Universidad de las Ciencias Informáticas
Facultad 4

Visor de contenidos para la propuesta de reproductor de archivos SCORM UPlayer_SCORM2004.

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

Autor
Darien Diego Hevia

Tutores
Ing. Arlan Galvez Alonso
Ing. Pablo Molina Toledo

La Habana, junio 2013
"Año 54 de la Revolución"

Declaración de Autoría

Por este medio declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI) para que haga el uso que estime pertinente con este trabajo. Para que así conste firmo la presente a los _____ días del mes de _____ del año 2013.

_____

Firma del Autor
Darien Diego Hevia

_____

Firma del Tutor
Ing. Arlan Galvez Alonso

_____

Firma del Tutor
Ing. Pablo Molina Toledo

_____

Firma del co-tutor
Ing. Yusdel Meriño Almaguer

Pensamiento

El camino siempre será difícil y requerirá el esfuerzo inteligente de todos. […] Ser tan prudentes en el éxito como firmes en la adversidad es un principio que no puede olvidarse. […][1]

[1] **Fidel Castro Ruz. La Habana, 18 de febrero de 2008.**

Agradecimientos

A mi familia por todo el apoyo que me brindaron a lo largo de mi vida.

A mis padres, sé que no tengo palabras para agradecerles todo los que han hecho por mí y porque hoy yo esté aquí parado, pero quiero empezar agradeciéndoles por toda la dedicación, el sacrificio y esmero, por confiar tanto en mí y por sus sabios consejos a lo largo de mi carrera, ahora me toca a mí hacer por ustedes. Papi gracias por guiarme siempre por el camino correcto, por estar siempre a mi lado y ayudarme con todos mis problemas, espero que estés orgulloso de mí. Esta es la parte más difícil de mi tesis, a ti mami gracias por la vida, por lo que soy hoy, todo esto te lo debo a ti y estoy muy contento de que estés aquí, acompañándome en este momento tan especial, porque eres la anfitriona de esta sala y ya vez, aquí tienes a tu hijo hecho un ingeniero.

A mis dos bebés que son lo que más amo en este mundo, por darme las fuerzas para seguir adelante y no defraudarlas, mis princesas las amo con todo mi corazón.

A mi hermana que adoro con el alma, por ser mi guía y mi protectora. Tata eres la mejor de las hermanas y quiero que sepas que te amo con toda mi vida y no quiero separarme nunca de ti, gracias por existir.

A mi abuelita que por razones de la vida hoy no se encuentra, sé que te sientes orgullosa de mí, quiero que sepas que te extraño mucho, te quiero mandar mil besos.

A mis tíos que adoro con el alma, a mi tío Iran que ha sido un padre para mí, gracias por todos los momentos que compartimos juntos, a mi tía, gracias por ser como eres por darme todo el amor que me distes y por poder contar siempre contigo para lo que quiera, te amo y te extraño mucho.

A mi primo Irancito por ser mi hermano, donde quiera que estés quiero que sepas que te amo con todo mi corazón y ojalá nos veamos pronto, un besote grande.

A mi hermanita Amandita, por ser mi asistente preferida, quiero decirte que lo eres todo para mi familia y te amamos con el alma.

A mi esposa, por darme el mayor tesoro de mi vida, por ayudarme para que la tesis saliera adelante, por la paciencia con que me escucha, por estar a mi lado en los momentos más duros de la carrera y darme ánimo para seguir adelante, te amo mi reina.

A mis primos Segrees, Jorge, primero por darme las primas más lindas del mundo, por dedicarle tiempo a todas mis dudas y por estar siempre apoyándome.

A la profe Gloria, por todos sus consejos, por ser la amiga incondicional que es de mi mamá y por estar siempre cuando más te necesitamos.

A la madrina de mi bebé, Maira gracias por todo el cariño que le tienes a Jade, y por tu apoyo incondicional.

A Gilbe, eres lo máximo no tengo palabras para agradecerte todo lo que haces por mi familia, gracias por ser el mejor padrino del mundo, sé que Jade va a poder contar contigo siempre, te quiero.

A los vecinos que siempre han confiado en mí y me han guiado hacia el bien, a Lázaro, Leticia, Marta, a mi abuelita Gise, Ramón, muchas gracias a todos.

A una personita muy especial, a mi amiga Ivis, gracias por dedicarme esas madrugadas, por tener paciencia, hoy estoy aquí gracias a ti y nunca te olvidaré.

A mis amigas Arianna y Jessica, por ser tan comprensibles y tan buenas, las adoro y quiero que sepan que estoy muy feliz de haberlas conocido y que van a estar en mi corazón el resto de mi vida, ah Jessi cuida mucho a esa sobri y háblale de mí, las amo.

Al vicedecano Antonio, que aunque es muy recto me protegió durante toda mi carrera y me enseñó a caminar por el camino correcto, un abrazo profe.

A mi decano Basulto por darme su apoyo, por confiar en mí y por arriesgarse en las decisiones que tomó respecto a mí bebe y mi esposa, gracias profe.

A todos los profesores que me hicieron mejor profesional cada día, al profe Fidel, Isie, Deiler, Yunesty, Vázquez y en especial a Yusdel que más que un profesor fue mi amigo y mi compañero, gracias a todos.

A mis amigos de la universidad, que me han hecho vivir innumerables momentos de alegría, risas, porque cada uno de ellos es parte de un grupo único en el que no existen misiones imposibles, gracias por hacerme la vida fácil, Ramel, Eduardo, Rafa, Danir, Luisi, Nesty, Bolo, Wichi, Pepe, Adriel, Osmel, Erick, Sol, Heriberto, Darlenis, Pily, Nani, Liliana, Adia, Anay, a todos los quiero mucho y los voy a extrañar.

Dedicatoria

A mi madre por ser la guía de mi vida y por ser la mejor del mundo, a quien amo y admiro con todo mi corazón…

A mi bebe hermosa, Jadecita, por ser mi alma, el aire que respiro, y el motor que me impulsó a culminar mi carrera, te amo mi ángel.

Resumen

La implantación de estándares es actualmente una necesidad en las plataformas de gestión del aprendizaje, fundamentalmente para el aprovechamiento de los recursos, independientemente del sistema a emplear, permitiendo la interoperabilidad. Entre los estándares que hoy en día existen en el área de la tecnología educativa, se destaca la especificación SCORM del inglés Sharable Content Object Reference Model, que no es más que un conjunto de estándares y especificaciones que permite crear objetos pedagógicos estructurados. En la Universidad de las Ciencias Informáticas desde hace algunos años se desarrollan Objetos de Aprendizaje empaquetados bajo este estándar, los cuales necesitan de un visor que permita su descompresión e interpretación, sin embargo los reproductores que existen en la actualidad no reconocen los Objetos de Aprendizaje empaquetados con el estándar SCORM 2004. Por tanto, el objetivo principal de la presente investigación es desarrollar un visor de contenidos para que los usuarios finales cuenten con una herramienta que les permita visualizar los contenidos de cualquier Objeto de Aprendizaje empaquetados bajo el estándar SCORM2004. Al finalizar el presente trabajo se obtuvo el reproductor de archivos UPlayer_SCORM2004, que permitirá reconocer e interpretar los paquetes de Objetos de Aprendizaje bajo el estándar SCORM2004 producidos en la Universidad de las Ciencias Informáticas.

Palabras claves: Objeto de Aprendizaje, visor, aplicación de escritorio, estándar, SCORM.

Abstract

The implementation of standards is now a necessity in the learning management platforms, primarily for the use of resources, regardless of the system to be used, enabling interoperability. Among the standards that exist today in the area of educational technology, highlights the SCORM specification of English Sharable Content Object Reference Model, which is nothing more than a set of standards and specifications for creating structured teaching objects. At the University of Computer Sciences in recent years developed Learning Objects packaged under this standard, which need a viewer that allows decompressing and interpretation, however players that exist today do not recognize the packaged Learning Objects with SCORM 2004. Therefore, the main objective of this research is to develop a content viewer for that end users have a tool that allows them to view the contents of any Learning Object SCORM2004 packaged under the standard. At the end of the present work was obtained UPlayer_SCORM2004 file player, which will recognize and interpret Learning Objects packages under standard SCORM2004 produced at the University of Information Sciences.

Keywords: Learning Object, viewer, desktop application, standard SCORM.

Índice

Índice de Figuras

Índice de Tablas

Introducción

Las Tecnologías de la Información y las Comunicaciones (TICs) contribuyen sustancialmente a mejorar la calidad de la educación y la formación de una sociedad basada en el conocimiento, si se utilizan adecuadamente. En el ambiente educativo, ha surgido un nuevo concepto que está causando cambios sustanciales en la forma de enseñar, al cual se le ha llamado Objeto de Aprendizaje (OA).

En el terreno de la enseñanza, la idea es que los profesores puedan contar con componentes educativos reutilizables, de tal manera que los Objetos de Aprendizaje serían pequeños componentes instruccionales que pudieran ser reutilizados en diferentes contextos de aprendizaje.

En la actualidad, diversas organizaciones desarrollan estos materiales educativos en formato electrónico, con el propósito de mejorar la usabilidad y disponibilidad de estos. Con el fin de lograr además, que sean usables por distintas plataformas de aprendizaje, es necesario que sigan un estándar.

Los estándares son normas internacionales que contienen especificaciones de los productos y servicios que se desarrollen, para competir internacionalmente en términos de calidad. Estos proporcionan modelos comunes de información, que permiten a los sistemas y a los cursos compartir datos. Con ello aumenta la oferta de cursos disponibles y se reducen los costos de adquisición o desarrollo de programas y cursos. (1)

Uno de los estándares con mayor uso para empaquetar Objetos de Aprendizaje es el SCORM (*Sharable Content Object Reference Model*), que es un conjunto de especificaciones para desarrollo, empaquetamiento y distribución de material educativo, en cualquier momento y en cualquier lugar. El estándar SCORM asegura que este material sea: reutilizable, accesible, interoperable y durable. (1)

El paquete SCORM es un fichero comprimido en formato *Zip* que contiene el Objeto de Aprendizaje, el manifiesto y las hojas de estilo que permiten interpretarlo.

En la Universidad de las Ciencias Informáticas (UCI) desde el año 2005 se desarrollan Objetos de Aprendizaje empaquetados bajo este estándar, ejemplo de esto son los desarrollados en el Proyecto ROX, luego llamado en el año 2007 Proyecto RHODA. Para visualizarlos, es necesario contar con un visor que permita descomprimir el paquete,

interpretarlo y mostrar al usuario los contenidos cuando éste no se encuentre conectado a la red.

En el mercado, existen reproductores de archivos SCORM, pero no reconocen los Objetos de Aprendizaje empaquetados con el estándar SCORM 2004, pues esta es la versión más reciente y los visores no pueden interpretar el XML del mismo, sólo los empaquetados bajo el estándar SCORM 1.2.

Por lo antes planteado se presenta la siguiente **problemática a resolver**: ¿Cómo interpretar paquetes que cumplan el estándar SCORM 2004 y visualizar los contenidos que se encuentren dentro de éste?

Objetivo General: Desarrollar el intérprete y visor de contenidos para interpretar Objetos de Aprendizaje empaquetados bajo el estándar SCORM2004.

El **objeto de estudio** son los reproductores de paquetes que cumplen con el estándar SCORM.

E**l campo de acción** son los reproductores de paquetes que cumplen con el estándar SCORM 2004.

Objetivos Específicos:

- Realizar un análisis bibliográfico asociado al estudio del estado del arte de las tendencias relacionadas con el objeto de estudio de la investigación.
- Investigar las tecnologías empleadas para el desarrollo de un reproductor de archivos SCORM 2004.
- Desarrollar los módulos intérprete y visor de contenidos para el UPlayer_SCORM2004.
- Validar los módulos desarrollados para el reproductor de contenidos.

Resultados Esperados:

Se esperan obtener los módulos intérprete y el visor de contenidos para el reproductor de archivos UPlayer_SCORM2004, permitiendo que este sea capaz de reconocer e interpretar los paquetes en el formato SCORM 2004, al igual que los Objetos de Aprendizaje desarrollados en la Universidad de las Ciencias Informáticas (UCI) bajo las especificaciones de dicho estándar. Además permitirá la secuenciación entre los recursos de manera lineal. Este será desarrollado bajo estándares libres de acuerdo a las políticas vigentes en la

Universidad y eliminará la necesidad de conexión a un servidor mediante la red para visualizar dichos paquetes.

Tareas a desarrollar:

- Análisis del estado del arte de los reproductores de archivos SCORM en la actualidad.
- Estudio de los diferentes estándares y especificaciones e-learning existentes.
- Análisis del estándar SCORM 2004.
- Investigación de las tecnologías y la metodología a utilizar en el proceso de desarrollo del software.
- Recopilación de requerimientos del sistema.
- Diseño de la interfaz no funcional de la aplicación.
- Desarrollo de los módulos intérprete y el visor de contenidos correspondientes para el UPlayer_SCORM2004.
- Integración de los módulos al reproductor UPlayer_SCORM2004.
- Aplicación de pruebas de software a la aplicación.

Los métodos de investigación que soportan la siguiente investigación es una combinación dialéctica entre los **métodos teóricos y empíricos**. Los empleados son los que se muestran a continuación:

Métodos Teóricos

- Histórico – Lógico: mediante el empleo de este método se realiza un estudio de las Plataformas Educativas que utilizan paquetes SCORM.
- Analítico – Sintético: sirve de base para la revisión de la bibliografía y materiales relacionados con los reproductores de paquetes que cumplen con el estándar SCORM.

Métodos Empíricos

Observación: Permite conocer la esencia del problema planteado, pues analizando desde varios puntos de vista la propuesta de solución y otras soluciones existentes, se permite identificar qué está hecho y qué falta por hacer.

Capítulo 1: Fundamentación teórica

Introducción

En el presente capítulo se define el estado del arte y los conceptos vinculados al objeto de estudio, con el fin de lograr una mejor comprensión del problema de la investigación. Se describe además, el lenguaje de programación y el análisis de sistemas similares para identificar funcionalidades o características a fines con el producto a desarrollar, así como las tendencias y tecnologías actuales sobre las que se basa.

1.1 Conceptos y definiciones asociados al problema de investigación

Para el desarrollo de la presente investigación, es necesario definir los conceptos básicos que permiten una mejor comprensión del problema de investigación.

