

Universidad de las Ciencias Informáticas
Facultad 5



**Estandarización de secuencias de tareas de una aplicación
Web para otros entornos virtuales de aprendizajes.**

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

Autores
Yuder Rodríguez Gutierrez.
Leonardo Morales Díaz.

Tutor
Lic. Luís Gabriel Vicedo Carabaloso.

Ciudad de la Habana
Junio 2007

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Leonardo Morales Díaz

Firma del Autor

Yuder Rodríguez Gutierrez

Firma del Autor

Lic: Luís Gabriel Viciado Carabaloso

Firma del Tutor

DATOS DEL CONTACTO

Lic. Luis Gabriel Viciado Caraballoso

Graduado de Licenciatura en Educación en la especialidad de Física en el Instituto Superior Pedagógico de la Ciudad de Camaguey. Realizó estudios de Postgrado en la Escuela de Física de la Universidad de La Habana, en la Universidad de Oriente y en Instituto Superior Politécnico José Antonio Echeverría. Profesor Auxiliar desde 1998. Impartió clases de Matemática Superior, Física y Computación en el Instituto Superior Agrícola de Ciego de Ávila. Graduado en Inglés, Francés, Ruso y Portugués. Ha presentado trabajos en eventos nacionales e internacionales relativos al uso de radiaciones ionizantes con fines mutagénicos, Dosimetría de Altas Dosis y uso de nuevas tecnologías con fines educativos. Realizó estudios de postgrado en el Centro de Referencia para la Educación de Avanzada de la CUJAE, sobre el empleo de laboratorios virtuales en las carreras de ingeniería. Imparte docencia en el Departamento de la Especialidad en las disciplinas de Programación y Sistemas Digitales.

Correo electrónico: viciado@uci.cu

Teléfono: 835-8757

AGRADECIMIENTOS

A nuestro comandante Fidel por brindarnos esta oportunidad única.

A nuestros padres y familiares que estuvieron apoyándonos en todo momento.

A nuestro tutor: Luís Gabriel Vicedo por brindarnos todo su apoyo y esfuerzo.

A nuestros amigos que comparten con nosotros desde un inicio este proyecto de la revolución.

A todos aquellos que de una forma u otra han contribuidos con nuestra formación.

A todos, GRACIAS.

DEDICATORIA

Leonardo:

Dedico el presente trabajo a mis padres Leonardo Morales Hernández y Odalis Díaz Rodríguez que tanto apoyo y esfuerzo han dedicado para que mis metas sean cumplidas, a mi hermana Yordailis, a mis abuelas Eloina y Angélica, a mi mujer Lisbety que tanto me ha ayudado. A mis tíos Manuel, Yamir, Misael, Juana, Segundito, Daisy, Senaida, Estrella, Ignacio, Midiala, a mi amigo y compañero de tesis Yuder. Agradezco igualmente a mis vecinos y amigos del barrio: Clara, Pepe, Herenia, la formidable Panchita, Pipi el Bernal, a Yulier. A todos aquellos que empezaron con nosotros y que ahora no están pero siguen en contacto, brindándonos su apoyo. A la penosa Rosayne, Yailin, a la salsera Lenna, a todos mis compañeros de cuarto: Anddy, Reinier, Alain, Omar, al Piti, al Lechón, a todo el colectivo del proyecto Laboratorios Virtuales, en fin, a todas esas personas que nunca olvidaremos y que de una forma u otra ayudaron a construir este sueño.

Yuder:

Dedico este trabajo a mis padres que durante todos estos años de estudios me han apoyado, especialmente a mi querida madre Margarita Gutierrez y mi padrastro Niño que siempre han confiado en mi y me han guiado para triunfar en mis estudios. A mis hermanos Pablo, Asiel y a la más pequeña Margarita que quiero con la vida. A Yariurmis la madre de mis niñas, por su apoyo, cariño y confianza. A mis niñas Lisdadis y Lianet que son mi tesoro más grande. A mi abuelita Saida por darme ánimos para seguir adelante. A mi prima Yumi por ser mi segunda madre, a mi tío Alberto y mi tía Marali por su confianza, amor, consejos, en fin a todos mis familiares que de una forma u otra contribuyeron a que este sueño se hiciera realidad. Agradezco igualmente mis amigos y hermanos Leo y Fofi por su apoyo, confianza, dedicación y preocupación porque todo saliera bien, a Yulio porque siempre estuvo presente dándome ánimos y fuerza. A Yeni por su locura y cariño, por buena amiga y compañera. A Jessy por su amor, cariño y amistad, por ser tan especial, por estar siempre presente, por su confianza, por saber ser lo mejor y por hacer mi sueño realidad. A mis compañeros de quinto año: a Yuriel, Raydel y Piti, por su apoyo, por darme confianza, por compartir buenos y malos momentos, por su maravillosa amistad durante todos estos años. A mis compañeros de proyectos, brigadas anteriores y mis compañeros de primer año, en fin a todos los que han hecho posible que hoy haya logrado cumplir esta gran meta.

RESUMEN

En el área educacional se encuentra el e-Learning, donde se destacan un conjunto de plataformas y aplicaciones que facilitan el aprendizaje a distancia, permitiendo la interactividad virtual entre el profesor y sus alumnos, facilitando el intercambio de información entre ellos. El problema que surge es cómo la información educacional puede ser funcional en diferentes entornos virtuales. En este contexto, surge la necesidad de estudiar y desarrollar soluciones al problema anteriormente mencionado, buscando la integración de nuevas tecnologías en aplicaciones Web.

Con esta investigación se definen los elementos funcionales necesarios para que una secuencia de tareas y sus contenidos sean reutilizados y reconocidos en entornos de aprendizaje cuyo diseño fue concebido de acuerdo a estándares internacionales definidos por la Tecnología Educativa. Cabe destacar, que el estándar utilizado es SCORM (Sharable Content Object Reference Model) de la iniciativa de ADL (Advanced Distributed Learning), el cual plantea las reglas para almacenar y ordenar los contenidos en forma jerarquizada, relacionándolos con un conjunto de archivos, denominados Objetos de Aprendizajes. También se realizó un análisis de las tendencias y tecnologías existentes, proponiéndose las más adecuadas para el trabajo. Se realiza todo un proceso de ingeniería que se inicia por el levantamiento de requisitos y pasa por la realización de diagramas en el proceso de análisis y diseño los cuales permiten presentar un modelo de la aplicación a desarrollar.

Palabras Clave:

Objeto de aprendizaje, SCORM, e-learning, entornos virtuales, estándares, Tecnología Educativa.

ÍNDICE

AGRADECIMIENTOS	I
DEDICATORIA	II
RESUMEN	III
INTRODUCCION	1
CAPÍTULO	1
1 FUNDAMENTACIÓN DEL TEMA	1
Introducción.....	1
1.1 Estándares para e- Learning.....	1
1.1.1 Antecedentes.....	2
1.1.2 Ventajas aportadas por los estándares.....	3
1.1.3 Organizaciones que se dedican a la estandarización.....	4
1.2 Learning Management System o LMS	5
1.2.1 Tipos de LMSs.....	5
1.3 Estándares más utilizados.....	6
1.3.1 IMS	6
R-1.3.1.1 IMS Content Packaging.....	6
1.3.2 SCORM.....	7
R-1.3.2.1 Soportes de SCORM.....	7
R-1.3.2.2 Componentes del Modelo de Contenidos SCORM.....	8
1.3.2.2.1 Asset.....	8
1.3.2.2.2 SCO (Sharable Content Object).....	9
R-1.3.2.3 Metadatos.....	9
1.3.2.3.1 Metadatos de SCO.....	10
1.3.2.3.2 Metadatos de Assets.....	10
1.3.2.3.3 Aplicación de los Metadatos.....	10
R-1.3.2.4 Paquete de Contenidos.....	11
R-1.3.2.5 Manifest.....	12
R-1.3.2.6 Componentes de un Manifest.....	13
1.3.2.6.1 Elemento Metadata.....	13
1.3.2.6.2 Elementos <organizations> y <organization>.....	13
1.3.2.6.3 Elemento <title>.....	14
1.3.2.6.4 Elemento <metadata>.....	14
1.3.2.6.5 Elemento <item>.....	15
1.3.2.6.5.1 Atributos de item:.....	15
1.3.2.6.6 Elemento <resources>.....	16
1.3.2.6.7 Elemento <file>.....	17
1.4 Propuesta del estándar a utilizar.....	17
Conclusión	17
CAPÍTULO	19
2 TENDENCIAS Y TECNOLOGÍAS ACTULES	19
Introducción.....	19
2.1 La política de migración hacia software libre.....	19
2.2 Lenguajes de programación.....	19
2.2.1 Personal Home Page (PHP).....	20
2.2.2 Java.....	21
2.2.3 eXtensible Markup Language, o Lenguaje de Marcado (XML)	21
2.2.4 Modos de Procesamiento de un documento XML.....	22
2.3 Parsers.....	22
2.3.1 Clasificación de los Parsers	22
2.4 IDE	23

2.4.1	NuSphere PHPEd.....	23
2.4.2	Zend Studio	23
2.5	Sistema gestor de bases de datos.	24
2.5.1	MySQL.....	24
2.5.2	SQL	25
2.5.3	ORACLE.....	25
2.5.4	PostgreSQL	26
2.6	Metodologías de Desarrollo de Software	26
2.6.1	El Proceso Unificado de Racional (RUP).	26
2.6.2	Programación Extrema (XP).....	27
2.7	Selección de las tecnologías a utilizar en la propuesta de solución.....	28
	Conclusiones.....	29
	CAPÍTULO	30
3	CARACTERÍSTICAS DEL SISTEMA.....	30
	Introducción.....	30
3.1	Modelo de dominio.	30
3.1.1	Diagrama de Modelo de Dominio.	31
3.1.2	Definición de las entidades y los conceptos principales.....	31
3.2	Requerimientos	32
3.2.1	Requerimientos Funcionales:	32
3.2.2	Requerimientos no Funcionales:	33
3.3	Actores del sistema	35
3.4	Casos de Usos del sistema.	35
3.4.1	Diagrama de Casos de uso del sistema.....	36
	Figura 7: Diagrama de Casos de uso del sistema.	36
3.4.2	Descripción de los Casos de Usos.....	36
	Tabla 4: Gestionar Organización.....	37
	Tabla 5: Gestionar Items.	39
	Conclusiones.....	42
	CAPÍTULO	43
4	CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA	43
	Introducción.....	43
4.1	Análisis	43
4.1.1	Diagramas de Clases de Análisis.....	44
4.2	Diseño.	46
4.2.1	Arquitectura de una aplicación Web.....	47
4.2.2	Diagramas de Clases del diseño.....	48
	Caso de Uso: Importar Contenido.....	48
	Caso de Uso: Gestionar Organización.....	49
	Caso de Uso: Gestionar item	50
	Caso de Uso: Gestionar Metadatos	51
	Caso de Uso: Empaquetar.....	52
4.3	Diagramas de Secuencia.	53
Caso de Uso: Importar Contenido.....		53
Caso de Uso: Gestionar Organización.....		53
Caso de Uso: Gestionar item		55
Caso de Uso: Gestionar Metadatos		57
Caso de Uso: Empaquetar		58
4.3.1	Descripción de las clases.....	59
4.3.2	Clases Persistentes.....	63
4.3.3	Modelo de Datos.	64
R-4.3.3.1	El diagrama del modelo de datos	65

Se corresponde con la representación física de la base de datos.....	65
Figura 32: El diagrama del modelo de datos.	65
R-4.3.3.2 Descripción de las tablas.....	66
4.3.4 Modelo de Despliegue.....	69
4.3.5 Principios de diseño.	69
R-4.3.5.1 Interfaz de usuario.....	70
R-4.3.5.2 Tratamiento de errores.....	70
Conclusión.....	70
CONCLUSIONES.....	72
RECOMENDACIONES.....	74
BIBLIOGRAFÍA.....	75
ANEXOS.....	76
GLOSARIO DE TERMINOS.....	86

INTRODUCCION

La educación se encuentra en un proceso de cambio en lo que respecta a la enseñanza por el empleo de Internet, donde se transita de la presentación de los contenidos tradicionales de aprendizaje con nuevos formatos y herramientas, a un enfoque que se orienta más a la planificación y a la aplicación de estándares tecnológicos y educativos.

En el ámbito académico esto provoca que las universidades y los centros de desarrollo tecnológico del primer mundo, marquen las pautas en cuanto a la manera de utilizar las tecnologías e intercambiar información en los servicios educativos.

Precisamente el e-Learning, surgido en estos centros es considerado un paradigma educativo que integra el uso de la Tecnología y la Didáctica, para lograr el desarrollo de un proceso de enseñanza-aprendizaje en la educación a distancia, la cual ofrece múltiples ventajas, entre las que se encuentran: la reducción de costos por ofrecer cursos a más participantes que lo tradicional y en la movilidad de dichas personas para recibirlos, así como en la flexibilidad de horarios, factor de suma importancia que permite al estudiante planificar el curso de la mejor forma posible. Además, la interacción entre los participantes en los cursos, despierta la motivación entre ellos, ayudándolos a ser más activos en el proceso de enseñanza.

El e-Learning engloba el desarrollo de cursos y entrenamientos a través de Internet, en una Intranet e incluso mediante CD-ROM, con contenidos diversos de acuerdo a los requerimientos específicos de cada individuo y organización. También integra el análisis estratégico sobre el uso de la información para el aprendizaje, lo que permite fusionar a los usuarios al mundo de la tecnología.

En un entorno de e-Learning la institución educativa debe proporcionar para el desarrollo del aprendizaje componentes de tipo informativo tales como texto, multimedia, video o audio, así como herramientas de comunicación a través de un sitio Web de acceso restringido. La atención personalizada de los profesores hacia los estudiantes y la interacción entre los sujetos se realiza por medio del correo electrónico, a través de los Chat, (Ej.: ICQ, MSN Messenger), foros de discusión o incluso videoconferencias.

En la Universidad de Ciencias Informáticas (UCI) se vienen desarrollando diferentes proyectos con el objetivo de contribuir a la mejora del aprendizaje, insertando al estudiantado en el mundo de la Tecnología Educativa. Entre esos proyectos se encuentra el de Laboratorios Virtuales, encargado de virtualizar prácticas de laboratorios que permitan el mejoramiento de habilidades en los estudiantes por intermedio de una aplicación Web. Esta aplicación debe permitir la interacción entre los sujetos que aprenden y propiciando en aprendizaje colaborativo.

Actualmente este proyecto está elaborando una aplicación Web con características similares a la aplicación LAMS (Learning Activities Management System), que es un entorno virtual de enseñanza-aprendizaje orientado a la construcción de secuencias de tareas, donde el profesor queda en libertad de organizar las mismas según su preferencia. Dichas secuencias constan de actividades simples: lectura de documentos, un debate en grupo, una sesión de charla online, la publicación de comentarios, etc.

Con esta nueva aplicación aparece la necesidad dentro del proyecto, de garantizar la interoperabilidad y reutilización de las secuencias de tareas, empleando los estándares ampliamente descriptos en la Tecnología Educativa, de manera que puedan ser usados en entornos virtuales de aprendizaje tales como Moodle o Blackboard.

Por lo tanto el problema científico radica en cómo garantizar tecnológicamente que las secuencias de tareas creadas en la aplicación Web sean totalmente funcionales en entornos virtuales de aprendizaje.

El objeto de estudio lo constituyen los Estándares Internacionales para la Tecnología Educativa.

El campo de acción es el Diseño del modelo SCORM para secuencias de tareas en prácticas de laboratorios virtuales.

El objetivo general de este trabajo es el de diseñar un componente de funcionalidad que permita la compatibilidad y reusabilidad de secuencias de tareas en la práctica de laboratorio virtual.

Se definieron los objetivos específicos a desarrollar para dar cumplimiento al trabajo:

1. Estudiar los estándares internacionales definidos por la Tecnología Educativa.
2. Analizar la situación actual de la compatibilidad entre diferentes entornos virtuales de aprendizaje.
3. Definir el estándar a usar para que las secuencias de tareas sean compatibles y reutilizables en entornos de aprendizaje.
4. Modelar los elementos del componente de funcionalidad definido por el estándar seleccionado.

Con este trabajo se indican los elementos esenciales para el diseño de un componente de funcionalidad, de manera que una secuencia de tareas diseñadas por un docente en la aplicación Web quede debidamente estandarizada y pueda reutilizarse en plataformas de aprendizaje que acepten el estándar internacional seleccionado.

El documento está estructurado en cuatro capítulos los cuales muestran los resultados obtenidos:

Capítulo 1 Fundamentación del tema: En el presente capítulo se realiza una descripción general de los estándares para el e-Learning, así como los componentes que define la iniciativa ADL-SCORM para construir recursos de aprendizaje.

Capítulo 2 Tendencias y Tecnologías Actuales: Se realiza un estudio referente a las tendencias y tecnologías existentes en la actualidad que deben considerarse para hacer la selección de aquellas que se utilizarán.

Capítulo 3 Características del Sistema: Se desarrolla el modelo de dominio, el cual sirve de guía para determinar qué se va a desarrollar. Se determinan las funcionalidades del sistema propuesto y se describen en detalle.

Capítulo 4 Construcción de la solución propuesta: Se abordan aspectos relacionados con la construcción de la solución propuesta, se modelan los diagramas de análisis y diseño utilizando las notaciones del UML, se plantea el modelo de datos del sistema así como el diagrama de despliegue.

1 FUNDAMENTACIÓN DEL TEMA

Introducción

En el presente capítulo se realiza una descripción general de los estándares para el e-Learning y se analizan sus ventajas desde el punto de vista tecnológico. Por otro lado, se indican las organizaciones dedicadas a estudio y divulgación de los estándares, así como los componentes que define la iniciativa ADL-SCORM para construir recursos de aprendizaje. Finalmente se describen los elementos que contiene el componente más importante de ADL-SCORM denominado manifiesto, dando una breve explicación de la funcionalidad de los mismos.

1.1 Estándares para e- Learning.

Al hablar sobre un estándar e-Learning, se está refiriendo a un conjunto de reglas en común para las compañías dedicadas a la tecnología e-Learning. Estas reglas especifican cómo los fabricantes pueden construir cursos on-line y las plataformas sobre las cuales son impartidos estos cursos de tal manera de que puedan interactuar unas con otras. Estas reglas proveen modelos comunes de información para cursos e-Learning y plataformas LMS, que básicamente permiten a los sistemas y a los cursos compartir datos o “hablar” con otros. Esto también da la posibilidad de incorporar contenidos de distintos proveedores en un solo programa de estudios.

En el mercado existen tanto LMS (Learning Management System) como Courseware (o Contenidos) de muchos fabricantes distintos. Por ello se hace necesaria una normativa que compatibilice los distintos sistemas y cursos a fin de lograr dos objetivos:

1. Que un curso de cualquier fabricante pueda ser cargado en cualquier LMS de otro fabricante.
2. Que los resultados de la actividad de los usuarios en el curso puedan ser registrados por el LMS.(MAURER 2002-2004)

Entre los diferentes estándares que se desarrollan hoy en día para la industria del e-Learning se pueden encontrar los siguientes tipos:

1. Sobre el Contenido o Curso: Ofrece estructuras de los contenidos, para el empaquetamiento de contenidos y seguimiento de los resultados.

2. Sobre el Alumno: Ofrece almacenamiento e intercambio de información con relación al alumno, competencias (habilidades) y aspectos relativos a la privacidad y la seguridad.
3. Sobre la interoperabilidad: Ofrece la integración de componentes del LMS y de interoperabilidad entre múltiples LMS.

1.1.1 Antecedentes.

Aunque las tecnologías aplicadas a la educación han evolucionado considerablemente en las últimas décadas, no existe consenso en esta evolución debido a que los LMS actuales son incapaces de intercambiar totalmente información entre ellos. Una de las dificultades radica en que la información se almacena en distintos sistemas de registros, lo que obstaculiza sus transferencias.

Para prevenir esta situación, es preciso disponer de estándares que garanticen la interoperabilidad de los LMS. Y no sólo para el contenido, sino también para la forma de empaquetarlos, definir su secuenciación y cualquier otra tarea que se precise para poder transferirlos entre distintas plataformas y entornos.

Desde hace algunos años varias organizaciones trabajan en la definición de estándares para los distintos modelos que forman la arquitectura de las plataformas de formación: alumnos, contenidos, pruebas, estrategias docentes, etc.

En el año 2000, aparecen varios proyectos entre ellos el Sistema de Gestión Instruccional o IMS (Instructional Management System), los cuales apoyan al grupo de trabajo: Metadatos de Objetos de Aprendizaje o LOM (Learning Objects Metadata), para obtener en el 2002, el primer estándar del Instituto de Ingenieros Eléctricos y Electrónicos o IEEE (Institute of Electrical and Electronic Engineers), sobre Objetos de Aprendizaje (ALVAREZ G., LUIS 2003, 24 noviembre).

La iniciativa ADL (Advanced Distributed Learning) unifica estándares anteriores, de forma que recoge las características más destacadas de cada una de ellos, de IMS recoge la descripción de cursos en XML, como forma de separar contenidos de presentación y de AICC (Aviation Industry CBT Committe), recoge el mecanismo de intercambio de información mediante APIs (Application Programming Interface). Con todo ello establece una proposición de estándar llamado SCORM, que define una serie de pasos a seguir para la implementación detallada de contenidos, que permita que los LMS que lo cumplan, puedan intercambiarlos, es decir, lograr la interoperabilidad, la reusabilidad y la adaptabilidad que se persigue.

La última versión divulgada por ADL es SCORM 2004, sin embargo la versión SCORM 1.2 es la más usada tanto en LMS como en herramientas certificadas. Técnicamente es especificada en tres libros: The SCORM Overview, The SCORM Content Agregation Model y The SCORM Run Time Environment.

Además, cabe destacar que ADL realiza dos acciones, que son: “Certifica” y “Valida”. Por otra parte, divide la tecnología e-learning en componentes funcionales. Los dos principales componentes son: Los LMS y los SCOs (Sharable Content Objects). Los SCO hacen referencia a objetos de aprendizaje reutilizables y estandarizados, los cuales son una combinación de “Asset” (ALVAREZ G., LUIS ; ESPINOZA, DANIELA 2001).

1.1.2 Ventajas aportadas por los estándares.

En el e-Learning, una de las principales funciones de los estándares es servir como facilitadores de la durabilidad y de la reutilización en el tiempo de los contenidos y de la interoperabilidad, es decir, facilitar el intercambio de los contenidos entre diversas plataformas y sistemas. Es importante señalar que los estándares no limitan la iniciativa o creatividad personal a la hora de desarrollar cualquier contenido. Su uso no es sólo en educación a distancia sino que también son útiles para otros planteamientos educativos, ya que la existencia de contenidos educativos reutilizables puede ser de gran ayuda para simplificar el trabajo de los docentes.

