativos sobre el marco de trabajo Xalix. Para garantizar que le componente se ajuste de manera independiente a la arquitectura basada en componentes Xalix, se desarrollará mediante la utilización de las herramientas y la base de tecnologías que se definen en la misma.

[12]El core de apache es la aplicación básica del servidor, es donde se agregan los distintos módulos que posee el servidor web Apache.

Capítulo 2: Propuesta de solución

En el presente capítulo se hará una descripción de la solución propuesta. Se presentarán los requerimientos que debe cumplir el sistema para satisfacer las necesidades del cliente, además se presentan los diagramas de diseño que se especifican en la metodología AUP-UCI y los patrones que se usarán en el desarrollo del componente para el trabajo colaborativo en la creación de recursos educativos.

Conceptos asociados al domino del componente

Para un mejor entendimiento de la propuesta solución es necesario conocer los principales conceptos que se involucran en el proceso de la creación de recursos educativos colaborativamente en la propuesta de solución.

A continuación, se presenta una descripción de los distintos conceptos identificados.

Grupo: es la unidad global dentro del componente para la colaboración en la creación de recursos educativos. Dentro de estos se desarrolla el proceso de creación de los recursos educativos mediante las convocatorias y permite que interactúen los distintos miembros

Convocatoria: las convocatorias son una especie de subgrupo donde se define que recurso educativo se creará específicamente. Dentro de la misma se define en que sección del recurso educativo trabajará cada uno de los miembros que pertenecen a la misma.

Recurso Educativo: el recurso educativo es el producto final que se desea obtener mediante la colaboración de diferentes usuarios, estos están estructurados en un conjunto de secciones.

Sección: es la mínima unidad de un recurso educativo y define un elemento o característica del recurso.

Usuario: es el actor que pertenece al sistema y dispone de los permisos para poder hacer uso del componente para la colaboración en la creación en recursos educativos.

Miembro: el miembro es el usuario del sistema, pero este posee un vínculo con un determinado grupo de trabajo colaborativo. Es el actor que realiza las funciones de creación del recurso educativo dentro del componente para el trabajo colaborativo.

Rol: es la característica que posee un miembro, y le proporciona una clasificación que le permite trabajar en una determinada sección de un recurso educativo.

Petición: la petición es lo que indica que usuario quiere pertenecer a que grupo de trabajo colaborativo, estas peticiones son gestionadas exclusivamente por el miembro que creó el grupo de trabajo. Las peticiones se pueden aceptar o rechazar o sea permitir o no que un usuario forme parte de un grupo de trabajo.

Solicitud: la solicitud es lo que indica que miembro quiere pertenecer a que convocatoria, las solicitudes son gestionadas exclusivamente por el miembro que creó la convocatoria. Las solicitudes se pueden aceptar o rechazar o sea permitir o no que un usuario forme parte de una convocatoria.

Modelo de dominio

El modelo de dominio se crea para representar los conceptos clave del dominio del problema e identifica las relaciones entre las entidades comprendidas. Según Martin Fowler un modelo de dominio es "una representación visual de las clases conceptuales u objetos de una situación real en un dominio" (33).

A continuación, se presenta el diagrama asociado al modelo del dominio del componente para la colaboración en la creación de recursos educativos.

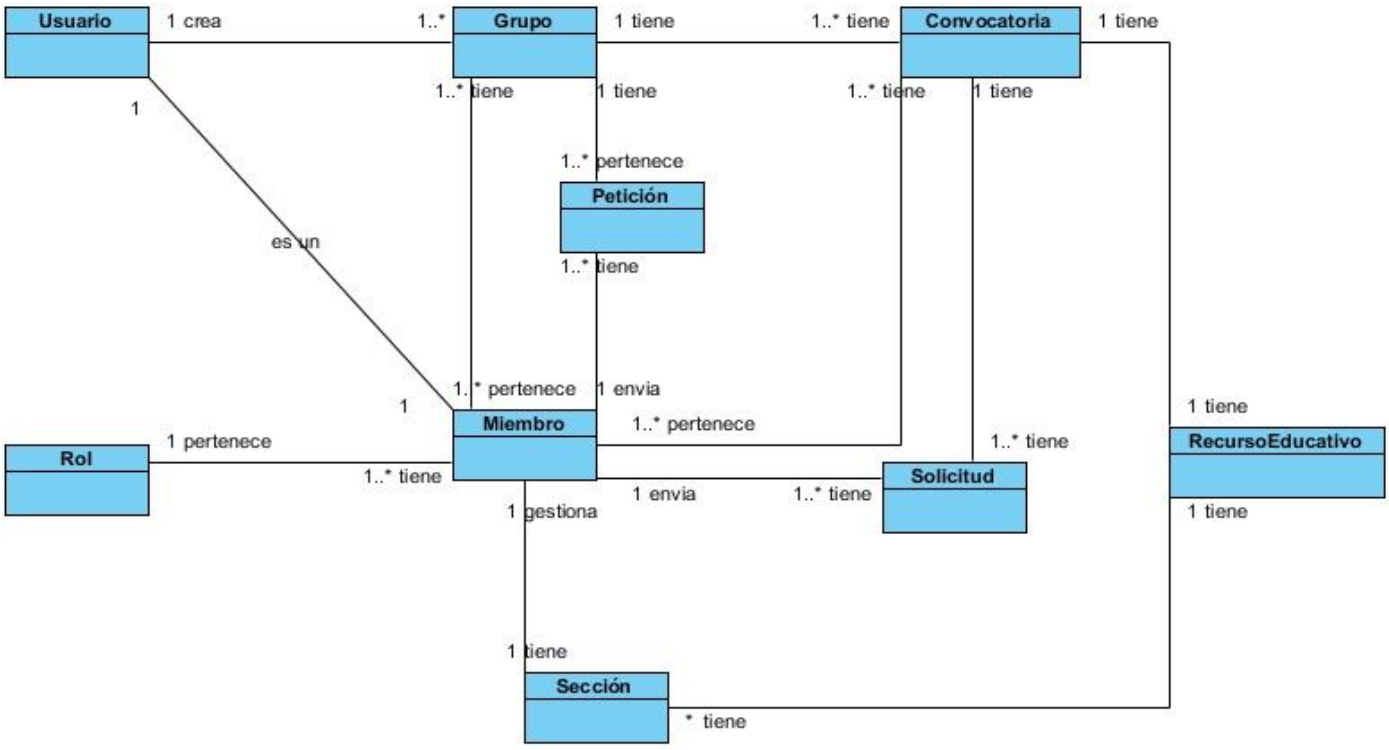


Ilustración 1 Diagrama del modelo del dominio

Descripción de la solución propuesta

El componente sobre el marco de trabajo Xalix, para la colaboración en la creación de recursos educativos constará de un conjunto de funcionalidades que permitirán a los usuarios organizarse en grupos de desarrollo, donde podrán mantener una constante comunicación. Dentro del grupo, los usuarios crearán convocatorias, que estarán orientadas a organizar el proceso de desarrollo mediante la asignación de las diferentes secciones de un recurso educativo, lo que permitirá que solamente los miembros asignados a la sección puedan trabajar en ella. Dentro de la aplicación se podrán crear roles que servirán como medida de control para el acceso a la configuración de los grupos, las convocatorias y el trabajo sobre el recurso educativo, además se controlará la subscripción en los grupos y las convocatorias mediante peticiones y solicitudes respectivamente.

El componente se desarrollará sobre el marco de trabajo Xalix, mediante la utilización de los estándares de codificación y las normas establecidas por el grupo de desarrollo del mismo, con el objetivo de obtener una completa integración.

Requerimientos del sistema

Los requisitos que debe cumplir la solución propuesta son la principal pauta para definir la estructura del sistema, además estos marcan el nivel de calidad que poseerá el producto con el objetivo de satisfacer las necesidades del cliente. Estos requerimientos definen las principales condiciones y restricciones necesarias para que el sistema funcione. Además, son fundamentales para establecer un punto medio de comunicación entre el equipo de desarrollo y el cliente.

Requerimientos funcionales

- RF 1 Crear grupo de trabajo: la aplicación debe permitir crear grupos de trabajo.
- RF 2 Editar grupo de trabajo: la aplicación debe permitir que se modifiquen los datos de los grupos de trabajo.
- RF 3 Listar grupos de trabajo: la aplicación debe permitir mostrar una lista con los grupos de trabajos creados.
- RF 4 Eliminar grupo de trabajo: la aplicación debe permitir eliminar grupos de trabajo.
- RF 5 Crear convocatoria: la aplicación debe permitir que crear convocatorias.
- RF 6 Editar convocatoria: la aplicación debe permitir modificar los datos de las convocatorias.

- RF 7 Listar convocatorias: la aplicación debe permitir mostrar una lista con las convocatorias.
- RF 8 Eliminar convocatoria: la aplicación debe permitir eliminar convocatorias.
- RF 9 Crear rol: la aplicación debe permitir crear roles.
- RF 10 Editar rol: la aplicación debe permitir modificar los datos de los roles.
- RF 11 Listar roles: la aplicación debe permitir mostrar una lista de los roles.
- RF 12 Eliminar rol: la aplicación debe permitir eliminar roles.
- RF 13 Enviar petición para formar parte de un grupo: la aplicación debe permitir que se envíen peticiones a los grupos de trabajo.
- RF 14 Aceptar petición: la aplicación debe permitir aceptar las peticiones.
- RF 15 Rechazar petición: la aplicación debe permitir rechazar las peticiones.
- RF 16 Enviar solicitud de subscripción a una convocatoria: la aplicación debe permitir enviar solicitudes.
- RF 17 Aceptar solicitud: la aplicación debe permitir aceptar las solicitudes.
- RF 18 Rechazar solicitud: la aplicación debe permitir rechazar las solicitudes.
- RF 19 Asignar rol a un miembro de un grupo: la aplicación debe permitir asignar un rol a los miembros de los grupos de trabajo.
- RF 20 Reasignar rol a un miembro de un grupo: la aplicación debe permitir reasignar un rol a los miembros de los grupos de trabajo.
- RF 21 Asignar sección de un recurso educativo a un miembro: la aplicación debe permitir asignar un miembro a una sección del recurso educativo.
- RF 22 Listar miembros: la aplicación debe permitir mostrar una lista de los miembros de los grupos de trabajo.
- RF 23 Eliminar un miembro: la aplicación debe permitir eliminar los miembros de los grupos de trabajo.

Requerimientos no funcionales

Los requisitos no funcionales son un conjunto de características y propiedades que definen si un sistema es confiable, utilizable, estable y otros rasgos que debe poseer el producto.

Seguridad

RNF 1 Garantizar la protección de información de accesos no autorizados.

RNF 2 Garantizar el acceso a las funcionalidades definidas para los usuarios de acuerdo a los roles que posean.

RNF 3 Mantener la aplicación disponible evitando que los mecanismos de seguridad impidan el acceso a la información requerida por los usuarios autorizados.

Usabilidad

RNF 4 Cumplir con las pautas de diseño establecidas en la Estrategia Marcaria de la Universidad.

RNF 5 Tener una interfaz amigable, de uso fácil e intuitivo.

Confiabilidad

RNF 6 Verificar las consecuencias asociadas a los fallos del sistema.

Disponibilidad

RNF 7 La aplicación debe estar accesible en cualquier momento para brindar acceso a la información.

Portabilidad

Debe ser instalada en un entorno con:

RNF 8 Servidor de bases de datos relacional PostgreSQL 9.4.x con memoria RAM: 16 GB, disco Duro: 100 GB y microprocesador: 6 x 800 Ghz.

RNF 9 Servidor de aplicaciones Ngnix: 2.7.x con memoria RAM: 16 GB, disco Duro: 500 GB y microprocesador: 6 x 800 Ghz.

Historias de usuario

Las historias de usuario son la técnica utilizada para especificar los requisitos del software. El cliente describe brevemente las características funcionales o no funcionales que el sistema debe poseer. El tratamiento de las historias de usuario es dinámico y flexible, en cualquier momento pueden eliminarse, reemplazarse por otras más específicas, añadirse nuevas o ser modificadas. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en poco tiempo.

A, continuación se presentan las historias de usuario correspondientes a los requerimientos crear grupos de trabajo y listar grupos de trabajo.

Tabla 1 Historia de usuario crear grupo de trabajo

Número: 1.1	**Nombre del requisito:** Crear grupo de trabajo
Programador: Roylan Tumbarell Quintana	**Iteración Asignada:** 1era
Prioridad: Alta	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo: Medio	**Tiempo Real:** 12 hosras

Descripción:

1- Objetivo:

Permitir crear un grupo de trabajo colaborativo.

2- Acciones para lograr el objetivo (precondiciones y datos):

Precondiciones:

- No existe un grupo de trabajo con las especificaciones deseadas.