1.1.1 E-learning

El e-learning es una forma de educación a distancia, dependiente de las tecnologías de virtualización, que permite la interacción entre un grupo de personas (dígase estudiantes, docentes, etc.) mediante una interconexión a través de ordenadores que no necesariamente deben tener acceso a Internet; figura como uno de los soportes principales de los procesos de enseñanza-aprendizaje actuales.
La propuesta de evolución de las TICs para el campo de la educación fue el surgimiento del e-learning que trajo consigo diversos beneficios (2):

- Reducción de costos: permite reducir y hasta eliminar gastos de traslado, alojamiento, material didáctico, etc.
- Rapidez y agilidad: Las comunicaciones a través de sistemas en la red confiere rapidez y agilidad a las comunicaciones.
- Acceso en tiempo real: los usuarios pueden acceder al contenido desde cualquier conexión a Internet, cuando les surge la necesidad.
- Flexibilidad de la agenda: no se requiere que un grupo de personas coincidan en tiempo y espacio, ni hay necesidad de la presencia física de un profesor.

Además permite (4):

- Producir cursos para la educación en línea ya sea a nivel formativo o de entrenamiento, módulos o unidades didácticas, objetos de aprendizaje y en general recursos educativos que vayan mucho más allá del contenido.
- Reduce el tiempo de aprendizaje, es aproximadamente un 40 % de ahorro al tiempo habitual utilizado frente a clases presenciales.
- Incrementa la asimilación y retención de conceptos del estudiante.

El creciente uso y desarrollo del aprendizaje electrónico y la necesidad de entrenamientos, clases y cursos mediante el uso de la red, ha dado lugar a la aparición de un nuevo concepto nombrado Objetos de Aprendizaje, con el fin de contar con materiales educativos que puedan ser reutilizados.

1.1.2 Objeto de Aprendizaje

El elemento central en la nueva forma de desarrollar los cursos en entornos educativos es el Objeto de Aprendizaje (OA), estos no se utilizan solamente en el aprendizaje electrónico, pueden ser usados de complementos a los profesores en clases semipresenciales o presenciales. La definición más citada en la literatura, es la del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE, Asociación técnico-profesional mundial dedicada fundamentalmente a la estandarización), propuesta en uno de los pocos estándares relacionados con e-learning que han sido aprobados:

"*Un objeto es cualquier entidad digital o no digital que puede ser usada, re-usada o referenciada para el aprendizaje soportado en tecnología*". (5)

"Un recurso educativo digital relativamente pequeño y auto-contenido, con una marcada intención formativa, compuesto por uno o varios objetos de información, con un objetivo, descrito con metadatos y con un comportamiento secuenciado que asegure el correcto enlace entre los elementos de su estructura didáctica; concebido para diferentes poblaciones según sus contextos socioculturales para lograr su reutilización e interoperabilidad en varios entornos b-learning, que contribuya a la transmisión de conocimiento y la formación de valores tanto en los profesores, diseñadores, como en los usuarios finales". (6)

El editor David Wiley también matiza que "…*se usa para designar material educativo diseñado y creado en pequeñas unidades con el propósito de maximizar el número de situaciones educativas en las que se puede utilizar dicho recurso"*. (7)

Esta idea está directamente recogida en la definición proporcionada por el editor Polsani que lo define como:

"Unidad didáctica de contenido, auto – contenida e independiente, predispuesta para su reutilización en múltiples contextos instruccionales." (8)

Después de la revisión y el análisis de los diferentes conceptos de Objetos de Aprendizajes, el autor de la presente investigación considera que el concepto más acabado es el de la Dra. Roxana Cañizares (2012).

En la concepción de un OA debe pensarse que sean recursos con atributos específicos para su interacción en un entorno e-learning, fácil de localizar, utilizar, almacenar y compartir. Para ello, estos recursos deben ser: (9)

Accesibles: Pueden ser indexados para una localización y recuperación más eficiente, utilizando esquemas estándares de metadatos.

Interoperables: Pueden operar entre diferentes plataformas de hardware y software.

Portables: Pueden moverse y albergarse en diferentes plataformas de manera transparente, sin cambio alguno en estructura o contenido.

Durables: Deben permanecer intactos a las actualizaciones de software y hardware.

1.2 Especificaciones y Estándares más utilizados en e-learning

Sobre las especificaciones de los sistemas e-learning más empleados en el empaquetamiento de cursos y recursos para el desarrollo de los LMS, se encuentran: IMS Content Packaging, IMS Question & Test Interoperability Specification, IMS Learning Design, IMS Learner Information Package Specification, IEEE Learning Object Metadata / IMS Learning Resources Metadata Specification, LOM y SCORM**.** De todas ellas la investigación se inclina para Learning Object Metadata (en adelante LOM) y SCORM, por constituir ella la base del desarrollo del presente trabajo de diploma.

1.2.1 IEEE Learning Object Metadata / IMS Learning Resource Metadata Specification

Actualmente IEEE Learning Object Metadata (Metadatos para Objetos de Aprendizaje), es el estándar de e-learning formalmente aprobado que goza de mayor aceptación, y que ha sido adoptado en la especificación de IMS Learning Resources Metadata. De hecho LOM se basa en los esfuerzos previos hechos para la descripción de recursos educativos en los proyectos ARIADNE, IMS y DublinCore. (10)

Los metadatos LOM, son información añadida a los materiales digitales que facilitan su clasificación y posterior recuperación. La especificación de metadatos adecuados para los materiales educativos es indispensable a fin de añadir valor a los mismos, en el sentido de facilitar su reutilización. Los materiales enriquecidos convenientemente con metadatos podrán almacenarse en bibliotecas digitales de contenidos educativos (por ejemplo, repositorios de Objetos de Aprendizaje). Estas bibliotecas soportarán, entonces, consultas

significativas que permitirán la recuperación de aquellos materiales almacenados que cubran una determinada necesidad pedagógica. Estos metadatos proporcionan descripciones, propiedades e información sobre los Objetos de Aprendizaje que permiten caracterizarlos, de forma que se simplifica su uso y gestión. De forma coloquial, lo que se busca mediante esta información complementaria es poder saber cuál es el contenido y el propósito de un OA sin tener que acceder a dicho contenido. Por tanto, los metadatos aportan información orientada a hacer más eficiente la búsqueda y utilización de los recursos. (10)

1.2.2 Estándares y especificaciones de contenido

Nombre: **IEEE Standard for Learning Object Metadata (Normas para metadatos de objetos de aprendizaje) (LOM)**

Organización: IEEE/LTSC
LOM-ES. Su principal objetivo es la creación de descripciones estructurales que permiten conocer el contenido del OA sin necesidad de abrirlo, lo que facilita la búsqueda y catalogación de OA de forma rápida y eficiente.
Su modelo de datos especifica alrededor de 80 campos que deberían ser descritos en un OA, agrupados en 9 categorías (16):

- General (Para aspectos generales como el título, palabras claves, etc.).
- Ciclo de Vida (Para tabular el ciclo de vida por el que pasa el OA).
- Meta-Metadatos (Aspectos relacionados con los metadatos versión, etc.).
- Aspectos Técnicos (Para información técnica como el formato, requisitos de software, etc.).
- Uso Educativo (Agrupa características de uso educativo como tipo de recurso, densidad semántica, tipo de usuario final, etc.).
- Derechos de Copia (Para aspectos de derecho de copia).
- Relación (Se utiliza para describir la relación entre el recurso en cuestión con otros recursos).
- Anotación (Registra las diferentes anotaciones que se le hacen al recurso o al OA que describe).
- Clasificación (Se utiliza para dar clasificaciones a los recursos que se describen).

Nombre: **Modelo Referenciado de Objetos de Contenido Compartible (SCORM).**
Organización: IMS, AICC, IEEE, ADL.

SCORM, es la unión de un conjunto de normas dirigidas al empaquetamiento, distribución, reutilización y ejecución de contenidos educativos, enfocado a lograr una mayor interoperabilidad entre las diferentes herramientas de un entorno e-learning. (11)

Esta norma cuenta con varias versiones, siendo la 1.2 y 2004 las más difundidas, es preferible elegir estas versiones ya que son más fáciles de implementar y probar los contenidos y la mayoría de las plataformas de e-learning la soportan. SCORM está dividido en 3 libros técnicos, cada tópico de los que a continuación se especifican constituye un libro de SCORM (17).

- Modelo de Agregación de Contenidos (CAM): Tiene como objetivo principal ofrecer un medio común que permita contener recursos educativos desde diversas fuentes, compartibles y reusables. Define un contenido educativo puede ser identificado, descrito y agregado dentro de un curso o parte de él y como puede ser compartido por diversos LMS o repositorios (15).
- Entorno de Tiempo de Ejecución (RTE), trata sobre 3 temas fundamentales: la administración del entorno en tiempo de ejecución, una API que permitirá programar las relaciones entre el contenido y el LMS, y un protocolo específico para el intercambio de contenidos web. También describe como el libro SCORM SN afecta al libro SCORM RTE (15).
- Secuencia y Navegación (SN), está basado en la especificación Simple Sequencing de IMS, y desarrolla un modelo para presentar los contenidos al alumno en función de las necesidades o capacidades del mismo. Describe las reglas de secuencia para que el desarrollador de contenidos logre trazar la ruta de lanzamiento de estos.

SCORM usa IMS-CP como modelo de agregación de contenido, como esquema de metadatos LOM u otro y la secuenciación de los contenidos está basada en los trabajos de la AICC en esta área.

1.3 Estándar de secuenciación simple

1.3.1 IMS-SS

Es un estándar (también desarrollado por IMS) de secuenciación. Básicamente, IMS SS trata de asociar a cada uno de los elementos que componen un recurso educativo (cada uno de los elementos del "índice") una información, indicando si el alumno puede acceder o no a él.

Puede parecer sencillo, pero con una secuenciación así podemos lograr objetivos como:

- Evitar que los alumnos comiencen a leer un recurso por la página 3.

- Evitar que los alumnos puedan volver atrás (por ejemplo, que vuelvan a leer los contenidos teóricos una vez han visto un examen).
- Establecer secuencias condicionales mediante reglas. Por ejemplo, "si el alumno no realiza correctamente la actividad 2, debe volver a ver un ejemplo, y luego intentar de nuevo la actividad, hasta que la realice de modo satisfactorio".

1.4 El estándar SCORM. Características

SCORM es un conjunto de normas técnicas en el ámbito de e-learning que hace posible que cualquier contenido se reproduzca en cualquier plataforma de gestión de formación. (20)

Un paquete SCORM es un archivo comprimido en formato .Zip, que contiene recursos como imágenes, videos, textos, animaciones, sonidos, objetos de evaluación y páginas web. En su raíz contiene un archivo llamado imsmanifiesto, que se describe a continuación. (20)

El imsmanifiesto contiene la información necesaria para describir la estructura del paquete, dividido en cuatro secciones:

- ✓ **Metadatos:** describen los objetos educativos, la versión del estándar que se utiliza, los recursos y los objetivos didácticos.

- ✓ **Organizaciones:** forma en que están organizados los contenidos y su desglose en actividades. Las actividades se encuentran conectadas con los recursos a través de su identificador. Dentro de las organizaciones se encuentran las instrucciones de secuenciación y navegación.

- ✓ **Recursos:** describen los recursos externos y locales que utiliza el paquete. Los recursos locales se encontrarán en el fichero comprimido. Para que un recurso pueda comunicarse con una plataforma debe ser un Objeto de Contenido Intercambiable, en inglés Sharable Content Object (en lo adelante SCO).

- ✓ **SubManifiestos:** está compuesto por otros manifiestos anidados.

En el modelo SCORM un SCO, es la unidad mínima de contenido, es un Objeto de Aprendizaje que contiene código JavaScript, que le permite comunicarse con un LMS al ser ejecutado. (21)

1.4.1 Evolución del estándar

SCORM 1.0 (enero de 2000): fue el inicio del estándar y contribuyó a formar lo que es hoy, se introdujo el concepto de SCO y el modelo de Interfaz de Programación de Aplicaciones,

en inglés Application Program Interface, en lo adelante API. SCORM 1.0 no es relevante hoy en día. (3)

SCORM 1.1 (enero de 2001): La primera versión, utilizaba el formato de la estructura de un curso en archivo XML basado en las especificaciones del Comité de Capacitación por Computadora de la Industria de la Aviación (AICC, por sus siglas en inglés) para describir la estructura del contenido, pero carecía de un manifiesto robusto de empaquetado y de un soporte para metadatos. Esta versión fue rápidamente sustituida por la versión SCORM 1.2. (3)

SCORM 1.2 (Octubre de 2001): La primera versión más utilizada. Actualmente sigue siendo muy empleada y es soportada por la mayoría de los LMS que existen. El problema de esta versión fue la necesidad de secuenciar de forma no lineal un recurso, proceso que estaba muy limitado e impedía que el estudiante (dependiendo de su desempeño) tuviera varias rutas de aprendizaje en el curso. (3)

SCORM 2004: la versión actual. Basada en los nuevos estándares de la IEEE para la Interfaz de Programación de la Aplicación API (por sus siglas en inglés) y para el funcionamiento entre el objeto de contenido y el ambiente de desempeño. Con muchas ambigüedades que las versiones anteriores no habían resuelto, incluye la posibilidad de especificar una secuencia de actividades flexible a partir de objetos de contenido. Asimismo permite compartir y usar información de acuerdo con el estatus de múltiples objetivos de aprendizaje o competencias alcanzados por los estudiantes en un mismo LMS. Con esto se pueden trazar rutas de aprendizaje personalizado. (3)

1.5 Visores de archivos

Dentro de la informática un visor es una aplicación utilizada para visualizar algún archivo o recurso, como por ejemplo:

- ✓ **Adobe Acrobat Reader:** permite leer, navegar e imprimir los ficheros de documentos en formato PDF. (22)
- ✓ **VLC media player v1.2.0:** reproduce videos y sonidos, es compatible con extensiones como MPEG-1, MPEG-2, MPEG-4, DivX, XviD, H.264, MP3 y OGG. (23)

En los visores anteriores se puede apreciar que las acciones de visualizar el recurso y navegar a través de él son aspectos comunes. También que los visores pueden brindar otras opciones, que estarán en dependencia de las características del recurso.
El estudio de los conceptos expuestos anteriormente, servirá para lograr un mejor entendimiento de los elementos que se analizan a continuación.