Existen infinidad de ventajas asociadas a la utilización generalizada de estándares de e-Learning para todas las partes implicadas en el proceso de enseñanza-aprendizaje. Entre ellas cabe mencionar las siguientes:

1. Desde el punto de vista del de los clientes o consumidores tanto institucionales como individuales, los estándares evitan quedarse atrapado por las tecnologías propietarias. Los costes se reducen al sustituir los desarrollos propios por tecnología “plug and play” de modo que, por ejemplo, una institución pueda cambiar de LMS sin tener que empezar desde el principio perdiendo toda o gran parte de la información que ya tenía en su LMS anterior.
2. Los estándares permiten, desde el punto de vista de los productores de contenidos educativos, que el formato de producción sea único y pueda ser utilizado en cualquier plataforma de distribución. Además la existencia de estándares facilita su labor, al tener acceso a almacenes de contenidos reutilizables, y les permite la creación de contenidos modulares de más fácil mantenimiento y actualización.
3. Desde el punto de vista de los alumnos, los estándares implican mayor posibilidad de elección del producto educativo. Además implican que los resultados de su aprendizaje (créditos o certificados) tengan mayor portabilidad.

Por otra parte, se destacan otras ventajas y propiedades beneficiosas que se obtienen con la aplicación de los estándares.

1. Interoperabilidad. Que se pueda intercambiar y mezclar contenido de múltiples fuentes y se pueda usar directamente en distintos sistemas. Que sistemas diferentes puedan comunicarse, intercambiar información e interactuar de forma transparente.
2. Reusabilidad. Que el contenido pueda ser agrupado, desagrupado y reutilizado de forma rápida y sencilla. Que los objetos de contenido puedan ensamblarse y utilizarse en un contexto distinto a aquél para el que fueron inicialmente diseñados.
3. Gestionabilidad. Que el sistema pueda obtener y trazar la información adecuada sobre el usuario y el contenido.
4. Accesibilidad. Que un usuario pueda acceder el contenido apropiado en el momento justo y en el dispositivo correcto.
5. Durabilidad. Que los consumidores no queden atrapados en una tecnología propietaria de una determinada empresa. Que no haya que hacer una inversión significativa para lograr la reutilización o la interoperabilidad.
6. Escalabilidad. Que las tecnologías puedan configurarse para aumentar la funcionalidad de modo que se pueda dar servicio a más usuarios respondiendo a las necesidades de la institución, y que esto no exija un esfuerzo económico desproporcionado.

1.1.3 Organizaciones que se dedican a la estandarización.

El AICC (Aviation Industry CBT Committee) El objetivo de esta organización es conseguir una formación eficiente, sostenible y con un coste eficaz (ajustado a los resultados).

El IEEE (Instituto de Ingenieros Eléctricos y Electrónicos), también ha desarrollado recomendaciones y estándares técnicos que recalcan las mejores prácticas. El estándar de uso más extendido de esta organización es LOM (Metadatos de Objetos de Aprendizaje), que define las características de los elementos y estructuras de aprendizaje.

El IMS (Instructional Management System) Su objetivo fue la creación de un formato que pusiese en práctica las recomendaciones de la IEEE y la AICC.

La iniciativa ADL (Advanced Distributed Learning) Este organismo recogió lo mejor de las anteriores iniciativas (el sistema de descripción de cursos en XML de la IMS, y el mecanismo de intercambio de información mediante una API de la AICC) y las refundió y mejoró en su propio estándar.

1.2 Learning Management System o LMS

Los LMS proporcionan la funcionalidad al e–Learning, ya que es el espacio en donde los estudiantes pueden planear, acceder, enviar y gestionar el aprendizaje por sí mismos, interactuar con otros alumnos, con los monitores y los facilitadores.

Básicamente se trata de un software para servidores de Internet e(o) Intranet que se ocupa de:

1. Gestionar los usuarios: inscripción, control de sus aprendizajes e historial, generación de informes, etc.
2. Gestionar y lanzar los cursos, realizando un registro de la actividad del usuario: tanto los resultados de los tests y evaluaciones que realice, como de los tiempos y accesos al material formativo.
3. Gestionar los servicios de comunicación: que son el apoyo al material online, foros de discusión, charlas, videoconferencia; programarlos y ofrecerlos conforme sean necesarios.

La existencia de instructores permiten en los LMS crear o brindar contenidos, monitorear la participación de los estudiantes dentro del sistema, además de evaluar la actuación y desarrollo de los mismos, al mismo tiempo, un LMS promueve en los estudiantes diferentes habilidades para el uso de herramientas interactivas como foros de discusión, videoconferencias, tele conferencias, chat y correo electrónico.

1.2.1 Tipos de LMSs.

Constantemente aparecen nuevas versiones de los LMS, tanto comerciales como de software libre o OSS (Software de código fuente abierto), que soportan el uso de algunas especificaciones y por lo menos, la importación o la exportación de cursos completos empaquetados según IMS o SCORM. Actualmente existen más de doscientos paquetes LMSs en el mercado. En cuanto a LMS comerciales cabe mencionar WebCT y Blackboard como dos de los más utilizados (actualmente están en proceso de fusión para crear una única plataforma), también están eCollege y LearningSpace.

En cuanto a LMS de software abierto destacan Moodle, LRN, Claroline, Dokeos y, últimamente, LAMS ya que incluye soporte para el desarrollo de unidades de aprendizaje.

1.3 Estándares más utilizados.

1.3.1 IMS

Actualmente IMS (Global Learning Consortium, Inc) es el principal promotor y desarrollador de especificaciones abiertas, y que cubren más aspectos de la enseñanza electrónica. Este trabajo, conjuntamente con el desarrollado por ADL en su modelo de referencia SCORM, y por IEEE LTSC con su propuesta de Metadatos para objetos de aprendizaje, son los que están teniendo una mayor repercusión en e- Learning.

El objetivo de IMS es definir especificaciones que hagan posible la interoperabilidad de aplicaciones y servicios de enseñanza distribuida.(MIGUEL, REBOLLO PEDRUELO Diciembre 2004).

R-1.3.1.1 IMS Content Packaging.

El objetivo de esta especificación es permitir la distribución de contenidos reutilizables e intercambiables, es decir, describe el modo en el que se debe empaquetar el contenido educativo para que pueda ser procesado por otro sistema LMS diferente. Ofrece una forma de empaquetar (en un archivo comprimido tipo .zip) los contenidos educativos tales como cursos individuales, conjuntos de cursos, o cualquier tipo de recurso necesario en el proceso educativo (por ejemplo, evaluaciones o exámenes). Al distribuir una serie de contenidos empaquetados según el Content Packaging de IMS, existe un documento fundamental que es el *Manifiesto*. Dicho documento es un fichero XML en el que se describe la estructura de los contenidos incluidos en el paquete. (MIGUEL, REBOLLO PEDRUELO Diciembre 2004).

Dicha descripción se realiza a dos niveles diferentes:

1. Por un lado, se describe cada uno de los *Recursos* del paquete. En una primera aproximación se puede hacer una relación casi directa entre un Recurso y un fichero con contenidos visualizables (un Objeto de Aprendizaje) como pueden ser ficheros HTML, animaciones en Flash, etc. En realidad, en cada Recurso se puede incluir información sobre los ficheros que componen dicho Recurso, el tipo de los mismos (que puede ser uno de los tipos ya definidos por el estándar o una extensión de los propuestos) y, opcionalmente, Metadatos con información adicional sobre dicho Recurso.
2. Por otro lado, en el Manifiesto se describe como están organizados dichos Recursos, es decir, como se estructura el contenido del paquete. Esto se implementa mediante las *Organizaciones*. Una organización es una vista (o recorrido) de una posible ordenación jerárquica (actualmente en forma de árbol) de los *Recursos* de un paquete. El estándar permite que un Manifiesto contenga distintas organizaciones sobre los *Recursos* del paquete, dando así lugar a distintas vistas o

“cursos” a partir de los mismos contenidos. El elemento básico de estructuración que se usa al definir las organizaciones son los Items. A cada Item se le puede asociar un Recurso, de modo que el árbol de Items es, efectivamente, una estructuración de los Recursos del paquete.

1.3.2 SCORM.

Corresponde a un conjunto de estándares técnicos interrelacionados para desarrollar enseñanza de contenidos vía WEB. Su estructura se basa en un Modelo de Agregación de Contenidos y en un Ambiente de Enseñanza en Tiempo Real.

Proporciona un marco de trabajo y una referencia de implementación detallada que permite a los contenidos y a los sistemas usar SCORM para “hablar” con otros sistemas, logrando así interoperabilidad, reusabilidad y adaptabilidad.

Al implementar SCORM se tiene la posibilidad de la libre movilidad (interoperabilidad) de contenidos desde una plataforma de administración de enseñanza (LMS) a otra, la reutilización de contenidos gracias a la interoperabilidad entre plataformas, una simple y eficiente administración de los cursos y de sus usuarios. Facilita la adaptación de contenidos (propios o importados) en cada plataforma. Permite la administración de los contenidos en repositorios temáticos, un fácil empaquetamiento de contenidos en cursos.

SCORM no cubre todos los aspectos relacionados con la tecnología e-Learning, por ejemplo no especifica cómo la información resultante del seguimiento de los alumnos debe ser almacenada, o qué modelos de aprendizaje deben ser usados. Tampoco cuenta con especificaciones relativas a la información de los alumnos.

Debido a todo esto, se puede decir que si bien SCORM es un estándar en potencia, su aplicación no estandariza la educación, lo único que persigue es establecer un formato común para los objetos que se manejan dentro de un sistema e-Learning de manera de posibilitar su intercambio con otros sistemas o ambientes.(MIGUEL, REBOLLO PEDRUELO Diciembre 2004)

R-1.3.2.1 Soportes de SCORM.

El estándar SCORM en particular es muy importante para permitir al LMS, publicar, distribuir, compartir y hacer seguimiento del conocimiento.

Ejemplos:

1. MOODLE: soporta SCORM 1.2 y no es completamente compatible con los siguientes estándares, SCORM 1.3; AICC; IMS, (IEEE) entre otros. MOODLE no tiene una característica activa de compartir y reutilizar contenidos.

2. BlackBoard 6: soporta los siguientes estándares: SCORM 1.2, IMS Metadata 1.2.1, IMS Content Packaging 1.1.2 y Microsoft LRN 3.0. Incluye herramientas para facilitar la migración de cursos desde distintas versiones del software.
3. WebCT: El fabricante soporta la migración entre las siguientes plataformas: Angel, BlackBoard, FirstClass, Prometheus IntraLearn. El sistema puede importar y exportar contenidos usando IMS Content Packaging standard. (MIGUEL, REBOLLO PEDRUELO Diciembre 2004)

R-1.3.2.2 Componentes del Modelo de Contenidos SCORM.

1.3.2.2.1 Asset

Son representaciones electrónicas de multimedia, tales como texto, imágenes, sonidos, videos o cualquier otro tipo de datos que pueda ser facilitado a un cliente Web. Los Asset individuales no son lanzados por un sistema LMS que cumpla con SCORM. En otras palabras un Asset es un archivo cualquiera que puede ser utilizado.

Un Asset puede ser descrito con Asset Metadata para permitir la búsqueda y descubrimiento sin la necesidad de que los datos sean almacenados, posibilitando así oportunidades de reusabilidad. El mecanismo para unir Asset con Metadata de Asset es el Content Package.

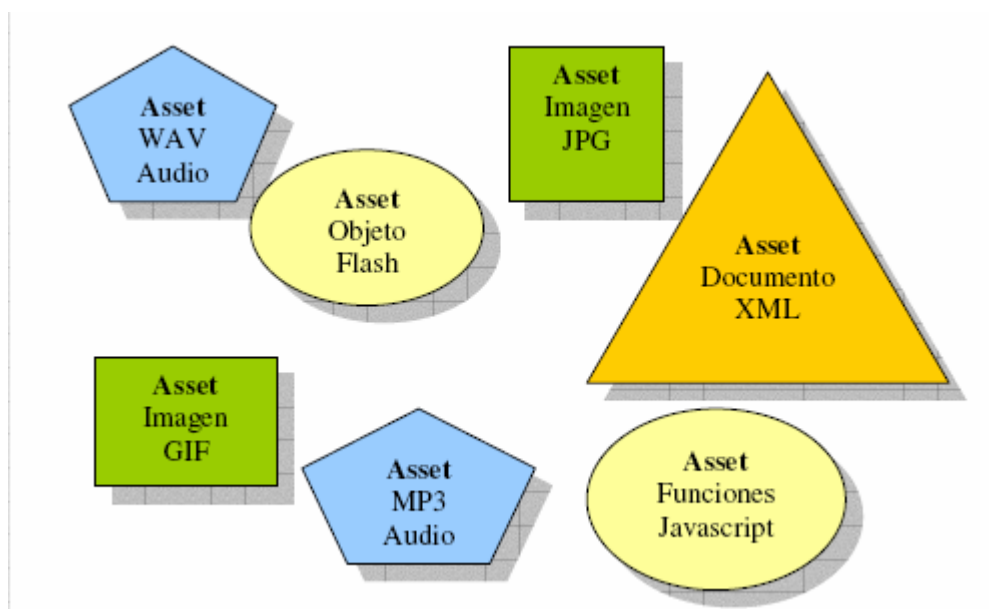


Figura 1: Ejemplo de Asset Cada figura representa un Asset.

1.3.2.2 SCO (Sharable Content Object).

Un SCO es un conjunto de recursos reutilizables, archivos u objetos de aprendizaje (Learning Object) reutilizables para que cumplan un fin específico dentro de una plataforma E- Learning. Estos archivos son casi siempre documentos, imágenes, videos y audio. Por lo tanto un SCO es un conjunto de Assets.

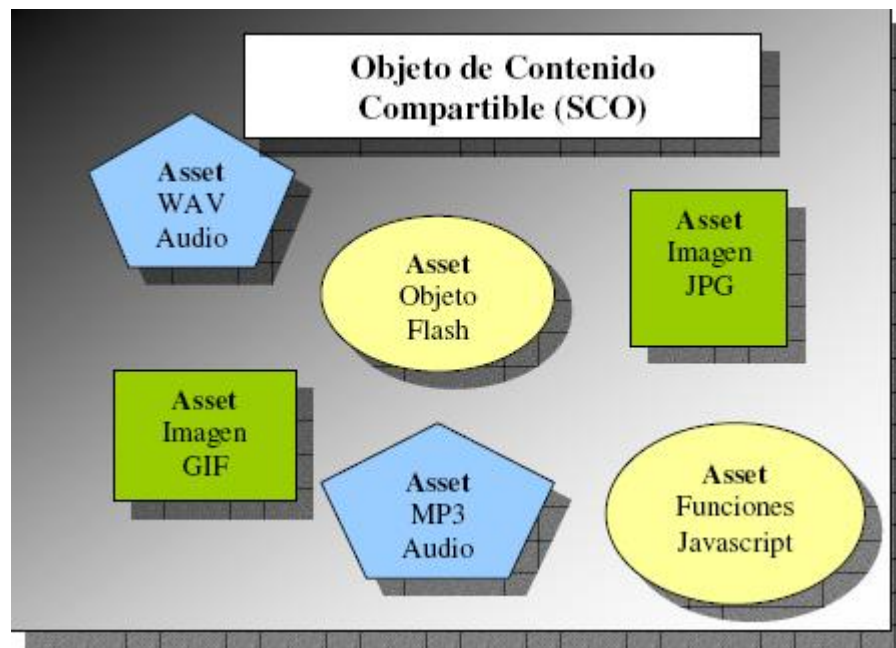


Figura 2: Ejemplos de un SCO. Todo el conjunto de archivos asociados corresponde a un SCO.

R-1.3.2.3 Metadatos

Otra de las partes importantes dentro del desarrollo de elementos que cumplan con las especificaciones de SCORM son los Metadatos.

Metadatos: es información sobre un objeto, ya sea físico o digital. Como el número de objetos crece exponencialmente y nuestras necesidades de expandir el aprendizaje es igualmente dramática, la falta de información o metadatos sobre objetos da lugar a un esfuerzo crítico y fundamental en nuestra habilidad para descubrir, manejar y usar objetos.

Learning Objects Metadata (LOM): La especificación más reconocida del trabajo del IEEE LTSC es la especificación de los Metadatos de los Objetos de Aprendizaje o Learning Object Metadata que define elementos para describir los recursos de aprendizaje.

La especificación del LOM entrega una guía sobre cómo los contenidos deben ser identificados o “etiquetados”, y sobre cómo se debe organizar la información de los alumnos de manera de que se puedan intercambiar entre los distintos servicios involucrados en un sistema de gestión de aprendizaje (LMS).

1.3.2.3.1 Metadatos de SCO.

Estos Metadatos pueden ser aplicados a los SCO para dar información descriptiva del contenido en el SCO independientemente de cualquier uso o potencial dentro de un contenido de cursos. Estos metadatos es usado para facilitar la reusabilidad y navegabilidad de los SCO.

1.3.2.3.2 Metadatos de Assets.

Estos Metadatos pueden ser aplicados a los Asset para dar información descriptiva acerca de los Asset independientemente de cualquier uso o potencial dentro de un contenido de cursos. Estos metadatos es usado para facilitar la reusabilidad y navegabilidad de los Asset.

1.3.2.3.3 Aplicación de los Metadatos.

El mecanismo de unión del modelo de componentes del Modelo de Contenidos es el paquete de contenidos. Existen cinco lugares para colocar los Metadatos dentro de un paquete de contenidos:

1. Manifest: en esta sección se guardan y almacenan los Metadatos del nivel manifest (por ejemplo, los metadatos de la Adición de Contenidos).debe estar conformada por las especificaciones dadas por la IEEE LTSC y LOM unidas pero sin restricciones adicionales impuestas por SCORM.
2. Organization: los metadatos en el nivel de organization describen la organización de contenidos como un todo. Estos Metadatos pueden representar a un curso, una unidad, un capítulo, etc.
3. Item: los metadatos en el nivel de Item describen una jerarquía anidada de actividades en un contexto de manera sensitiva.
4. Resource: los metadatos de este nivel describen recursos de SCO o de Assets.

5. File: los metadatos en el nivel de archivos (file), describen a un Asset en particular.

R-1.3.2.4 Paquete de Contenidos.

El Empaquetado de Contenidos de SCORM es un set de requerimientos específicos y guías, o perfiles de aplicación de la Especificación de Empaquetado de Contenidos de la IMS. Los paquetes de Contenidos de SCORM adhieren estrictamente al de la especificación de la IMS y provee requerimientos explícitos adicionales y guía de implementación para empaquetamiento de Assets, SCOs y Organización de Contenidos.

Un Empaquetado de Contenidos de la IMS contiene dos componentes principales:

1. Un documento XML que describe la estructura de contenidos y recursos asociados del empaquetado llamado el archivo manifest (manifest.xml).
2. Archivos físicos que generan paquetes de contenidos.

La siguiente figura es un diagrama conceptual que ilustra los componentes de un paquete de Contenidos de la IMS.

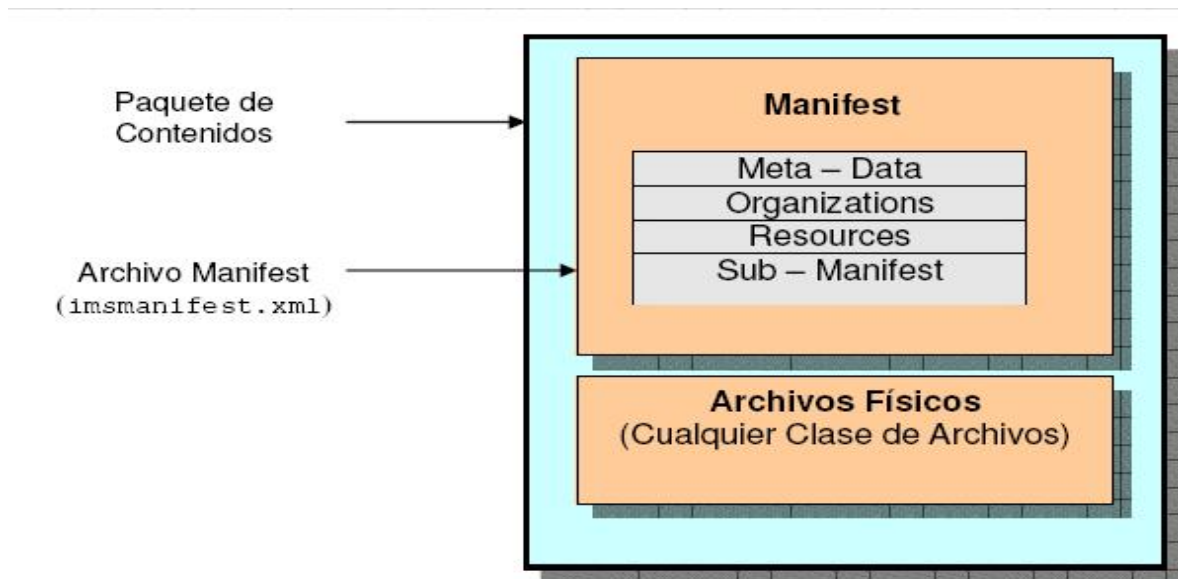


Figura 3. Árbol de Organización de Contenidos.

R-1.3.2.5 Manifest.

Un manifiesto es un documento XML que contiene inventarios estructurados de contenidos de un paquete. Si el paquete de contenidos está pensado para ser liberado a un usuario final, también contiene información acerca de cómo el contenido está organizado.

El alcance de un manifest es elástico. Un manifest puede describir parte de un curso que puede ser definido por si mismo fuera del contexto de curso (objeto “instruccional”), un curso entero, una colección de cursos o solo una colección de contenidos que es enviado de un sistema a otro. Cuando se empaqueta una colección de cursos, tal como un paquete de contenido, típicamente debería tener la opción de ser desagregado en orden para llegar a los usuarios de aprendizaje en un sistema LMS práctico.

La regla general es que un paquete siempre debe contener un nivel máximo (un nodo padre) de manifest que pueden contener a uno o más sub manifest. El nodo padre de un manifest siempre describe al paquete.

El documento manifest debe tener los siguientes requerimientos:

1. Debe ser llamado `imsmanifest.xml` o debe tener esa estructura.
2. El documento `imsmanifest.xml` y cualquiera de sus archivos de validación de soporte (DTD, XSD) deben estar en la raíz del paquete de contenidos. Si las extensiones se usan para describir características organizacionales definidas y estas características se representan en XML, entonces cualquiera de los archivos de control también es requerido para estar en la raíz del paquete. Esto incluya cualquier archivo que sirva para validar instancias XML las referenciadas al elemento `<adlcp: location>`.

En el anexo 6 encontramos un ejemplo de cómo quedaría estructurado el elemento `imsmanifest.xml`.

R-1.3.2.6 Componentes de un Manifest.

El archivo Manifest representa la información necesaria para describir los contenidos del paquete.



Figura 4: Componentes de un Manifiesto.

1.3.2.6.1 Elemento Metadata.

Este elemento contiene los datos principales del sistema como un todo, tales como Keyword (palabra clave), Title (título), autor, etc., siempre y cuando esté ubicada al principio del documento manifest, debido a que los metadatos también pueden estar ubicados en las etiquetas que están relacionadas con organizaciones y sus nodos hijos.

Tomando en cuenta que el elemento <Metadata> constituye el nodo raíz de:

<schema>

<schemaversion>

(Metadatos)

En el anexo 1 se encuentra un ejemplo de cómo quedaría la estructura del elemento metadata.

1.3.2.6.2 Elementos <organizations> y <organization>

Al elemento <organizations> que es único, se le puede englobar todos los elementos <organization>, lo que quiere decir que el elemento <organizations> es el nodo padre de todos los elementos

<organization>. Los elementos organization como son fijos y libres de nivel, se les puede asignar como los representantes de los cursos.