Pos-condiciones:

- Se crea un grupo de trabajo con los datos especificados por el usuario.

3- Comportamientos válidos y no válidos (flujo central y alternos):

Flujo central:

- El usuario selecciona la opción crear nuevo grupo de trabajo.
- El usuario llena el formulario con los datos deseados para el grupo de trabajo.
- El usuario selecciona la opción guardar.
- Se muestra la vista del grupo creado.
- Termina el requisito.

Flujos alternos:

- El usuario selecciona la opción crear nuevo grupo de trabajo.
- El usuario selecciona la opción cancelar.
- No se crea el grupo de trabajo.
- Termina el requisito.

Observaciones:

Prototipo de interfaz:

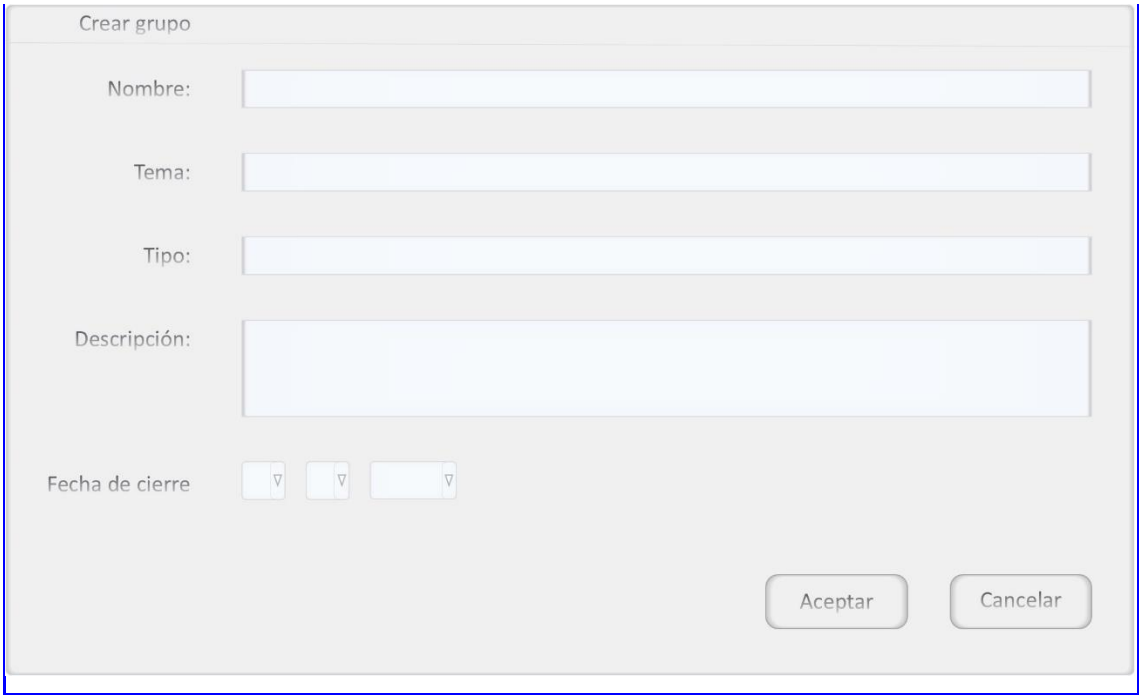


Tabla 2 Historia de usuario listar grupo de trabajo

Número: 1.2	**Nombre del requisito:** Listar los grupos de trabajo
Programador: Roylan Tumbarell Quintana	**Iteración Asignada:** 1era
Prioridad: Media	**Tiempo Estimado:** 1 hora
Riesgo en Desarrollo: Medio	**Tiempo Real:** 1 hora

Descripción:

1- Objetivo:

Mostrar al usuario los grupos creados en un listado donde puede acceder enviar una petición de entrada al grupo de trabajo.

2- Acciones para lograr el objetivo (precondiciones y datos):

Precondiciones:

- Existen grupos de trabajo creados.

Pos-condiciones:

- Se muestra una lista de grupos de trabajo creados.

3- Comportamientos válidos y no válidos (flujo central y alternos):

Flujo central:

- El usuario selecciona la opción listar grupos de trabajo.
- Se muestra una lista de los grupos de trabajo existentes.
- Termina el requisito.

Flujos alternos:

- N/A

Observaciones:

Prototipo de interfaz:

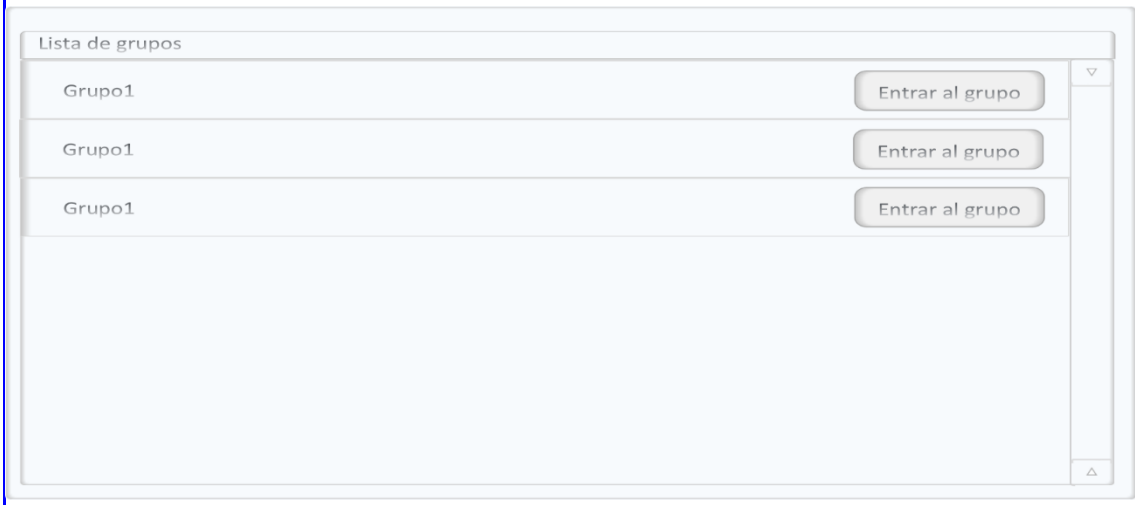


Diagrama de clases del análisis

Una clase de análisis representa una abstracción de una o varias clases del diseño del sistema. Según Ivar Jacobson las clases del análisis siempre encajan en uno de los tres estereotipos básicos: clase de interfaz, de control y de entidad.

A continuación, se presentan los diagramas de clases del análisis gestionar grupos y gestionar convocatorias.

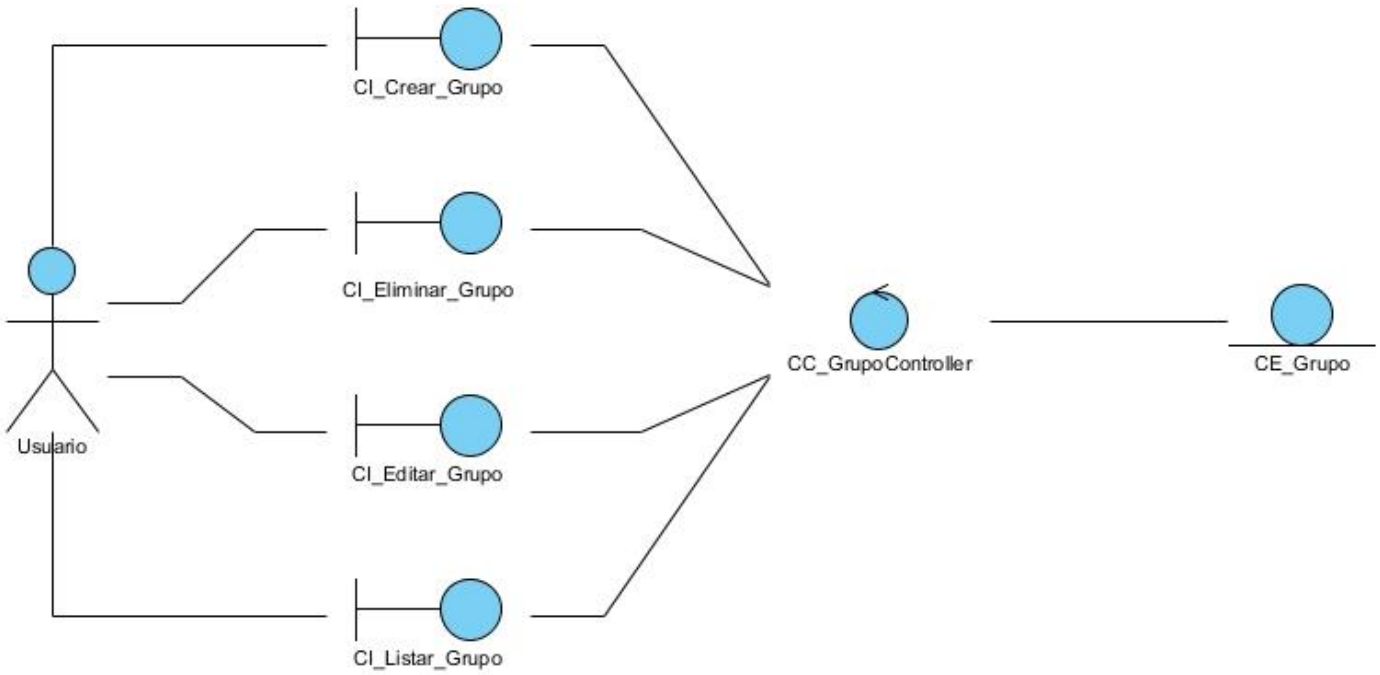


Ilustración 2 Diagrama de clase de análisis gestionar grupo

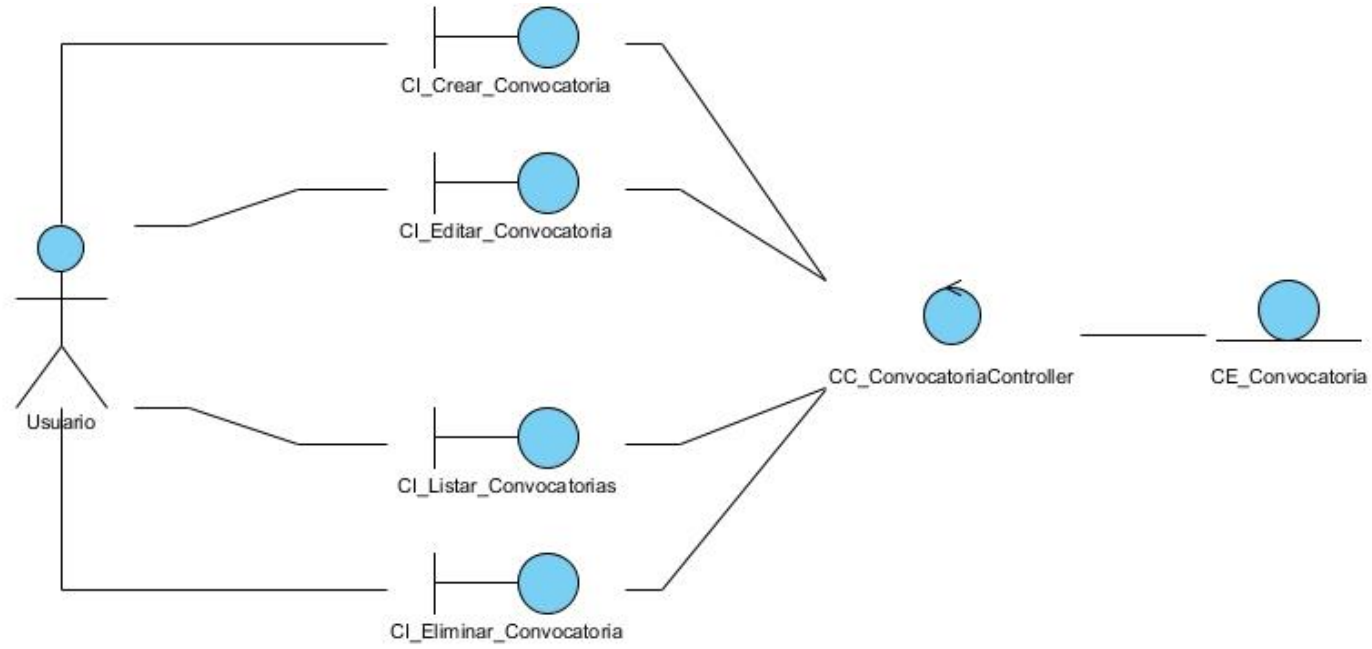


Ilustración 3 Diagrama de clase del análisis gestionar convocatoria

Diagramas de colaboración del análisis

Los diagramas de colaboración del análisis modelan mediante enlaces y mensajes las interacciones entre las clases u objetos. Según Ivar Jacobson los diagramas de colaboración del análisis "muestran las interacciones entre objetos creando enlaces entre ellos y añadiendo mensajes a esos enlaces. El nombre de un mensaje debería denotar el propósito del objeto invocante en la interacción con el objeto invocado" (35).