1.5.1 Visores de archivos SCORM

Los reproductores o visores de paquetes SCORM permiten navegar por el contenido de un Objeto de Aprendizaje elaborado en una herramienta de autor. En la actualidad se han confeccionado varios reproductores de paquetes SCORM y estos pueden ser aplicaciones de escritorio, o estar integrado a repositorios de Objetos de Aprendizaje o a plataformas. A continuación se explican algunas de las características de algunos visores existentes.

1.5.1.1 Visores de archivos SCORM usados por plataformas educativas

El uso de visores de paquetes SCORM se ha extendido entre varias plataformas. Para la presente investigación se analizarán cuatro de ellas. Se comenzará por Sakai, la cual se actualiza cada tres o cuatro meses y se le añaden herramientas constantemente para su integración, (24) como los visores SCORM. Luego a Moodle por ser “la plataforma educativa libre más conocida y extendida, con 67 000 sitios registrados, que ofrecen 5,5 millones de cursos, en los que participan 54 millones de usuarios”, (25) a continuación Blackboard, por tener uno de sus puntos focales más importantes, la compatibilidad con los estándares globales, como SCORM. (26) La última analizada fue Dokeos, esta permite la importación y exportación de archivos SCORM. (26)

Sakai

Sakai es una plataforma en línea desarrollada por una comunidad de educadores para facilitar la enseñanza y el aprendizaje colaborativo en instituciones educativas. El proyecto originado en la Universidad de Michigan, fue anunciado oficialmente en EDUCAUSE, en noviembre del 2003. La plataforma es distribuida gratuitamente como un programa de código abierto bajo la licencia de comunidad educativa. (27)
La plataforma es utilizada para el manejo de cursos, pero también es un ambiente de aprendizaje y colaboración que añade herramientas para el desarrollo de contenidos. Permite colocar el prontuario (anotación o resumen) y presentaciones dentro de la plataforma, y ofrece plantillas para diseñar lecciones de manera secuencial, para crear asignaciones, exámenes y concursos en línea. (27)

La plataforma cuenta con el visor SCORM que se describe a continuación:

Icodeon

Es un visor robusto, la configuración es flexible y de sencilla adaptación. La integración añade un nuevo tipo de contenido a la herramienta de “Recursos” que permite subir un paquete SCORM y lo registra en Icodeon. Una funcionalidad importante es que Icodeon registra eventos en las tablas de Sakai, aunque todavía no existen informes ni otras características que aprovechen esta información. (28)

Tiene como ventajas que es basado en el estándar de la ADL, reproduce todas las versiones de SCORM, en castellano, es flexible, personalizable, tiene un completo entorno de ejecución diseñado para ser integrado fácilmente a sistemas de gestión de aprendizaje, soporta la secuencia y navegación de los contenidos que propone la última versión del estándar, también tiene excelente soporte técnico y es distribuible, pero hay que pagar por este producto (28).

Moodle

Moodle es un Sistema de Gestión de Cursos de Código Abierto, en inglés Open Source Course Management System (CMS), conocido también como Sistema de Gestión de Aprendizaje o como Entorno de Aprendizaje Virtual. Es una herramienta para producir cursos basados en Internet, páginas web y procedimientos que permitan fácilmente la comunicación a través de Internet y el trabajo colaborativo. (29)

Flex SCORM Player 2.0

Es el visor que muestra los SCO construidos a través del Flex SCORM Builder, otorgándole la apariencia de un libro virtual. Está compuesto por una serie de herramientas que hacen del estudio una experiencia amigable y cómoda, optimizando los recursos que el alumno tiene a su disposición. Es extremadamente sencillo, rápido y ágil de usar. El libro virtual se basa en un documento en formato PDF para su visualización. Cumple con el estándar SCORM 2004. (30)

Blackboard

Blackboard se fundó con la visión de transformar Internet en un potente entorno educativo. Es el principal proveedor de soluciones de educación electrónica. Presta sus servicios a varios niveles de enseñanza como primaria, secundaria y educación superior, así como a agencias del gobierno o de las empresas. Blackboard tiene su sede central en Washington D.C. y cuenta con oficinas y personal en Norteamérica, Europa y Asia. (31)

Para darle soporte al estándar SCORM, cuenta con el visor siguiente:

Content Player Building Block

El Content Player Building Block de Blackboard fue desarrollado internamente por medio de los equipos de ingenieros y seguridad de Blackboard, y cuenta con certificado ADL. Permite a un instructor agregar contenidos que se ajusten a SCORM 1.2 y 2004, IMS, o las normas NLN a un curso. Los instructores pueden ver los IMS, SCORM y NLN dependiendo de lo que el administrador haya puesto a disposición. (31)

La UCI tiene un **Repositorio de Objetos de Aprendizaje (ROA)** que almacena a los objetos creados en la herramienta de autor CRODA, en formato .Zip siguiendo el estándar SCORM. Para acceder al contenido de uno de estos objetos o cursos se debe seleccionar la opción que permite visualizarlo, mostrándose en una nueva pestaña del navegador. En la pestaña se mostrará el árbol de contenido y opciones que faciliten la navegación. Otra de las opciones que brinda el repositorio es la de exportar los Objetos de Aprendizaje a SCORM 1.2 y SCORM 2004.

1.6 Visores de escritorio

Reload Scorm Player

Verifica que el archivo imsmanifest.xml esté correcto, simula la comunicación con una plataforma educativa y almacena los resultados hasta que se reinicien las pruebas. Es una herramienta muy útil para probar contenidos SCORM, es libre, gratuita y multiplataforma. Permite importar un paquete SCORM 1.2, ver el árbol de contenidos descrito en el manifiesto y ejecutar los SCO importados. Como es una aplicación de escritorio es muy útil para visualizar paquetes SCORM creados sin la necesidad de subirlo a plataformas que lean SCORM. (32)

Vi-Sco

Vi-Sco es un visor de paquetes SCORM que muestra los Objetos de Aprendizaje que son descargados de Agrega en formato SCORM1.2 o que han sido creados en herramientas como eXeLearning. Este visor al ser una aplicación de escritorio, para emplearla solo hay que ejecutarla y cargar el archivo SCORM. Luego se genera un menú con toda la secuencia de contenidos que contiene el paquete SCORM, que se emplea para la navegación. (33)

1.6.1 Resultados del análisis

Luego del estudio realizado a los Visores de archivos SCORM, se toma como referencia el Reload Scorm Player, ya que éste reproductor cuenta con características factibles, afines posibles a utilizar en el desarrollo de la propuesta de solución.

1.7 Aplicaciones de escritorio

Una aplicación Desktop (también llamada de Escritorio) es aquella que está instalada en el ordenador del Usuario, que es ejecutada directamente por el sistema operativo, ya sea Microsoft Windows, Mac OS X, Linux o Solaris, y cuyo rendimiento depende de diversas configuraciones de hardware como memoria RAM, disco duro, memoria de video, etc. (32)

Ventajas: (32)

- Habitualmente su ejecución no requieren comunicación con el exterior, sino que se realiza de forma local. Esto repercute en mayor velocidad de procesamiento, y por tanto en mayores capacidades a la hora de programar herramientas más

complicadas o funcionales.

- Suelen ser más robustas y estables que las aplicaciones Web.
- Rendimiento: el tiempo de respuesta es muy rápido.
- Seguridad: pueden ser muy seguras (dependiendo del desarrollador).

A continuación se muestra una breve comparación entre las aplicaciones Web y las de Escritorio:

Para la ejecución de una aplicación web se necesita tener acceso a un servidor (ya sea local o remoto) y de un navegador, a diferencia de una de escritorio que no necesita estar conectada a ningún servidor, pero sí depende del sistema operativo para el cual fue diseñada.

La posibilidad de no tener que instalar ningún sistema operativo, solo conectarte desde tu navegador web y usar tu aplicación es más que suficiente para hacer voltear la cara a cualquier empresario, pero ellos son los primeros en extrañar la interface de las aplicaciones de escritorio. (34)

El problema de estar siempre conectado en una aplicación web es algo que influye mucho en el rendimiento de dicho software, ya que no depende ni del cliente ni del servidor sino de la compañía que brinda el servicio de internet. (34)

Por todo lo planteado anteriormente se decide desarrollar el visor como una aplicación de escritorio, atendiendo además a las facilidades que ofrecen las mismas.

1.8 Metodología de desarrollo

Un proceso de *software* detallado y completo según Iván Jacobson suele denominarse "**Metodología"**. Este término lo define como: Quién debe hacer Qué, Cuándo y Cómo debe hacerlo. Las metodologías se desarrollan con el objetivo de dar solución a los problemas existentes en la producción de *software*, que cada vez son más complejos. Estas engloban procedimientos, técnicas, documentación y herramientas que se utilizan en la creación de un producto de *software*. No existe una metodología de *software* universal (35), las características de cada proyecto (equipo de desarrollo, recursos técnicos y humanos, tiempo de desarrollo, tipo de sistema) exigen que el proceso sea configurable para lograr un éxito en el proceso de desarrollo.

1.8.1 RUP (Rational Unified Process)

"El Proceso Unificado de Rational (en lo adelante RUP, por sus siglas en inglés ***R**ational **U**nified **P**rocess*) se encarga del desarrollo de software que ofrece un conjunto de actividades para transformar los requerimientos de un usuario en un sistema informático. (35)

Es además, un conjunto de metodologías adaptables al contexto y necesidades de cada nacionalización, que en conjunto con el Lenguaje Unificado de Modelado (en lo adelante UML, por sus siglas en inglés ***U**nified **M**odeling **L**anguage*), constituye la metodología estándar más utilizada para el diseño, implementación y documentación de sistemas informáticos. (35)

Sus características fundamentales proponen un desarrollo iterativo e incremental, centrado en la arquitectura y dirigido por caso de uso. Es una de las metodologías estándar más eficaz para la implementación y documentación del sistema, pero al ser una metodología tradicional, plantea la creación de una cantidad de documentación excesiva, mucho más de la que se necesita lograr para cumplir el objetivo de la presente investigación.
Está compuesto por cuatro fases que representan un ciclo de desarrollo en la vida de un producto de software ellas son: Inicio, Elaboración, Construcción y Transición. (35)

1.8.2 SCRUM

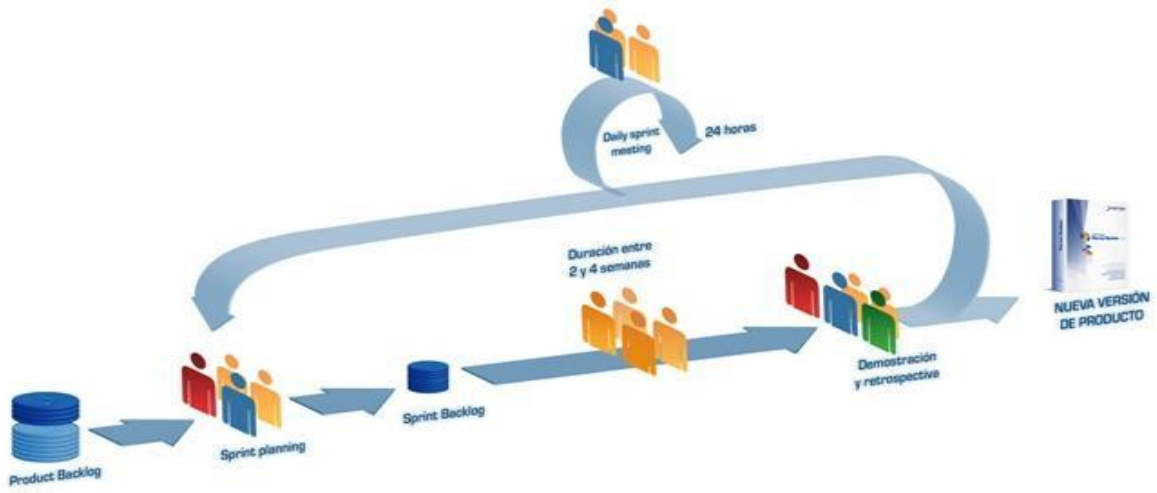


Figura 1. Proceso de la metodología SCRUM

SCRUM es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos (36).

La metodología SCRUM se divide en tres fases las cuales son: Planificación, Desarrollo y Entrega, en las cuales se efectúan varias reuniones, donde se elabora la lista de todos los cambios requeridos sobre un producto, se determina que tareas deben desempeñarse para cumplir el objetivo de cada sprint (iteración), entre otros artefactos que se genera en cada sprint.

En SCRUM se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, SCRUM está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales (36).

SCRUM permite a las organizaciones eliminar los impedimentos en el desarrollo de los proyectos, aumentando la satisfacción de los clientes mediante las entregas de resultados tangibles e integrándolos activamente en el ciclo de desarrollo, potencia la formación de equipos de trabajos autosuficientes y multidisciplinares, proporciona a los miembros del equipo un entorno amigable y productivo para desarrollar sus habilidades al máximo (36). Permite combinarse con otras metodologías y marcos de gestión de proyectos, además es un modelo de referencia en el que se definen un conjunto de prácticas y roles, y que puede tomarse como punto de partida para definir el proceso de desarrollo que se ejecuta en un proyecto. En cambio otras fuentes consultadas lo definen como marco de trabajo para la gestión y desarrollo de software en un proceso iterativo e incremental empleado en entornos basados en el desarrollo ágil de software.

1.8.3 SXP

SXP es un híbrido cubano de metodologías ágiles que tiene como base las metodologías SCRUM y XP, las que permiten actualizar los procesos de desarrollo de software para el mejoramiento de su producción. Esta metodología ayuda a fortalecer el trabajo en equipo, enfocados en una misma dirección, permitiendo además seguir de forma clara el avance de las tareas a realizar, a partir de la inserción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la producción, aumentando el nivel de interés del equipo (37). En la figura se describe el Esquema de la metodología SXP.