Los elementos Organization cuentan con los siguientes atributos:

1. identifier: obligatorio, es un identificador para cada elemento de la organización en el caso particular de la implementación del sistema este atributo podría indicar el código del curso.
2. structure: opcional, describe la forma de la organización.
3. adlseq, describe un secuenciamiento de la estructura. Es opcional.

En el anexo 2 encontramos un ejemplo de cómo quedaría estructurado el elemento organización.

1.3.2.6.3 Elemento <title>

Describe el título o nombre descriptivo de la organización

Dentro de una organización (<organization>) solo puede haber un solo título (<title>). Por ejemplo:

```
<organization identifier="CC1463">
```

```
<title>redes</title>
```

```
</organization>
```

Ejemplo: Estructura del elemento <title>

El elemento <title> podría representar el nombre descriptivo del curso, o el campo descriptivo del curso, como en el caso anterior, el curso redes.

1.3.2.6.4 Elemento <metadata>

Este elemento fue descrito con anterioridad. Corresponde a la misma etiqueta metadata mencionada anteriormente, con la única diferencia es que esta hace mención al elemento <organization>

El elemento Metadata tiene el mismo comportamiento y estructura en todos los otros elementos donde se encuentra.

En el anexo 3 encontramos un ejemplo de cómo quedaría estructurado el elemento metadata que hace mención a una organización

1.3.2.6.5 Elemento <item>

Es el elemento hijo más importante dentro de un elemento <organization>.

Corresponde a la descripción de un contenido, de un módulo o una lectura dentro de un curso especificado, teniendo en cuenta que un curso es un elemento <organization>.

Un elemento <item> puede tener a otro elemento <item> como hijo, sencillamente porque un contenido puede tener a otro contenido dentro de sí. También un item puede ser una lectura o un módulo. En la estructura habría que especificar el tipo de elemento item a través de un identificador de tipo.

En el anexo 4 encontramos un ejemplo de cómo quedaría estructurado el elemento item.

1.3.2.6.5.1 Atributos de item:

1. identifier: Corresponde a la identificación del Item(módulo, contenido o lectura). Es de carácter obligatorio.
2. identifierref: Corresponde a una identificación del Item en relación con un Learning Object.
3. isvisible: Quiere decir si el item es visible o no.

```
<item identifier="1" identifierref="rec1" isvisible="true">
```

```
<title>lectura 1</title>
```

```
</item>
```

Ejemplo 6. Ejemplo de atributos de un elemento "item". Sub Elementos de <item>

```
<title>
```

```
<item>
```

```
<metadata>
```

```
<adlcp:timeLimitAction>
```

```
<adlcp:dataFromLMS>
```

Los elementos más importantes son los tres primeros, <title> corresponde a un nombre o título para el item, el segundo corresponde a un <item> hijo que tiene todas las propiedades de item padre y <metadata> corresponde a todos los metadatos de Item.

Por lo tanto un elemento <item> padre puede tener a n elementos <item> como hijos.

En el anexo 5 encontramos un ejemplo de cómo quedaría estructurado el Sub Elemento item.

1.3.2.6.6 Elemento <resources>

En este elemento se van a asociar a todos los recursos físicos del sistema.

Tiene un solo elemento hijo llamado <resource>, en donde se guardan los archivos y elementos físicos correspondientes de cada item. (Archivos de cualquier tipo, Learning Objects, SCO, Assets).

El elemento <resource> es aquel que almacena las propiedades de localización de los Objetos de Aprendizaje, tiene relación directa con el elemento <item>, siempre y cuando este le haga referencia.

Cuenta con dos atributos:

1. identifier: es un identificador del elemento resource, es de carácter obligatorio.
2. href: corresponde a una URL donde se encuentran ubicados los LO, Assets, o cualquier tipo de archivo.

Ejemplo

```
<resource identifier="rec1"
href="http://localhost:8080/tesis/LO/audio/diopush.mp3">
<file href="diopush.mp3"/>
</resource>
```

Ejemplo: De elemento resource.

También <resource> cuenta con un elemento. Este elemento se llama "<file>"

1.3.2.6.7 Elemento <file>

Es el elemento que hace mención directa al archivo físico ocupado por el sub.-elemento <item>, su atributo principal es "ahref", Ejemplo:

```
<resource identifier="rec2"
ahref="http://localhost:8080/tesis/LO/images/images.htm">
<file ahref="images.htm"/>
<file ahref="robotech.jpg"/>
<file ahref="untitled.bmp"/>
<file ahref="tigres.jpg"/>
<file ahref="vinniepaul.jpg"/>
</resource>
```

Ejemplo: Elemento file dentro de elemento <resource>.

1.4 Propuesta del estándar a utilizar.

Con el estudio realizado de las características que poseen cada uno de los estándares y debido a la gran aceptación y las ventajas que proporcionan en cuanto a la reutilización, compatibilidad, interoperabilidad de contenidos de aprendizaje y al uso de tecnologías como XML, que gracias a su flexibilidad y su gran capacidad de reutilización va asociada a algunos de estos estándares, se tomara la propuesta de la iniciativa de ADL-SCORM como modelo de estandarización para lograr el objetivo planteado.

Conclusión

En este capítulo se describe el objeto de estudio del trabajo, centrándose en los estándares para e-Learning mostrando la importancia y necesidad que tienen los mismos en el sector educativo. Se especifican los estándares y sus ventajas, así como las organizaciones que se dedican a este proceso. Se muestran los Sistemas de Gestión de Aprendizajes. Se da una propuesta del estándar a utilizar realizando un análisis del mismo y se definen algunos de sus componentes. Se explica el modelo de empaquetado

que utiliza de forma estricta la especificación IMS, ya que de esta manera se dispone de una forma estandarizada para intercambiar contenido entre distintas plataformas. Se realiza una amplia descripción del documento imsmanifest.xml, explicando cada uno de los elementos que los conforman, mostrando al detalle su estructura y composición.

2 TENDENCIAS Y TECNOLOGÍAS ACTULES

Introducción

En el presente capítulo se hace un análisis de las tecnologías y tendencias que existen en la actualidad a nivel mundial, lenguajes de programación, sistemas gestores de bases de datos y la metodología de desarrollo que pueden ser adecuadas para la construcción del sistema que se pretende desarrollar. Se tienen en cuenta la política de migración hacia el software libre en el ámbito nacional.

2.1 La política de migración hacia software libre

Con desarrollo de aplicaciones para Internet ha surgido la necesidad de disponer de estándares para el intercambio de información y de datos de procesamiento. HTML y XML han llegado a ser los estándares más utilizados por la industria sin que ninguna empresa las haya creado ni disponga de los derechos sobre estos lenguajes y esta es la tendencia que sigue el país y la Universidad en específico, como institución de avanzada en el campo de la informática, procurando un mayor nivel en el uso del software libre. Cada vez se promociona más la migración desde los sistemas con licencia comercial que están en poder de unos pocos monopolios de la rama de la informática, hacia aquellos que se denominan “libres”, es decir, aquellos cuyo uso por parte del interesado permite la modificación de los códigos y cuyas licencias están exentas de pago. En el presente trabajo se propone al proyecto de Laboratorios Virtuales de nuestra universidad, un componente funcional haciendo uso de herramientas y tecnologías libres.

2.2 Lenguajes de programación.

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente. Dentro de estos se tienen según la forma de ejecución los compilados, y los interpretados. Los lenguajes compilados son lenguajes de alto nivel en los que las instrucciones se traducen del lenguaje utilizado a código máquina

para una ejecución rápida. Por el contrario un lenguaje interpretado es aquel en el que las instrucciones se traducen o interpretan una a una siendo típicamente unos 10 veces más lentos que los programas compilados.

2.2.1 Personal Home Page (PHP)

“PHP, acrónimo de "PHP: Hypertext Preprocessor", es un lenguaje "Open Source" interpretado y de alto nivel, especialmente pensado para desarrollos Web. La mayoría de su sintaxis es similar a C, Java y Perl y es extremadamente fácil de aprender. La meta de este lenguaje es permitir escribir a los creadores de páginas Web, páginas dinámicas de una manera rápida y fácil.

El código escrito en PHP es independiente de la plataforma, por lo que puede ser ejecutado en cualquier sistema operativo, gracias a que existen interpretes de PHP para los principales sistemas operativos del mercado, incluyendo Linux, muchas variantes Unix (incluido HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS, entre otros. PHP es soportado por la mayoría de servidores Web de hoy día, incluyendo Apache, Microsoft Internet Information Services, Personal Web Server, Netscape e iPlanet, O'Reilly Website Pro Server, Caudium, Xitami y OmniHTTPd. Este lenguaje es muy rápido, con una gran librería de funciones y mucha documentación, además es interpretado y embebido en el HTML.

Este lenguaje resulta muy atractivo y está respaldado por una gran comunidad de desarrolladores, debido a la perfecta combinación que proporcionan su simpleza para los programadores inexpertos, y a la vez su poder para los programadores profesionales.

Con PHP4 se han logrado aumentos de entre 5 y 10 veces en la velocidad de ejecución de páginas PHP. Estos resultados le han puesto por delante de ASP, la tecnología de Microsoft, permitiéndole competir directamente con ASP en la plataforma de Microsoft. La plataforma de desarrollo Apache+PHP en entornos Microsoft ha logrado rendimientos superiores a IIS+ASP.

Actualmente se encuentra en su versión 5. Es evidente que se ha convertido en la gran tendencia en el mundo de Internet. Las estadísticas arrojan que cada mes su uso crece en un 15% y cuenta con una de las comunidades más grandes de Internet, lo cual facilita encontrar ayuda, documentación, y otros recursos relacionados.(BAKKEN 2003).

2.2.2 Java

Es un lenguaje de programación orientado a objetos. A diferencia de los lenguajes de programación convencionales, que generalmente están diseñados para ser compilados a código nativo, Java es compilado en un bytecode que es interpretado (usando normalmente un compilador JIT), por una máquina virtual Java. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos mucho más simple y elimina herramientas de bajo nivel como punteros. Java está sólo lejanamente emparentado con JavaScript, aunque tengan nombres similares y compartan una sintaxis al estilo de C algo parecida.

Ofrece toda la funcionalidad de un lenguaje potente, elimina muchas de las características de otros lenguajes como C++, para mantener reducidas las especificaciones del lenguaje y añadir características muy útiles como el garbage collector (reciclador de memoria dinámica), reduce en un 50% los errores más comunes de programación. Además, el intérprete completo de Java que hay en este momento es muy pequeño, solamente ocupa 215 Kb de RAM. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo. Java se ha construido con extensas capacidades de interconexión TCP/IP. Existen librerías de rutinas para acceder e interactuar con protocolos como http y ftp. Esto permite a los programadores acceder a la información a través de la red con tanta facilidad como a los ficheros locales. La verdad es que Java en sí no es distribuido, sino que proporciona las librerías y herramientas para que los programas puedan ser distribuidos, es decir, que se corran en varias máquinas, interactuando.(LUCAS)

2.2.3 eXtensible Markup Language, o Lenguaje de Marcado (XML)

Extensible, muy parecido en estructura a HTML, el cual fue diseñado para describir datos. XML es un lenguaje de meta marcado que ofrece un formato para la descripción de datos estructurados. Esto facilita unas declaraciones de contenido más precisas y unos resultados de búsquedas más significativos en varias plataformas. La principal característica de XML es que sus etiquetas son definidas por el propio desarrollador, usando palabras del lenguaje natural para describir y representar los datos.

Aunque se parece mucho al HTML, es mucho más estricto con las comillas, los indicadores correctamente finalizados y otros detalles similares. XML no define nombres de indicadores, por lo que los autores de documentos deberán inventar sus propios conjuntos de indicadores, o fijarse en una organización

formalizada que defina un lenguaje de marcado XML adecuado. Un lenguaje de marcado es básicamente un conjunto de indicadores personalizados que poseen un significado semántico; XSLT es un lenguaje de este tipo puesto que se expresa utilizando la sintaxis XML.(MONTERO 1999)

2.2.4 Modos de Procesamiento de un documento XML

Existen varias razones para utilizar la tecnología XML. La más importante está relacionada con la composición del documento Manifest, donde los Cursos, Contenidos, Módulos, Lecturas y Objetos de Aprendizaje (Learning Objects) quedan representados de una manera sencilla y jerarquizada a través de Metadatos. La otra razón de usar tecnologías XML es su extraordinaria flexibilidad, es por eso que para su procesamiento utilizamos los Parsers facilitando su elaboración.

2.3 Parsers

Un Parser es un módulo, biblioteca o programa que se ocupa de transformar un archivo de texto en una representación interna. En el caso de XML, como el formato siempre es el mismo, no necesitamos crear un parser cada vez que hace un programa, hay muchos parsers disponibles. (CARLOS 2004, Diciembre)

2.3.1 Clasificación de los Parsers

En XML encontramos dos modos principales de procesamiento: SAX y DOM.

SAX se utiliza para hacer recorridos secuenciales de los elementos del documento XML y DOM implica la creación de un árbol en memoria que contiene el documento XML, y con él en memoria se puede hacer cualquier tipo de recorrido y acciones con los elementos que queramos.

Si se quiere observar, buscar o procesar datos desde el contenido de un archivo “imsmanifest.xml”, se logra ocupar cualquiera de estos dos parsers. Incluso se puede generar un documento manifest a través de estos métodos.



Figura 5: El modo de procesamiento de un documento XML

2.4 IDE

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI. Los IDEs pueden ser aplicaciones por si solas o pueden ser parte de aplicaciones existentes.

2.4.1 NuSphere PHPEd

PHPEd es un editor para programadores con soporte para múltiples formatos, similar a otras aplicaciones como PHP Coder.

PHPEd facilita tu trabajo de programación con numerosas características de gran utilidad entre las que destacan:

1. Completo sistema de ayuda.
2. Plantillas de documentos y de fragmentos de código frecuentes.
3. Código de colores para comandos en PHP, Perl, Javascript, SOL, HTML y más.

Además, esta herramienta incluye un cliente de FTP y un servidor Web integrados, totalmente configurables según tus necesidades de trabajo. (SOFTONIC 2005).

2.4.2 Zend Studio

Zend Studio es uno de los ambientes de desarrollo integrado o *Integrated Development Environment* (IDE) disponible para desarrolladores profesionales, agrupa todos los componentes necesarios para ciclo de

desarrollo de aplicaciones PHP. A través de un comprensivo conjunto de herramientas de edición, depurado, análisis, optimización y bases de datos, Zend Studio acelera los ciclos de desarrollo y simplifica los proyectos complejos.(MIGUEL, ANGEL ALVAREZ 03.11.2003).

2.5 Sistema gestor de bases de datos.

Los Sistemas Gestores de Bases de Datos son un tipo de software muy específico, dedicado a servir de interfaz entre las bases de datos y las aplicaciones que la utilizan. En la actualidad existe una gran variedad de SGBD, tanto de tipo comercial como libre.

2.5.1 MySQL

Desde sus inicios el MySQL se ha convertido en el Gestor de Bases de datos de código abierto más popular de Internet, principalmente por su simpleza y robustez. Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, atrajo a los desarrolladores de páginas Web con contenido dinámico, justamente por su simplicidad; aquellos elementos faltantes fueron llenados por la vía de las aplicaciones que la utilizan.

Poco a poco los elementos faltantes en MySQL están siendo incorporados tanto por desarrollos internos, como por desarrolladores de software libre.

Entre las características disponibles en las últimas versiones se puede destacar:

1. Amplio subconjunto del lenguaje SQL.
2. Algunas extensiones son incluidas igualmente.
3. Disponibilidad en gran cantidad de plataformas y sistemas.
4. Diferentes opciones de almacenamiento según si se desea velocidad en las operaciones o el mayor número de operaciones disponibles.
5. Transacciones y claves foráneas.
6. Conectividad segura.
7. Replicación.

8. Procedimientos almacenados.

(MURUGARREN 2006)

2.5.2 SQL

Debido a la diversidad de lenguajes y de bases de datos existentes, la manera de comunicar entre unos y otras sería realmente complicada a gestionar de no ser por la existencia de estándares que permiten realizar las operaciones básicas de una forma universal.

El Structured Query Language (SQL) es una herramienta para organizar, gestionar y recuperar datos almacenados en una base de datos informática. Como su propio nombre indica, SQL es un lenguaje informático que se puede utilizar para interactuar con una base de datos y más concretamente con un tipo específico llamado base de datos relacional.

SQL es a la vez un lenguaje fácil de aprender y una herramienta completa para gestionar datos. Las peticiones sobre los datos se expresan mediante sentencias, que deben escribirse de acuerdo con unas reglas sintácticas y semánticas de este lenguaje. Su aprendizaje sirve para todas las aplicaciones gestoras de bases de datos existentes en el mercado que soporten este lenguaje (que son muchas) ya que es un lenguaje estándar por haberse visto consolidado por el Instituto Americano de Normas (ANSI) y por la Organización de Estándares Internacional (ISO). (S.L. 2006)

2.5.3 ORACLE

Oracle es una potente herramienta cliente/servidor para la gestión de Bases de Datos. Para desarrollar en Oracle utilizamos PL/SQL un lenguaje de 5ª generación, bastante potente para tratar y gestionar la base de datos, también por norma general se suele utilizar SQL al crear un formulario. Es posible lógicamente atacar a la base de datos a través del SQL plus incorporado en el paquete de programas Oracle para poder realizar consultas, utilizando el lenguaje SQL.

Oracle es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hace que sólo se vea en empresas muy grandes y multinacionales, por norma general. En el desarrollo de páginas Web pasa lo mismo: como es un sistema muy caro no está tan extendido como otras bases de datos, por ejemplo, Access, MySQL, SQL Server. (*Wikipedia. La Enciclopedia Libre. Oracle.*)

2.5.4 PostgreSQL

Postgres intenta ser un sistema de bases de datos de mayor nivel que MySQL, a la altura de Oracle, Sybase o Interbase, con licencia BSD. Por su arquitectura de diseño, escala muy bien al aumentar el número de CPUs y cantidad de RAM. Soporta transacciones y desde la versión 7.0 claves ajenas (con comprobaciones de integridad referencial) Tiene mejor soporte para triggers y procedimientos en el servidor. Soporta un subconjunto de SQL92 MAYOR que el que soporta MySQL. Además, tiene ciertas características orientadas a objetos. (*Wikipedia. La Enciclopedia Libre. PostgreSQL.*)

Inconvenientes:

1. Consume BASTANTES mas recursos y carga mas el sistema. Limite del tamaño de cada fila de las tablas a 8k!!! (Se ampliar a 32k recopilando, pero con un coste añadido en el rendimiento). Es de 2 a 3 veces más lenta que MySQL.
2. Menos funciones en PHP.

2.6 Metodologías de Desarrollo de Software

Una metodología tiene como objetivo aumentar la calidad del software que se produce en todas y cada una de sus fases de desarrollo. Se han desarrollado dos corrientes en lo referente a las metodologías de desarrollo de software, las llamadas “pesadas” y las llamadas “ligeras o ágiles”. Las primeras se basan en la idea de conseguir el objetivo común por medio de orden y documentación, mientras que las segundas tratan de lograrlo por medio de la comunicación directa e inmediata entre aquello que intervienen en el proceso.(JACOBSON 2000)

2.6.1 El Proceso Unificado de Racional (RUP).

RUP es una de las metodologías más generales y más usadas de las que existen en la actualidad, pues está pensada para adaptarse a cualquier proyecto. Constituye además, una propuesta de proceso para el desarrollo de software orientado a objeto, utilizando UML (del inglés Unified Model Language), para describir todo el proceso, basándose en componentes. Este lenguaje es estándar, con él se puede visualizar, especificar, construir y documentar los artefactos de un sistema.

Las principales características de esta metodología son:

1. Centrado en los Modelos: Los diagramas son un vehículo de comunicación más expresivo que las descripciones en lenguaje natural. Se trata de minimizar el uso de descripciones y especificaciones textuales del sistema.
2. Guiado por los casos de uso: Los casos de uso son el instrumento para validar la arquitectura del software y extraer los casos de prueba.
3. Iterativo e incremental: Durante todo el proceso de desarrollo se producen versiones incrementales (que se acercan al producto terminado) del producto en desarrollo.
(WIKIPEDIA® 2007).

2.6.2 Programación Extrema (XP)

La Programación Extrema, o Extreme Programming, es otra de las metodologías de desarrollo de software que existen en la actualidad. “Mientras que el RUP intenta reducir la complejidad del software por medio de estructura y la preparación de las tareas pendientes en función de los objetivos de la fase y actividad actual, XP, como toda metodología ágil, lo intenta por medio de un trabajo orientado directamente al objetivo, basado en las relaciones interpersonales y la velocidad de reacción.” (MOLPECERES 05/01/2007)

Supone la disposición en todo momento, por parte del equipo de trabajo, de un representante competente del cliente, que debe estar en condiciones de dar una respuesta rápida y correcta a cualquier pregunta del equipo de desarrollo de forma que no se retrase la toma de decisiones.

La base para el desarrollo del software que usa esta metodología son las llamadas User Stories, historias escritas por el cliente en las que describe escenarios sobre el funcionamiento del sistema y que no sólo están limitados a la interfaz de usuario, sino que también pueden describir modelos, dominio, etc. Estas User Stories junto a la arquitectura que se persigue, sirve de base para crear un plan de “entregas de software” entre el equipo de desarrollo y el cliente, para cada una las cuales se definen objetivos y las iteraciones (generalmente cortas) necesarias para cumplirlos. Las User Stories y los casos de pruebas son la base sobre la que se asienta el trabajo del desarrollador.

Esta metodología apuesta por iteraciones cortas que generan software que el cliente puede ver.

La codificación del software se realiza siempre en parejas (dos programadores, un ordenador), las cuales no son fijas sino que rotan a lo largo del proyecto, y el código que escriben no les pertenece sólo a ellos sino al equipo completo. El objetivo ideal sería que cada integrante del equipo trabaje al menos una vez con cada uno de los demás integrantes y con cada componente software, de forma que el conocimiento de la aplicación completa lo posea el equipo entero y no unos pocos miembros. Se programa sólo la funcionalidad requerida para la entrega en curso, se trabaja en función de las necesidades del momento, por lo que no se le da importancia al análisis como fase independiente.(MOLPECERES 05/01/2007).

2.7 Selección de las tecnologías a utilizar en la propuesta de solución.

Después de haber realizado un estudio de las principales características de las tecnologías mas usadas en la actualidad, se propone el uso PHP como lenguaje del lado del servidor, por todas las ventajas que presenta por su propia esencia: PHP no soporta directamente punteros, como el C, de forma que no existen los problemas de depuración provocados por estos, Se pueden hacer grandes cosas con pocas líneas de código. Lo que hace que merezca la pena aprenderlo, Viene acompañado por una excelente biblioteca de funciones que permite realizar cualquier labor (acceso a base de datos, encriptación, envió de correo, gestión de un e-commerce, xml, creación de PDF...), Al poderse encapsular dentro de código html se puede recoger el trabajo del diseñador gráfico e incrustar el código php posteriormente, Está siendo utilizado con éxito en varios millones de sitios web, Hay multitud de aplicaciones php para resolver problemas concretos (weblogs, tiendas virtuales , periódicos , ...) listas para usar, Es multiplataforma, funciona en todas las plataformas que soporten apache, Es software libre. Se puede obtener en la web y su código está disponible bajo la licencia GPL.