A continuación, se muestran los diagramas de colaboración del análisis asociados a los requerimientos crear grupo de trabajo y crear convocatoria.

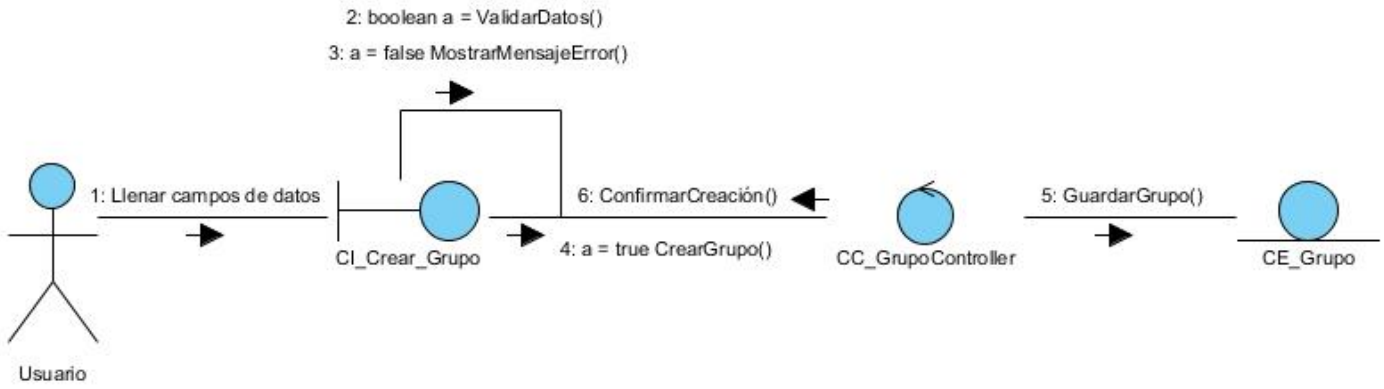


Ilustración 4 Diagrama de colaboración del análisis asociado al RF crear grupo de trabajo

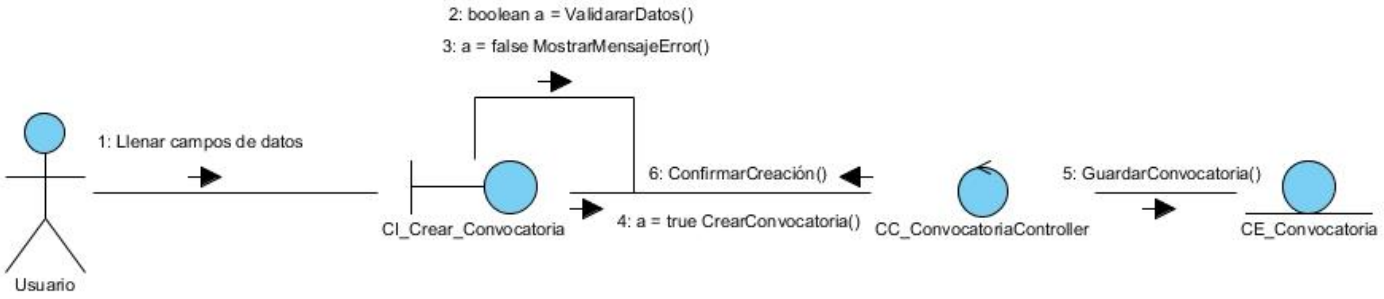


Ilustración 5 Diagrama de colaboración del análisis asociado al RF crear convocatoria

Modelo de diseño

El modelo de diseño es la etapa de desarrollo de software donde se definen las principales características que poseerá el producto final. Es la fase que proporciona detalles sobre arquitectura del software, estructura de datos, interfaces y componentes que se necesitan para implementar el sistema, aunque todo está expenso a cambios a medida que se avanza en el desarrollo.

Patrón arquitectónico

En el proceso de resolver los problemas identificados en la estructura interna del software, se utilizarán patrones arquitectónicos que están orientados a las interacciones que se establecen entre los elementos que conforman todo el funcionamiento. Se representa mediante un esquema de organización estructural que contiene subsistema, sus responsabilidades e interacciones. Algunos ejemplos de patrones arquitectónicos son:

Programación por capas: es una arquitectura cliente-servidor en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño. La principal característica de esta arquitectura es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de algún cambio, solo se realiza en el nivel requerido sin tener que revisar todo el código.

Arquitectura en pizarra: consta de múltiples elementos funcionales, denominados agentes, y un instrumento de control denominado pizarra, donde los agentes suelen estar especializados en una tarea concreta o elemental, todos ellos cooperan para alcanzar una meta común; y la pizarra coordina a los distintos agentes y facilita su intercomunicación.

Orientada a servicios (SOA del inglés *Service Oriented Architecture*): es un paradigma de arquitectura para diseñar y desarrollar sistemas distribuidos. Son un conjunto de ordenadores independientes conectados entre sí que brindan servicios como un solo ordenador.

Modelo Vista Controlador (MVC): es un patrón arquitectónico que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo para gestionar los eventos y las comunicaciones.

El patrón arquitectónico que se usará para la implementación de la solución propuesta es el MVC, el cual realiza un desglose de los principales componentes de un sistema web, lo cual garantiza independencia en el funcionamiento del sistema. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado, define componentes para la representación de la información, y por otro lado para la interacción del usuario. Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.(37)

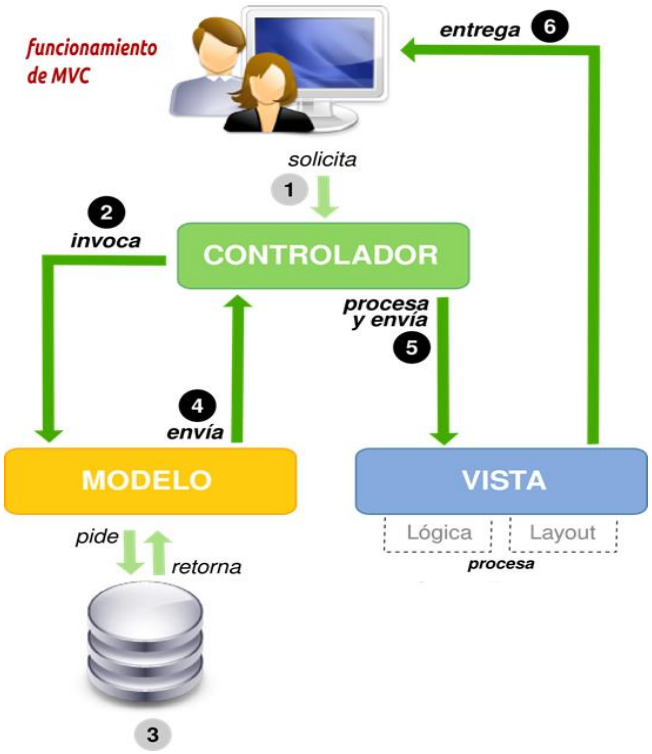


Ilustración 6 Esquema Modelo Vista Controlador

Patrones de diseño

Los patrones de diseño son un conjunto experiencias acumuladas que tienen como objetivo dar solución a problemas y lograr que estas soluciones sean reutilizables en situaciones específicas, para así lograr un ahorro en tiempo. Según Pressman "son formas de describir las mejores prácticas, buenos diseños, y encapsulan la experiencia de tal forma que es posible para otros el reutilizar dicha experiencia."(39)

Existen diferentes tipos de patrones de diseños. Estos están definidos según el lugar donde se apliquen, entre estos se encuentran los patrones de diseño en el nivel de componentes, los cuales están orientados a resolver uno o varios problemas extraídos del modelo de requerimientos; también se encuentran los patrones de diseño en la interfaz de usuario que se centran en hacer que la navegación y la manipulación de los datos por parte del usuario del sistema sea un proceso sencillo; otros son los patrones de diseño de webapp[13], que están clasificados en dos criterios, en cuanto a diseño y el nivel de granularidad del patrón,

[13] Aplicación web.

donde el diseño identifica que aspecto del diseño es importante en el desarrollo del sistema y la granularidad se refiere al nivel de abstracción del patrón, o sea se aplica a todo el sistema o simplemente a lugares específicos.(39)

También existen los patrones **GOF** y **GRASP**, los cuales se utilizarán en el desarrollo de componente para el trabajo colaborativo en la creación de recursos educativos, dado que estos definen estructuras y modos de organización, que brindan un mayor rendimiento del sistema en cuanto a funcionamiento.

Los patrones **GOF** deben su nombre al grupo *The Gang of Four*, los cuales definieron en sus investigaciones 23 patrones que se clasifican en cuanto a dos criterios, propósito y alcance y se organizan en tres grupos:

Creacionales: se ocupan del proceso de creación de clases y objetos, son los encargados de abstraer el proceso de instanciación o creación de objetos, ayudan a que el sistema sea independiente de cómo sus objetos son creados, integrados y representados. Los patrones que responden a esta categoría son *Factory Method, Abstract Factory, Builder, Prototype y Singleton.*(41)

Estructurales: tratan de la composición de clases y objetos, se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes. Estos patrones se encargan de que los cambios que se realizan en los requisitos del sistema no ocasionen cambios en las relaciones entre los objetos. Los patrones que corresponden a esta categoría son *Adapter, Bridge, Composite, Decorator, Façade, Flyweight y Proxy*.(41)

Comportamiento: caracterizan las formas en que las clases o los objetos interactúan y distribuyen la responsabilidad. Son los encargados de las opciones de comportamiento de la aplicación, permitiendo que el comportamiento varíe en tiempo de ejecución, sin estos patrones cada comportamiento tendría que diseñarse e implementarse por separado. Los restantes 11 patrones se agrupan en esta categoría, estos son *Interpreter, Template Method, Chain of Responsibility, Command, Iterator, Mediator, Memento, Observer, State, Strategy, Visitor.*(41)

Los patrones **GRASP** se definen en los siguientes cinco tipos:

Experto en información: La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. Este patrón se evidencia en las entidades, debido a que estas son las que contienen toda la información necesaria sobre determinado objeto en el sistema.(43)

Bajo acoplamiento: debe haber pocas dependencias entre las clases. Como evidencia del uso de este patrón, en el sistema se definen, en la gran mayoría, clases y objetos que, en concreto, estructuran y definen sus responsabilidades sin depender del uso de las funcionalidades de las demás clases. Además, se evita el uso de excesivo de la herencia de clases, evitando así que un cambio en las clases no repercuta en el funcionamiento básico de las demás.(43)

Alta cohesión: Cada elemento de nuestro diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable. El uso de este patrón en el desarrollo de la solución propuesta se evidencia en la creación de paquetes de clases que realizan funciones similares, pero sobre distintos objetos del sistema, lo cual garantiza un mayor entendimiento en el momento de la reutilización, además de la independencia en la obtención de información y funcionamiento del sistema.(43)

Creador: Se le asigna la responsabilidad de crear un objeto de la clase a otra cuando esta contiene, usa, almacena, tiene los datos de inicialización o es una agregación o composición de la clase. La creación de instancias es una de las actividades comunes en un sistema orientado a objetos. En consecuencia, es útil contar con un principio general para la asignación de las responsabilidades de creación. Si se asignan bien el diseño puede soportar un bajo acoplamiento, mayor claridad, encapsulación y reutilización.(43)