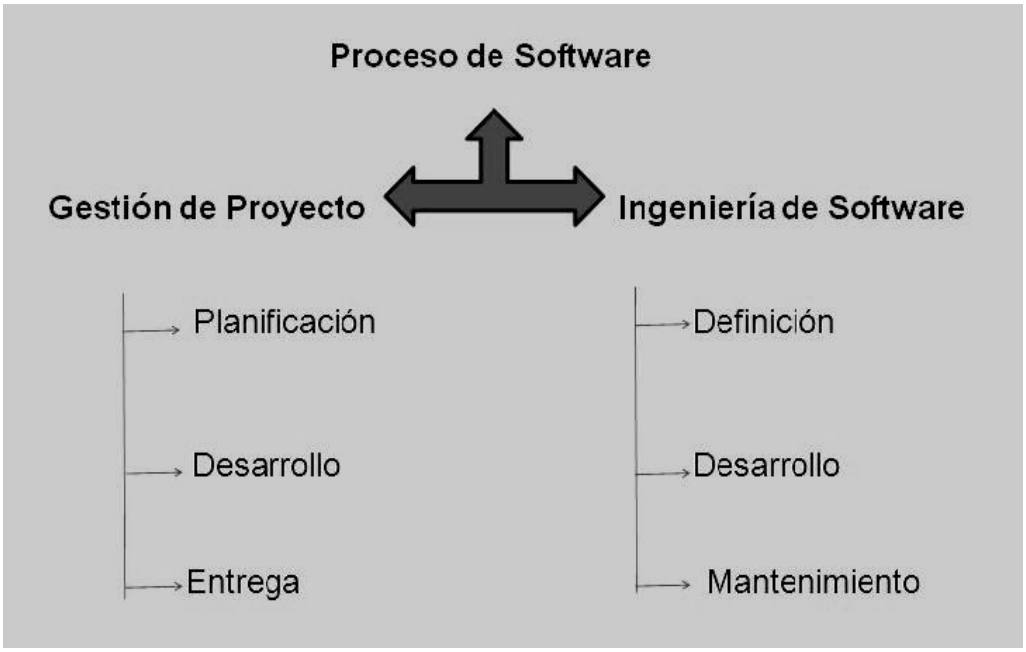


Figura 2. Esquema de la metodología SXP.

Esta metodología se divide en cuatro fases: Planificación-Definición, Desarrollo, Entrega y Mantenimiento, cada una de estas fases está compuesta por una serie de actividades que son las que generan los artefactos que quedan incluidos en el nuevo expediente de proyecto.

1.8.4 XP (eXtreme Programming)

XP es una metodología ligera de desarrollo de software que se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado (35).

Esta metodología tiene tres fases las cuales son: Definición, Desarrollo y Mantenimiento. En cada fase se genera una serie de artefactos, que permiten producir rápidamente versiones del sistema que sean operativas, para que cada entrega sea lo más corta posible y reduzca el riesgo de errores.

Las características más notables de la metodología XP son (35):

- Desarrollo iterativo e incremental.
- Establece mejoras en cada una de las iteraciones del proyecto de forma incremental.
- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión.
- Frecuente interacción del equipo de programación con el cliente o usuario.

- Corrección de todos los errores antes de añadir una nueva funcionalidad y realización de entregas frecuentes.

- Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad pero sin modificar su comportamiento.

- Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto.

- Simplicidad en el código.

El estudio de las metodologías arrojó como resultado que en el caso de RUP donde su principal característica es que está directamente enfocado a proyectos grandes y además genera numerosa documentación por ser una metodología robusta, no es factible para el desarrollo de la aplicación, por su parte SCRUM, no es apta para todos los proyectos y además requiere de un "agile champion", experto en la metodología que monitorice su cumplimiento, por tanto se seleccionó XP pues cuenta con un proceso de diseño disciplinado en el que se combina la disciplina con la adaptabilidad, por lo que, según Martin Fowler, consiste en la más desarrollada de todas las metodologías adaptables. En gran medida se debe al énfasis que hace en las pruebas, poniendo la comprobación como fundamento para el desarrollo, donde cada programador escribe sus pruebas mientras codifica. Dichas pruebas se integran en el proceso de integración continua y construcción, brindando una plataforma estable sobre la cual se construye un proceso de diseño evolutivo basado en la Refactorización de un sistema simple en cada iteración.

1.9 Lenguaje de Programación

Un lenguaje de programación es un lenguaje artificial usado para controlar el comportamiento de una máquina, especialmente una computadora.

Con el objetivo de identificar el lenguaje de programación que será empleado en el desarrollo de la aplicación, en primer lugar se realizó un análisis para seleccionar los lenguajes más usados en la implementación de aplicaciones enmarcadas dentro del mismo dominio de aplicación de la herramienta a desarrollar.

1.9.1 C++

Creado en los Laboratorios Bell por BjarneStroustrup a mediados de 1980. Es un poderoso sucesor de C. Añade a C el concepto de clases, un mecanismo para proveer tipos de datos

definidos por el usuario, también llamado tipo de datos abstractos. C++ soporta el paradigma de la Programación Orientado a Objetos (en lo adelante POO). (51)

Al compilar las aplicaciones realizadas en este lenguaje, se genera código objeto, nativo para cada sistema operativo. Por tal razón no necesita intérpretes por ser compilado, obteniéndose como resultado aplicaciones sorprendentemente rápidas. (51)

Otro punto a tener presente es que permite un control de la memoria y una capacidad de programación de bajo nivel sorprendente, con lo cual se provee un acceso completo al sistema. Esta característica provee una increíble velocidad a la aplicación, pues estaría interactuando directamente con el sistema operativo sin necesidad de intérprete. (51)

Su principal desventaja frente a los lenguajes más actuales, es la ausencia de un “recolector de basura”, teniendo que encargarse el programador de cerrar las referencias de memoria que no vayan a ser usadas. Esta cualidad convierte la tarea de desarrollo algo tediosa, pues mayor control en tu programa implica más dificultad en el desarrollo. Otro punto en contra es que las rutinas de bajo nivel no son portables (que puedan ejecutarse en otro sistema operativo), por lo cual limita el uso de las aplicaciones en otros entornos. (51)

Es muy utilizado también en la creación de aplicaciones de Inteligencia Artificial. Se han desarrollado librerías para la implementación de algoritmos genéticos, como es el caso de la librería GAlib, desarrollada en 1996 por Matthew Wall y su equipo en el MIT en Estados Unidos; la librería aiParts, utilizada para desarrollar la Inteligencia Artificial de múltiples problemas de decisión; y la librería EO (EvolvingObjects) utilizada en la creación de algoritmos evolutivos. (51)

Poseeun framework de desarrollo llamado Qt, que incluye clases, librerías y herramientas para la producción de aplicaciones de interfaces gráficas, que pueden operar en varias plataformas; incluye soporte de nuevas tecnologías como OpenGL, XML, Base de Datos, programación para redes, internacionalización, entre otras. Dispone de herramientas que facilitan la creación de formularios, botones y ventanas de diálogos con el uso del ratón. Otra librería muy utilizada para el desarrollo de aplicaciones gráficas con C++ es winbgim.h. Esta librería tiene como objetivo emular la librería *graphics.h*de Borland C++. (51)

1.9.2 Python

Fue creado por Guido Van Rossum en el año 1990. Surgió como un sucesor del lenguaje de programación ABC, capaz de manejar excepciones e interactuar con el sistema operativo Amoeba. (51)

Es un lenguaje de programación multi-paradigma, permite varios estilos, es dinámico y orientado a objetos. El principal objetivo que persigue es la facilidad, tanto de lectura, como de diseño, además agregar que es de libre distribución. Actualmente se usa en grandes plataformas como: YouTube y Google. (51)

Permite la programación estructural y funcional y se desarrolla como un proyecto de código abierto, administrado por la Python Software Foundation. Tiene gran soporte e integración con otros lenguajes y herramientas. Entre sus principales ventajas está la integración de varias bibliotecas estándar, es rápido de desarrollar, sencillo y sus bibliotecas hacen gran parte del trabajo. (51)

Como desventaja se puede decir que es un lenguaje interpretado y es más lento en comparación con C++ y Java.

1.9.3 C#

Lenguaje altamente expresivo que se ajusta al paradigma de la Programación Orientada a Objeto (POO). Su sintaxis es similar a C++ y Java. Fue desarrollado en gran parte por Anders Hejlsberg. (51)

En C# no existe el concepto de función global o variable fuera de una clase u objeto. Por su buen apego a la POO, es posible sobrecargar métodos y operadores. Soporta la definición de interfaces. Ninguna clase puede poseer más de un padre (no se permite la herencia múltiple), pero sí puede suscribir un contrato con diversas interfaces. Permite la definición de estructuras, pero, a diferencia de C++, aquí no son tan parecidas a las clases y poseen ciertas restricciones. (51)

El código C# se compila como código administrado, lo que significa que se beneficia de los servicios de CommonLanguageRuntime (librería común en tiempo de ejecución, CLR por sus siglas en inglés). Estos servicios incluyen la interoperabilidad de lenguajes, recolección de basura, seguridad mejorada y compatibilidad de versiones mejoradas. (51)

Algunas de las principales desventajas que se derivan del uso de este lenguaje es que, en primer lugar se tiene que conseguir una versión reciente del framework .NET para tener actualizaciones de la librería, es necesario además tener algunos requerimientos mínimos del sistema operativo para poder trabajar adecuadamente (como contar con Windows NT 4 o superior, tener más de 3GB de espacio libre para la instalación, etc.) (51)

1.9.4 Java

Es un lenguaje de programación Open Source (código abierto) que cumple con el paradigma de la POO. Fue desarrollado por Sun Microsystems a principios de los años 90. A partir del 13 de noviembre de 2006, ya es un proyecto completamente libre, tiene la licencia GPL v2.

Toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Es un lenguaje robusto. La gestión de memoria y punteros es realizada por el propio lenguaje y no por el programador. Otro punto a su favor es que contiene estructuras para la detección de excepciones y obliga al programador a escribir código fiable mediante la declaración de excepciones posibles para una determinada clase reutilizable.

Otras de sus características esenciales, es que incorpora Multi-Threading (ejecución de tareas concurrentes dentro de un mismo programa). Fue diseñado “partiendo de cero”, es decir, no necesitaba ser compatible con versiones anteriores de ningún lenguaje como ocurre con C y C++.

Una de las características más importantes es que los programas “ejecutables”, creados por el compilador de Java, son independientes de la plataforma, es decir, se ejecutan indistintamente en una gran variedad de equipos con diferentes microprocesadores y sistemas operativos. Es un lenguaje de desarrollo público y se puede tener acceso gratis al conjunto de herramientas de desarrollo de aplicaciones de Java (Java Developer's Kit, JDK según sus siglas en Inglés). Posee gran conectividad con Bases de Datos, ERP y otros sistemas.

Una ventaja importante que se debe tener presente para el tipo de herramienta que se va a desarrollar, es su trabajo con modelos 3D, aspecto en el que Java tiene un buen desempeño, ya que presenta gran variedad de librerías libres con buena potencia gráfica tales como jGL, JOGL y Java3D, por solo citar algunas (31). Además, se caracteriza por ser un lenguaje muy utilizado en la implementación de aplicaciones que incorporen técnicas de inteligencia artificial. Ejemplo de ello es la gran variedad de publicaciones y software realizados que emplean librerías como JGAP (usada en la implementación de algoritmos genéticos). Añadir también, la existencia de frameworks como JADE (plataforma para el desarrollo de sistemas multi-agentes) que facilitan el desarrollo de este tipo de aplicaciones. (31)

Existe además gran documentación y una amplia comunidad sobre este lenguaje. Sin embargo, resulta importante señalar que los programas realizados en Java suelen ser un poco lentos y para el manejo de código de bajo nivel, se deben utilizar métodos nativos, disminuyendo así su portabilidad. (31)

1.9.5 Selección del lenguaje de programación

Se utilizará Java como lenguaje de implementación de la solución debido a que es un lenguaje de desarrollo de propósito general, para realizar todo tipo de aplicaciones.

Uno de los principales elementos que hacen diferente a Java es su portabilidad, pues permite que la misma aplicación escrita con este lenguaje se pueda ejecutar en cualquier máquina, independientemente del sistema operativo y de la configuración de hardware, gracias a la máquina virtual sobre la que se ejecutan los programas. En el caso de C#, la plataforma Mono permite la ejecución de las aplicaciones en Linux y Mac, diseñadas para Microsoft .NET en entorno Windows. Esta plataforma solo implementa los componentes cubiertos bajo el estándar ECMA aprobado por Microsoft, el cual cubre parte de la librería de clases, el entorno de ejecución y el lenguaje C#. Con C++, sin embargo, si no se utilizan funciones específicas para una determinada plataforma, el mismo código puede ser funcional para todas las arquitecturas de los sistemas operativos.

En cuanto a la potencia gráfica, C++ supera a Java en este sentido, en cambio esto no es una característica determinista pues la aplicación a desarrollar no requiere de grandes prestaciones gráficas.

Al contrario de los demás lenguajes analizados, Java está apoyado por un gran número de librerías que son soportadas por diversas empresas como IBM y Oracle, además de encontrarse otra gran cantidad desarrolladas por terceros, simplificando el desarrollo y la implementación de las aplicaciones.

1.10 Plataforma de Cliente Rico

Una Plataforma de Cliente Rico (Rich Client Platform, en lo adelante RCP), es un entorno para manejar el ciclo de vida de una aplicación, es la base para la creación de aplicaciones de escritorio. (40)

La mayoría de las aplicaciones de escritorio tienen características similares como menús, barras de herramientas, barras de estado, progreso de visualización, visualización de datos, cargar y guardar datos específicos y configuración, internacionalización y sistemas de ayuda, y mucho más. Por estas y otras características típicas de cada aplicación una RCP provee un marco de trabajo con esas características que pueden ser rápida y sencillamente adicionadas a la aplicación. (35, 41)

Este tipo de plataformas están caracterizadas por poseer una:

- Arquitectura de aplicación flexible y modular.
- Adaptabilidad al usuario final.

- Habilidad para trabajar tanto en línea como desconectada.
- Distribución simplificada al usuario final.

El aspecto más importante de una RCP es su arquitectura. Las aplicaciones basadas en RCP son escritas en forma de módulos, con una lógica coherente entre sus partes independientes. Un módulo es descrito declarativamente y automáticamente cargado por la plataforma. (31)

1.10.1 Plataforma NetBeans

Esta plataforma está completamente basada en la Interfaz de Programación de Aplicación (Application Programming *Interface,* API por sus siglas en inglés) de Java utilizando los componentes AWT y Swing, y utilizando además los conceptos de *Java Standard* Edition (JSE) (31). Facilita en gran medida el proceso de desarrollo de aplicaciones de escritorio. El ejemplo más fiel del desarrollo de aplicaciones es el caso de NetBeans, el cual ha sido desarrollado a sí mismo en sobre la plataforma antes mencionada.