Como sistema gestor de base de datos se seleccionó MySQL ya que junto con Apache y PHP forman un buen equipo para servir páginas web con contenido dinámico, discusiones, noticias, etc. Es capaz de implementar funcionalidades Web que permiten un acceso a los datos, seguro y fácil, desde Internet. Además es uno de los SGBD más populares, desarrollado bajo la filosofía de código abierto. También brinda sistemas en los que la velocidad y el número de accesos concurrentes es algo primordial.

Como ambiente de desarrollo se escogió NuSphere PHPEd, y como metodología de desarrollo RUP por todas las ventajas de organización que brinda y por venir acompañada de una potente herramienta que soporta todos los procesos básicos de RUP: Suite del Rational. Pero siempre se pueden comparar si se

definen determinados criterios, en cuanto a la forma en que se capturan los requisitos, RUP y XP crean como base los use cases (casos de uso), los cuales describen los requerimientos de la aplicación desde el punto de vista del cliente, definen los requisitos técnicos sin abordar detalles de implementación.

Conclusiones

En este capítulo se hizo un estudio de las tecnologías a utilizar en el desarrollo de la propuesta de solución así como algunos conceptos y tendencias que se deben tener en cuenta. Se fundamentó la elección del lenguaje de programación, el sistema gestor de bases de datos y la metodología de desarrollo de software; así como el uso de otras técnicas.

3 CARACTERÍSTICAS DEL SISTEMA

Introducción

El presente capítulo describe la propuesta de solución, desarrollando el modelo de dominio del negocio donde se describen los principales conceptos del mismo. Se plantean los requisitos funcionales y no funcionales y se modela la misma en términos de casos de uso de sistema.

3.1 Modelo de dominio.

El objetivo del Proceso Unificado, dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental; es guiar a los desarrolladores de cualquier sistema software, en la implementación y distribución eficiente de sistemas que se ajusten a las necesidades de los clientes.

Teniendo en cuenta que los procesos del negocio no son visibles y no es posible establecer fronteras bien definidas acorde al objeto de estudio, y una poca estructuración de estos procesos, se desarrolla un proceso de identificación de conceptos, entidades y sus correspondientes relaciones; los cuales se agrupan en un Modelo de Objeto o Modelo de Dominio.

Para esto se realiza la descripción del modelo del dominio a través de un diagrama de clases UML, mostrando a los clientes, usuarios, revisores y a otros desarrolladores las principales clases conceptuales y cómo se relacionan una con otras mediante asociaciones.

3.1.1 Diagrama de Modelo de Dominio.

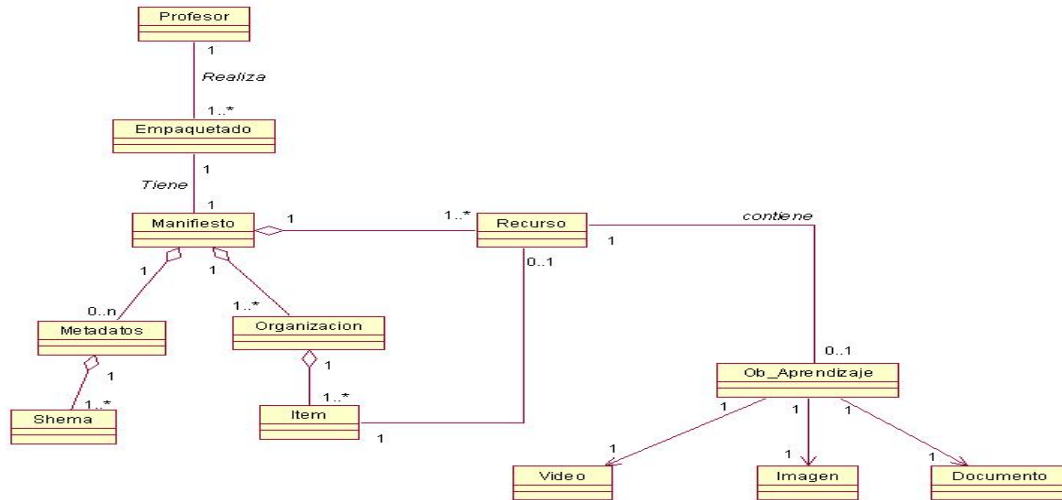


Figura 6: Modelo de Dominio.

3.1.2 Definición de las entidades y los conceptos principales.

En el Dominio se identificadas las siguientes entidades y conceptos:

Profesor

Es el usuario encargado de realizar todas las gestiones en el sistema.

Empaquetado

Contiene la descripción de la estructura y localización de materiales en línea, así como algunas definiciones específicas de contenido.

Manifiesto

Es un documento XML donde quedan reflejados los Metadatos, es decir, la información sobre la estructura en que se organizan los objetos de aprendizaje.

Metadato

Información que describe el paquete como un todo, indicando, por ejemplo, el estándar (schema) que se utiliza, su versión o el lenguaje del contenido.

Organización

Representa la Organización de Contenidos y su descomposición en actividades (item).

Recurso

Es aquel que almacena las propiedades de localización de los Objetos de Aprendizaje, tiene relación directa con el elemento item, siempre y cuando este le haga referencia.

Item

Es el elemento hijo más importante dentro de un elemento organización.

Corresponde a la descripción de un contenido, de un módulo o una lectura dentro de un curso especificado, teniendo en cuenta que un curso es un elemento organización.

Objeto de Aprendizaje

Es cualquier entidad, digital o no digital, que puede ser utilizada o referenciada durante el aprendizaje.

3.2 Requerimientos

3.2.1 Requerimientos Funcionales:

A través de los requisitos funcionales, los cuales se describen a continuación, se puede expresar una especificación más detallada de las responsabilidades del sistema. Con ellos se pretende determinar de manera clara y concisa lo que debe hacer el sistema siguiendo un enfoque funcional.

R-1 Permitir capturar y mostrar los archivos a empaquetar.

R-2. La generación de los archivos y manifiesto en la carpeta a estandarizar.

Gestión de organizaciones.

R-3 El sistema debe ser capaz de crear organizaciones.

R-4 El sistema debe mostrar la lista de organizaciones creadas.

R-5 El sistema debe permitir eliminar una o varias organizaciones.

R-6 El sistema debe facilitar poder ordenar las organizaciones a través de una lista donde se encuentren las mismas.

Gestión de los elementos items.

R-7 El sistema debe posibilitar seleccionar en una lista la organización en la que desea insertar el elemento item.

R-8 El sistema debe posibilitar, seleccionando la organización una lista de los elemento items para así obtener aquel que se vaya a eliminar.

R-9 El sistema debe posibilitar, seleccionando la organización una lista de los elemento items para así obtener aquellos que se vayan a organizar para una mejor estructuración del documento a generar.

R-10 El sistema debe posibilitar la lista de recursos para poder referenciar los items creados.

Mostrar recursos.

R-11 El sistema debe permitir ver la lista de recursos importados.

Gestión de los Metadatos.

R-12 El sistema debe ser capaz de crear Metadatos.

R-13 Seleccionando la organización el sistema permitirá editar Metadatos correspondientes a la misma.

R-14 El sistema debe posibilitar eliminar los Metadatos creados.

R-15 El sistema debe ser capaz de salvar las actividades en un formato .zip conjunto al Manifiesto.

3.2.2 Requerimientos no Funcionales:

Los requerimientos no funcionales describen alguna forma o restricción para la realización de algún requerimiento (funcionalidad) o conjunto de ellas e inclusive todos los requerimientos.

Los requerimientos no funcionales son los atributos del sistema, cualidades que debe tener el producto, los cuales aparecen a continuación.

Apariencia o interfaz externa:

Diseño orientado a proporcionar una navegación sencilla brindándole al usuario posibilidades de adaptación al sistema.

Usabilidad:

El sistema podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora y de un ambiente Web en sentido general.

Rendimiento:

Tiempos de respuestas no mayor de 2 segundos, al igual que la velocidad de procesamiento de la información.

Soporte:

Se requiere un servidor de bases de datos con soporte para medianos volúmenes de datos y velocidad de procesamiento. Con un tiempo de respuesta de no más de 3 segundos en accesos concurrentes.

1. Versión de PHP 5.0.x o superior.
2. Servidor Web Apache 2.0.x o superior

Portabilidad:

Necesidad de que el sistema sea multiplataformas.

Seguridad:

Que garantice que la información sea editada únicamente por el profesor. Que contenga verificación sobre acciones irreversibles (eliminaciones).

Confiabilidad:

La herramienta de implementación a utilizar tiene soporte para recuperación ante fallos y errores.

Funcionalidad:

Reducir al mínimo el tiempo en que carga el sistema.

Software:

1. Sistema operativo superior a Windows NT
2. MySQL 2.3.x.

- 3. Apache 2.0.x
- 4. PHP 5.0.x

3.3 Actores del sistema

Un Actor es un rol que un usuario juega con respecto al sistema. Es importante destacar el uso de la palabra rol, pues con esto se especifica que un Actor no necesariamente representa a una persona en particular, sino más bien la labor que realiza frente al sistema.

Tabla 1: Actores del Sistema.

Nombre	Justificación
Profesor	Este usuario hace el rol de profesor, tiene todas las atribuciones de administrar y manipular datos del sistema.

3.4 Casos de Usos del sistema.

Los casos de uso se utilizan para modelar el funcionamiento o cómo el cliente desea que funcione el sistema. Utilizando las facilidades que brinda el UML, se capturan los requisitos funcionales del sistema y se representan mediante un diagrama de casos de uso.

Tabla 2: Casos de Usos del Sistema.

Referencia	Caso de Uso	Prioridad
CU-1	Importar Contenido	Critico
CU-2	Gestionar Organización	Critico
CU-3	Gestionar item	Critico
CU-4	Añadir Metadatos	Secundario
CU-5	Empaquetar	Critico

3.4.1 Diagrama de Casos de uso del sistema.

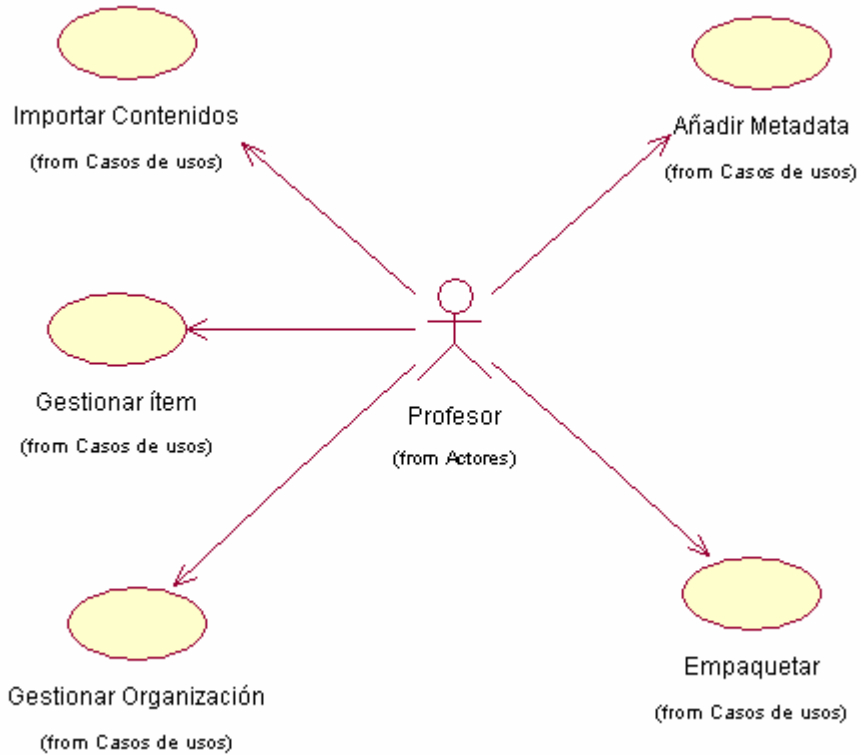


Figura 7: Diagrama de Casos de uso del sistema.

3.4.2 Descripción de los Casos de Usos.

Tabla 3: Importar Contenido

CU	Importar Actividades
Actor	Profesor
Resumen	El caso de uso se inicia cuando el Profesor necesita cargar la carpeta que se va a estandarizar y así incorporarla al panel de trabajo.
Referencia	R-1; R-2
Precondiciones	Tener ubicado en una misma carpeta todos los archivos que se

	van a estandarizar.
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1- El Profesor entra al sistema y selecciona la opción Importar Actividades.	1.1 El sistema muestra una ventana de búsqueda.
2- El Profesor selecciona la carpeta creada con los archivos a estandarizar.	2.1- El sistema crea automáticamente en la carpeta seleccionada los archivos necesarios para el manejo del manifiesto.
	2.2- El sistema muestra en el panel de trabajo el listado de los archivos de la carpeta seleccionada.
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
	2.1- Si en la carpeta seleccionada existen ya los archivos del manifiesto el programa muestra una ventana de error indicando la existencia del mismo.
Poscondiciones	
Prioridad	Crítico

Tabla 4: Gestionar Organización.

CU	Gestionar Organización
Actor	Profesor
Resumen	El caso de uso se inicia cuando el Profesor necesita referenciar los recursos a una organización.

Referencia	R-3;R-4;R-5;R-6	
Precondiciones	Debe haberse importado las actividades de la carpeta a estandarizar.	
Flujo Normal de Eventos		
Acción del Actor	Respuesta del Sistema	
1- El Profesor selecciona la opción de Gestionar Organización.	1.1- El sistema muestra una ventana donde permite ponerle un nombre a la organización y la opción de crear, eliminar y mover.	
2- El Profesor crea una organización dando clic en el botón crear.	2.1- El sistema muestra una ventana con el listado de las organizaciones creadas.	
3- El Profesor puede seleccionar las opciones de eliminar y mover una organización dando clic en el botón eliminar o mover.	3.1- El sistema actualiza el listado de las organizaciones.	
Flujos Alternos		
Acción del Actor	Respuesta del Sistema	
3- Para eliminar una organización.	2.1- El sistema debe haber creado una organización.	
3- Para mover una organización.	2.1- El sistema debe haber creado al menos dos organizaciones.	
Poscondiciones		
Prioridad	Crítico	

Tabla 5: Gestionar Items.

CU	Gestionar Items	
Actor	Profesor	
Resumen	El caso de uso se inicia cuando el Profesor necesita referenciar cada recurso a un Item.	
Referencia	R-7; R-8; R-9;R-10;R-11	
Precondiciones	Debe haberse creado una organización.	
Flujo Normal de Eventos		
Acción del Actor		Respuesta del Sistema
1- El Profesor selecciona la opción de Gestionar Items.		1.1- El sistema muestra una ventana donde permite ponerle un nombre al Item y la opción de crear, eliminar, referenciar y mover.
2- El Profesor crea un Item dando clic en el botón crear.		2.1- El sistema muestra una ventana con el listado de los Items creados.
3- El Profesor puede seleccionar las opciones de eliminar y mover un Item dando clic en el botón eliminar o mover.		3.1- El sistema actualiza el listado de los items.
4- El Profesor puede seleccionar la opción de referenciar dando clic en el botón referenciar.		4.1- El sistema muestra una ventana donde aparece el listado de los recursos y el de los items creado. 4.2- El sistema permite seleccionar el item y el recurso a referenciar.
Flujos Alternos		
Acción del Actor		Respuesta del Sistema

3 Para eliminar y referenciar un ítem.	2.1- El sistema debe haber creado un ítem.
3 Para mover un ítem.	2.1- El sistema debe haber creado al menos dos ítem.
Poscondiciones	
Prioridad	Crítico

Tabla 6: Añadir Metadatos.

CU	Añadir Metadatos	
Actor	Profesor	
Resumen	Este caso de uso se inicia cuando el Profesor decide agregar Metadatos al manifiesto para una mejor lectura y organización.	
Referencia	R-12; R-13;R-14	
Precondiciones	Por lo meno debe haberse creado una organización.	
Flujo Normal de Eventos		
Acción del Actor	Respuesta del Sistema	
1- El Profesor selecciona la opción de Añadir Metadatos.	1.1- El sistema muestra una ventana que permite seleccionar la organización a la cual será añadida los Metadatos y las opciones de crear, eliminar, editar y añadir Shemas y Shema versión. 1.2- El sistema incorpora los Metadatos al manifiesto.	
2- El Profesor selecciona añadir Shemas y Shema versión.	2.1- El sistema incorpora los Shemas con los datos estándares que necesitan.	

3- El Profesor selecciona Editar Metadatos.	3.1- El sistema muestra una ventana donde el Profesor podrá llenar los campos correspondientes a los Metadatos.
4- El Profesor selecciona eliminar Metadatos.	4.1- El sistema muestra una ventana donde permitirá seleccionar la organización a la cual se le eliminara los Metadatos.
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
3- Para eliminar, editar un Metadato y añadir Shemas y Shema versión.	1.2- El sistema debe haber creado un Metadato.
Poscondiciones	
Prioridad	Secundario

Tabla 7: Empaquetar.

CU	Empaquetar
Actor	Profesor
Resumen	El caso de uso se inicia cuando esté listo el paquete para ser estandarizado.
Referencia	R-15
Precondiciones	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1- El Profesor selecciona la opción de	1.1- El sistema muestra una ventana en la

Empaquetar.	cual dice que el documento fue modificado.
2- El Cliente selecciona guardar modificación.	2.1- El sistema muestra una ventana que permite seleccionar la carpeta donde va a guardar el paquete.
	2.2- El sistema muestra una ventana que le especifica al Cliente si desea o no realizar los cambios y guardarlos.
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
Poscondiciones	
Prioridad	Crítico

Conclusiones

En el desarrollo de este capítulo se profundizó en la propuesta de solución, obteniéndose las funcionalidades que debe tener el sistema, las cuales fueron modeladas mediante un diagrama de casos de uso. Además se describieron todas las acciones que realiza el actor y las respuestas del sistema. Partiendo de los requisitos presentados en este capítulo se modelará el sistema.

4 CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA

Introducción

En el presente capítulo se abordará la construcción de la solución propuesta a través de los flujos de trabajo de análisis y diseño de la aplicación en los aspectos más importantes en función de satisfacer los requerimientos esperados, se confecciona el diagrama de clases de análisis y los diagramas de interacción, se completa el modelo de clases de diseño y se aplica el diseño Web para las especificaciones de la aplicación. Se realiza además el diseño del modelo de datos para el sistema.

4.1 Análisis

Durante el análisis, analizamos los requisitos que se describieron en la captura de requisitos, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que ayude a estructurar el sistema entero, incluyendo su arquitectura. El lenguaje que utilizamos en el análisis se basa en un modelo de objetos conceptual, que llamamos modelo de análisis.

El modelo de análisis ayuda a refinar y analizar los requisitos. Ofrece una especificación más precisa de los requisitos que la que se tiene como resultado de la captura de requisitos, incluyendo al modelo de casos de uso. Describe utilizando el lenguaje de los desarrolladores, y se puede por tanto introducir un mayor formalismo y ser utilizado para razonar sobre los funcionamientos internos del sistema. Estructura los requisitos de un modo que facilita su comprensión, su preparación, su modificación, y en general, su mantenimiento, y puede considerarse como una primera aproximación al modelo de diseño (aunque es un modelo por sí mismo), y es por tanto una entrada fundamental cuando se da forma al sistema en el diseño y en la implementación.

4.1.1 Diagramas de Clases de Análisis

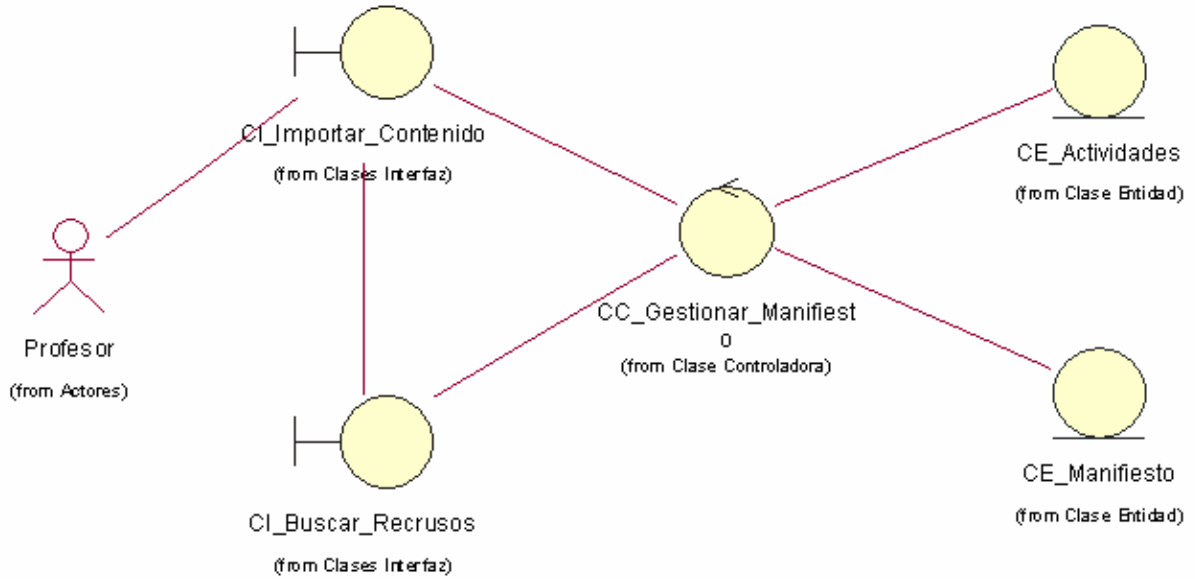


Figura 8: Diagrama de Clases de Análisis: Importar Contenido.

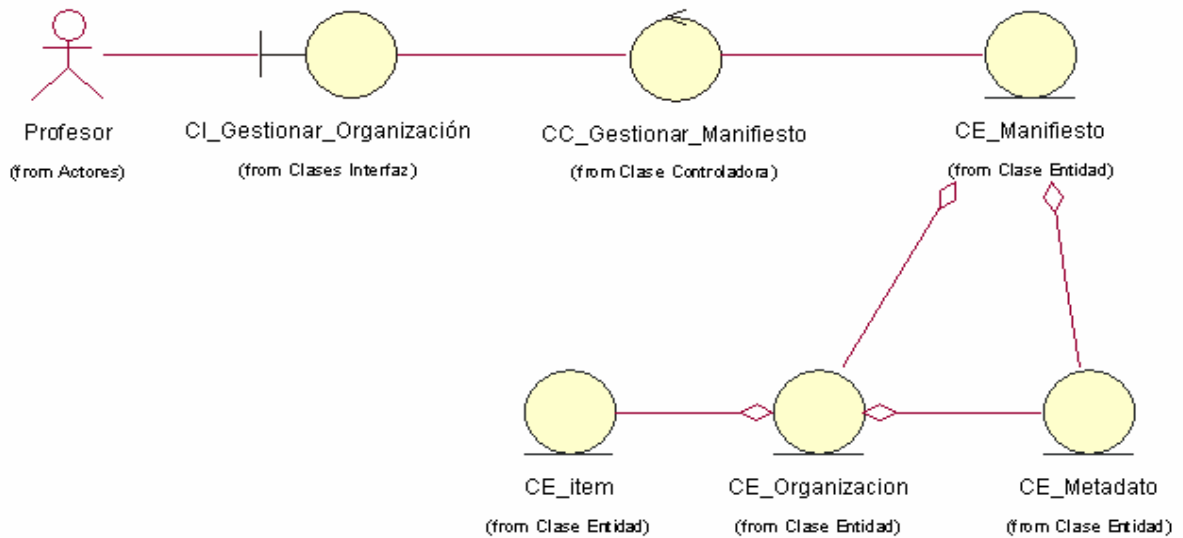


Figura 9: Diagrama de Clases de Análisis: Gestionar Organización.