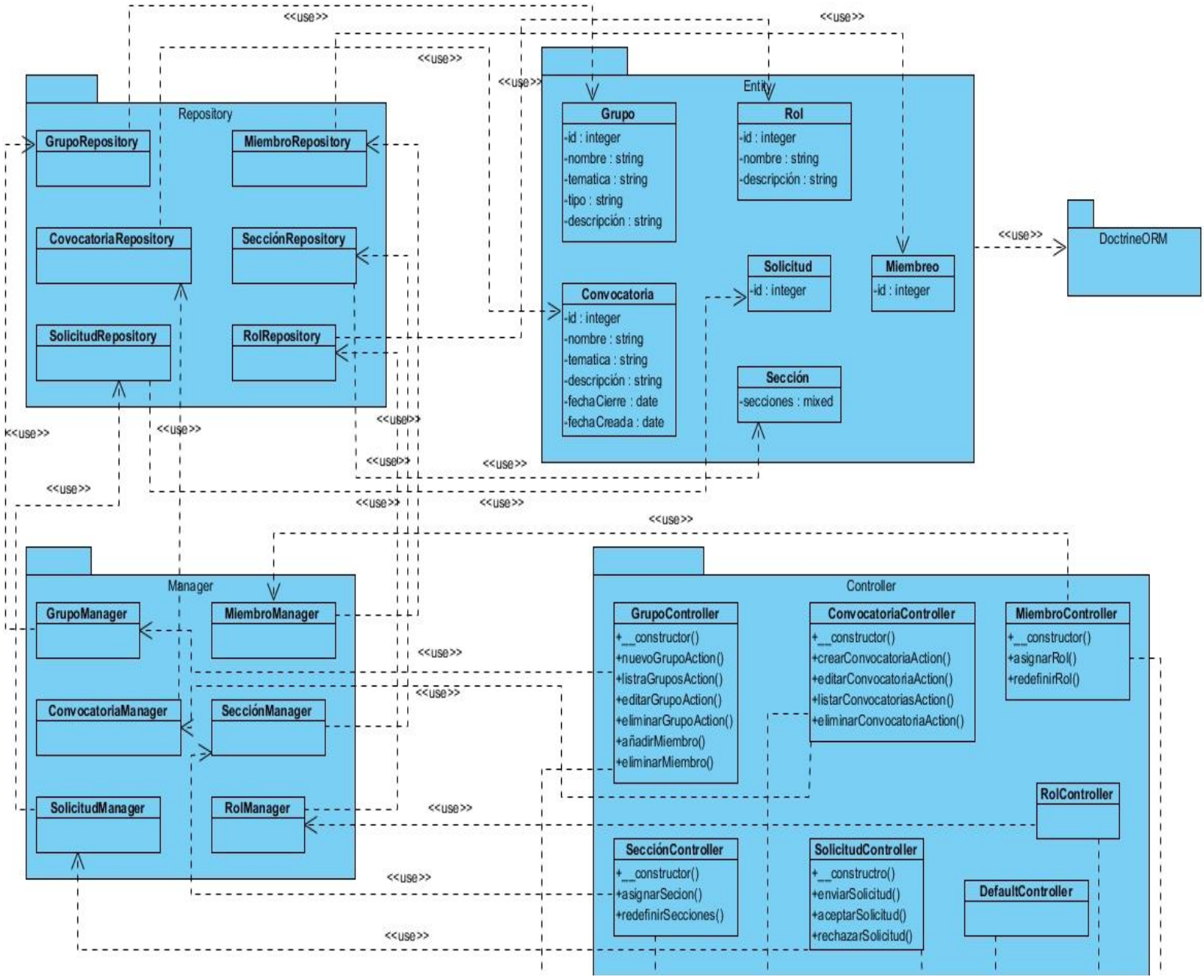
Controlador: Asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. El controlador no debe realizar la actividades y funciones que son propias del sistema como por ejemplo la seguridad, las validaciones de datos, etc., estas funciones se realizan en otras clases con las cuales el controlador posee una alta cohesión.(43)

El utilizar estos patrones de diseño permite que se obtenga un modelo que represente la estructura necesaria en el componente para el trabajo colaborativo en la creación de recursos educativos, para brindar soluciones a los principales problemas asociados al modelado de los requerimientos y sus relaciones. Se establece con mayor claridad que problemas se resolverán en el nivel de arquitectura del software.

Diagrama de clases del diseño

El diagrama de clases del diseño describe gráficamente las descripciones de las clases de software y de las interfaces en una aplicación mediante la representación de clases, métodos, dependencias, interfaces y las asociaciones entre todas ellas.(45)

A continuación, se presenta el diagrama de clases del diseño del componente para el trabajo colaborativo en la creación de recursos educativos.


Repository
GrupoRepository
MiembroRepository
CovocatoriaRepository
SecciónRepository
SolicitudRepository
RolRepository
Entity
Grupo
-id : integer
-nombre : string
-tematica : string
-tipo : string
-descripción : string
Rol
-id : integer
-nombre : string
-descripción : string
Convocatoria
-id : integer
-nombre : string
-tematica : string
-descripción : string
-fechaCierre : date
-fechaCreada : date
Solicitud
-id : integer
Miembreo
-id : integer
Sección
-secciones : mixed
DoctrineORM
Manager
GrupoManager
MiembroManager
ConvocatoriaManager
SecciónManager
SolicitudManager
RolManager
Controller
GrupoController
+__constructor()
+nuevoGrupoAction()
+listraGruposAction()
+editarGrupoAction()
+eliminarGrupoAction()
+añadirMiembro()
+eliminarMiembro()
ConvocatoriaController
+__constructor()
+crearConvocatoriaAction()
+editarConvocatoriaAction()
+listarConvocatoriasAction()
+eliminarConvocatoriaAction()
MiembroController
+__constructor()
+asignarRol()
+redefinirRol()
RolController
SecciónController
+__constructor()
+asignarSecion()
+redefinirSecciones()
SolicitudController
+__constructro()
+enviarSolicitud()
+aceptarSolicitud()
+rechazarSolicitud()
DefaultController
<<use>>

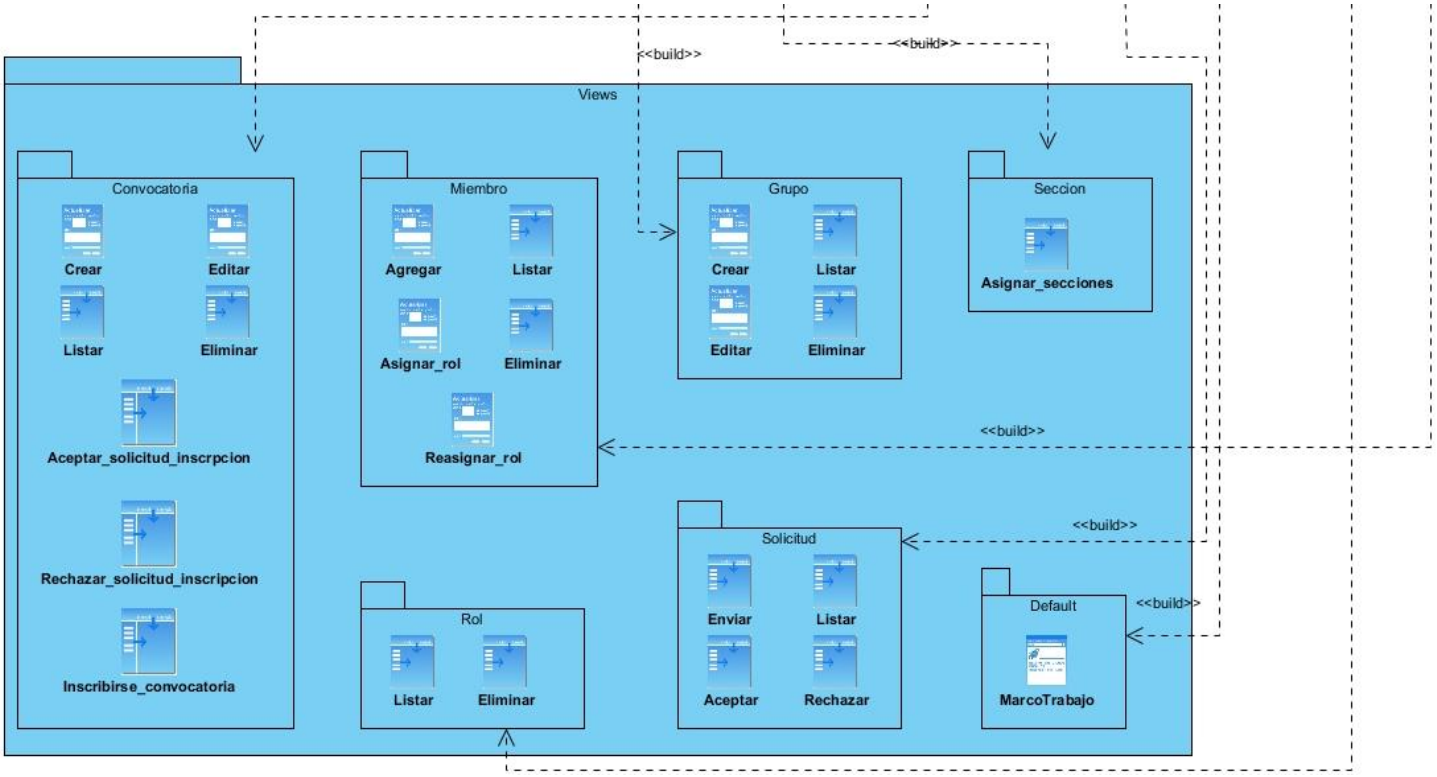


Ilustración 7 Diagrama de clases del diseño

Diagrama de secuencia del diseño

Los diagramas de secuencia muestran las interacciones entre objetos mediante transferencia de mensajes entre objetos o subsistemas. El nombre del mensaje debería indicar una operación del objeto que recibe la invocación o de una interfaz que el objeto proporciona.(45)

A continuación, se muestran los diagramas de secuencia del diseño correspondientes a los requisitos funcionales crear grupo y crear convocatorias dentro componente para la colaboración en la creación de recursos educativos.

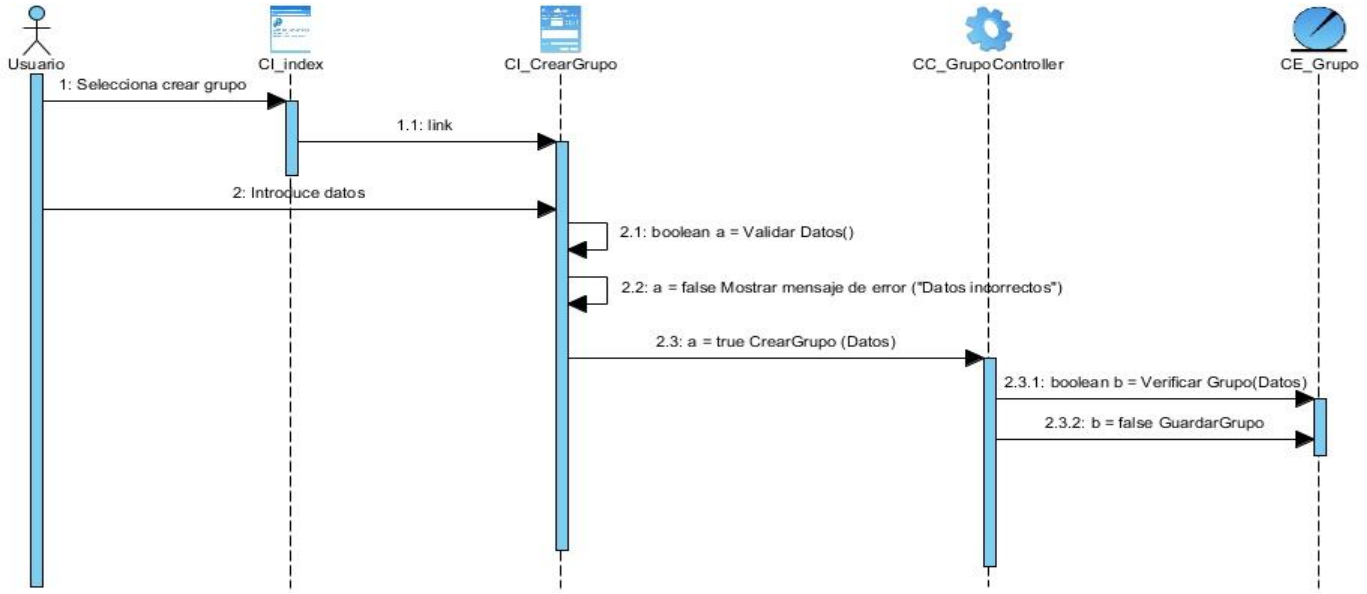


Ilustración 8 Diagrama de secuencia del diseño asociado RF crear grupo de trabajo