Provee además disímiles API que reducen el tiempo de desarrollo. Contiene además un módulo que hace mucho más fácil y eficiente instalar o actualizar los módulos de la aplicación del usuario final. Está basada en estándares y componentes reusables, y como resultado las aplicaciones pueden ser desplegadas en múltiples sistemas, tales como Windows, Linux, Mac, Solaris, etc. Contiene gran variedad de módulos que pueden ser reutilizados en el desarrollo de las aplicaciones; si estos módulos no concuerdan completamente con los requisitos de la aplicación, pues entonces se puede extender de estos componentes a través de sus puntos de extensión. Los componentes de interfaz de usuario como es el caso de los menús, las ventanas, las barras de tarea y otros componentes de la propia plataforma son reutilizables en el desarrollo de aplicaciones. La integración de un marco de trabajo para la creación de wizards (asistentes) que ayudan al usuario en pasos tan complejos como la instalación; el soporte de métodos y clases para la internacionalización de las aplicaciones y un excelente sistema de ayuda, son algunas de las principales características que se pueden encontrar en esta plataforma y que la hacen una excelente elección para el proceso de desarrollo. (31)

1.10.2 Plataforma Eclipse

Eclipse comenzó como una aplicación modular integrada a un IDE. En 2004 la versión 3.0 de Eclipse fue liberada, esta versión soportaba la reutilización de la Plataforma Eclipse para construir aplicaciones basadas en la misma tecnología que el IDE Eclipse.

Las aplicaciones basadas en Eclipse son construidas sobre una arquitectura de plugins. Lo cual proporciona crear aplicaciones que hereden de Eclipse con componentes adicionales.
La base de la arquitectura de Eclipse, su sistema modular, es la especificación OSGi. Este estándar altamente adoptado, permite la creación de módulos dinámicos para Java. Los plugins desarrollados se auto-describen usando un archivo de metadatos.
La interfaz de usuario de Eclipse está basada en SWT y JFace. Este maduro grupo de herramientas provee un amplio conjunto de elementos tales como visores, formularios, vinculación de datos, asistentes, etc. Estas herramientas nativas dotan de un excelente rendimiento a las aplicaciones. Existen muchas herramientas para la construcción de interfaces de usuario (GUI) para el IDE Eclipse, pero la mayoría de ellas son para uso comercial.

Provee un conjunto de puntos de extensión y clases para la comunicación entre módulos, así como funcionalidades para la actualización de las aplicaciones y sistemas de ayudas. Esta plataforma está bajo la licencia Eclipse Public License y la ICU4J. (31)

1.10.3 Selección de la plataforma de desarrollo

Ambas plataformas proveen un sistema de módulos con administración de dependencias, módulos dinámicos y rutas de clases privadas para los mismos. Facilitan además un excelente servicio de infraestructura y una leve carga de los servicios de infraestructura. Las dos proveen excelentes sistemas con soporte para la actualización e integración de sistemas de ayuda y mucho más.
Sin embargo, se decide hacer uso de la Plataforma NetBeans para el desarrollo de la herramienta producto de las características que se describen a continuación en la Tabla 1.

Tabla 1: Comparación entre las plataformas RCP de NetBeans y Eclipse.

	Plataforma NetBeans	**Plataforma Eclipse**
Herramientas de interfaz de usuario	Conjunto de herramientas estándar Swing	SWT
Diseño de interfaz de usuario	Libre y apoyado sobre la herramienta Matisse GUI Builder	Alternativas comerciales para Eclipse
Sistema modular	Sistema modular basado en el estándar OSGi y/o	Sistema de módulos basado en el estándar

	sistema de módulos específico de NetBeans	OSGi
Soporte en JDK	La aplicación VisualVM, una aplicación basada en esta plataforma está incluida en el JDK, por lo tanto, muchas de las librerías de la plataforma NetBeans están incluidas en el JDK	Ninguna de las librerías que requiere para el funcionamiento están incluidas en el JDK, por lo que requiere carga adicional de archivos
Entrenamiento	Entrenamientos basados en una comunidad libre para organizaciones no comerciales	Sin soporte equivalente

1.11 Entornos de desarrollo

1.11.1 NetBeans

NetBeans IDE es uno de los productos fundamentales del proyecto open-source NetBeans junto a Rich Client PLatform. Es reconocido por muchos desarrolladores como el IDE original para Java. Al estar escrito en Java puede ser usado en múltiples plataformas. Brinda completamiento inteligente de código, el cual incluye abreviaturas CamelCase. Además soporta un amplio conjunto de acciones de refactorización de código. Muestra la documentación incluida en la Javadoc mediante ventanas pop-up. (35, 20, 42)

Profiling integrado: permite monitorizar el rendimiento de las aplicaciones basado en el uso que hacen del CPU, la memoria consumida. Para ello permiten la colocación de puntos de control en determinados fragmentos del código fuente llamados profiling points, así como mantener un seguimiento de la memoria usada. Además permite identificar los puntos críticos en cuanto al consumo de la CPU (35, 20, 42)

Java Swing GUI Builder (Matisse): cuenta con un inspector de propiedades y componentes. Además presenta una paleta de componentes extensible con componentes Swing y AWT preinstalados. (35, 20, 42)

1.11.2 Eclipse

Cuando se descarga el Eclipse SDK, se obtiene un equipo de instrumentos para desarrollo en Java o Java DevelopmentToolkit (JDT) para escribir y depurar programas en Java; además se obtiene un ambiente de desarrollo de plugins (Plugin DevelopmentEnviroment) para heredar de Eclipse. Si todo lo que se quiere es un IDE para Java, no se necesita nada además que el JDT. Esto es para lo que la mayoría las personas usan Eclipse. Aunque Eclipse es escrito en Java y su principal uso es como IDE para Java, este es un lenguaje neutral. El soporte para desarrollo en Java es proveído por un componente embebido o plugin, pero además están disponibles plugins para otros lenguajes, como C/C++, Cobol, C# y ActionScript. En principio permite ejecutar un programa sobre cualquier plataforma. Es una extensible plataforma de código abierto (*open* source) para desarrollar herramientas. Administrado y dirigido por un consorcio de compañías de desarrollo de software con un interés comercial en promover Eclipse como plataforma compartida para herramientas de desarrollo de software. (51)

1.11.3 Selección del entorno de desarrollo

El uso de la Plataforma NetBeans para el desarrollo de la aplicación, producto de las ventajas que facilita su uso y su amplia documentación, inducen directamente a la selección de NetBeans como IDE para el desarrollo. A pesar de que ambos IDEs poseen características similares para el trabajo, NetBeans es más fácil e intuitivo de usar, está mejor integrado con Maven y tiene un excelente editor de interfaz de usuario.

1.12 Conclusiones

El estudio y análisis del estado del arte permitió concluir que las aplicaciones de escritorio existentes para la reproducción de paquetes SCORM, no mostraban el contenido de los Objetos de Aprendizaje empaquetados bajo el estándar SCORM 2004, por ser ésta la última versión del mismo.

Para el desarrollo de la solución se seleccionaron varias tecnologías y herramientas, como IDE de desarrollo se decide utilizar NetBeans, como lenguaje de programación Java, además se utiliza como plataforma de desarrollo NetBeans Platform. Como metodología de desarrollo para la guía del proceso se seleccionó XP.

Capítulo 2: Desarrollo de la propuesta de solución

Introducción

Todo proyecto que utiliza metodología XP inicia con una o varias reuniones con el cliente, donde se recogen las necesidades a través de las Historias de Usuario (HU). Sirve también como base para crear una metáfora del sistema con el cual todo el equipo de trabajo tendrá una idea general de la aplicación a implementar.

El objetivo es describir la solución propuesta para resolver el problema planteado. Siguiendo la metodología XP se realiza la planificación incluyendo a los clientes, programadores y coordinadores. Se comienza con la recopilación de los datos que serán registrados en las Historias de Usuario (HU), los programadores evalúan el tiempo de desarrollo de cada una de ellas, se establece un plan de entregas con el cliente del cual surge el número inicial de iteraciones y duración de las mismas y de ahí comienza la fase de las iteraciones que debe iniciar con una reunión en la que se da claridad a las tareas a desarrollar, obteniendo un plan que servirá como hoja de ruta para el transcurso de la iteración. Se confecciona las tarjetas CRC (Clase-Responsabilidad-Colaboración) que identifica las responsabilidades asociadas a las clases del sistema.

2.1 Metáfora del Sistema

En XP no se enfatiza la definición temprana de una arquitectura estable para el sistema. Dicha arquitectura se asume evolutiva y los posibles inconvenientes que se generarían por no contar con ella explícitamente en el comienzo del proyecto se solventan con la existencia de una metáfora. El sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema. Kent Beck plantea que "…una metáfora del sistema es algo que todos entienden, sin necesidad de mayores explicaciones". Las metáforas ayudan con la abstracción y modelado del sistema. Estas en XP reemplazan a lo que se llama arquitectura. (44)

La tarea de elegir una metáfora para el sistema permite mantener una coherencia de nombres de todo aquello que se va a implementar, por lo que la metáfora propuesta es: "Un visor de contenidos donde se muestren los recursos de los Objetos de Aprendizaje

empaquetados bajo el estándar SCORM2004 para que los usuarios puedan interactuar con los mismos". (44)

2.2 Propuesta de solución

Para contribuir con el aprendizaje de los usuarios, se propone la realización del Visor UPLAYER_SCORM2004 que mostrará los Objetos de Aprendizaje empaquetados bajo el estándar SCORM2004.

Esta aplicación se realizará con el objetivo de que los usuarios finales cuenten con una herramienta que les permita visualizar los contenidos de cualquier OA empaquetado bajo éste estándar.

La aplicación brindará al usuario la posibilidad de abrir el fichero comprimido (Zip) que constituye el OA, el cual será visualizado estructuralmente por el sistema en forma de árbol, al seleccionar un elemento del mismo y ejecutar clic derecho Mostar Recurso, el visor mostrará el navegador web externo por defecto del sistema operativo con el contenido del recurso, además al ejecutar clic en los botones Anterior y Siguiente de la barra de herramientas se permitirá la secuenciación lineal entre los mismos.

2.3 Exploración

En esta etapa el cliente crea las HU que son de interés para la entrega del producto. Además, el equipo de desarrollo se familiariza con las herramientas y tecnologías que se utilizarán en el proyecto.

2.4 Historias de usuarios

Las HU tienen el mismo propósito que los casos de uso. Estas se escriben desde la perspectiva del cliente aunque los desarrolladores pueden brindar también su ayuda en la identificación de las mismas. El contenido de estas debe ser concreto y sencillo.

A continuación se muestran las **Historias de Usuario identificadas por el cliente**:

Tabla 2: Historias de Usuarios identificadas por el cliente.

#	Historias de Usuario	Prioridad en negocio
1	Cargar Objeto de Aprendizaje(OA)	Alta
2	Mostrar Recurso	Alta
3	Secuenciar recursos del OA	Media

A continuación la plantilla que se usará para definir las **Historias de Usuario**:

Tabla 3: Estructura de una Historia de Usuario.

Historia de Usuario	
Número: Número sucesivo a partir de uno.	**Usuario:** El usuario del sistema que utiliza o protagoniza la historia**.**
Nombre historia: Identifica la Historia de Usuario.	
Prioridad en negocio: Define la relevancia e impacto de la Historia de Usuario para el negocio de acuerdo a las necesidades del usuario.	**Iteración asignada:** Precisa la iteración a la que pertenece la historia de usuario.
Descripción: Explica en qué consiste la Historia de Usuario, teniendo en cuenta las acciones realizadas por el usuario y la respuesta brindada por el sistema.	

En esta fase de exploración se llegaron a identificar 3 historias de usuario las cuales se describen a continuación:

Tabla 4: HU Cargar Objeto de Aprendizaje.

Historia de Usuario	
Número: 1	**Usuario:** Cliente
Nombre historia: Cargar Objeto de Aprendizaje(OA)	
Prioridad en el negocio: Alta	**Iteración asignada:** 1
Descripción: Se le brinda al usuario la posibilidad de abrir el fichero comprimido (Zip) que constituye el OA, el cual será visualizado estructuralmente por el sistema en forma de árbol.	

Tabla 5: HU Mostrar Recursos.

Historia de Usuario	
Número: 2	**Usuario:** Cliente
Nombre historia: Mostrar Recurso	
Prioridad en el negocio: Alta	**Iteración asignada:** 2

Descripción: Al seleccionar un elemento del árbol y ejecutar clic derecho Mostrar Recurso el visor mostrará el navegador web externo por defecto del sistema operativo con el contenido del recurso del OA.

Tabla 6: HU Secuenciación entre los recursos.

Historia de Usuario	
Número: 3	**Usuario:** Cliente
Nombre historia: Secuenciar recursos del OA.	
Prioridad en el negocio: Media	**Iteración asignada:** 2
Descripción: Al ejecutar clic en los botones Anterior y Siguiente de la barra de herramientas se muestra en orden secuencial lineal los recursos del OA abierto en el navegador web externo por defecto del sistema operativo.	

2.5 Estimación de esfuerzos por Historias de Usuarios

Las estimaciones del esfuerzo para implementar las HU permiten tener una medida real de la velocidad de progreso del proyecto y brindan una guía a la cual ajustarse. Los resultados obtenidos se muestran a continuación:

Tabla 7: Estimación de esfuerzos por Historia de Usuario.

No.	Historia de Usuario	Puntos de Estimación (semanas)
1	Cargar Objeto de Aprendizaje(OA)	3
2	Mostrar Recurso	3
3	Secuenciar recursos del OA	1
Total		**7**

2.6 Plan de Iteraciones

Para facilitar la creación del sistema, la metodología divide el proceso en etapas, que por lo general son tres, las cuales toman el nombre de iteraciones. Para cada iteración se define un módulo o conjunto de HU que se van a implementar. Al final de la iteración se obtiene una entrega del módulo correspondiente, el cual debe haber superado las pruebas de

aceptación que establece el cliente para verificar el cumplimiento de los requisitos. Esto se registra en el Plan de Iteraciones.

Iteración 1: En la primera iteración se realizará la primera Historia de Usuario ya que es de prioridad alta:

1- Cargar Objeto de Aprendizaje(OA)

Iteración 2: En esta iteración se realizan las restantes Historias de Usuarios que son importantes para el cliente:

1- Mostrar Recursos.
2- Secuenciar recursos del OA.

2.7 Plan de duración de las iteraciones

El plan de duración de las iteraciones se realiza luego de tener el estimado en días que demora implementar cada HU. Se tendrá en cuenta la prioridad que el cliente le asigna a cada HU y la complejidad que estas tengan.

Tabla 8: Plan de duración de las iteraciones.