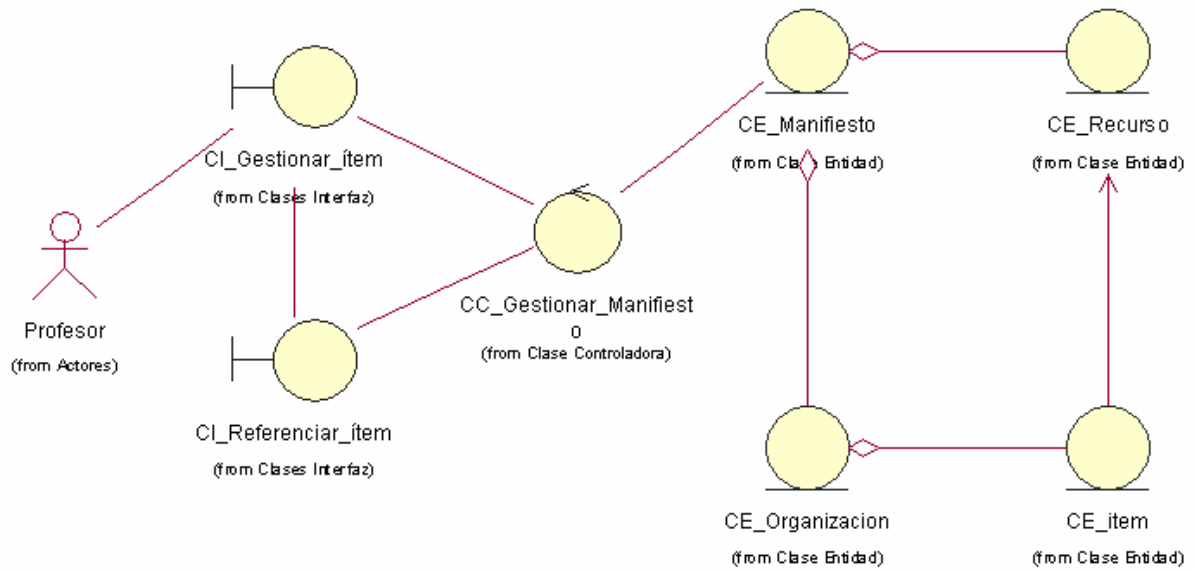


Figura 10: Diagrama de Clases de Análisis: Gestionar item.

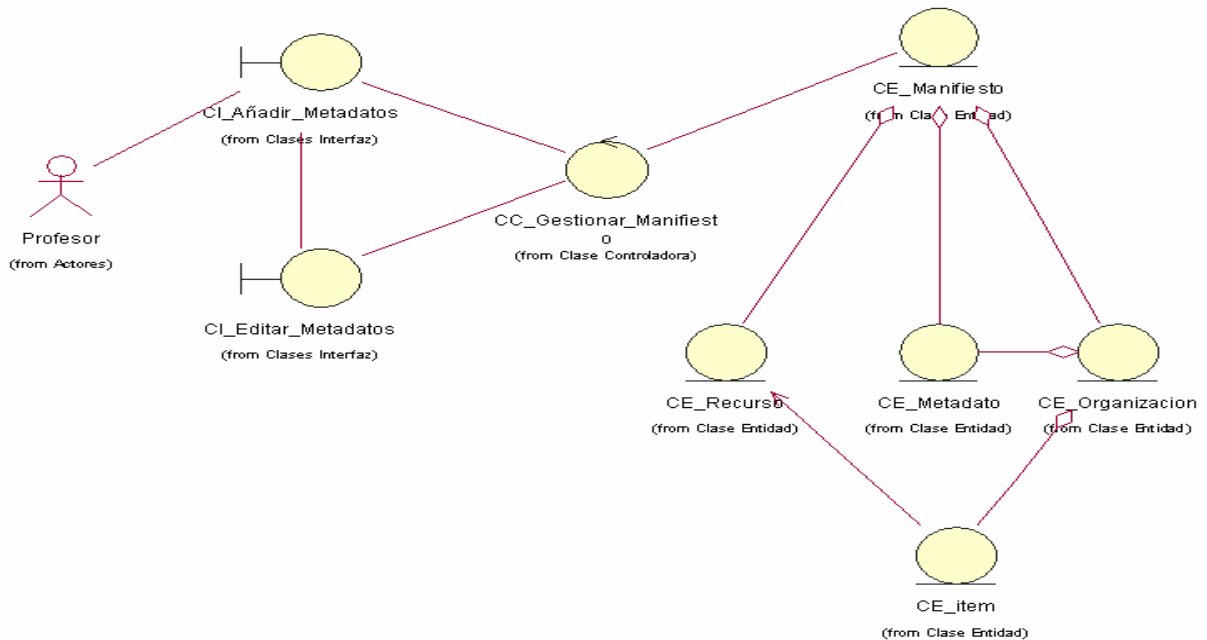


Figura 11: Diagrama de Clases de Análisis: Gestionar Metadatos.

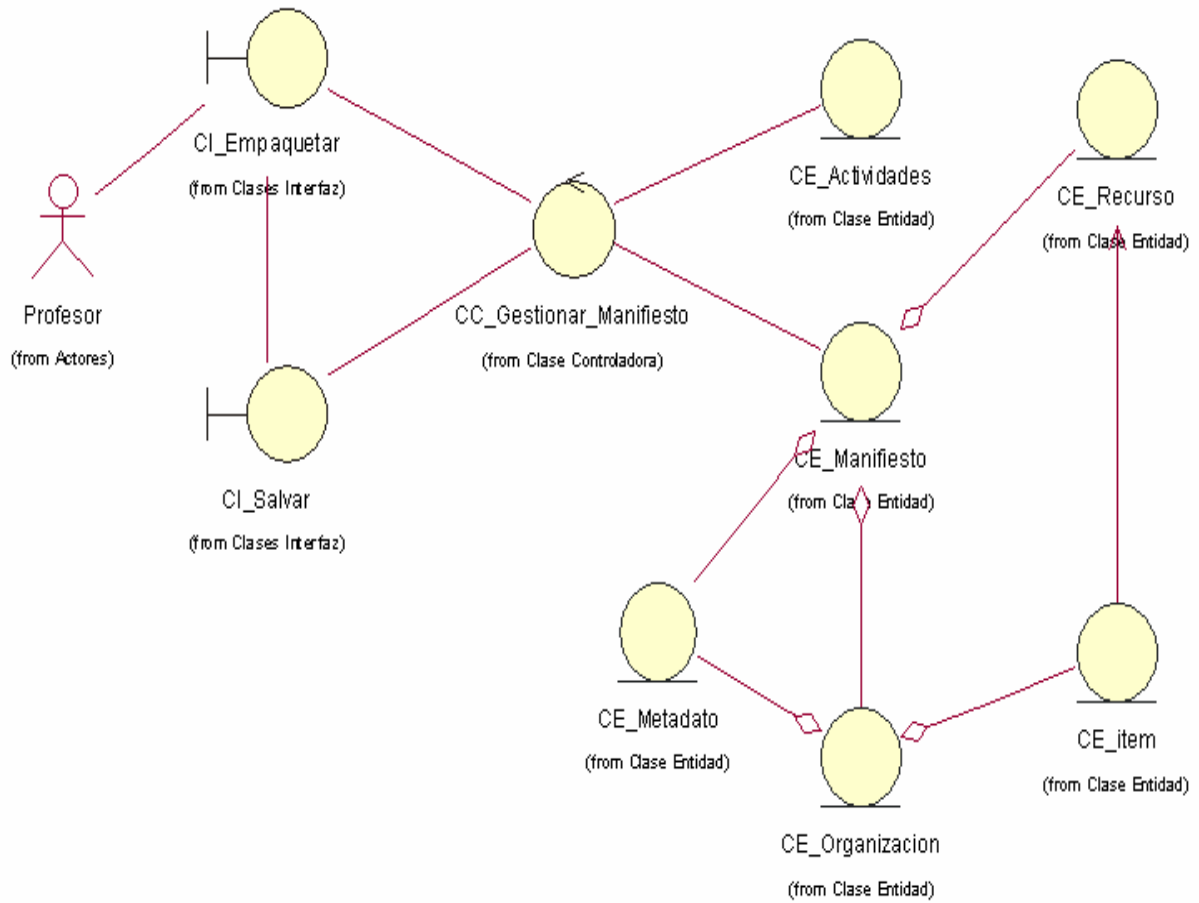


Figura 12: Diagrama de Clases de Análisis: Empaquetar.

4.2 Diseño.

Durante el diseño modelamos el sistema y su arquitectura para que soporte los requisitos funcionales y no funcionales. Una entrada esencial al diseño es el modelo de análisis. El diseño es el centro de atención al final de la fase de elaboración y comienzo de las iteraciones de construcción.

4.2.1 Arquitectura de una aplicación Web.

Las aplicaciones Web, por lo general, suelen estar estructuradas siguiendo un modelo basado en tres niveles.

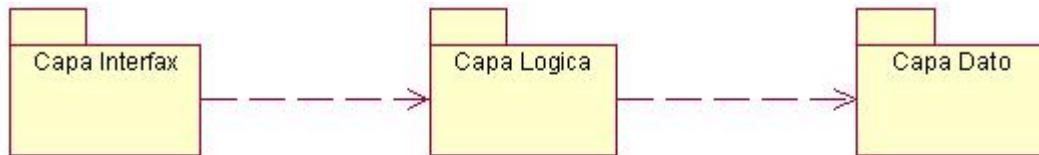


Figura 13: Arquitectura de una aplicación Web.

El primer nivel es la interfaz. Las funciones que se realizan en él se limitan únicamente a enviar las peticiones/consultas que genera el usuario, obtener los datos generados por la aplicación Web (servidor) y finalmente representar los datos en el navegador.

En el segundo nivel se sitúa la lógica. En éste se encuentra el núcleo principal de la aplicación y es el encargado de dotar a la aplicación Web del contenido dinámico.

Las funciones que pueden ser realizadas en este nivel son:

1. interpretación de las consultas generadas por el cliente
2. ejecución de la lógica necesaria para esas consultas
3. obtención/almacenaje de datos
4. envío de los datos generados al cliente

En el tercer nivel estarían los datos. Éste podría estar formado por una base de datos, por ficheros XML o por cualquier otro tipo de sistema de almacenaje de datos dependiendo de las necesidades de la aplicación.

4.2.2 Diagramas de Clases del diseño.

La forma tradicional de modelar clases no es aplicable a la hora de diseñar una aplicación Web, es por esto que Rational creó una extensión para UML que se adapta a la arquitectura de este tipo de sistemas.

Se elaboró un diagrama de clases Web para cada caso de uso del sistema de forma tal que se facilite la comprensión de cómo se relacionan los distintos componentes en la realización de cada uno de ellos.