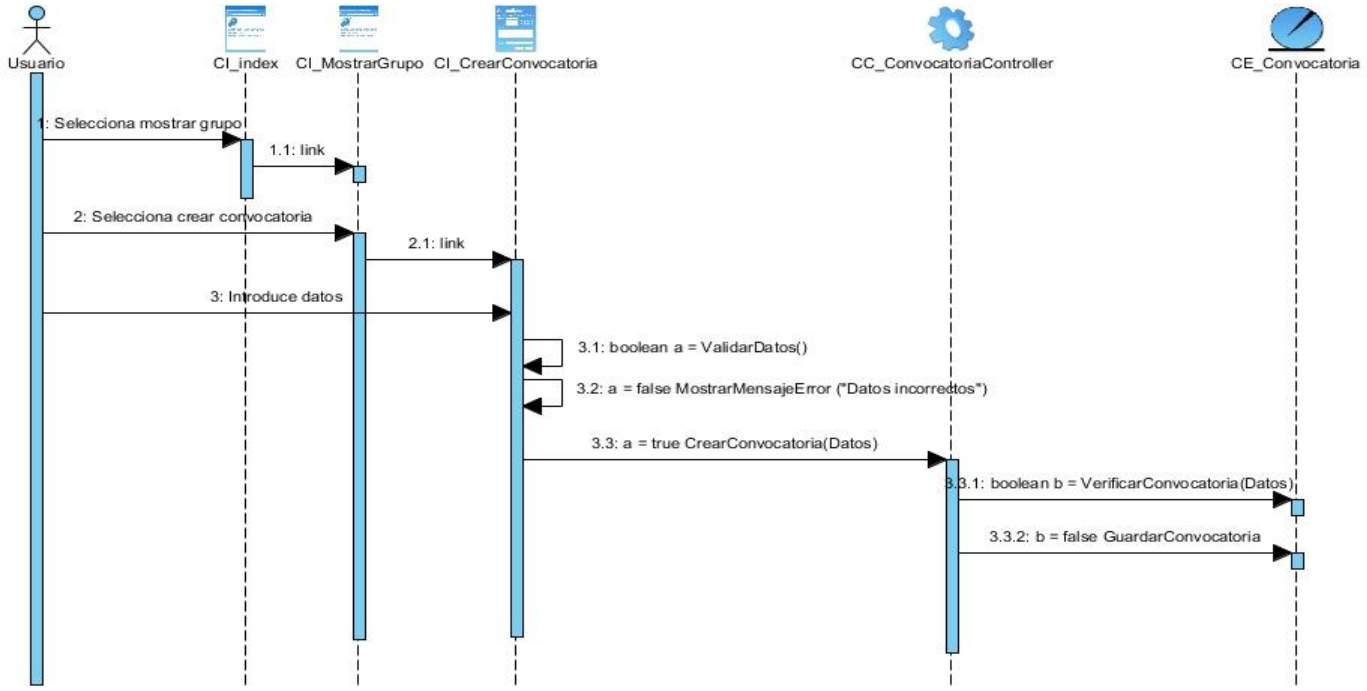


Ilustración 9 Diagrama de secuencia del diseño asociado al RF crear convocatoria

Modelo de datos

El modelado de datos es el momento del desarrollo de software donde se definen y representan gráficamente mediante el diagrama de entidad relación, los distintos objetos de datos que conformarán la fuente de información del sistema. Según Roger Pressman el diagrama de modelo de datos “define todos los datos que se introducen, se almacenan y se producen dentro de una aplicación” (39)

A continuación, se presenta el diagrama de modelo de datos correspondiente al componente para el trabajo colaborativo en la creación de recursos educativos, donde se hace una representación de los objetos de datos correspondientes con la información que se gestiona dentro del mismo.

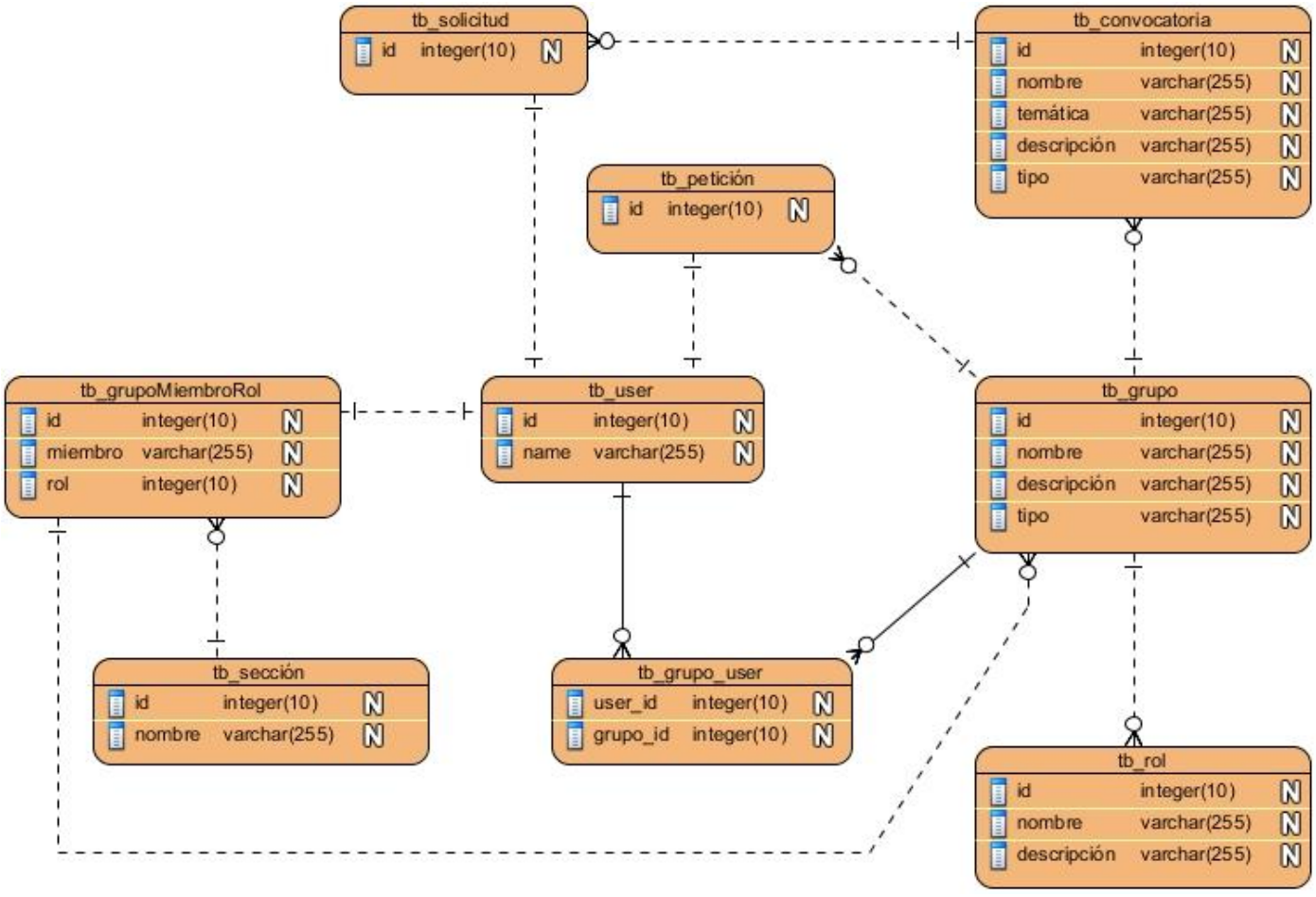


Ilustración 10 Diagrama de modelo de datos

A continuación, se presenta una descripción de las tablas tb_grupo, tb_convocatoria y tb_rol:

Tabla 3 Descripción de la tabla tb_grupo

tb_grupo		
Esta tabla tb_grupo almacena la información que describe un grupo dentro de la base de datos.		
Atributo	Tipo de dato	Descripción
id	integer	El valor id identifica un grupo específico dentro de la tabla que contiene todos los grupos. El valor se genera automáticamente cuando se inserta un nuevo grupo en la base de datos, el tipo de datos es numérico.
nombre	varchar (255)	El valor nombre contiene el nombre que se especificó en la creación del grupo. El campo almacenará una cadena de caracteres con una cantidad máxima de 255 caracteres y acepta caracteres alfanuméricos.
descripción	varchar (255)	El valor descripción contiene la descripción del grupo que se especificó en la creación. El campo almacenará una cadena de caracteres con una cantidad máxima de 255 caracteres y acepta caracteres alfanuméricos.
tipo	varchar (255)	El valor tipo almacena el tipo de grupo que se escogió en la creación. El campo almacenará una cadena de caracteres con una cantidad máxima de 255 caracteres, pero los posibles valores ya están predefinidos.

Tabla 4 Descripción de la tabla tb_convocatoria

tb_convocatoria		
La tabla tb_convocatoria contiene la información que describe una convocatoria dentro de la base de datos		
Atributo	Tipo de dato	Descripción
id	integer	El valor id identifica una convocatoria específica dentro de la tabla que contiene todas las convocatorias. El valor se genera automáticamente

		cuando se inserta una nueva convocatoria en la base de datos, el tipo de datos es numérico.
nombre	varchar (255)	El valor nombre contiene el nombre que se especificó en la creación de la convocatoria. El campo almacenará una cadena de caracteres con una cantidad máxima de 255 caracteres y acepta caracteres alfanuméricos.
temática	varchar (255)	El valor temática contiene la temática de la convocatoria que se especificó en la creación. El campo almacenará una cadena de caracteres con una cantidad máxima de 255 caracteres y acepta caracteres alfanuméricos
Descripción	varchar (255)	El valor descripción contiene la descripción de la convocatoria que se especificó en la creación. El campo almacenará una cadena de caracteres con una cantidad máxima de 255 caracteres y acepta caracteres alfanuméricos
tipo	varchar (255)	El valor tipo almacena el tipo de convocatoria que se escogió en la creación. El campo almacenará una cadena de caracteres con una cantidad máxima de 255 caracteres, pero los posibles valores ya están predefinidos.

Tabla 5 Descripción de la tabla tb_rol

tb_rol		
La tabla tb_rol contiene la información que describe un rol dentro de la base de datos.		
Atributo	Tipo de dato	Descripción
id	integer	El valor id identifica un rol específico dentro de la tabla que contiene todos los roles que se definen para los grupos de trabajo colaborativo. El valor se genera automáticamente cuando se inserta un nuevo rol en la base de datos, el tipo de datos es numérico.

nombre	varchar (255)	El valor nombre contiene el nombre que se especificó en la creación del rol. El campo almacenará una cadena de caracteres con una cantidad máxima de 255 caracteres y no acepta caracteres alfanuméricos.
descripción	varchar (255)	El valor descripción contiene la descripción del rol que se especificó en la creación. El campo almacenará una cadena de caracteres con una cantidad máxima de 255 caracteres y acepta caracteres alfanuméricos.

Diagrama de despliegue

El diagrama de despliegue muestra las relaciones físicas entre los componentes software y hardware en el sistema final, la configuración de los elementos de procesamiento en tiempo de ejecución y los procesos y objetos que se ejecutan en ellos.

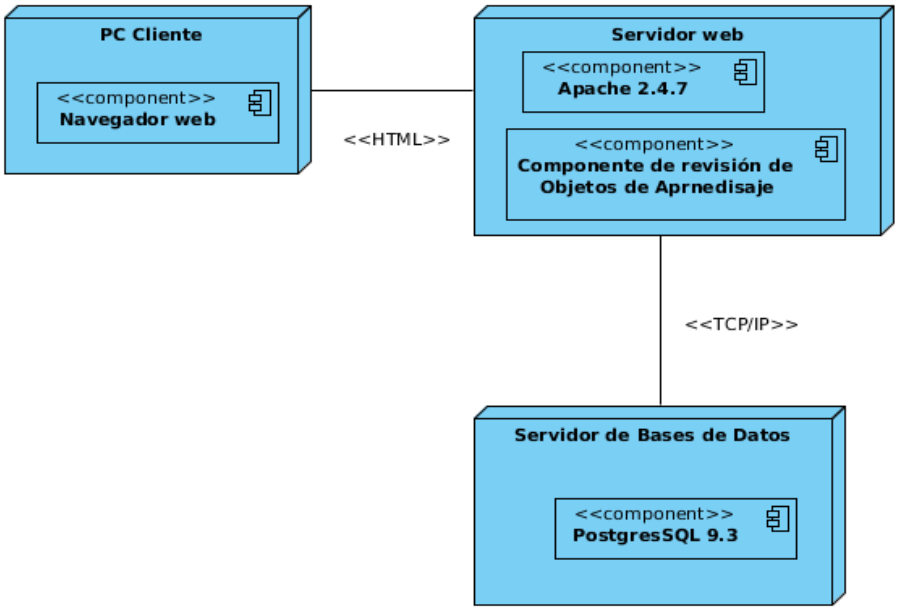


Ilustración 11 Diagrama de despliegue

Conclusiones parciales

Luego de realizar el análisis de las diferentes metodologías de desarrollo de software, el uso de las diferente herramientas y lenguajes de programación junto con la utilización de la herramienta de modelado Visual Paradigm 8.0, se obtienen los diferentes artefactos que brindan apoyo y marcan un punto de partida para la implementación de la solución propuesta. También se crean los diferentes diagramas correspondientes a la estructura de la base de datos que utilizará el sistema, se describen mediante el diagrama de clases del análisis y el diagrama de clases del diseño las posibles clases que interactuarán en el funcionamiento del sistema.

También se definen que patrones de diseño y arquitectura serán utilizados con el objetivo de que se genere una estructura de código que permita la reutilización del sistema.