Iteración	Orden de las HU	Duración total
1	➢ Cargar Objeto de Aprendizaje(OA)	3 semanas
2	➢ Mostar Recursos ➢ Secuenciar recursos del OA	4 semanas

2.8 Plan de entregas

En el plan de entregas se establecen el orden y las HU que se agrupan para conformar una entrega. Este plan tiene como finalidad reflejar la duración de cada iteración, lo que ayuda a tener una idea aproximada del tiempo que durará la confección del sistema en su totalidad. Este cronograma será el resultado de una reunión entre el equipo de desarrollo y el cliente, permitiendo alcanzar un mayor entendimiento en la implementación del sistema.

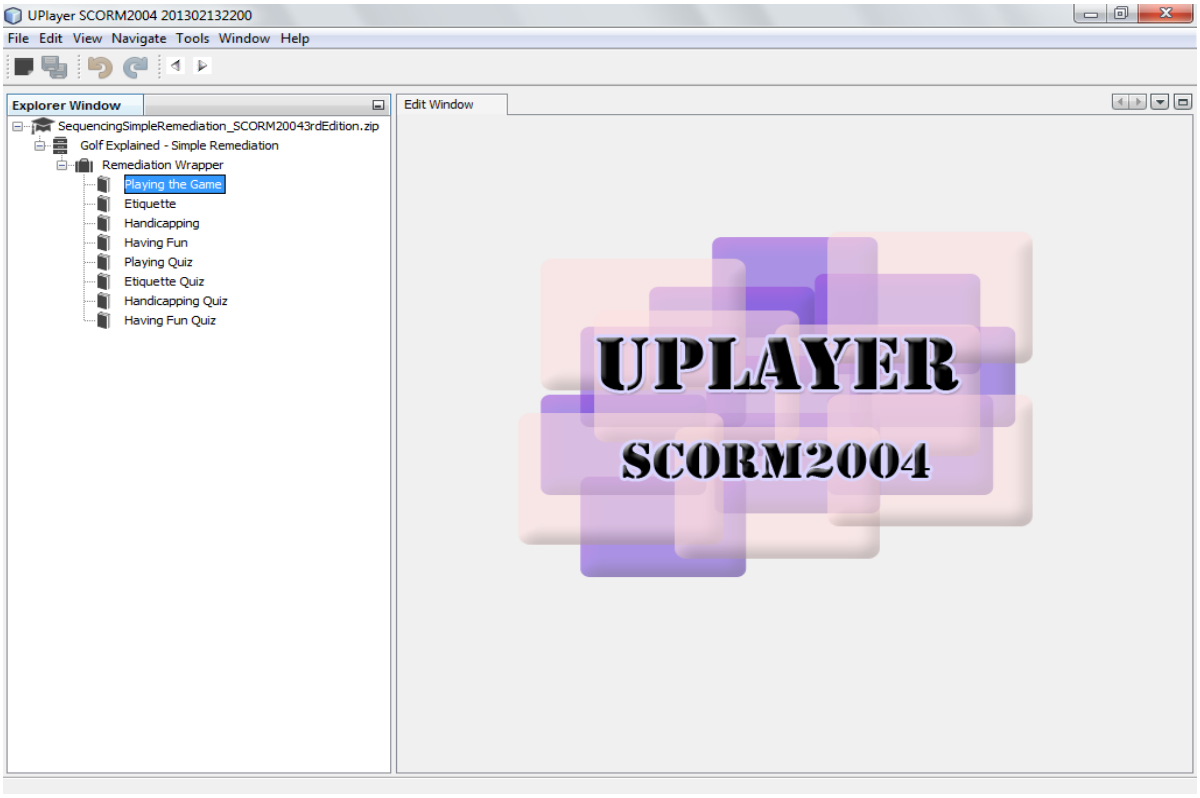
Tabla 9: Plan de entregas.

Historia de Usuario	1ra Iteración	2da Iteración
Cargar Objeto de Aprendizaje(OA)	V 1.0	Finalizado
Mostrar Recurso	-	V 1.1
Secuenciar recursos del OA	-	V 1.2

2.9 Prototipo de interfaz de usuario

Para mostrar una vista preliminar del sistema que se va a desarrollar se crea un prototipo de interfaz, el cual es de fácil modificación conteniendo todas las características definidas de la propuesta de solución. La propuesta que se muestra a continuación parte de la interfaz inicial que presentará el visor de contenidos UPLAYER_SCORM2004. Esta propuesta puede cambiar en algún momento, según los criterios del cliente.

Tabla 10: Interfaz de usuario.



2.10 Tarjetas CRC

La metodología no requiere la presentación del sistema mediante diagramas de clases utilizando notación UML, en su lugar se crean las tarjetas CRC que permiten trabajar con una metodología basada en objetos y posibilitan que el equipo completo contribuya en la tarea del diseño. Una tarjeta CRC representa un objeto, módulo o clase y debe cumplir tres principios básicos, Cargo o Clase, Responsabilidad y Colaboración. Estas tarjetas son llamadas CRC (clases, responsabilidad, colaboración). Son sencillas de utilizar y entender y permiten al equipo de trabajo en su totalidad participar en el diseño del sistema. Además estas tarjetas son una herramienta de reflexión en el diseño de software orientado a objeto. (45)

Tabla 11: Tarjeta CRC Manager.

Manager	
Responsabilidades	**Colaboradores**
✓ read	✓ Reader ✓ KnowObject ✓ Exception

Tabla 12: Tarjeta CRC Reader.

Reader	
Responsabilidades	**Colaboradores**
✓ read ✓ extract ✓ find ✓ unZip ✓ saveFile ✓ getFileName	✓ KnowObject ✓ ObjectType ✓ Resource ✓ Exception

Tabla 13: Tarjeta CRC KnowObject.

KnowObject	
Responsabilidades	**Colaboradores**
✓ addPropertyChangeListener ✓ getName ✓ setName ✓ getItems ✓ setItems ✓ setResource ✓ getResource ✓ getType ✓ setType ✓ getParameters ✓ setParameters ✓ getHref	✓ KnowObject ✓ ObjectType ✓ Resource

Tabla 14: Tarjeta CRC Resource.

Resource	
Responsabilidades	**Colaboradores**
✓ getId ✓ setId ✓ getType ✓ setType ✓ getHref ✓ setHref	

Tabla 15: Tarjeta CRC EmbeddedTomcat.

EmbeddedTomcat	
Responsabilidades	**Colaboradores**
✓ stopTomcat ✓ startTomcat	✓ Tomcat ✓ Exception

Tabla 16: Tarjeta CRC ManagerNode.

ManagerNode	
Responsabilidades	**Colaboradores**
✓ getTomcat ✓ getPrevItem ✓ getNextItem ✓ getInstance ✓ addPropertyChangeListener ✓ removePropertyChangeListener ✓ getNodes ✓ add ✓ getPropertyChangeSupport ✓ extractLeaf	✓ EmbeddedTomcat ✓ KnowObject ✓ ObjectType ✓ Resource ✓ Exception

Tabla 17: Tarjeta CRC ExplorerNodeFactory.

ExplorerNodeFactory

Responsabilidades	Colaboradores
✓ createNodes ✓ propertyChange ✓ addNotify	✓ KnowObject ✓ ExplorerNode ✓ Exception

Tabla 18: Tarjeta CRC ExplorerNode.

ExplorerNode	
Responsabilidades	**Colaboradores**
✓ getActions ✓ setIcon	✓ KnowObject ✓ ObjectType ✓ ExplorerNodeFactory ✓ Exception

Tabla 19: Tarjeta CRC NextItemAction.

NextItemAction	
Responsabilidades	**Colaboradores**
✓ actionPerformed	✓ KnowObject ✓ Exception

Tabla 20: Tarjeta CRC PrevItemAction.

PrevItemAction	
Responsabilidades	**Colaboradores**
✓ actionPerformed	✓ KnowObject ✓ Exception

Tabla 21: Tarjeta CRC OpenFileAction.

OpenFileAction	
Responsabilidades	**Colaboradores**
✓ actionPerformed	✓ KnowObject ✓ Exception

Tabla 22: Tarjeta CRC ViewAction.

ViewAction	
Responsabilidades	**Colaboradores**
✓ actionPerformed	✓ KnowObject ✓ Exception

2.11 Conclusiones

En el presente capítulo se describió la propuesta de solución para el desarrollo de los módulos intérprete y visor de contenidos para el Uplayer_SCORM2004.

Las fases de Planificación y Diseño posibilitaron la confección de las historias de usuario, la planificación y estimación de estas, diseño de las tablas para la base de datos y tarjetas CRC.

El empleo de la metodología de desarrollo XP permitió minimizar el tiempo de trabajo, en cuanto a reducir la cantidad y complejidad de los artefactos a modelar.

Capítulo 3: Implementación y Prueba.

Introducción

En el presente capítulo se exponen los aspectos correspondientes a las fases de Desarrollo y Pruebas de la metodología XP, haciendo énfasis en las tareas desarrolladas por cada iteración, así como los resultados de las pruebas efectuadas a la propuesta de solución.

3.1 Fase de Implementación

Como resultado del desarrollo se obtuvo de manera paulatina, la implementación de las Historias de Usuarios según la planificación realizada para cada iteración. Al principio de estas, se lleva a cabo una revisión del plan de iteraciones y se modifica en caso de ser necesario.

3.2 Plan de Tareas

Como parte del plan de iteraciones, se desglosaron las Historias de Usuarios en tareas de desarrollo asignándole al programador la responsabilidad de su implementación. Estas tareas son para el uso estricto del programador, pueden ser escritas en lenguaje técnico y no necesariamente entendible por el cliente.

Tabla 23: Plan de Tareas.

Iteraciones	Historias de Usuarios	Tareas
Iteración 1	Cargar Objeto de Aprendizaje(OA)	T1: Cargar el fichero Zip. T2: Descomprimir el fichero Zip. T3: Extraer, interpretar y organizar el OA. T4: Adicionar OA.
Iteración 2	Mostrar Recurso	T5: Iniciar el servidor tomcat embebido.
	Secuenciar recursos del OA	T6: Ejecutar servidor T7: Extraer y organizar recursos del OA. T8: Mostrar recurso Siguiente del OA. T9: Mostrar recurso Anterior del OA.

Iteraciones

Iteración 1: se implementó la Historia de Usuario Cargar Objeto de Aprendizaje (OA) una de las más importantes para la estructura y el diseño de la aplicación, es decir, las funcionalidades principales del sistema que dan soporte a la implementación de las demás funcionalidades. Con el fin de obtener una primera versión del producto para ser mostrado al cliente y realizar nuevos cambios en caso necesario.

Tabla 24: Tarea 1 Cargar el fichero Zip.

Tarea	
No. de Tarea: 1	**No. de Historia de Usuario:** 1
Nombre de la Tarea: Cargar el fichero Zip.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 08/01/2013	**Fecha Fin:** 29/01/2013
Programador Responsable: Darien Diego Hevia	
Descripción: Se da clic en el botón Open File, el sistema muestra una ventana de selección de ficheros, se realiza la búsqueda del fichero Zip, y una vez que se le de Aceptar se le pasa la ruta del fichero a la clase Reader.	

Tabla 25: Tarea 2 Descomprimir el fichero Zip.

Tarea	
No. de Tarea: 2	**No. de Historia de Usuario:** 1
Nombre de la Tarea: Descomprimir el fichero Zip.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 08/01/2013	**Fecha Fin:** 29/01/2013
Programador Responsable: Darien Diego Hevia	
Descripción: Una vez cargado el fichero Zip se ejecuta el método **unZip** encargado de descomprimir el fichero en la ruta correspondiente a la carpeta base del servidor web.	

Tabla 26: Tarea 3 Extraer Objeto de Aprendizaje.

Tarea	
No. de Tarea: 3	**No. de Historia de Usuario:** 1
Nombre de la Tarea: Extraer, interpretar y organizar el OA.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1

Fecha Inicio: 08/01/2013	**Fecha Fin:** 29/01/2013
Programador Responsable: Darien Diego Hevia	
Descripción: Se obtiene el fichero imsmanifiest.xml del fichero Zip y se ejecuta el método **extract** encargado de traducir el xml y encapsularlo en un objeto de tipo KnowObject.	

Tabla 27: Tarea 4 Adicionar Objeto de Aprendizaje.

Tarea	
No. de Tarea: 4	**No. de Historia de Usuario:** 1
Nombre de la Tarea: Adicionar OA.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 08/01/2013	**Fecha Fin:** 29/01/2013
Programador Responsable: Darien Diego Hevia	
Descripción: Adiciona el OA al árbol de estructura.	

Iteración 2: se continuó con el desarrollo de las restantes historias de usuarios, ofreciéndole al cliente la posibilidad de consultar detalles y características de los contenidos de la aplicación. La versión de pruebas de esta iteración además de la anterior, fueron mostradas al cliente con el objetivo de obtener su valoración y nuevas indicaciones.

Tabla 28: Tarea 5 Iniciar el servidor tomcat embebido.

Tarea	
No. de Tarea: 5	**No. de Historia de Usuario:** 2
Nombre de la Tarea: Iniciar el servidor tomcat embebido.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 29/01/2013	**Fecha Fin:** 27/02/2013
Programador Responsable: Darien Diego Hevia	
Descripción: Se ejecuta el método startTomcat de la clase EmbeddedTomcat.	

Tabla 29: Tarea 6 Ejecutar servidor.

Tarea	
No. de Tarea: 6	**No. de Historia de Usuario:** 2
Nombre de la Tarea: Ejecutar servidor.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 29/01/2013	**Fecha Fin:** 27/02/2013
Programador Responsable: Darien Diego Hevia	

Descripción: Ejecutar el explorador web con la dirección (URL) del recurso seleccionado en el árbol de estructura.

Tabla 30: Tarea 7 Extraer y organizar recursos del OA.

Tarea	
No. de Tarea: 7	**No. de Historia de Usuario:** 3
Nombre de la Tarea: Extraer y organizar recursos del OA.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 29/01/2013	**Fecha Fin:** 27/02/2013
Programador Responsable: Darien Diego Hevia	
Descripción: Extrae linealmente los nodos hojas del árbol asociados a recursos del OA.	

Tabla 31: Tarea 8 Mostrar recurso Siguiente del OA.

Tarea	
No. de Tarea: 8	**No. de Historia de Usuario:** 3
Nombre de la Tarea: Mostrar recurso Siguiente del OA.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 29/01/2013	**Fecha Fin:** 27/02/2013
Programador Responsable: Darien Diego Hevia	
Descripción: De los nodos hojas extraídos linealmente se muestra el nodo actual y secuencialmente los siguientes.	