Caso de Uso: Importar Contenido

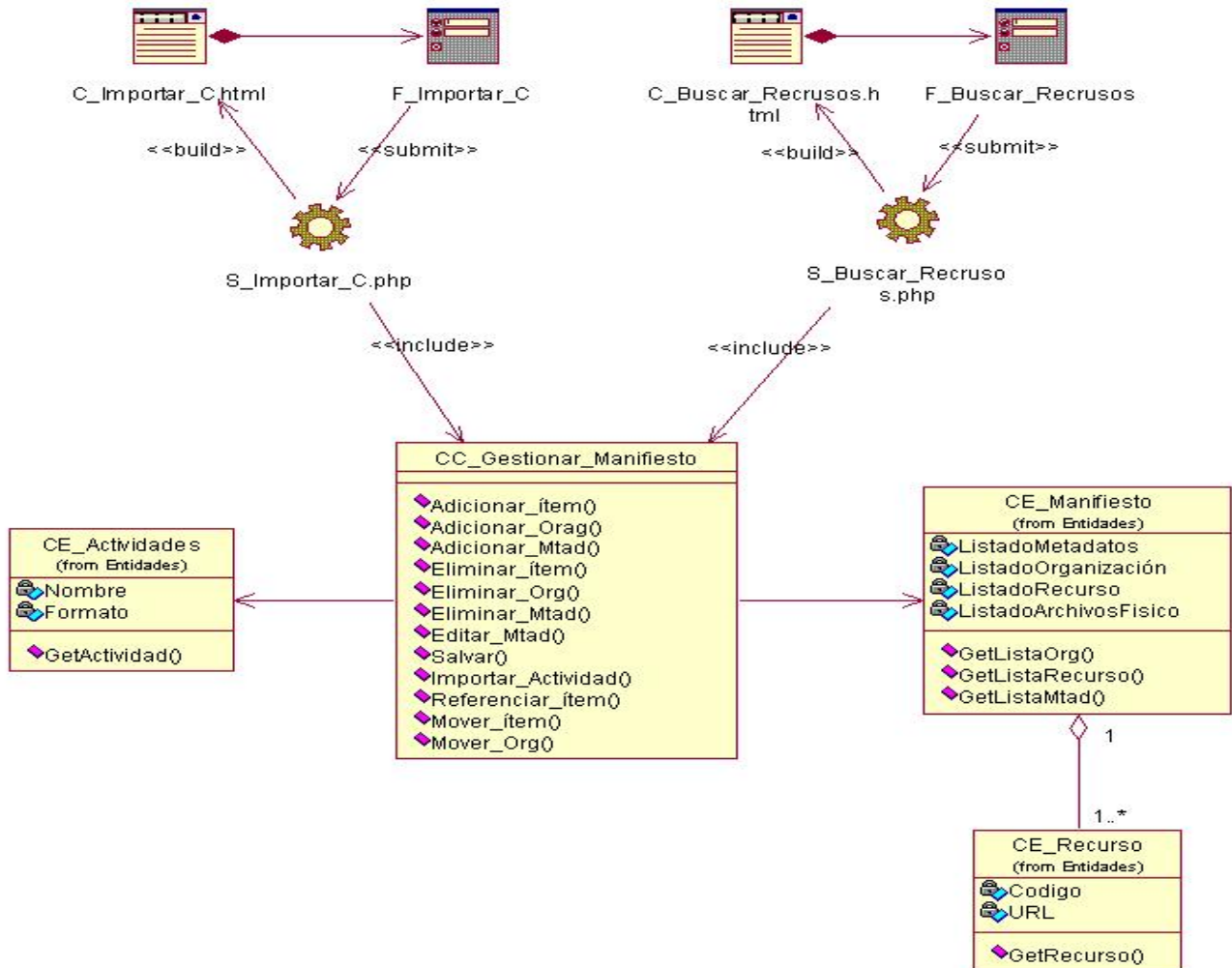


Figura 14: Diagrama de Clase del diseño: Importar Contenido

Caso de Uso: Gestionar Organización

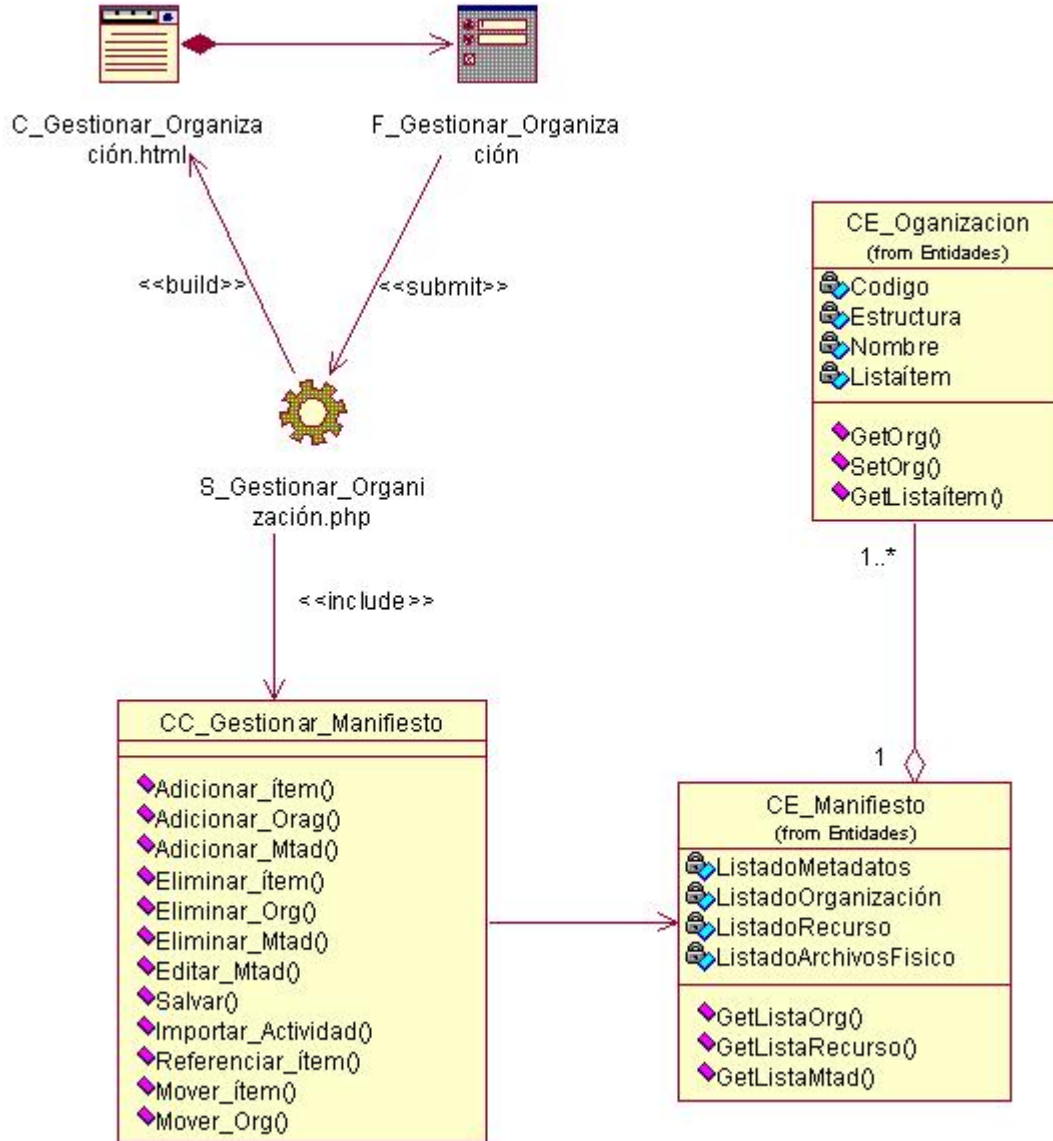


Figura 15: Diagrama de Clase del diseño: Gestionar Organización

Caso de Uso: Gestionar item

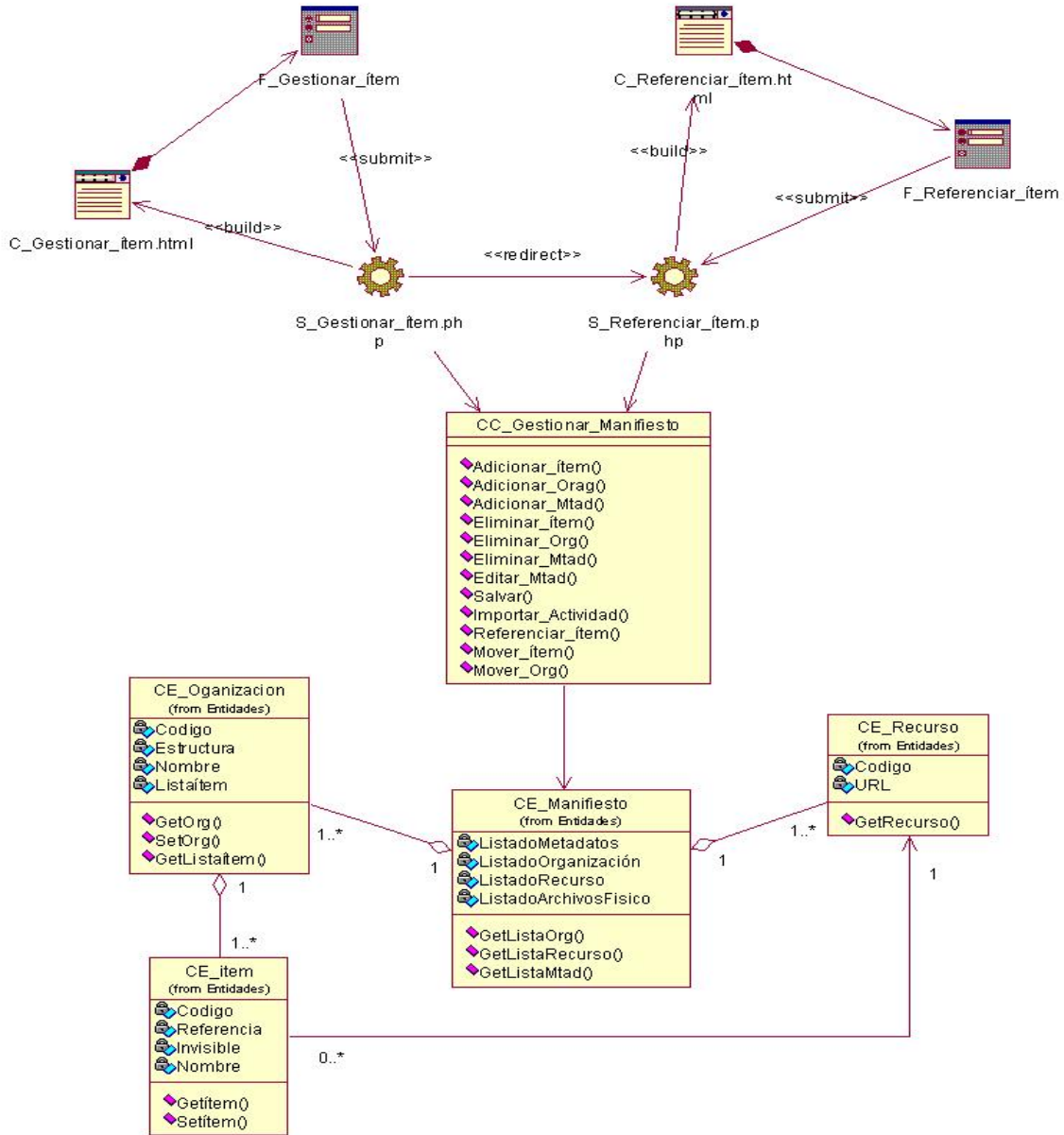


Figura 16: Diagrama de Clase del diseño: Gestionar item

Caso de Uso: Gestionar Metadatos

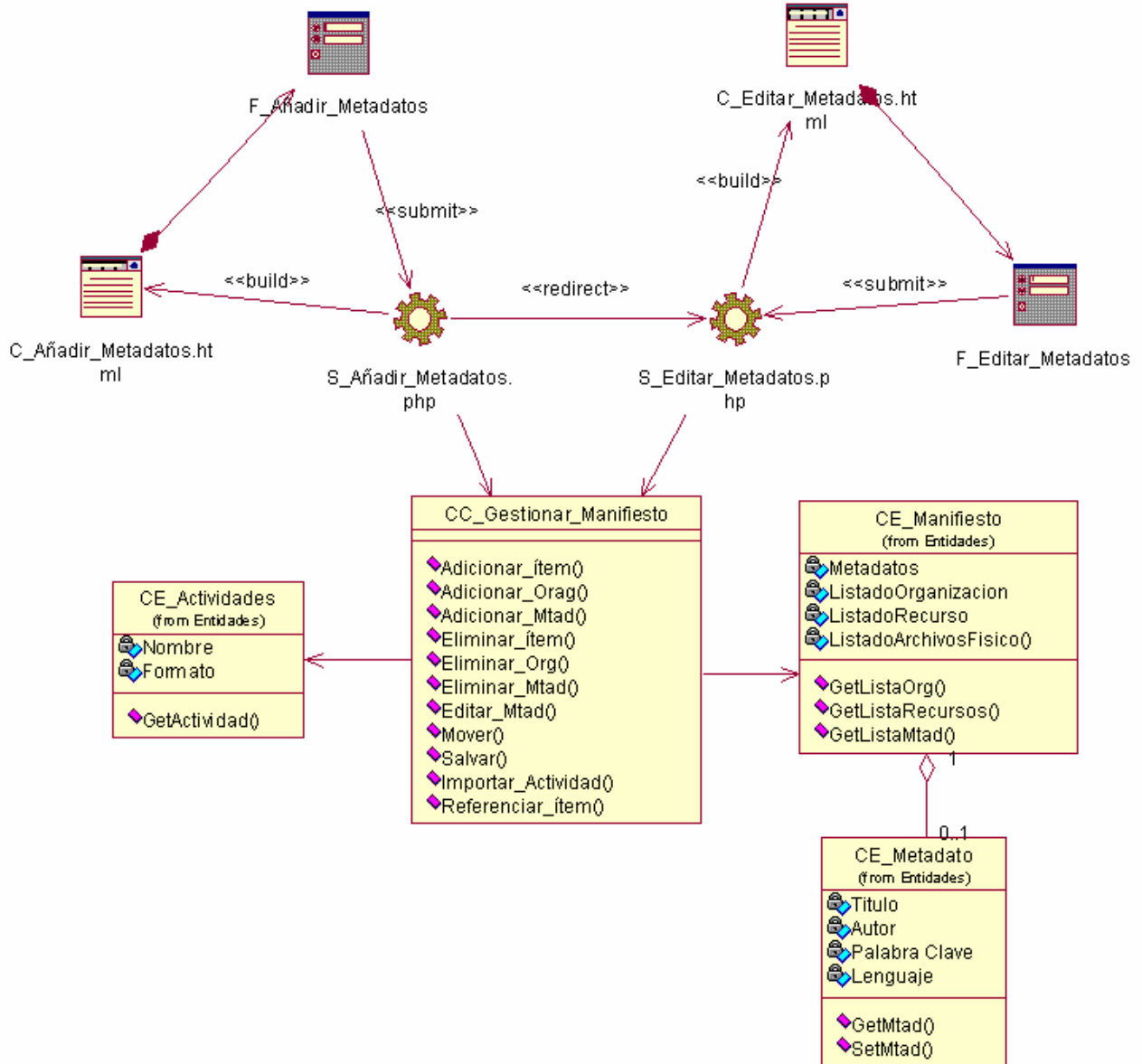


Figura 17: Diagrama de Clase del diseño: Gestionar Metadatos

Caso de Uso: Empaquetar

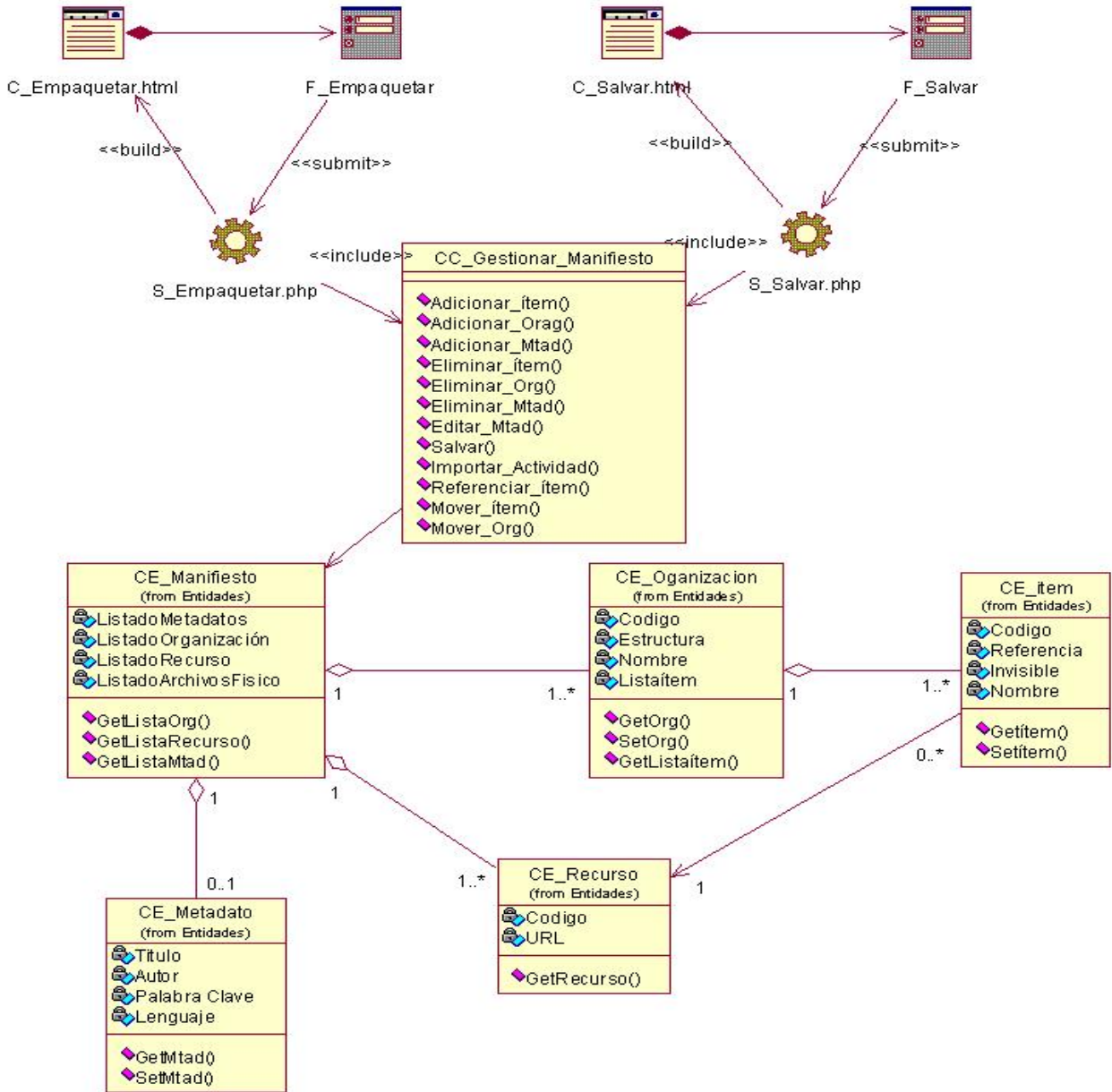


Figura 18: Diagrama de Clase del diseño: Empaquetar

4.3 Diagramas de Secuencia.

Caso de Uso: Importar Contenido

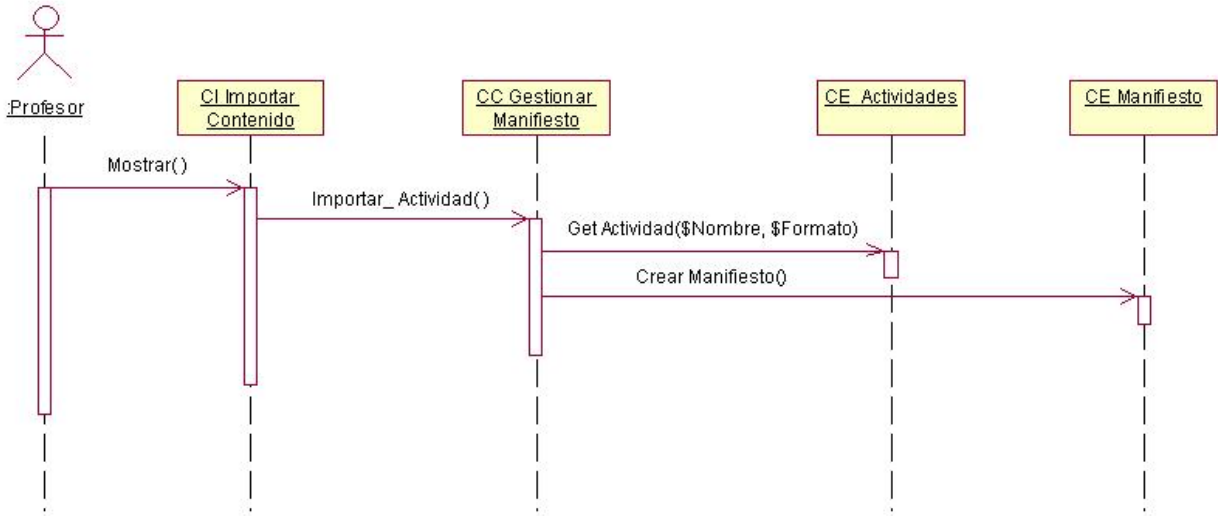


Figura 19: Diagrama de Secuencia del diseño: Importar Contenido

Caso de Uso: Gestionar Organización

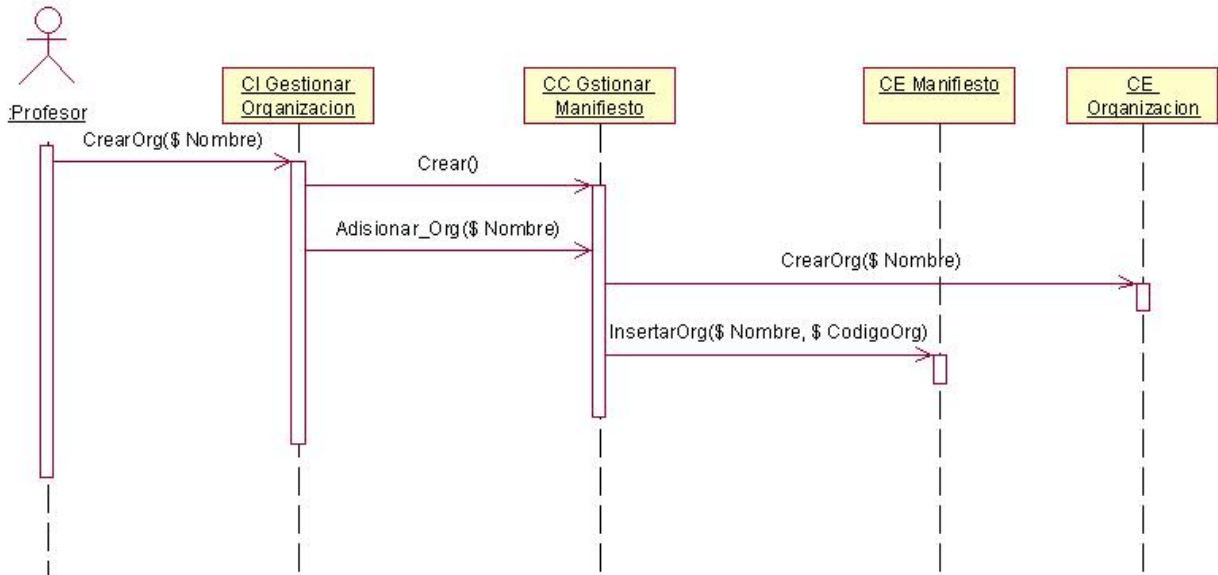


Figura 20: Diagrama de Secuencia del diseño: Crear Organización.

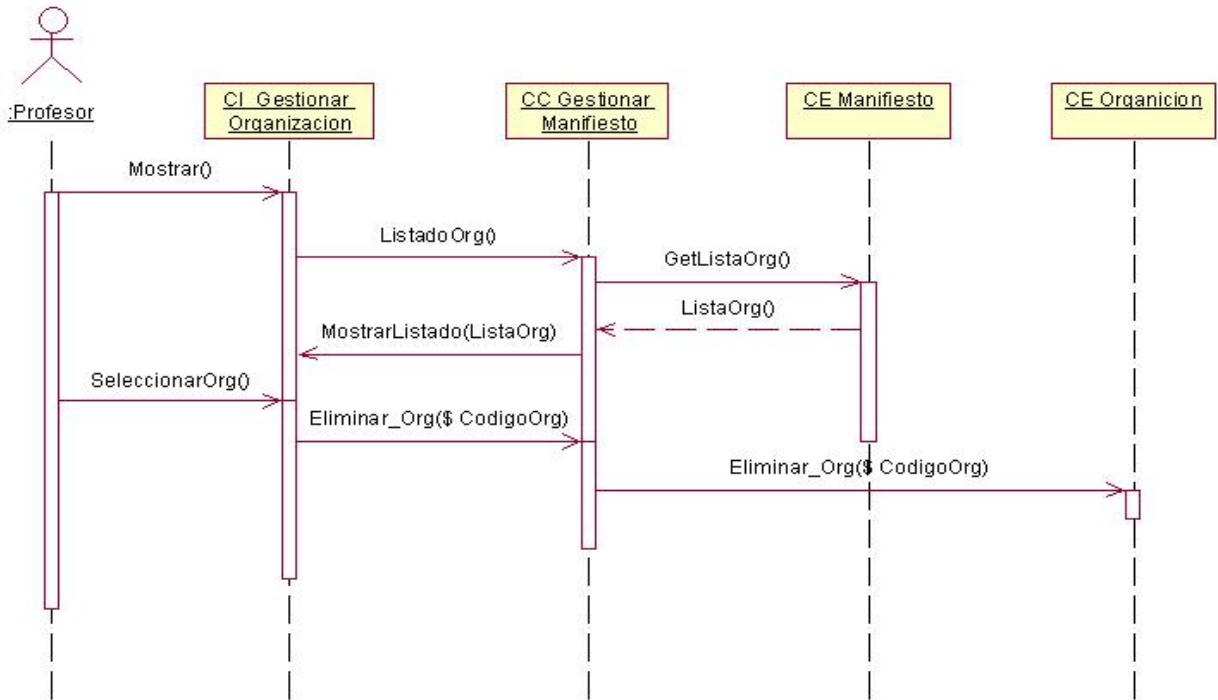


Figura 21: Diagrama de Secuencia del diseño: Eliminar Organización.

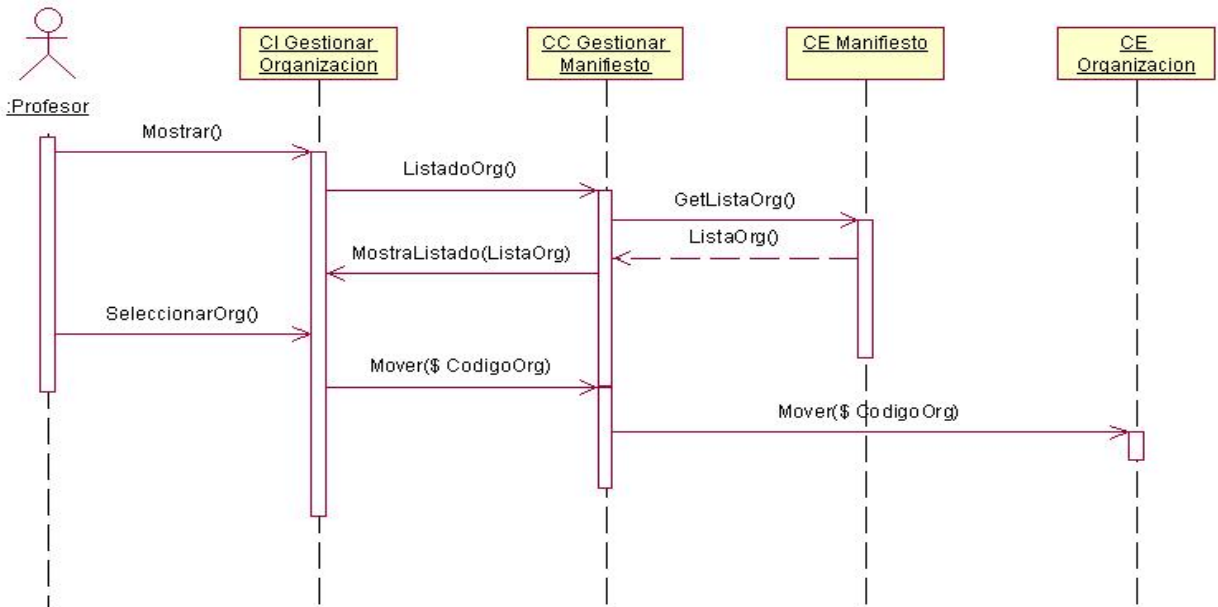


Figura 22: Diagrama de Secuencia del diseño: Mover Organización

Caso de Uso: Gestionar item

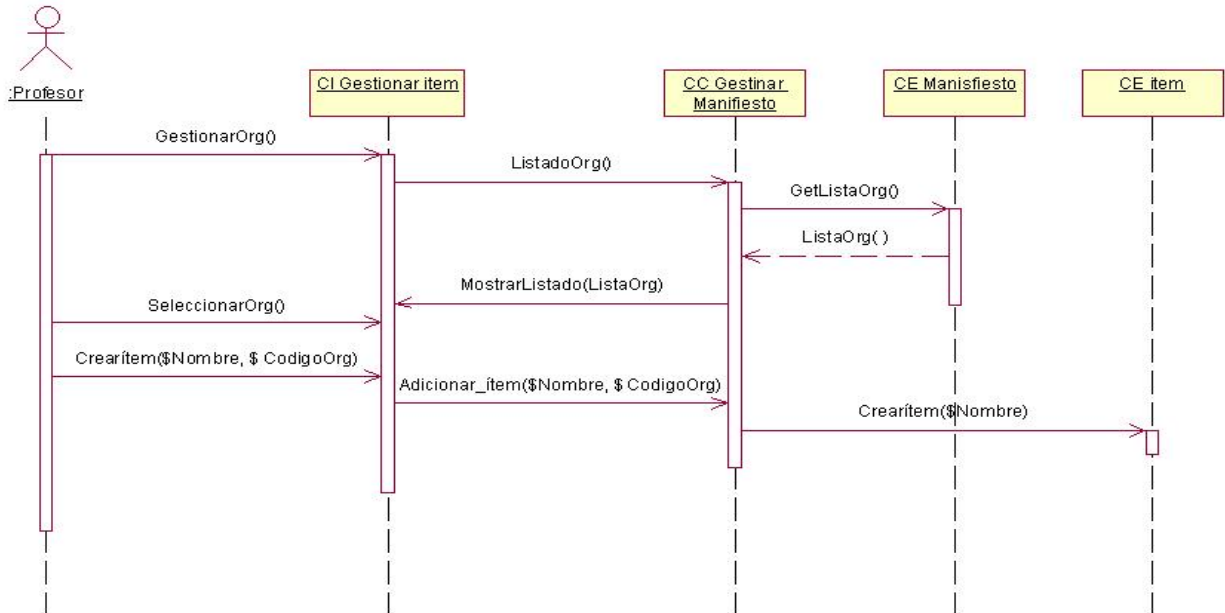


Figura 23: Diagrama de Secuencia del diseño: Crear item.

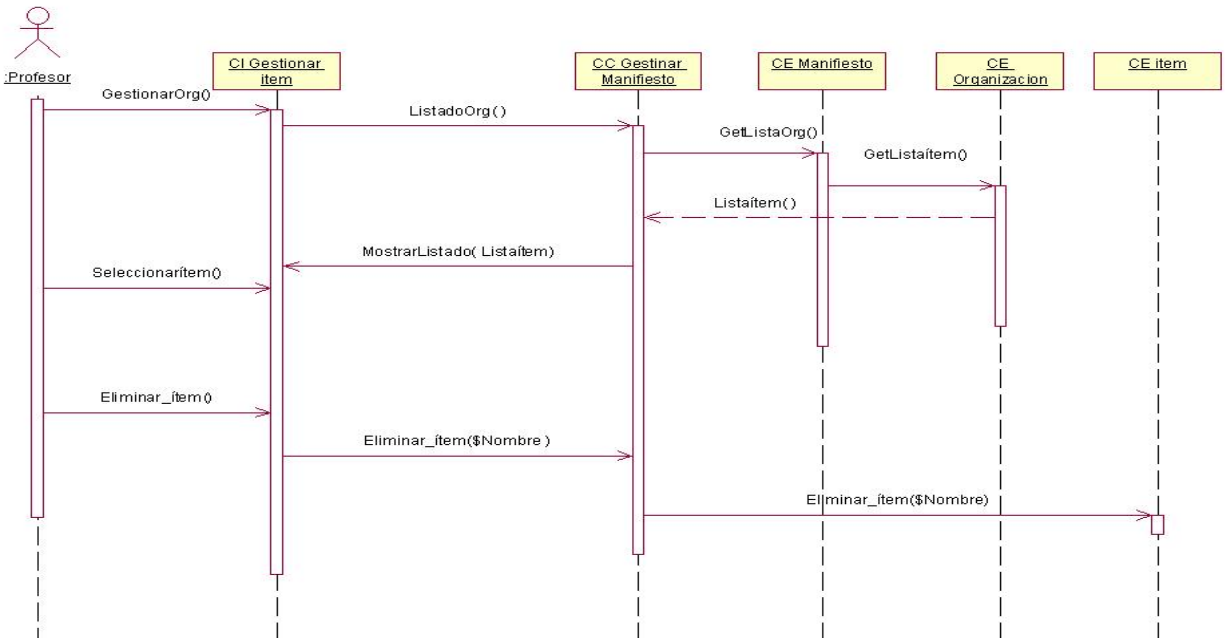


Figura 24: Diagrama de Secuencia del diseño: Eliminar item.

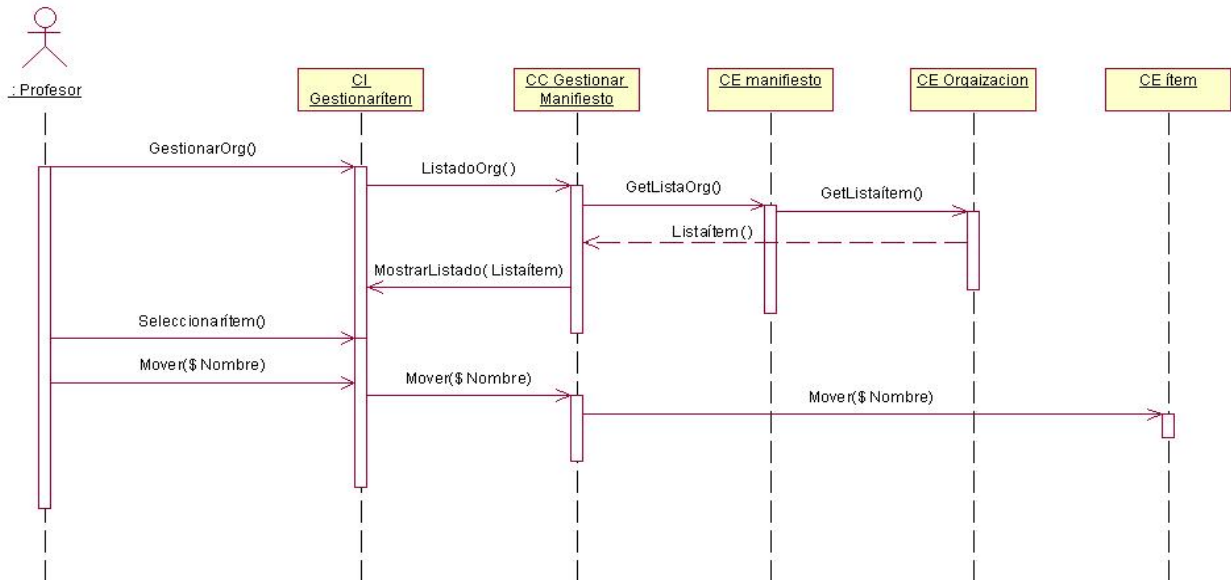


Figura 25: Diagrama de Secuencia del diseño: Mover item.