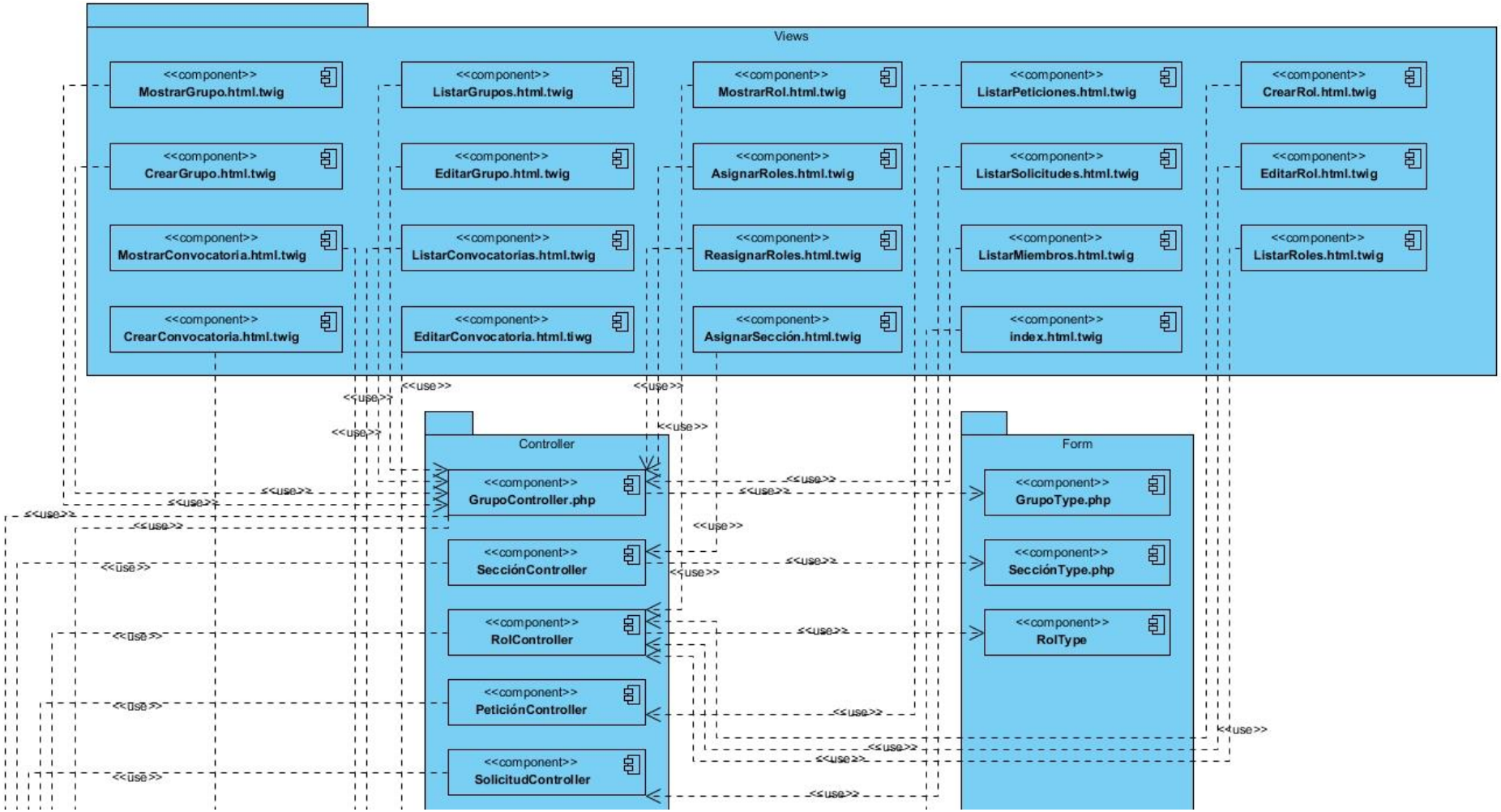
Capítulo 3: Implementación y pruebas

En este capítulo se hace una descripción del proceso de implementación que se desarrollará en base a los resultados obtenidos luego de concluido el capítulo anterior. También se presentan las pruebas realizadas al sistema junto con los resultados obtenidos en pos de obtener un producto con la calidad requerida para satisfacer las necesidades del cliente.

Modelo de implementación

Un componente es un objeto independiente que compone un software, según Pressman un componente es "una unidad de software independiente que puede estar compuesta por otros componentes y que se utiliza para crear un sistema de software" (39). También plantea que estos poseen dos características importantes: el componente es una entidad ejecutable independiente donde el código fuente no está disponible, por lo que no se tiene que compilar antes de usarse con otros componentes del sistema, y los servicios de un componente están disponibles a través de una interfaz, y todas las interacciones son mediante esa interfaz.

A continuación, se presenta el diagrama de componentes correspondiente al componente para el trabajo colaborativo en la creación de recursos educativos.


Views
<<component>>
MostrarGrupo.html.twig
<<component>>
ListarGrupos.html.twig
<<component>>
MostrarRol.html.twig
<<component>>
ListarPeticiones.html.twig
<<component>>
CrearRol.html.twig
<<component>>
CrearGrupo.html.twig
<<component>>
EditarGrupo.html.twig
<<component>>
AsignarRoles.html.twig
<<component>>
ListarSolicitudes.html.twig
<<component>>
EditarRol.html.twig
<<component>>
MostrarConvocatoria.html.twig
<<component>>
ListarConvocatorias.html.twig
<<component>>
ReasignarRoles.html.twig
<<component>>
ListarMiembros.html.twig
<<component>>
ListarRoles.html.twig
<<component>>
CrearConvocatoria.html.twig
<<component>>
EditarConvocatoria.html.tiwg
<<component>>
AsignarSección.html.twig
<<component>>
index.html.twig
Controller
<<component>>
GrupoController.php
<<component>>
SecciónController
<<component>>
RolController
<<component>>
PeticiónController
<<component>>
SolicitudController
Form
<<component>>
GrupoType.php
<<component>>
SecciónType.php
<<component>>
RolType
<<use>>

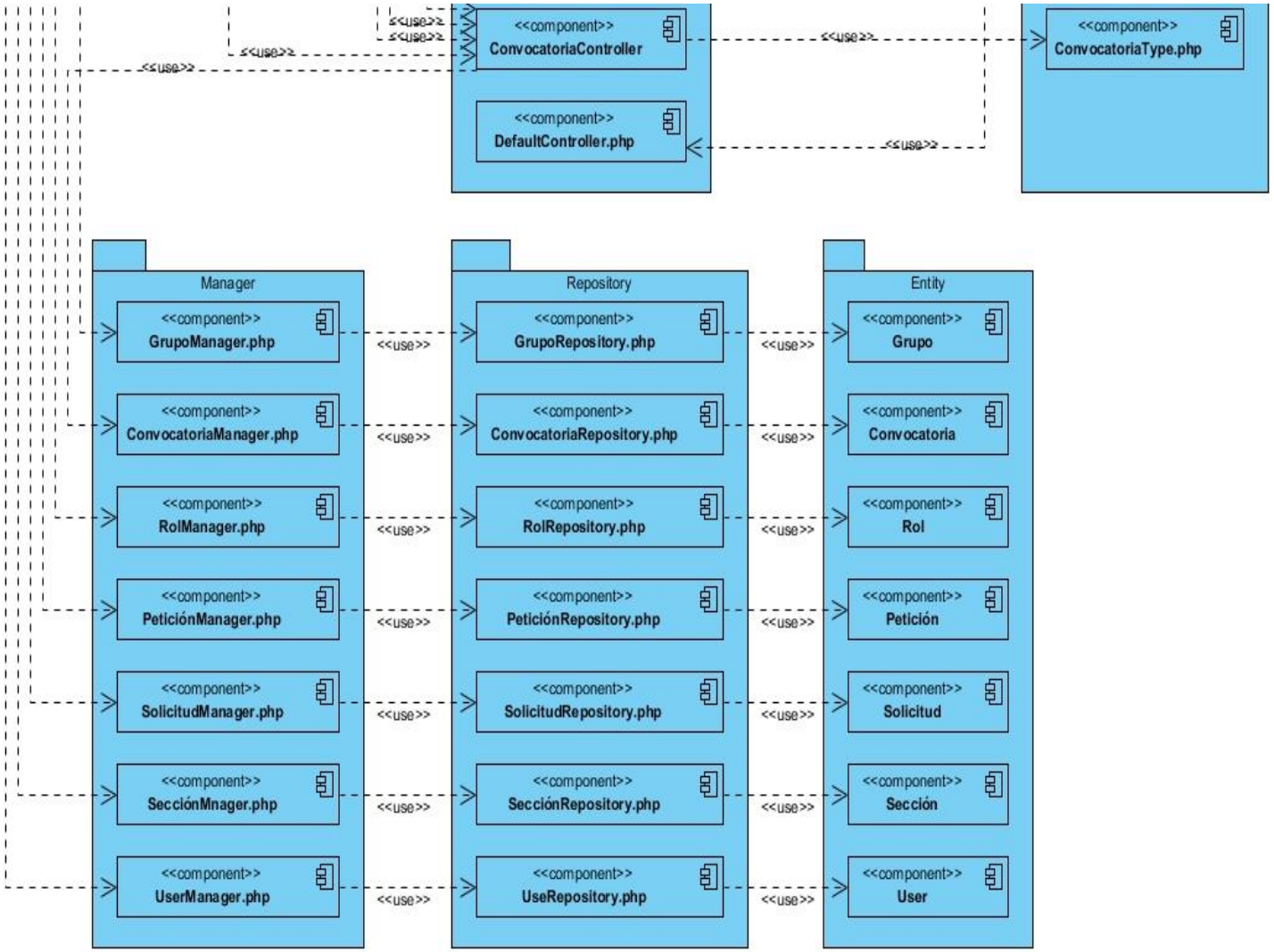


Ilustración 12 Diagrama de componentes

Estándares de codificación

Los estándares de codificación son pautas establecidas para la organización y estructura del código del software que se desea desarrollar. Estos estándares tienen como objetivo definir una organización común que facilite la comprensión del código en pos de que la reutilización, el mantenimiento y la edición del mismo se realice de manera sencilla.

Para el establecer los estándares de codificación en el desarrollo del componente para el trabajo colaborativo en la creación de recursos educativos, se hace uso del documento Estándares de Codificación, creado por el equipo de desarrollo del marco de trabajo Xalix.

En el documento Estándares de Codificación se definen grupos de estándares agrupados en cuanto a diferentes categorías las cuales son:

Estructura

- Añade un solo espacio después de cada delimitador coma.
- Añade un solo espacio alrededor de los operadores (==, &&, …).
- Añade una coma después de cada elemento del arreglo en un arreglo multilínea, incluso después del último.
- Añade una línea en blanco antes de las declaraciones return, a menos que el valor devuelto solo sea dentro de un grupo de declaraciones (tal como una declaración if).
- Usa llaves para indicar la estructura del cuerpo de control, independientemente del número de declaraciones que contenga.
- Define una clase por archivo — esto no se aplica a las clases ayudantes privadas, de las cuales no se tiene la intención de crear una instancia desde el exterior y por lo tanto no les preocupa la norma PSR-0.
- Declara las propiedades de clase antes que los métodos.
- Declara los métodos públicos (*public*) primero, luego los protegidos (*protected*), y finalmente los privados (*private*).
- Utiliza paréntesis cuando se instancien clases, independientemente del número de argumentos que tenga el constructor.
- Los mensajes de excepción se deben concatenar utilizando la función sprintf.

Nomenclatura

- Utiliza mayúsculas intercaladas (sin guiones bajos) en nombres de variable, función, método o argumentos.
- Usa guiones bajos para nombres de opción y nombres de parámetro.
- Utiliza espacios de nombres para todas las clases.
- Prefija las clases abstractas con Abstract. Por favor, ten en cuenta que algunas de las primeras clases de Symfony2 no siguen esta convención y no se han rebautizado por razones de compatibilidad hacia atrás. No obstante, todas las nuevas clases abstractas tienen que seguir esta convención de nomenclatura.
- Sufija las interfaces con Interface.
- Sufija las características con Trait.
- Sufija las excepciones con Exception.\
- Utiliza caracteres alfanuméricos y guiones bajos para los nombres de archivo.

Documentación

- Añade bloques PHPDoc a todas las clases, métodos y funciones.
- Omite la etiqueta @return si el método no devuelve nada.
- Las anotaciones @package y @subpackage no se utilizan.

Codificación: Cuando un objeto tiene una relación «principal» con muchas «cosas» relacionadas (objetos, parámetros, ...), los nombres de los métodos están normalizados:

- get()
- set()
- has()
- all()
- replace()
- remove()
- clear()
- isEmpty()
- add()
- register()

- count()
- keys()

Estos son los principales estándares utilizados en el desarrollo del componente para el trabajo colaborativo en la creación de recursos educativos, con el objetivo de obtener un mayor entendimiento dentro del equipo de desarrollo del proyecto.