Tabla 32: Tarea 9 Mostrar recurso Anterior del OA.

Tarea	
No. de Tarea: 9	**No. de Historia de Usuario:** 3
Nombre de la Tarea: Mostrar recurso Anterior del OA.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 29/01/2013	**Fecha Fin:** 27/02 /2013
Programador Responsable: Darien Diego Hevia	
Descripción: De los nodos hojas extraídos linealmente se muestra el nodo actual y secuencialmente los Anteriores.	

3.3 Aplicación de los Patrones de diseño

Los patrones de diseño son soluciones simples y elegantes a problemas específicos y comunes del diseño, basados en la experiencia y en la evidencia de su funcionamiento. (46)

GRASP

GRASP es un acrónimo de General Responsibility Assignment Software Patterns (patrones generales de software para asignar responsabilidades). El nombre se eligió para sugerir la importancia de aprender (grasping en inglés) estos principios para diseñar con éxito el software orientado a objetos (47), describen los principios fundamentales del diseño de objetos y la asignación de responsabilidades.

Experto: se utiliza con frecuencia en la asignación de responsabilidades; es un principio de guía básico que se utiliza continuamente en el diseño de objetos. El Experto no pretende ser una idea oscura o extravagante; expresa la "intuición" común de que los objetos hacen las cosas relacionadas con la información que tienen (47). Para poner en práctica este patrón en la construcción del sistema se definieron en cada clase las funcionalidades relacionadas directamente con la información contenida en ésta, se evidencia en la clase KnowObject

```
public class KnowObject {

    private String name;
    private List<KnowObject> items;
    private Resource resource;
    private ObjectType type;
    private String parameters;

    public KnowObject(String name, ObjectType type) {
        this.name = name;
        this.type = type;
        this.items = new LinkedList<KnowObject>();
        this.resource = null;
        this.propertyChangeSupport = new PropertyChangeSupport(this);
    }
…
```

```
}
```

Creador: guía la asignación de responsabilidades relacionadas con la creación de objetos, una tarea muy común. La intención básica del patrón Creador es encontrar un creador que necesite conectarse al objeto creado en alguna situación (47). Está relacionado con el patrón experto, le asigna a una clase la responsabilidad de crear aquellos objetos que contiene y les brinda la información necesaria para su creación, se evidencia en la clase Reader

```
private Resource find(String idref) {
    NodeList resources = documentElement.getElementsByTagName(RESOURCE_TAG_NAME);
    for (int i = 0; i < resources.getLength(); i++) {
      Element e = (Element) resources.item(i);
      if (idref.equals(e.getAttribute("identifier")))  {
        Resource resource = new Resource(
            e.getAttribute("identifier"),
            (temporal.getAbsolutePath().replace("C:\\" + TEMPORAL_FILE_NAME, "http://localhost:9999") + D_SLASH + e.getAttribute("href")).replace(D_SLASH,  "/"),
            e.getAttribute("type"));
        return resource;
      }
    }
    return null;

  }
```

Bajo Acoplamiento: El acoplamiento es una medida de la fuerza con que un elemento está conectado a otros elementos (47). El bajo acoplamiento disminuye la dependencia entre los componentes del sistema y el impacto de los cambios externos a ellos garantizando una mayor reutilización de éstos. Está presente en los Actions y en las relaciones entre los módulos de la solución propuesta.

Alta Cohesión: existe alta cohesión si una clase tiene una responsabilidad moderada en un área funcional y colabora con otras clases para llevar a cabo las tareas. Como regla empírica, una clase con alta cohesión tiene un número relativamente pequeño de métodos, con funcionalidad altamente relacionada, y no realiza mucho trabajo. Colabora con otros objetos para compartir el esfuerzo si la tarea es extensa. (47)

GoF

Los patrones de la "pandilla de los cuatro" (Gang-of-Four) deben su nombre a Gamma, Helm, Johnson, Vlissides, quienes publicaron en 1995 el trabajo "Design Patterns", en 1995, subtitulado "Elementos de software orientado a objetos reusable". Estos autores son los que toman el trabajo de hacer un catálogo de los patrones de diseño que hasta ese momento habían aparecido, y en una lista de 23 patrones, tratan de enumerar el conocimiento acumulado sobre el tema. En el diseño de la solución propuesta se utilizan varios de estos patrones, es el caso del patrón, Singular (Singleton) que garantiza una única instancia de la clase Manager

```
public class Manager {

    private static Manager instance;

    private Reader reader;

    /**
     *
     */
    private Manager() {
        reader = new Reader();
    }

    /**
     * @return the instance
     */
    public static Manager getInstance() {
        if (instance == null) instance = new Manager();
        return instance;
    }
…
}
```

, Observador (Observer)

```
public class ExplorerNodeFactory extends Children.Keys<KnowObject> implements
PropertyChangeListener {
```

```
    …
    @Override
    public void propertyChange(PropertyChangeEvent evt) {
        if (evt.getPropertyName().equals(ManagerNode.NEW_NODE)) {
            setKeys(knowObjects);
        }
    }
…
}
```

Este patrón se utiliza para notificar al árbol de estructura que se ha adicionado un nuevo nodo y por lo tanto debe pintarlo. (47)

3.4 Arquitectura de la aplicación

La IEEE define la arquitectura de software como la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución. José A. San Gil define el estilo basado en componentes como una rama de la Ingeniería de software en la cual se trata con énfasis la descomposición del software en componentes funcionales. Esta descomposición permite convertir componentes pre-existentes en piezas más grandes de software (48). El patrón arquitectónico empleado para el diseño de la aplicación es Arquitectura basadas en Componentes.

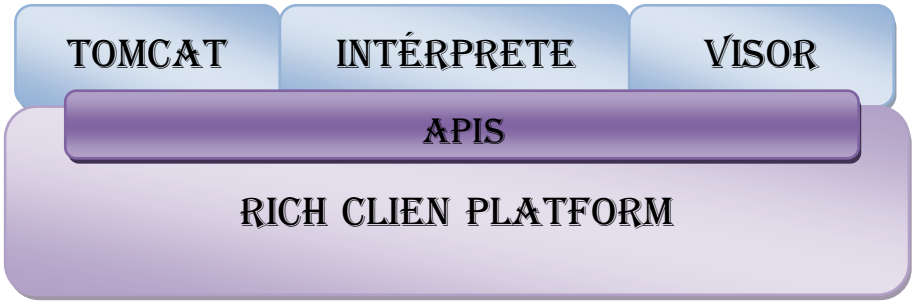


Figura 3: Arquitectura de la aplicación.

3.5 Fase de Pruebas

La forma más adecuada para evaluar un producto de software son las pruebas de calidad que se le aplican al mismo. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos. (49)

En XP es de suma importancia el proceso de pruebas de calidad de software. XP anima a probar constantemente tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones. (50)

XP divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores, las pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñada por el cliente. (50)

3.5.1 Pruebas de Aceptación

Se considera que las pruebas de aceptación poseen un peso superior a las unitarias, pues las mismas muestran a los desarrolladores la satisfacción del cliente. Por este motivo se puede decir que estas ponen fin a una iteración dando inicio a la siguiente.
Las pruebas de aceptación se describen mediante una tabla llamada Caso de Prueba de Aceptación y son destinadas a evaluar si al final de cada iteración se obtuvo la funcionalidad requerida y son creadas sobre la base de las Historias de Usuarios. El cliente desde su punto de vista creará los casos de prueba en conjunto con los desarrolladores para probar que una Historias de Usuarios ha sido implementada correctamente. Por cada Historias de Usuarios se elaborarán todas las pruebas de aceptación que se necesiten para asegurar su correcto funcionamiento. (51,52)

A continuación se muestra la estructura de un caso de prueba de aceptación:

Tabla 33: Estructura de un caso de prueba de aceptación.

Caso de prueba de aceptación	
Código: Código que identifica la prueba.	**Historia de usuario:** Número de la historia de usuario relacionada con la prueba que se realiza.
Nombre: Definición de la prueba que se realiza.	
Descripción: Breve descripción del objetivo con que se realiza la prueba.	

Entrada/Pasos de Ejecución: Se describen los pasos de ejecución de la prueba en cuestión.
Resultado Esperado: Proporciona las expectativas ideales para las cuales fue pensada la prueba.
Evaluación de la Prueba: Calificación que recibe la prueba de acuerdo a los resultados obtenidos.

A continuación se muestran los casos de pruebas de aceptación realizados para las Historias de Usuarios:

Tabla 34: Caso de Prueba de Aceptación HU1_P1.

Caso de prueba de aceptación	
Código: HU1_P1	**Historia de usuario:** 1
Nombre: Mostrar el árbol de estructura del OA.	
Descripción: Prueba para la funcionalidad que le permite al usuario visualizar el árbol de estructura del OA.	
Entrada/Pasos de ejecución El usuario, debe dar Clic en Archivo, Abrir OA, buscar el fichero comprimido (Zip) del OA y "Aceptar", automáticamente la aplicación mostrará la estructura en forma de árbol en el componente Explorador que se encuentra en la parte izquierda del visor.	
Resultado esperado: El árbol de estructura del Objeto de Aprendizaje se mostrará con éxito.	
Evaluación de la prueba: Satisfactoria.	

Tabla 35: Caso de Prueba de Aceptación HU2_P2.

Caso de prueba de aceptación	
Código: HU2_P2	**Historia de usuario:** 2

Nombre: Mostrar los recursos del OA.
Descripción: Prueba para la funcionalidad que le permite al usuario visualizar los recursos que contiene el OA.
Entrada/Pasos de ejecución El usuario, debe dar Clic derecho encima del recurso, mostrar recurso y automáticamente la aplicación mostrará el contenido en el navegador web que trae por defecto el sistema operativo.
Resultado esperado: El recurso se mostrará con éxito.
Evaluación de la prueba: Satisfactoria.

Tabla 36: Caso de Prueba de Aceptación HU3_P3.

Caso de prueba de aceptación	
Código: HU3_P3	**Historia de usuario:** 3
Nombre: Secuenciar recursos del OA	
Descripción: Prueba para la funcionalidad que le permite al usuario visualizar los recursos Siguientes o Atrás que contiene el OA.	
Entrada/Pasos de ejecución El usuario, debe ejecutar clic en los botones Anterior y Siguiente de la barra de herramientas, automáticamente se mostrará en orden secuencial lineal los recursos del OA abierto en el navegador web externo por defecto del sistema operativo.	
Resultado esperado: El recurso se mostrará con éxito.	
Evaluación de la prueba: Satisfactoria.	

3.5.2 Resultados de las Pruebas

Durante el desarrollo de la aplicación se realizaron las pruebas necesarias para comprobar el cumplimiento y la calidad de las funcionalidades del mismo, planteadas por el cliente mediante las Historias de usuarios, para lo cual se realizaron pruebas de aceptación. Por cada historia de usuario se realizó un caso de prueba. Fueron detectadas un total de 8 no

conformidades en las 2 iteraciones realizadas, recogido de la siguiente manera: en la primera iteración se detectaron 6 no conformidades, 3 significativas, 2 no significativas y 1 recomendación, en la segunda iteración se detectaron 2 no conformidades, 1 no significativa y 1 recomendación. Las no conformidades detectadas se centraron más en los errores ortográficos como son: comas, puntos, paréntesis, mayúsculas o minúsculas, ya que esta aplicación requiere de mucha documentación. Las significativas en errores de validación y cambios del sistema, pero luego de realizar un análisis exhaustivo se logró detectar el problema y solucionarlo. Las no conformidades encontradas no influyeron en que la aplicación cumpliera con todos los objetivos propuestos por el cliente logrando así el éxito de la misma y la evaluación satisfactoria de todas las pruebas.

A continuación se muestra una gráfica de barras que ilustra la cantidad de no conformidades encontradas por tipo en cada iteración.

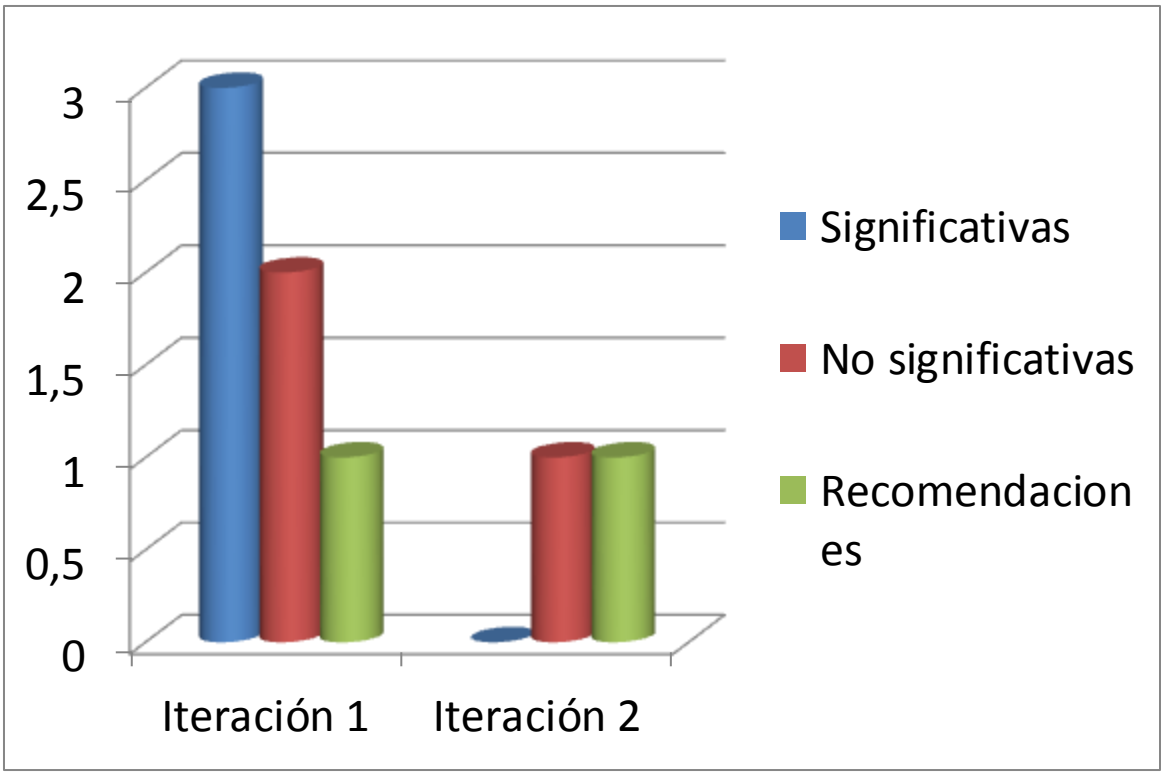


Figura 4: Registro de no conformidades detectadas.