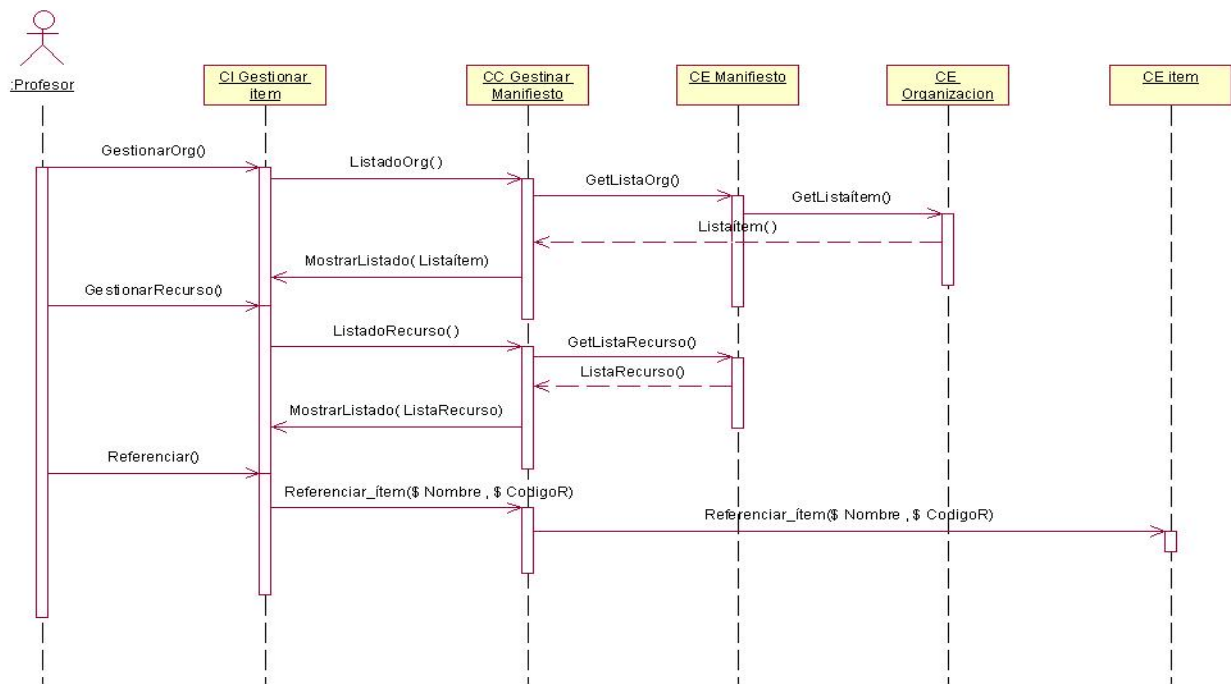


Figura 26: Diagrama de Secuencia del diseño: Referenciar item

Caso de Uso: Gestionar Metadatos

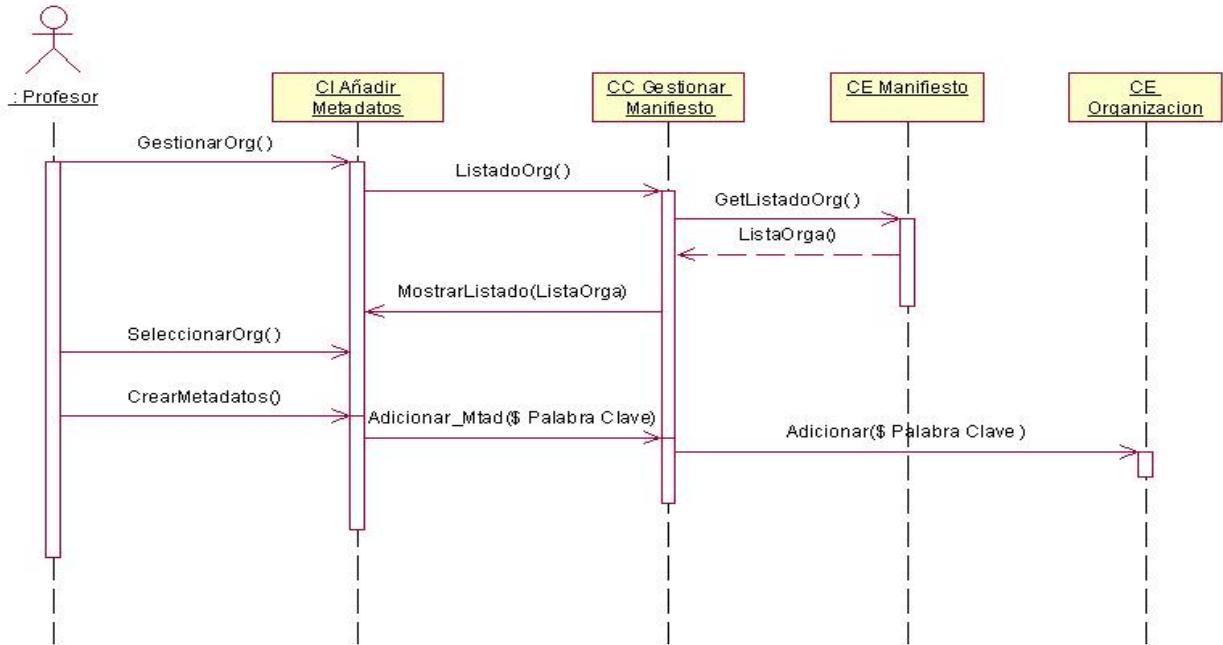


Figura 27: Diagrama de Secuencia del diseño: Crear Metadatos.

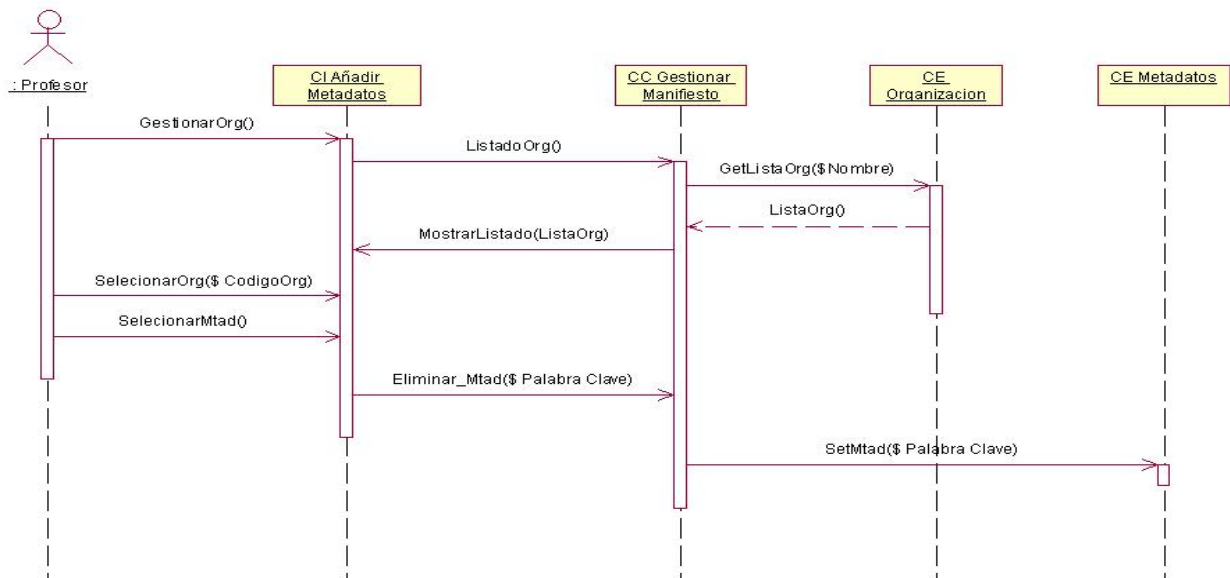


Figura 28: Diagrama de Secuencia del diseño: Eliminar Metadatos.

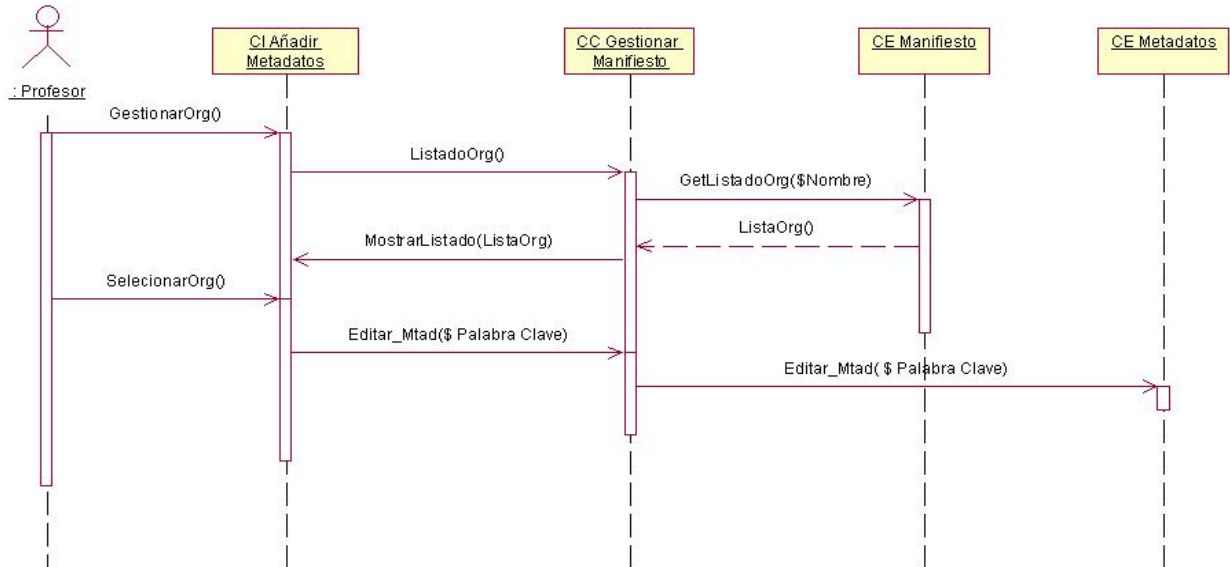


Figura 29: Diagrama de Secuencia del diseño: Editar Metadatos.

Caso de Uso: Empaquetar

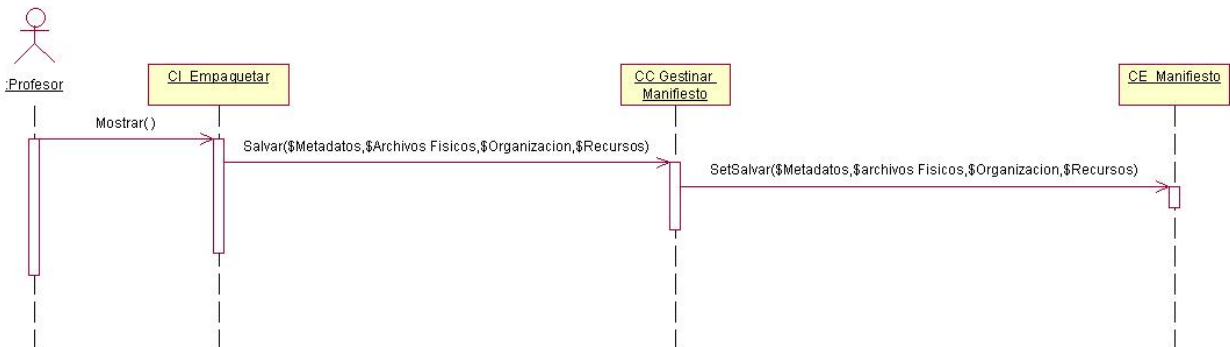


Figura 30: Diagrama de Secuencia del diseño: Empaquetar

4.3.1 Descripción de las clases.

Tabla 8: Gestionar Manifiesto.

Nombre: CC_ Gestionar Manifiesto	
Tipo de clase (controladora)	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	Adicionar_item(\$Nombre, \$CodigoOrg)
Descripción:	Inserta un item en una organización
Nombre:	Adicionar_Org(\$ Nombre)
Descripción:	Inserta una organización
Nombre:	Adicionar_Mtad(\$ Palabra Clave)
Descripción:	Inserta Metadatos en el elemento seleccionado
Nombre:	Eliminar_item(\$Nombre, \$CodigoOrg)
Descripción:	Elimina al elemento item de la organización seleccionada
Nombre:	Eliminar_Org(\$CodigoOrg)
Descripción:	Elimina una organización
Nombre:	Eliminar_Mtad(\$ Palabra Clave)
Descripción:	Elimina los Metadatos del elemento seleccionado
Nombre:	Editar_Mtad(\$ Palabra Clave)
Descripción:	Muestra los campos a llenar
Nombre:	Mover_item(\$ Nombre)

Descripción:	Organiza ascendente o descendente al item seleccionado.
Nombre:	Salvar(\$Metadatos,\$ Archivos Físicos, \$Organización, \$Recursos)
Descripción:	Guarda todos los elementos en el documento manifiesto
Nombre:	Referenciar_item(\$ Nombre , \$CodigoR)
Descripción:	Asocia un elemento item a uno recurso
Nombre:	Mover_org(\$CodigoOrg)
Descripción:	Organiza ascendente o descendente a la Organización seleccionada.
Nombre:	Importar_Actividad(\$Nombre, \$Formato)
Descripción:	Captura toda la información de la carpeta a empaquetar

Tabla 9: Manifiesto.

Nombre: CE_Manifiesto	
Tipo de clase (entidad)	
Atributo	Tipo
Listado Metadatos	STRING
Listado Organización	STRING
Listado Recurso	STRING
Listado Archivos Físicos	STRING
Para cada responsabilidad:	
Nombre:	GetLsitaOrg()
Descripción:	Devuelve un listado con todas las organizaciones creadas
Nombre:	GetLsitaMtad()

Descripción:	Devuelve el Metadato seleccionado
Nombre:	GetLsitaRecurso()
Descripción:	Devuelve un listado con todos las recursos importados

Tabla 10: Organización.

Nombre: CE_Organización	
Tipo de clase (controladora)	
Atributo	Tipo
Código	STRING
Nombre	STRING
Estructura	STRING
Listaitem	STRING
Para cada responsabilidad:	
Nombre:	GetOrg()
Descripción:	Devuelve una organización
Nombre:	SetOrg()
Descripción:	Crea una organización
Nombre:	GetListaitem()
Descripción:	Devuelve un listado con los elementos item

Tabla 11: item.

Nombre: CE_item

Tipo de clase (entidad)	
Atributo	Tipo
Código	STRING
Referencia	STRING
Nombre	STRING
Invisible	BOOL
Para cada responsabilidad:	
Nombre:	Getitem()
Descripción:	Devuelve un elemento item
Nombre:	Setitem()
Descripción:	Crea al elemento item

Tabla 12: Metadatos.

Nombre: CE_Metadatos	
Tipo de clase (entidad)	
Atributo	Tipo
Título	STRING
Autor	STRING
Palabra Clave	STRING
Lenguaje	STRING
Para cada responsabilidad:	

Nombre:	GetMtad()
Descripción:	Devuelve un elemento Metadato
Nombre:	SetMtad()
Descripción:	Crea al elemento Metadato

Tabla 13: Recursos.

Nombre: CE_Recurso	
Tipo de clase (controladora)	
Atributo	Tipo
Código	STRING
URL	STRING
Para cada responsabilidad:	
Nombre:	GetRecurso()
Descripción:	Devuelve un elemento Recurso

4.3.2 Clases Persistentes.

El diagrama de clases persistentes es usado para modelar la estructura lógica de la base de datos con clases representando tablas, y atributos de clase representando columnas.

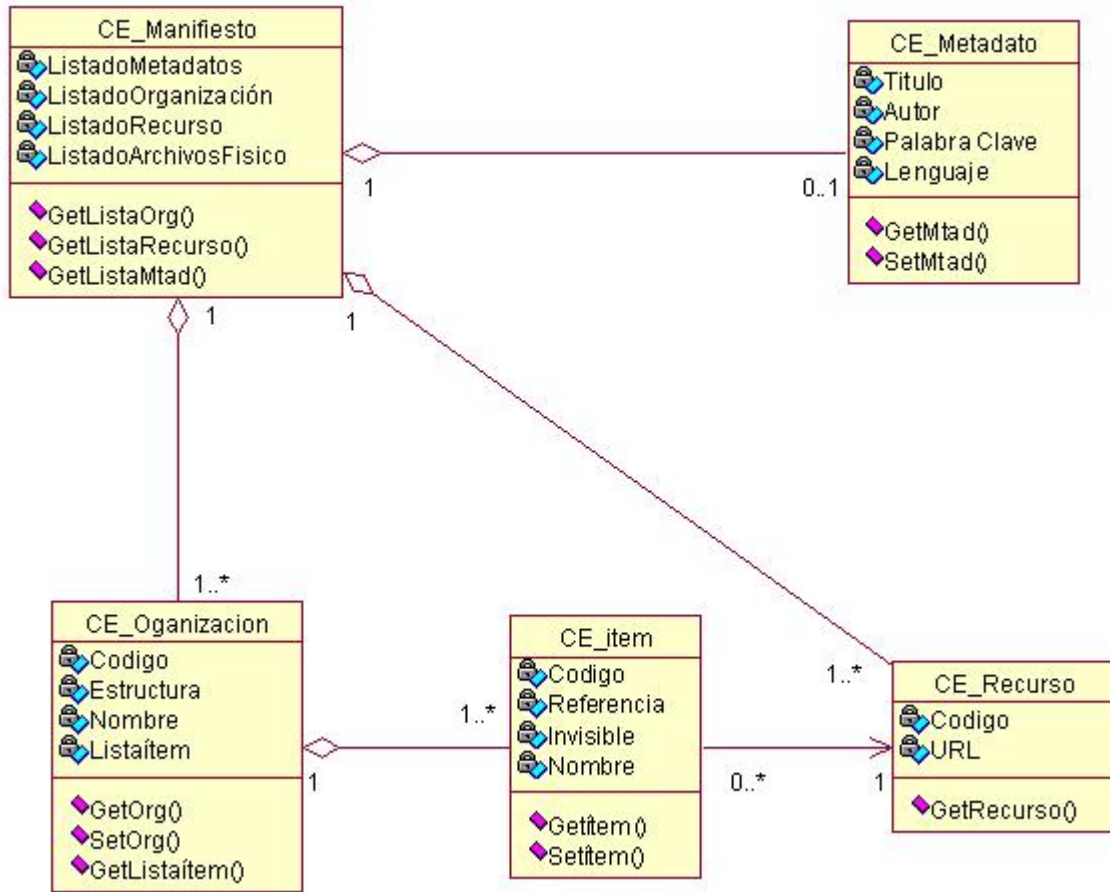


Figura 31: Diagrama de Clases Persistentes.

4.3.3 Modelo de Datos.

En el modelo de datos es necesario realizar algunas consideraciones a la hora de la implementación, es necesario conformar una serie de consultas para cada una de las entidades participantes en el modelo que jugarán un papel determinante para el funcionamiento de la aplicación, estas consultas comprenderán las responsabilidades básicas fundamentales de cada entidad con el fin de facilitar la implementación del lado del negocio y del cliente.

R-4.3.3.1 El diagrama del modelo de datos

Se corresponde con la representación física de la base de datos.

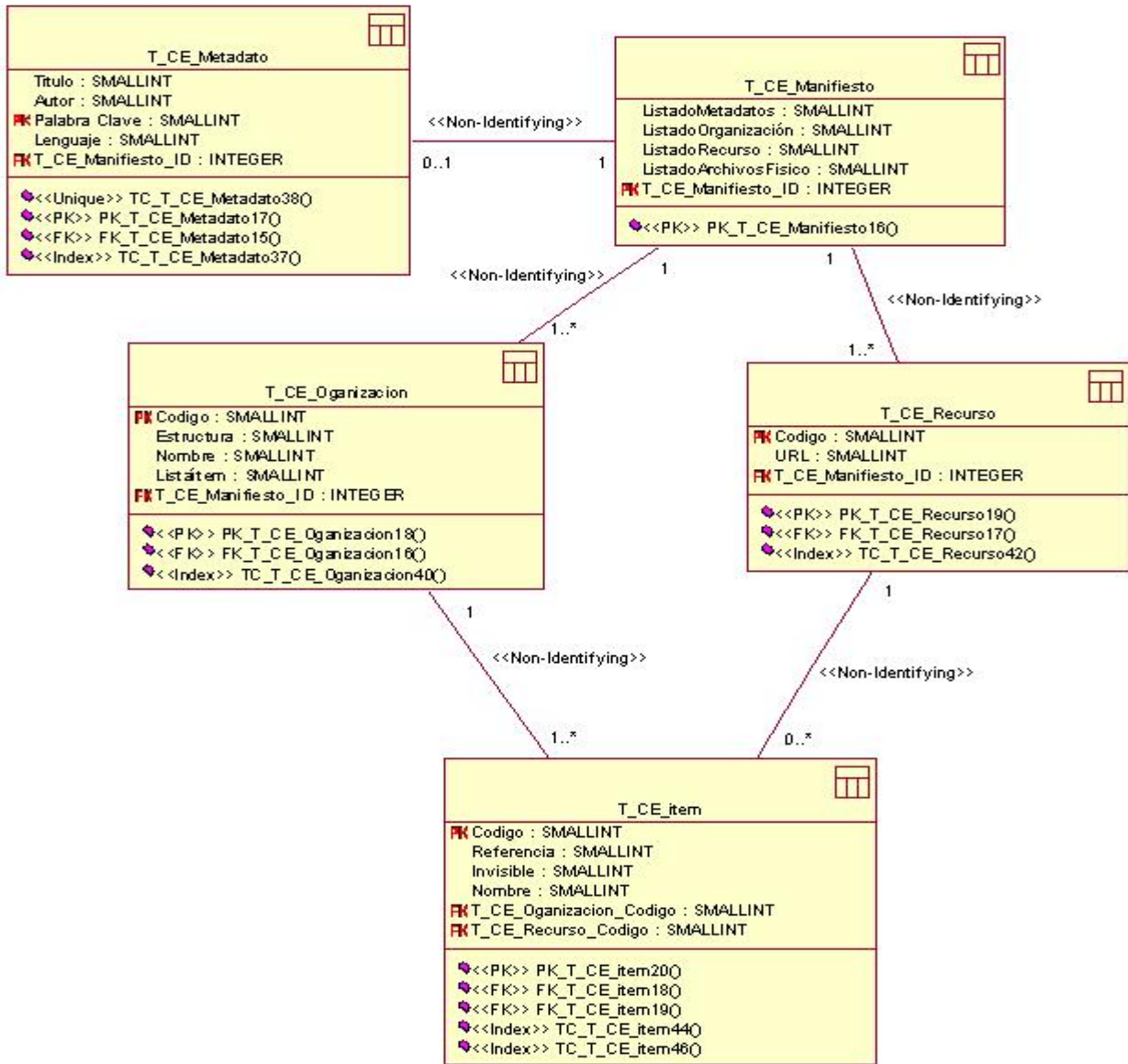


Figura 32: El diagrama del modelo de datos.