Pruebas de software

El modelo de prueba describe principalmente cómo se prueban los componentes ejecutables en el modelo de implementación con pruebas de integración, pruebas de sistema, pruebas unitarias entre otras. Constituye una colección de CP, procedimientos de prueba y componentes de prueba.

Métodos y técnicas de pruebas

Los métodos y técnicas de prueba brindan diferentes criterios para realizar un caso de prueba, el cual, pueda ocasionar un fallo en el sistema a evaluar. Las técnicas se agrupan en dos categorías:

Pruebas de caja blanca o estructurales: estas pruebas están destinadas a realizar una evaluación de los detalles procedimentales del código, estas tratan de garantizar que: se ejecuten al menos una vez cada uno de los caminos independientes de cada módulo, o sea, todas las posibles opciones de flujo de datos, que se utilizan las decisiones en su parte verdadera y su parte falsa, se ejecuten todos los bucles en sus límites y que se usan todas las estructuras de datos internas.

Pruebas de caja negra o funcionales: son pruebas que se realizan sobre la interfaz del software, obviando el funcionamiento interno y la estructura del sistema, mediante las entradas que recibe y las salidas o respuestas que produce. Se enfoca en que es lo que hace y no en como lo hace. Estas pruebas tienen como objetivo demostrar que: las funciones del software son operativas, las entradas se aceptan de forma correcta, se produce una salida correcta y que se mantiene la integridad de la información externa.

Para la evaluación del componente para la colaboración en la creación de recursos educativos se realizaron pruebas de caja negra teniendo en cuenta el método de particiones equivalentes que divide el dominio de entrada de un sistema en clases de datos, a partir de las cuales derivan los casos de prueba. Cada una de estas clases de equivalencia representa un conjunto de estados válidos o inválidos para las condiciones de entrada.

Diseño de casos de pruebas

Los casos de pruebas se orientan a encontrar todos los posibles errores y deficiencias en las funcionalidades implementadas y se realizan en base a los casos de uso del sistema.

A continuación, se presentan los diseños de los casos de pruebas asociados a los casos de uso Crear grupo de trabajo y Listar grupos de trabajo.

Tabla 6 Caso de prueba crear grupo de trabajo

Escenario	Descripción	Nombre	Tema	Tipo	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción crear grupo de trabajo	Selecciona la opción crear grupo para crear un grupo de trabajo.					Brinda la posibilidad de introducir de manera obligatoria los siguientes datos del grupo: -nombre(n) -tema(t) -tipo(p) -descripción(d) Permite además la opción cancelar la operación y la opción aceptar los datos.	Colaboración/Grupos/CrearGrupo
EC 1.2 Opción de cancelar	Selecciona la opción de cancelar					Redirecciona a la vista anterior.	Colaboración/Grupos/CrearGrupo/Cancelar
EC 1.3 Opción de aceptar con dato incorrectos	Selecciona la opción aceptar	I	V	V	V	Muestra un error en los campos que no tienen valores correctos. Regresa al escenario 1.1	Colaboración/Grupos/CrearGrupo/Aceptar
		V	I	V	V		
		V	V	I	V		
		V	V	V	I		

EC 1.4 Opción aceptar con datos válidos.	Selecciona la opción aceptar	V	V	V	V	Muestra el grupo creado.	

Tabla 7 Caso de prueba listar grupo de trabajo

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción listar grupos de trabajo	Selecciona la opción listar grupos para mostrar una lista de grupos de trabajo.	Brinda una lista de los grupos creados, la opción de ver los datos del grupo deseado y la opción de enviar una petición de entrada a cada grupo.	Colaboración/Grupos/ListarGrupos
EC 1.2 Opción de enviar petición de entrada al grupo	Selecciona la opción subscribirse	Envía una petición de entrada al administrador del grupo.	Colaboración/Grupos/ListarGrupos/Subscribirse
EC 1.3 Opción de ver grupo	Selecciona la opción ver grupo	Muestra los datos del grupo seleccionado.	Colaboración/Grupos/ListarGrupos/Mostrar

Resultados de las pruebas

Mediante las pruebas realizadas a la aplicación se obtuvieron los siguientes resultados en la primera iteración.

Tabla 8 Resultados de la primera iteración de pruebas realizada a la aplicación.

No.	Casos de prueba	No conformidades			Total
		Validación	Funcionalidad	Interfaz de usuario	
1	Gestionar grupo	1	-	2	3
2	Gestionar rol	-	-	-	-
3	Gestionar convocatoria	3	2	4	9
4	Enviar peticiones	-	-	1	1
5	Aceptar petición	1	1	-	2
6	Rechazar petición	-	-	-	
8	Enviar solicitud	-	1	1	2
9	Aceptar solicitud	1	-	-	1
10	Rechazar solicitud	-	-	-	
12	Asignar rol	2	2	2	6
13	Reasignar rol	1	-	1	2
14	Asignar sección	3	3	2	8
15	Listar miembros	1	-	2	3

16	Eliminar miembro	-	-	-	-
	Total	13	9	15	37

A continuación, se muestra un gráfico con los resultados de las tres iteraciones de pruebas realizadas a la aplicación.

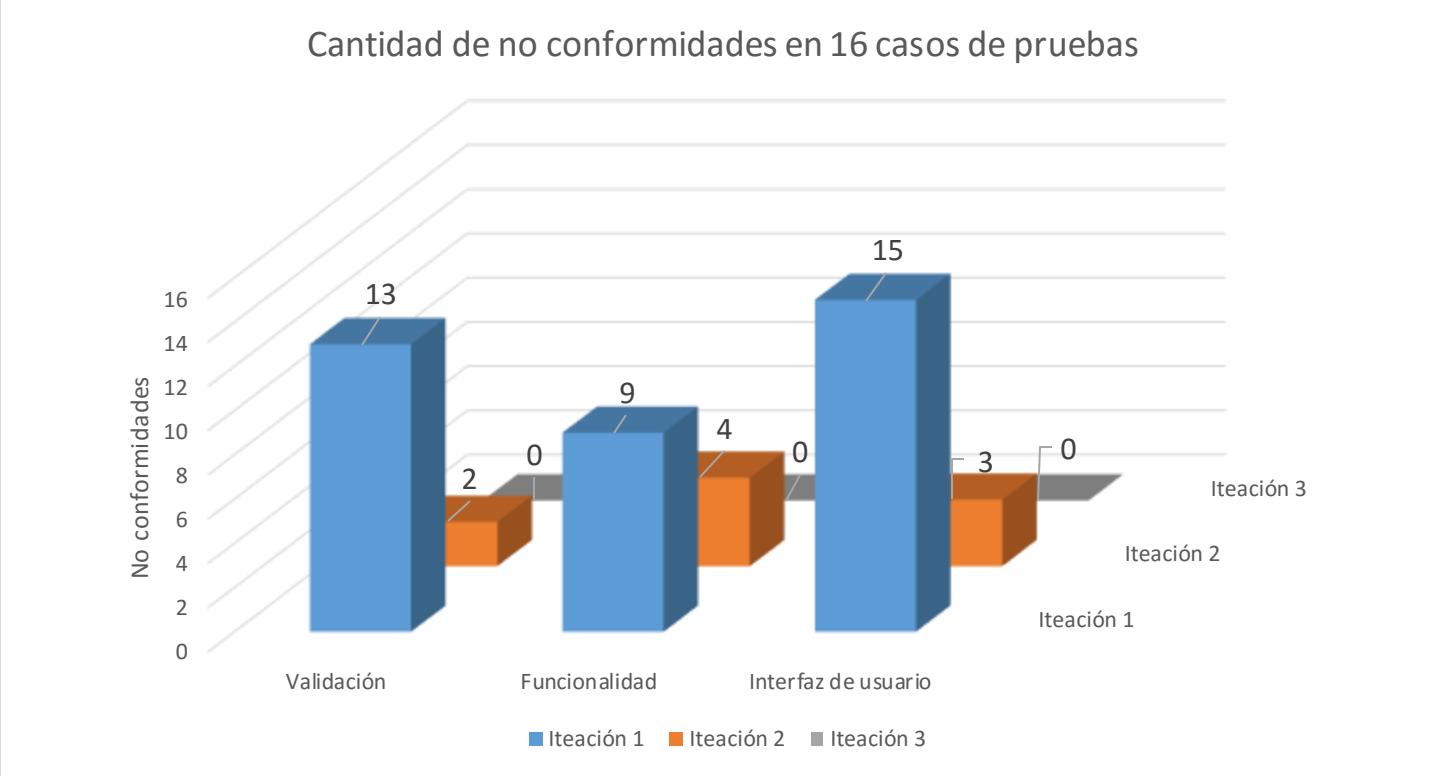


Ilustración 13 Resultados de las iteraciones de pruebas

Conclusiones parciales

En el capítulo se presentó el diagrama de componentes que facilitó la implementación del componente para el trabajo colaborativo en la creación de recursos educativos. Se describen los estándares de codificación que se utilizaron en el desarrollo de la solución, los cuales permitieron que la obtuviera una aplicación compatible con la arquitectura basada en componentes Xalix.

Los métodos de pruebas de caja negra permitieron validar funcionamiento del componente para el trabajo colaborativo en la creación de recursos educativos, mediante la comprobación de las funcionalidades, lo cual permitió obtener un producto que satisface las necesidades del cliente.

Conclusiones

Como resultado de la investigación, se desarrolló el componente sobre en el marco de trabajo Xalix, para el trabajo colaborativo en la creación de recursos de educativos, obteniéndose las siguientes conclusiones:

- La investigación del proceso de creación de recursos educativos de forma colaborativa comprobó que: el mismo se realiza con mayor organización, se obtienen gran calidad en los recursos educativos creados y que además de fomentar la comunicación en un grupo de desarrollo.
- Mediante el análisis y diseño llevado a cabo en el proceso de investigación asociado al desarrollo del componente para la colaboración en la creación de recursos educativos, se obtuvieron los diferentes artefactos que se definen en la metodología usada. Los diagramas elaborados facilitaron la implementación de la solución propuesta debido a que los mismos manifiestan de forma organizada todos los elementos, las relaciones y el funcionamiento de la solución obtenida.
- La implementación del componente sobre el marco de trabajo Xalix, satisface la necesidad de poder crear recursos educativos de manera colaborativa mediante la participación conjunta de los usuarios y que el mismo se puede acoplar de forma independiente a la arquitectura basada en componentes del centro FORTES.
- Las pruebas realizadas al sistema y la corrección de las no conformidades permitieron determinar que las funcionalidades se desarrollaron correctamente.

Recomendaciones

- Incluir a la aplicación funcionalidades que permitan avisar de manera automática a los usuarios cuando son aceptados en un grupo o una convocatoria.
- Incluir a la aplicación funcionalidades que notifiquen a los administradores de los grupos cuando tienen peticiones sin revisar.
- Funcionalidades de mensajería instantánea.

Referencia Bibliográfica

1. MARTHA ZAPATA. Recursos educativos digitales: Conceptos básicos. *Programa Integración de Tecnologías a la Docencia* [online]. September 2012. [Accessed 25 June 2016]. Available from: http://aprendeenlinea.udea.edu.co/boa/contenidos.php/d211b52ee1441a30b59ae008e2d31386/845/estilo/aHR0cDovL2FwcmVuZGVlbmxpbmVhLnVkZWEuZWR1LmNvL2VzdGlsb3MvYXp1bF9jb3Jwb3JhdGl2by5jc3M=/1/contenido/

2. DAVID H. JHONASO and LUCIA ROHRER-MURPHY. *Activity Jonasson* [online]. 1999. [Accessed 18 June 2016]. Available from: ftp://ftp.uwc.ac.za/users/DMS/CITI/New%20PHd%20folder/PHd%20PDF's/Activity_Jonasson.pdf

3. BENKLER, Yochai. *Commons-based Peer Production and Virtue* [online]. 4 November 2006. [Accessed 30 June 2016]. Available from: https://www.nyu.edu/projects/nissenbaum/papers/jopp_235.pdf

4. Centro de Formación Permanente. *e-Learning* [online]. 2007. [Accessed 18 June 2016]. Available from: http://www.cfp.us.es/area-de-empresas/formacion/e-learning

5. BADRUL HUDA KHAN. *Web-based Instruction*. Educational Technology, 1997. ISBN 978-0-87778-296-4.

6. E LÓPEZ BALLESTEROS and L TORRES. *Las plataformas virtuales: escenarios alternativos para la formación* [online]. 2004. [Accessed 18 June 2016]. Available from: http://www.lmi.ub.es/edutec2004/pdf/195.pdf

7. Definición de e-Learning. [online]. 2010. [Accessed 18 June 2016]. Available from: http://www.e-abclearning.com/definicione-learning

8. OMG. *OMG Unified Modeling Language (OMG UML), Superstructure* [online]. January 2011. Available from: http://www.omg.org/spec/UML/2.4/Superstructure/

9. BABELTIC. BABELTIC. [online]. [Accessed 18 June 2016]. Available from: http://www.babeltic.eu/herramientasdeautor.htm

10. DAVID H. JONASSEN. *Diseño de entornos constructivistas de aprendizaje U III*.

11. ¿Qué es un Recurso Didáctico? | Pedagogía. [online]. 24 March 2006. [Accessed 30 June 2016]. Available from: http://www.pedagogia.es/recursos-didacticos/¿Qué es un Recurso Didáctico?

12. UNESCO. Recursos educativos abiertos | Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura. [online]. [Accessed 30 June 2016]. Available from: http://www.unesco.org/new/es/communication-and-information/access-to-knowledge/open-educational-resources/

13. *Objectivism and Constructivism* [online]. [Accessed 18 June 2016]. Available from: http://www.myweb.ttu.edu/ddao/assets/Objectivism%20and%20Constructivism.pdf

14. ZULEY CRUZ. Tecnologia. *calameo.com* [online]. [Accessed 25 June 2016]. Available from: http://www.calameo.com/books/004744352e2cd30bfadbe

15. STEPHEN J. FALLOWS and RAKESH BHANOT. *Quality Issues in ICT-based Higher Education*. Psychology Press, 2005. ISBN 978-0-415-33521-8.

16. EDUARDO HERNÁNDEZ. *Estándares y Especificaciones de E-learning: Ordenando el Desorden* [online]. [Accessed 18 June 2016]. Available from: http://cvonline.uaeh.edu.mx/Cursos/Especialidad/Modulo1_PDF/ESTEM01T05E04.pdf

17. DAVID GRIFFITHS, SERGIO SAYAGO, ROCÍO GARCÍA and JOSEP BLAT. La aportación de IMS Learning Design a la creación de recursos pedagógicos reutilizables. . 18 October 2010.

18. SOTO CARDINAULT, Cynthia Guadalupe, MENÉNDEZ DOMÍNGUEZ, Víctor Hugo and VERA AGUILAR, Raúl Antonio. Interoperabilidad entre el LMS Moodle y las aplicaciones educativas de propósito específico utilizando servicios del IMS-LTI. *Apertura (Guadalajara, Jal.)*. 2016. Vol. 7, no. 2, p. 24–34.

19. JOSÉ L. MONTERO O'FARRILL and ELSA HERRERO TUNIS. *Las Herramientas de Autor en el proceso de producción de cursos en formato digital* [online]. July 2008. [Accessed 18 June 2016]. Available from: http://www.sav.us.es/pixelbit/pixelbit/articulos/n33/4.pdf

20. PABLO, Moreno Ger, IVÁN, Martínez Ortiz, LUIS, Sierra Rodríguez, José, PABLO, Fernández Manjón, Baltasar and PÉREZ SANZ, ANTONIO. *Estándares en e-learning y diseño educativo*. Ministerio de Educación, 2011. ISBN 978-84-369-5082-3.

21. EDWARD R. JONES. *Implications of SCORM and Emerging E-learning Standards On Engineering Education* [online]. [Accessed 18 June 2016]. Available from: http://aseegsw.com/past%20Proceedings/IB5.pdf

22. EDUARDO HERNÁNDEZ. Unidades de Aprendizaje, una propuesta de complemento a los Objetos de Aprendizaje. [online]. [Accessed 18 June 2016]. Available from: http://campus.usal.es/~teoriaeducacion/rev_numero_06_2/n6_02_art_hernandez.htm

23. IMS GLOBAL. IMS Learning Design Information Model Version 1.0 Final | IMS Global Learning Consortium. [online]. 20 January 2003. [Accessed 18 June 2016]. Available from: http://www.imsglobal.org/learningdesign/ldv1p0/imsld_infov1p0.html

24. JOSÉ H. CANÓS, PATRICIO LETELIER and CARMEN PENADÉS. *Metodologías ágiles en el Desarrollo de Software* [online]. [Accessed 18 June 2016]. Available from: http://ima.udg.edu/Docencia/07-08/3105200728/TodoAgil.pdf

25. DANIEL BURGOS and ROB KOPE. *Comunidades Virtuales, grupos y proyectos de investigación sobre IMS Learning Design. Status Quo, factores clave y retos inmediatos* [online]. 2005. [Accessed 18 June 2016]. Available from: http://www.uv.es/RELIEVE/v11n2/RELIEVEv11n2_6.pdf

26. OCHOA, M., BRITOS, Paola Verónica and GARCÍA MARTÍNEZ, Ramón. Una protofase de entendimiento del negocio para metodologías de desarrollo de sistemas. In : *XII Congreso Argentino de Ciencias de la Computación* [online]. October 2006. [Accessed 18 June 2016]. Available from: http://hdl.handle.net/10915/22831

27. DANIEL BURGOS. *Qué es IMS Learning Design y cómo modela Unidades de Aprendizaje* [online]. [Accessed 18 June 2016]. Available from: file:///F:/School/Tesis/Bibliography/UoL/OEM2005_IntroduccionAIMSLD.pdf

28. QUINTERO, Juan Bernardo, PÁEZ, Raquel Anaya de, MARÍN, Juan Carlos and LÓPEZ, Alex Bilbao. Un estudio comparativo de herramientas para el modelado con UML. *Revista Universidad EAFIT*. 6 June 2012. Vol. 41, no. 137, p. 60–76.

29. DUARTE, Ailin Orjuela and C, Mauricio Rojas. Las Metodologías de Desarrollo Ágil como una Oportunidad para la Ingeniería del Software Educativo. *Avances en Sistemas e Informática* [online]. 2 July 2009. Vol. 5, no. 2. [Accessed 29 June 2016]. Available from: http://www.revistas.unal.edu.co/index.php/avances/article/view/10037

30. CARLA MARTÍN VILLALBA. *Lenguajes de programación* [online]. Universidad Nacional de Educación a Distancia, UNED, 2011. [Accessed 18 June 2016]. Available from: https://dialnet.unirioja.es/servlet/libro?codigo=573586

31. RODRÍGUEZ SÁNCHEZ, Tamara. *Metodología de desarrollo para la Actividad productiva de la UCI*.

32. TERRENCE, PRATT and ZELKOWITZ, MARVIN. Lenguajes de programación: diseño e implementación. [online]. [Accessed 18 June 2016]. Available from: http://dspace.ucbscz.edu.bo/dspace/handle/123456789/1354

33. FOWLER, Martin and SCOTT, Kendall. *UML gota a gota*. Pearson Educación, 1999. ISBN 978-968-444-364-8.

34. Lenguajes de programación. [online]. [Accessed 18 June 2016]. Available from: http://www.monografias.com/trabajos-pdf/lenguajes-programacion/lenguajes-programacion.shtml

35. GRADY BOOCH, JIM RUMBAUGH and IVAR JACOBSON. El Lenguaje Unificado de Modelado. [online]. [Accessed 18 June 2016]. Available from: file:///C:/Users/Roy/AppData/Local/Temp/3E-UML.pdf

36. LUKE WELLING and LAURA THOMSON. *Desarrollo Web con PHP y MySQL* [online]. 2005. [Accessed 18 June 2016]. Available from: https://dialnet.unirioja.es/servlet/libro?codigo=314756

37. Arquitectura Modelo/Vista/Controlador. *Exequiel Catalani* [online]. 20 August 2007. [Accessed 18 June 2016]. Available from: https://exequielc.wordpress.com/2007/08/20/arquitectura-modelovistacontrolador/

38. LUIS MIGUEL CABEZAS GRANADO. *PHP 5* [online]. 2004. [Accessed 18 June 2016]. Available from: https://dialnet.unirioja.es/servlet/libro?codigo=318320

39. ROGER S. PRESSMAN. *Ingeniería del Software. Un Enfoque Práctico*. 7ma. 2005.

40. GAUCHAT, Juan Diego. *El gran libro de HTML5, CSS3 y Javascript*. Marcombo, 2012. ISBN 978-84-267-1782-5.

41. GUERRERO, Carlos A., SUÁREZ, Johanna M. and GUTIÉRREZ, Luz E. Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web. *Información tecnológica*. 2013. Vol. 24, no. 3, p. 103–114. DOI 10.4067/S0718-07642013000300012.

42. FRANGANILLO, Jorge. Html5: el nuevo estándar básico de la Web. *Anuario ThinkEPI*. 6 October 2011. Vol. 5, no. 0, p. 261–265.

43. TABARES, Ricardo Botero. Patrones Grasp y Anti-Patrones: un Enfoque Orientado a Objetos desde Lógica de Programación. *Entre Ciencia e Ingeniería*. 23 August 2011. Vol. 0, no. 8, p. 161–173.

44. DAVID FLANAGAN. *JavaScript. La Guía Definitiva* [online]. 2007. [Accessed 18 June 2016]. Available from: https://dialnet.unirioja.es/servlet/libro?codigo=316551

45. SOMMERVILLE, Ian. *Ingeniería del software*. 7ma. Pearson Educación, 2005. ISBN 978-84-7829-074-1.

46. SCOTT DUFFY. *How to do everything with JavaScript* [online]. 2003. [Accessed 18 June 2016]. Available from: https://dialnet.unirioja.es/servlet/libro?codigo=368588

47. RODRIGUEZ, Abraham Gutierrez and MARTINEZ, Raul. *Xml a Traves De Ejemplos*. Alfaomega Grupo Editor, 2002. ISBN 978-970-15-0716-2.

48. MICHAEL MORRISON. *XML al descubierto* [online]. Pearson Educación, S.A., 2000. [Accessed 18 June 2016]. Available from: https://dialnet.unirioja.es/servlet/libro?codigo=41709

49. MIGUEL ANGEL ALVAREZ. *Manual de jQuery*.

50. JAVIER J. GUTIÉRREZ. *¿Qué es un framework web?* [online]. [Accessed 18 June 2016]. Available from: http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf

51. RUBÉN LEDESMA. *Introduccción al Bootstrap* [online]. 2008. [Accessed 18 June 2016]. Available from: http://mail.tqmp.org/Content/vol04-2/p051/p051.pdf

52. MARC GIBERT GINESTÀ and OSCAR PÉREZ MORA. *Base de datos en PostgreSQL*.

53. MOHAMMED J. KABIR. *Servidor Apache 2* [online]. 2003. [Accessed 18 June 2016]. Available from: https://dialnet.unirioja.es/servlet/libro?codigo=318885

54. MICHAEL GLASS, YANN LE SCOUARNEC and ELIZABETH NARAMORE. *Desarrollo Web con PHP, Apache y MySQL* [online]. 2004. [Accessed 18 June 2016]. Available from: https://dialnet.unirioja.es/servlet/libro?codigo=314754

55. SÁEZ, Christina Holgado. Las Webquest en la docencia universitaria: aprendizaje colaborativo con LAMS. *Revista de Educación a Distancia* [online]. 2010. Vol. 0, no. 24. [Accessed 18 June 2016]. Available from: http://revistas.um.es/red/article/view/125281

56. ROBYN PHILIP and JAMES DALZIEL. *Designing Activities for Student Learning Using the Learning Activity Management System (LAMS)*.

57. PEREZ SUAREZ, José Javier. Teambox Software Appliance. [online]. 13 June 2011. [Accessed 18 June 2016]. Available from: http://openaccess.uoc.edu/webapps/o2/handle/10609/8119

58. Teambox gestor de tareas y proyectos online. [online]. [Accessed 18 June 2016]. Available from: http://www.elchecibernetico.com/software/teambox-programa-para-gestion-de-proyectos

59. ALVARO MATÍNEZ GUAITA. Teambox, gestor de proyectos colaborativo. [online]. 25 April 2011. [Accessed 18 June 2016]. Available from: http://www.desarrolloweb.com/de_interes/teambox-gestor-proyectos-5138.html

60. MARTÍ, Jordi. Com8s, una gran herramienta colaborativa para Educación. [online]. 12 August 2010. [Accessed 18 June 2016]. Available from: http://www.xarxatic.com/com8s-una-gran-herramienta-colaborativa-para-educacion/

61. CRAIG LARMAN. *UML y Patrones*. 2da. [no date].

62. MACARIO POLO USAOLA. *Mantenimiento Avanzado de sistemas de Información. Pruebas del Software* [online]. [Accessed 18 June 2016]. Available from: http://moodle.univo.com.mx/ingenieria/moodledata/temp/backup/1293679217/course_files/semana4/pruebas_avanzadas_de_software.PDF

63. THE APACHE SERVER FOUNDATION. Welcome to The Apache Software Foundation! [online]. 2016. [Accessed 18 June 2016]. Available from: http://apache.org/

64. GITHUB PROJECT. Bootstrap. [online]. [Accessed 18 June 2016]. Available from: http://getbootstrap.com/2.3.2/