R-4.3.3.2 Descripción de las tablas.

Tabla 14: Modelo de Dato Manifiesto.

Nombre: Manifiesto		
Descripción: Esta tabla recoge todos los datos que se necesitan sobre los demás elementos, como Metadatos, Organización, Recurso, Archivos Físicos.		
Atributo	Tipo	Descripción
ListadoMetadatos	STRING	En este atributo se guarda la información de todos los Metadatos creados.
ListadoOrganización	STRING	En este atributo se guarda la información de todas las Organizaciones creados.
ListadoRecurso	STRING	En este atributo se guarda la información de todos los Recursos creados.
ListadoArchivoFisico	STRING	En este atributo se guarda la información de todos los Archivos Físicos.

Tabla 15: Modelo de Dato Organización.

Nombre: Organización		
Descripción: Esta tabla recoge todos los datos que se necesitan sobre las organizaciones, como su Nombre, Estructura, Código, Lista de item.		
Atributo	Tipo	Descripción
Código	STRING	Este atributo es un identificador para cada elemento de la organización y es de carácter obligatorio.
Nombre	STRING	En este atributo

Estructura	STRING	En este atributo se describe la forma de la organización
Listaitem	STRING	En este atributo se guarda la información de todos los items creados.

Tabla 16: Modelo de Dato item.

Nombre: item		
Descripción: Esta tabla recoge todos los datos que se necesitan sobre los Items creados, como su Nombre, Titulo, Código, Invisible, Referencia.		
Atributo	Tipo	Descripción
Código	STRING	Este atributo es un identificador para cada elemento item y es de carácter obligatorio.
Nombre	STRING	En este atributo se guarda el nombre del elemento
Referencia	STRING	En este atributo corresponde a una identificación del item en relación con un Learning Object.
Invisible	BOOL	En este atributo representa si el item es visible o no.

Tabla 17: Modelo de Dato Metadatos.

Nombre: Metadato		
Descripción: Esta tabla recoge todos los datos que se necesitan sobre los Metadatos, como su Palabra Clave, Autor, Lenguaje, Titulo.		

Atributo	Tipo	Descripción
Título	STRING	En este atributo corresponde a un nombre o título para el Metadato.
Autor	STRING	En este atributo representa el nombre del cliente.
Palabra Clave	STRING	En este atributo un nombre que se asocia al ítem.
Lenguaje	STRING	En este atributo se representa el lenguaje en que está creado.

Tabla 18: Modelo de Dato Recurso.

Nombre: Recurso		
Descripción: Esta tabla recoge todos los datos que se necesitan sobre los Recursos, como su URL, Código.		
Atributo	Tipo	Descripción
Código	STRING	Este atributo es un identificador para cada elemento Recurso y es de carácter obligatorio.
URL	STRING	En este atributo representa la dirección física de cada recurso.

4.3.4 Modelo de Despliegue.

El modelo de despliegue describe la distribución física del sistema, muestra como están distribuidos los componentes de software entre los distintos nodos de cómputo. Permite comprender la correspondencia entre la arquitectura software y la arquitectura hardware.

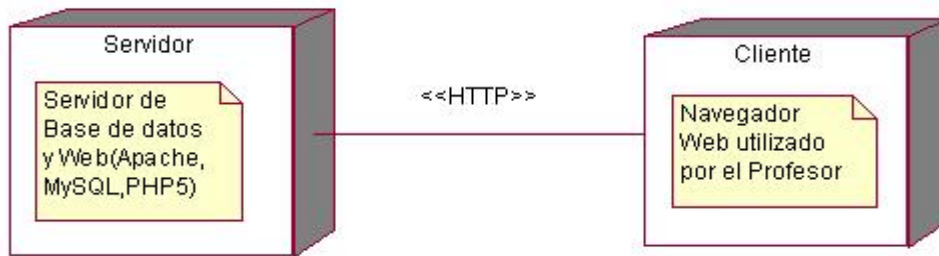


Figura 33: Diagrama del Modelo de Despliegue.

4.3.5 Principios de diseño.

El diseño de la interfaz es uno de los puntos fundamentales a tratar a la hora de la presentación de la aplicación teniendo en cuenta que es lo que ve el usuario y por lo tanto debe ser lo más amigable y comprensible posible. Una aplicación con una interfaz bien diseñada debe tener, además de un buen diseño gráfico, una buena navegabilidad, usabilidad y distribución de los contenidos.

Para el diseño de la interfaz de usuario de este sistema se han seguido los siguientes principios: Permitir su utilización por cualquier persona que tenga conocimientos básicos de informática.

1. Permitir al usuario una interfaz atractiva y flexible facilitando exactitud y precisión.
2. Los objetos que componen la interfaz den a entender al usuario de su existencia y su funcionalidad.

3. Brindar a los usuarios una amplia legibilidad de la información esencial.
4. Brindar a los usuarios avisos eficaces y métodos de respuesta durante y tras la finalización de la tarea, que elimine la complejidad innecesaria.

R-4.3.5.1 Interfaz de usuario

Con el objetivo de lograr un diseño consistente de la interfaz de la aplicación, se respetó en todas las páginas el esquema Cabecera-Navegador-Contenido. La cabecera contiene el nombre de la aplicación en la esquina superior izquierda y el logo de la organización que puede ser fácilmente configurable. En el navegador se incluyen varios bloques. En el área del contenido se muestran los restantes contenidos (formularios de entrada, las salidas, etc.).

R-4.3.5.2 Tratamiento de errores

El tratamiento de errores una vez que ocurre una excepción el cliente es redireccionado a una página de error con el mensaje correspondiente.

Por otra parte el módulo verifica que las actividades a empaquetar no contengan ya el manifiesto creado utilizando un mensaje de confirmación si el mismo está. Por último, se utilizan errores en forma de mensajes de texto en la misma página donde se ejecutó la acción, de forma que el usuario pueda corregir más fácilmente y continuar.

Conclusión

En el presente capítulo se obtuvo la solución al sistema propuesto, mostrando los resultados de la etapa de diseño. Se modelaron los componentes como clases utilizando las extensiones del UML, y se presentaron a través de diagramas de clases Web. A partir del refinamiento del diagrama conceptual se obtuvo un modelo físico de datos en correspondencia con los requerimientos planteados en el capítulo anterior.

Se realizó el diseño del diagrama de despliegue donde se describen los nodos de procesamiento en tiempo real donde se ejecutará la aplicación y los vínculos entre ellos. Todos estos elementos obtenidos son claves para la correcta implementación del sistema propuesto.

CONCLUSIONES

En este trabajo de diploma se realizó un estudio sobre el proceso de estandarización en el campo del e-Learning. Fueron obtenidas muchas concepciones adaptables a cualquier aplicación Web que se desarrolle en el Proyecto de Laboratorios Virtuales.

1. De las cuatro principales iniciativas de estandarización existentes en la actualidad: AICC, IEEE, IMS y ADL, se ha seleccionado la propuesta de ADL SCORM, por ser el estándar más usado en la industria de los contenidos educativos.
2. Se muestran las ventajas de la estandarización en LMS, ya que proporcionan reglas de estructuración para los contenidos y facilita la tarea tanto a los desarrolladores de software educativo, como al resto de los implicados en el proceso de enseñanza-aprendizaje.
3. El e-Learning ofrece ventajas comparativas respecto a otros modos de aprendizaje ya que no importa el lugar o distancias donde se encuentren los usuarios ni la actividad que realicen, si se cuenta con un acceso a Internet, los usuarios pueden acceder a cualquier contenido de aprendizaje.
4. XML provee grandes ventajas a los programadores y desarrolladores, ya que es un lenguaje extremadamente sencillo y flexible, solo se centra en los datos y no en la presentación. En el caso del desarrollo de aplicaciones Web bajo la estandarización ADL-SCORM, el desarrollador define que etiquetas son las que almacenarán los datos de Contenidos, Módulos o Lecturas y los Objetos de Aprendizaje Asociados.
5. La definición de los requerimientos del sistema permite estructurar el modelo de casos de uso del sistema y contribuye en la descripción de los caso de uso para una mejor comprensión de la funcionalidad brindada, además de guiar las funcionalidades a plasmar en un prototipo de interacción con el cliente.
6. Se realizó el Modelo de Dominio definiendo las principales clases a través del modelo conceptual. Se estructuró el modelo de casos de uso del sistema, describiéndose los caso de uso principales para una mejor comprensión de las funcionalidades que brindan. Se diseñó el sistema, a través de diagramas de clases Web, el diagrama de clases persistentes, entre otros. Se estructuró el modelo

de datos, que es la representación física de la base de datos. Posteriormente se elaboró el modelo de despliegue.

7. En el futuro podría implementarse una estandarización ADL-SCORM en la aplicación Web del proyecto de Laboratorios Virtuales en nuestra universidad, para el tratamiento de contenidos de Aprendizaje, específicamente en secuencias de tareas. La estandarización ADL-SCORM trata acerca de la reusabilidad de los objetos de aprendizaje. Almacena a estos objetos y sus características en repositorios para su posterior uso. La idea es tener un lugar de almacenamiento en el cual se encuentren todos los archivos disponibles para el profesor y puedan ser utilizados y aplicados más de una sola vez.

RECOMENDACIONES

Continuar con el estudio del tema tratado, con el objetivo de encontrar nuevas funcionalidades para refinar el proceso de estandarización en las condiciones previstas dentro del proyecto Laboratorios Virtuales.

Iniciar el ciclo de implementación de una primera versión de la funcionalidad correspondiente a la estandarización de secuencias de tareas en la aplicación Web prevista en el Proyecto.

Dada la complejidad en el desarrollo de la funcionalidad en la aplicación Web prevista, los autores recomiendan más de un equipo para distribuir las tareas de implementación, proponiendo como metodología de desarrollo de software a RUP, adaptándolo a las condiciones del proyecto.

BIBLIOGRAFÍA

- ALBERTO, C. C. C. *Generalidades de la Perspectiva Tecnológica del e-Learning*, 2004. [Disponible en: http://www.monografias.com/trabajos901/generalidades-perspectiva-tecnologica-e-learning/generalidades-perspectiva-tecnologica-e-learning.shtml#_Toc131657775]
- ALVAREZ G., L. *REPOSITORIO DE OBJETO DE APRENDIZAJE COLABORATIVO*, 2003, 24 noviembre. [Disponible en: http://www.tise.cl/archivos/tise2003/papers/repositorio_de_objetos.pdf]
- ALVAREZ G., L. E., DANIELA *Empaquetamiento de Objetos de Aprendizaje Bajo el Estandar SCORM*, 2001. [Disponible en: <http://www.tise.cl/archivos/tise2005/08.pdf>]
- BAKKEN, S. S. *Manual de PHP* 2003. [Disponible en: http://www1.us.es/pautadatos/publico/personal/pas/2166/1498/php_manual.pdf]
- CARLOS, C. *Parsing XML*, 2004, Diciembre.
- JACOBSON, I. B., G.; RUMBAUGH, J. *"El Proceso Unificado de Desarrollo de Software"*. 2 Tomos. Addison Wesley., 2000. p.
- JOAQUIN, G. *Vista general de ASP.NET*, 2004. [Disponible en: <http://www.webestilo.com/aspnet/aspnet01.phtml>]
- LUCAS. *Introducción a Java*. Disponible en: <http://www.monografias.com/trabajos/java/java.shtml>
- MAURER, W. *Estándares elearning*, 2002-2004. [Disponible en: <http://fgsnet.nova.edu/cread2/pdf/Maurer.pdf>]
- MIGUEL, A. A. *Evaluando Zend Studio*, 03.11.2003. [Disponible en: <http://www.maestrosdelweb.com/editorial/zendstudio/>]
- MIGUEL, R. P. *El estándar SCORM para EaD*, Diciembre 2004. [Disponible en: <http://mrebollo.webs.upv.es/pubs/tesina.pdf>]
- MOLPECERES, A. *Procesos de desarrollo: RUP, XP y FDD.*, 05/01/2007. [Disponible en: <http://www.javahispano.org/articles.article.action?id=76>]
- MONTERO, J. B. *XML*, 1999. [Disponible en: <http://www.programacion.net/html/xml/principal.htm>]
- MURUGARREN, J. G. *Manual de MySQL*, 2006. [Disponible en: <http://www.webestilo.com/mysql/>]
- S.L., E. G. M. *Tutorial de SQL*, 2006. [Disponible en: <http://www.desarrolloweb.com/manuales/9/>]
- SOFTONIC, E. D. *NuSphere PHPEd "Editor de programación para PHP y otros formatos"*, 2005 [Disponible en: <http://nusphere-phped.softonic.com/>]
- Wikipedia. *La Enciclopedia Libre. Oracle*. Disponible en: <http://es.wikipedia.org/wiki/Oracle>
- Wikipedia. *La Enciclopedia Libre. PostgreSQL*. Disponible en: <http://es.wikipedia.org/wiki/PostgreSQL>
- WIKIPEDIA®. *Proceso Unificado de Rational*, 2007. [Disponible en: http://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational]

ANEXOS

Anexo 1: Los metadatos del sistema

```

<metadata>
<schema>ADL Scorm</schema>
<schemaversion>CAM 1.3</schemaversion>
<lom>
<general>
<identifier>
<catalog>URI</catalog>
<entry>http://www.adlnet.org/CO_01</entry>
</identifier>
<author>Claudio Guzmán Herrera</author>
<title>
<string language="sp">Generación y Administración de
Contenidos</string>
</title>
<language>sp</language>
<description>
<string language="en">textual description</string>
</description>
<keyword>
<string language="en">learning objects</string>
</keyword>
<coverage>
<string language="en">Circa,16th Century</string>
</coverage>
<structure>
<source>LOM v1.0</source>
<value>atomic</value>
</structure>
<aggregationLevel>
<source>LOM v1.0</source>
<value>2</value>
</aggregationLevel>
</general>
</lom>
</metadata>

```

Anexo 2: Los elemento Organizations y Organization pueden quedar empaquetados de la siguiente forma:

```

<organizations>

<organization identifier="CC1463">

<title>redes</title>

```



```
</organization>
<organization identifier="CCM1425">
<title>redes</title>
</organization>
<organization identifier="OFG2525">
<title>Etica Cristiana</title>
</organization>
.....
.....
</organizations>
```

Ejemplo 2. Estructura de los elementos <organizations> y <organization>

Elementos hijos del elemento <organization>
Los nodos hijos del elemento organization son los siguientes:

```
<title>
<item>
<metadata>
<imsss:sequencing>
```

En el desarrollo de este trabajo, sólo se van a tomar en cuenta los tres primeros elementos nombrados.

Anexo 3: Elemento Metadatos.

```
<organization identifier="curso1">
<title>teleprocesos</title>
```

```
<metadata>
<lom>
<general>
<author>Claudio Guzman Herrera</author>
<keyword>contenido 1</keyword>
<language>sp</language>
<description>
<string language="sp">Esta es una descripción textual para el
contenido 1</string>
</description>
</general>
</lom>
</metadata>
```

Ejemplo 4. Estructura del elemento <metadata> dentro del elemento <organization>.

Anexo 4: Elemento <item>.

```
<item identifier="1">
<title>módulo 1</title>
<item identifier="1" identifierref="rec1">
<title>lectura 1</title>
</item>
<item identifier="2" identifierref="rec2">
<title>lectura 2</title>
</item>
```

</item>

Ejemplo 5. Ejemplo de elemento <item>.

Anexo 5: Sub Elementos de <item>.

```
<organization identifier="curso2">
```

```
<title>redes</title>
```

```
<item identifier="2">
```

```
<title>contenido 2</title>
```

```
<metadata>
```

```
<general>
```

```
<author>Claudio Guzman Herrera</author>
```

```
<keyword>contenido 2</keyword>
```

```
<language>sp</language>
```

```
<description>
```

```
<string language="sp">Esta es una descripción textual para
```

```
el contenido 2</string>
```

```
</description>
```

```
</general>
```

```
</metadata>
```

```
<item identifier="1" identifierref="rec100">
```

```
<title>módulo1</title>
```

```
</item>
```

```
</item>
```

```
<metadata/>
```

</organization>

Ejemplo 7. Estructura del elemento <metadata> dentro del elemento <organization>.

Anexo 6: Ejemplo de un documento "imsmanifest.xml".

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<manifest>
```

```
<metadata>
```

```
<schema>ADL Scorm</schema>
```

```
<schemaversion>CAM 1.3</schemaversion>
```

```
<lom>
```

```
<general>
```

```
<identifier>
```

```
<catalog>URI</catalog>
```

```
<entry>http://www.adlnet.org/CO_01</entry>
```

```
</identifier>
```

```
<title>
```

```
<string language="en">title for Learning
```

```
Objects</string>
```

```
</title>
```

```
<language>en</language>
```

```
<description>
```

```
<string language="en">textual description</string>
```

```
</description>
```

```
<keyword>
```

```
<string language="en">learning objects</string>
</keyword>
<coverage>
<string language="en">Circa,16th Century</string>
</coverage>
<structure>
<source>LOM v1.0</source>
<value>atomic</value>
</structure>
<aggregationLevel>
<source>LOM v1.0</source>
<value>2</value>
</aggregationLevel>
</general>
</lom>
</metadata>
<organizations>
<organization identifier="curso1">
<title>teleprocesos</title>
<item identifier="1">
<title>contenido 1</title>
<metadata>
</lom>
```

```
<general>
<author>Claudio Guzman Herrera</author>
<keyword>contenido 1</keyword>
<language>sp</language>
<description>
<string language="sp">Esta es una descripción
textual para el contenido 1</string>
</description>
</general>
</lom>
</metadata>
<item identifier="1">
<title>módulo 1</title>
<item identifier="1" identifierref="rec1">
<title>lectura 1</title>
</item>
<item identifier="2" identifierref="rec2">
<title>lectura 2</title>
</item>
<item identifier="3" identifierref="rec3">
<title>lectura 3</title>
</item>
</item>
```

```
<item identifier="2" identifierref="rec1">
<title>módulo 2</title>
</item>
<item identifier="3" identifierref="rec2">
<title>módulo 3</title>
</item>
</item>
<metadata/>
</organization>
<organization identifier="curso2">
<title>redes</title>
<item identifier="2">
<title>contenido 2</title>
<metadata>
<general>
<author>Claudio Guzman Herrera</author>
<keyword>contenido 2</keyword>
<language>sp</language>
<description>
<string language="sp">Esta es una descripción
textual para el contenido 2</string>
</description>
</general>
```

```
</metadata>
<item identifier="1" identifierref="rec100">
<title>módulo1</title>
</item>
</item>
<metadata/>
</organization>
<organization identifier="curso3">
<title>hardware de computadores</title>
<item identifier="3">
<title>contenido 3</title>
<metadata>
<general>
<author>Oriol Herrera</author>
<keyword>contenido 3</keyword>
<language>sp</language>
<description>
<string language="sp">Esta es una descripción
textual para el contenido 3</string>
</description>
</general>
</metadata>
</item>
```



```
<metadata/>
</organization>
</organizations>
<resources>
<resource identifier="rec1"
ahref="http://localhost:8080/tesis/LO/audio/diopush.mp3">
<file ahref="diopush.mp3"/>
</resource>
<resource identifier="rec2"
ahref="http://localhost:8080/tesis/LO/images/images.htm">
<file ahref="images.htm"/>
<file ahref="robotech.jpg"/>
<file ahref="untitled.bmp"/>
<file ahref="tigres.jpg"/>
<file ahref="vinniepaul.jpg"/>
</resource>
<resource identifier="rec3">
</resource>
<resource identifier="rec100">
</resource>
</resources>
<sub_manifest/>
</manifest>
```

GLOSARIO DE TERMINOS

1. Estándar: Los estándares son acuerdos internacionales documentados o normas establecidas por consenso mundial. Contienen las especificaciones técnicas y de calidad que deben reunir todos los productos y servicios para cumplir internacionalmente en condiciones de igualdad. Es decir, sin impedimento de las barreras técnicas que pudieran obedecer a diferentes formatos según las especificaciones de cada país.(ALBERTO 2004).
2. E-Learning: El e-Learning se define como la utilización de las nuevas tecnologías multimediales y de Internet, para mejorar la calidad del aprendizaje facilitando el acceso a recursos y servicios, así como los intercambios y la colaboración a distancia.(ALBERTO 2004).
3. Courseware o Contenidos: Los contenidos para e-Learning pueden estar en diversos formatos, en función de su adecuación a la materia tratada. El más habitual es el WBT (Web Based Training), cursos online con elementos multimedia e interactivos que permiten que el usuario avance por el contenido evaluando lo que aprende. Sin embargo, en otros casos puede tratarse de una sesión de aula virtual, basada en videoconferencia y apoyada con una presentación en forma de diapositivas tipo PowerPoint, o bien en explicaciones en una pizarra virtual. En este tipo de sesiones los usuarios interactúan con el docente, dado que son actividades sincrónicas en tiempo real. Lo habitual es que se complementen con materiales online tipo WBT o documentación accesoria que puede ser descargada e impresa.(MAURER 2002-2004)
4. AICC: (Aviation Industry CBT Committee). Comité de formación por computador de la Industria de la Aviación. Asociación Internacional de profesionales relacionados con la formación basada en tecnologías que desarrolla líneas de acción de formación para la industria de la aviación.
5. ADL: (Advanced Distributed Learning Network). Iniciativa del Departamento de defensa estadounidense para conseguir interoperabilidad entre computadores y software de aprendizaje basado en Internet, a través del desarrollo de un marco técnico común que almacena el contenido en forma de objetos de aprendizaje reutilizables.
6. IEEE: (Institute of Electrical and Electronics Engineers). Instituto de Ingenieros Eléctricos y Electrónicos (USA).

7. IMS: (Instructional Management System). Sistema de gestión Instruccional, Consorcio de aprendizaje global. Coalición de organizaciones gubernamentales dedicadas a definir y distribuir especificaciones de interoperabilidad de arquitectura abierta para productos de elearning.
8. LMS: (Learning Management System). Software que automatiza la administración de acciones de formación. Un LMS registra usuarios, organiza los diferentes cursos en un catálogo, almacena datos sobre los usuarios, también provee informes para la gestión. Un LMS es diseñado generalmente para ser utilizado por diferentes editores y proveedores. Generalmente no incluye posibilidades de autoría (creación de cursos propios), en su lugar, se centra en gestionar cursos creados por gran variedad de fuentes diferentes. Generalmente también se le conoce como plataforma.
9. LO: (Learning Object). Objetos de aprendizaje: Unidad reusable de información independiente de los medios. Bloque modular de contenido para elearning.
10. SCORM: (Shareable Courseware Object Reference Model). Resultado de la iniciativa de Aprendizaje avanzado distribuido (ADL) del Departamento de Defensa Estadounidense. Los elementos de la plataforma de SCORM pueden ser combinados fácilmente con otros.