3.6 Conclusiones

La propuesta de solución permitió desarrollar las funcionalidades descritas a partir de las herramientas y lenguaje seleccionados, tomando como punto de partida el diseño propuesto en el capítulo anterior. Se validó el sistema propuesto mediante la realización de pruebas de

aceptación a las historias de usuarios, a través de dos iteraciones. Las pruebas fueron evaluadas de satisfactorias por parte del cliente.

Conclusiones generales

La investigación desarrollada y los resultados obtenidos permiten al autor plantear las siguientes conclusiones:

- ✓ El estudio y análisis del estado del arte sobre las tecnologías empleadas para el desarrollo de un reproductor de archivos SCORM arrojó como resultado que las herramientas existentes para estos fines a nivel internacional no permiten la interpretación de Objetos de Aprendizaje empaquetados bajo las versiones más actuales del estándar SCORM.
- ✓ El desarrollo de los módulos intérprete y visor de contenidos para el Uplayer_SCORM2004 permitió obtener una herramienta que contribuirá al aprendizaje de los usuarios, a través de la visualización de los Objetos de Aprendizaje empaquetados bajo el estándar SCORM2004.
- ✓ La aplicación fue validada mediante la realización de pruebas de aceptación al cliente, las no conformidades detectadas fueron resueltas en tiempo, obteniéndose un producto con calidad.

Recomendaciones

Como parte del proceso desarrollador llevado a cabo por el programador se recomienda:

- ✓ Implementar el estándar de secuenciación simple IMS-SS.
- ✓ Agregarle nuevas funcionalidades para hacerlo más portable, usable y manejable.
- ✓ Realizar la internalización de la aplicación para varios idiomas, específicamente para el español.
- ✓ Documentar el código para que pueda ser comprendido por futuros desarrolladores.
- ✓ A la comunidad científica, tomar el presente trabajo como material de estudio en el desarrollo de aplicaciones similares.

Referencias Bibliográficas

1. RODRÍGUEZ, Oscar. Unidad 3:El estándar SCORM — UNIA OpenCourseWare. In: [online]. 2012. [Accessed 11 June 2013]. Available from: http://ocw.unia.es/innovaciondocente_formacionprofesorado/diseno-de-contenidos-educativos-multimedia/contenidos_ud3.

2. e-ABC es e-Learning Sin Límites. In: *e-ABC* [online]. 2011 2010. [Accessed 21 February 2013]. Available from: http://www.e-abclearning.com/.

3. SCORM - SCORM Versions - An eLearning Standards Roadmap » SCORM -. In: [online]. [Accessed 11 June 2013]. Available from: http://scorm.com/es/scorm-explic%C3%B3/negocio-de-scorm/versiones-scorm/.

4. SUELEN, Oseida. 1 August 2011. [Accessed 21 February 2013]. Available from: http://www.slideshare.net/yplarosa/newsfeed.

5. JACOBS, Mark. IEEE - The world's largest professional association for the advancement of technology. In: [online]. 2013. [Accessed 22 February 2013]. Available from: http://www.ieee.org./index.html.

6. CAÑIZARES, Roxana, FEBLES, Juan Pedro and ESTRADA, Vivian. Los objetos de aprendizaje, una tecnología necesaria para las instituciones de la educación superior en Cuba | Cañizares González | Revista Cubana de Información en Ciencias de la Salud. In: [online]. 2012. [Accessed 22 February 2013]. Available from: http://www.acimed.sld.cu/index.php/acimed/article/view/248.

7. D. WILEY, Marcus. *A definition, a metaphor and a taxonomy.* S.l.: s.n., 2000.

8. POLSANI, R. Pithamber. *Use and Abuse of Reusable Learning Objects*. 2003. S.l.: s.n.

9. MASON, REHAK. D. *Journal of Interactive Media in Education*. 2003. S.l.: s.n.

10. Uso de estándares aplicados a Tic en educación. In: [online]. [Accessed 23 February 2013]. Available from: http://ares.cnice.mec.es/informes/16/contenido/8.htm#up.

11. FERNANDEZ-MAJÓN, Baltasar. *Especificaciones y estándares en e-learning.* [online]. S.l.: s.n., 2006. Available from: http://reddigital.cnice.mec.es/6/Articulos/Articulo_resumen.php.

12. *Real Academia Española* [online]. 2010. S.l.: s.n. Available from: www.rae.es. RAE

13. SALVADOR BROCHE, Orlando Felipe and SOLER MARTÍN, Javier. *Implementación de un Repositorio de Objetos de Aprendizaje para la Universidad de las Ciencias Informáticas. Ciudad de La Habana*. S.l.: s.n., 2010.

14. IGLESIAS BORJA, Manero. *Estudio de la propuesta IMS de estandarización de enseñanza asistida*. 2003. S.l.: s.n.

15. GONZÁLEZ, Hilera, RAMÓN, José and HOYA MARÍN, Rubén. *ESTÁNDARES DE E-LEARNING*. 2010. S.l.: s.n.

16. CAÑIZARES, Roxana. *Framewok para la gestión de contenidos educativos*. 2008. S.l.: s.n. Trabajo de Diploma en opción al título Ingeniero en Ciencias Informáticas.

17. *ADL* [online]. 2002. S.l.: s.n. Available from: http://xml.coverpages.org/ADLRepositoryTIR.pdf. Emerging and Enabling Technologies for the design of Learning Object.

18. NGUEN, Van Nhu. *Binding the Simple Query Interface.* 2007. S.l.: s.n.

19. BALTASAR FERNÁNDEZ, Manjón, MORENO GER, Pablo, SIERRA RODRÍGUEZ, José Luis and MARTÍNEZ ORTÍZ, Iván. Centro Nacional de Información y Comunicación Educativa (CNICE-MEC). In: [online]. [Accessed 11 June 2013]. Available from: http://ares.cnice.mec.es/informes/16/contenido/13.htm.

20. DIÉGUEZ, Jorge. La norma SCORM, un acercamiento práctico | Raccoon. In: [online]. [Accessed 11 June 2013]. Available from: http://www.raccoon-learning.com/conocimiento_raccoon/la-norma-scorm-un-acercamiento-practico/.

21. Unidad 5: Herramientas para la creación de SCORM para Moodle — UNIA OpenCourseWare. In: [online]. [Accessed 24 February 2013]. Available from: http://ocw.unia.es/innovaciondocente_formacionprofesorado/diseno-de-contenidos-educativos-multimedia/contenidos_ud5/skinless_view.

22. [En línea] http://descargas.mundo-r.com/ficha/1141648416060_es.html..

23. Warez., Argentina. [En línea] http://www.argentinawarez.com/programas-gratis/1542091-descarga-gratis-vlc-media-player-v1-2-0-nightly-portable.html..

24. Informática, Alumnos del Master de Formación del Profesorado en la especialidad de. Instalación de LMS. . [En línea] 2009. joseperezlozano.com/lms/?page_id=178..

25. ysalinas. Cosolig comunidad de software libre. [En línea] 2012. http://www.cosolig.org/2012/04/13/blackboard-no-pudo-con-la-competencia-y-compra-moodle/..

26. PR Newswire. . [En línea] 2013. http://www.prnewswire.co.uk/news-releases/blackboard-supera-el-test-de-certificado-adlscorm-gracias-a-su-producto-content-player-building-block-155162835.html..

27. Hernández., Nitza M. López. TECNOINFO. . [En línea] 14 de Octubre de 2010. http://tecnoinfo6300.blogspot.com/2010/10/sakai-plataforma-educativa-de-codigo.html..

28. Icodeon Scorm 2004 Player. [En línea] 2004-2009. http://www.icodeon.com/..

29. Torre, Aníball de la. adelat. [En línea] 2006. www.adelat.org/media/docum/moodle/docum/23_cap01.pdf..

30. Blue Core Software. . [En línea] http://bluecoresoftware.es/mediapool/103/1033199/data/flex_scorm_player.pdf..

31. RELOAD Projects: SCORM Player. . [En línea] www.reload.ac.uk/scormplayer.html ..

32. Centro Aragonés de Tecnologías para la educación - Herramientas de Autor. [En línea] http://www.catedu.es/webcatedu/index.php/descargas/herramientas-de-autor..

33. [En línea] http://xmeele.wordpress.com/2009/07/17/aplicaciones-web-vs-aplicaciones-de-escritorio/.

34. EcuRed. . [En línea] http://www.ecured.cu/index.php/Proceso_Unificado_de_Desarrollo..

35. Jacobson, Ivar, Booch, Grady Booch y Rumbaugh, James. .El Proceso Unificado de Desarrollo de Software. . [En línea] 1999.

36. Object Management Group - UML. [En línea] 1997-2013. www.uml.org..

37. Project, M. Cross platform, open source .NET development framework. . [En línea] 2010. http://www.mono-project.com/Main_Page..

38. Oracle Technology. In: [online]. . [En línea] 2012. http://www.oracle.com/technetwork/java/javase/index.html.

39. ExtJs 3.4- Sencha. [En línea] http://www.sencha.com/products/extjs..

40. Hernández., Maylen Fiffe. EcuRed. . [En línea] http://www.ecured.cu/index.php?title=Patrones_en_Symfony&oldid=585394..

41. Halgamuge, M.N., T.-K. Chan, and P. Mendis. Experiences of Deploying an Indoor Building Sensor Network. in Sensor Technologies and Applications, 2009. SENSORCOMM '09. Third International Conference on. 2009. [En línea]

42. ConectaTutoriales. . [En línea] http://www.tutorialesadobe.conectatutoriales.com/blog/noticias-y-articulos/195-adobe-flash-professional-cs6-cree-y-distribuya-experiencias-sofisticadas-y-atractivas-en-diversos-dispositivos..

43. Desarrollo de software. Tarjetas CRC | Jummp. Desarrollo de software. . [En línea] http://jummp.wordpress.com/2012/01/10/desarrollo-de-software-tarjetas-crc/..

44. Patrón de diseño. . [En línea] http://sourcemaking.com/patr%C3%B3n-de-dise%C3%B1o..

45. 49. LARMAN, Csaig. UML y Patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado. Segundaedición. [En línea] 2002.

46. Gil, Jose A. San. Scribd. [En línea] http://es.scribd.com/doc/14704374/Arquitectura-Basada-en-Componentes..

47. Las pruebas de software forman el único instrumento adecuado para determinar la calidad de software. [En línea] http://pruebasdesoftware.com/laspruebasdesoftware.htm. .

48. Pruebas del Sistema en Programación Extrema . [En línea] http://www.lsi.us.es/~javierj/investigacion_ficheros/PSISEXTREMA.pdf. .

49. Pérez, Fernández. Revista Cubana de Ciencias Informáticas. . [En línea] http://rcci.uci.cu/index.php/rcci/art.

50. Prueba de Aceptación. [En línea] http://pruebasdesoftware.com/pruebadeaceptacion.htm..

51. Estévez, Ing. Henry Rodríguez y Molina Toledo, Ing. Pablo. Arquitectura de la Herramienta Andrómeda. 2012.

Glosario de Términos

Objetos de aprendizaje: son recursos digitales dedicados a la enseñanza y que presentan metadatos que permiten su localización e interoperabilidad.

TIC: Tecnologías de la Información y las Comunicaciones.

IMS: Especificación de pruebas de interoperabilidad.

LOM: Metadatos de Objetos de Enseñanza.

SCORM: Contenido Compartido modelo de objetos de referencia.

IMS: Learning Resources Metadata: metadatos de recursos de enseñanza-aprendizaje.

ARIADNE: Alianza de distancia, autoría de Instrucción y Distribución de Redes para Europa.

DublinCore: Es un modelo de metadatos elaborado y auspiciado por la DCMI (Dublin Core Iniciativa de Metadatos).El estándar The Dublin Core Metadata Element Set de ISO recoge el conjunto de metadatos definidos por la iniciativa, un foro abierto dedicado al desarrollo de estándares de metadatos de propósito general enfocado principalmente a la localización y catalogación de recursos.

IEEE/LTSC: Learning Technology Standardization Committee (Comité de estandarización de tecnologías aplicadas al aprendizaje). Su principal objetivo es desarrollar estándares, prácticas recomendadas y guías para componentes de software.

LOM-ES: describe un perfil de aplicación que adapta al sistema educativo español, el perfil extiende LOM con nuevos elementos y vocabulario, lo que ha hecho necesario la nueva implementación de esquemas XML para el uso y adaptación de las herramientas de autorías existentes.

AICC: Aviation Industry Computer Based Trainig Comitte (Comité de la Industria de la Aviación para el aprendizaje basado en Computadoras), fue creado en 1992 y fue el primer organismo creado para desarrollar un conjunto de normas que permitiese el intercambio de cursos CBT (Computer Based-Training).

ADL: Advanced Distributed Learning es una iniciativa creada por el departamento de Defensa de EEUU y la oficina de Ciencia y Tecnología de la Casa Blanca lanzada en

Noviembre de 1997. ADL ha sido una de las organizaciones más activas en el esfuerzo de la estandarización de las tecnologías de aprendizaje, en colaboración con otras iniciativas principalmente IEEE, IMS y AICC.

CEN: Comité Europeo de Normalización es la organización regional europea de estandarización. Fue fundado en 1961 y su principal objetivo es fomentar la economía europea en el negocio global, el bienestar de ciudadanos europeos y el medio ambiente.

XP: Programación Extrema.

ECMA: Ecma International es una organización internacional basada en membresías de estándares para la comunicación y la información.

Jbuilder: es un entorno de desarrollo para el lenguaje de programación Java de Borland.

NLN: National Learning Network, se encarga de proveer una infraestructura de red robusta y una amplia gama de programas de apoyo, información y asesoramiento, así como de desarrollar materiales digitales para la enseñanza y el aprendizaje.

LMS: Learning Manager Sistem (Sistema de Gestión de Aprendizaje).

B-Learning: (formación combinada, del inglés blended learning) consiste en un proceso docente semipresencial; esto significa que un curso dictado en este formato incluirá tanto clases presenciales como actividades de e-learning.

Anexo1

Tabla 37: Observaciones y resultados del método empírico.

Observaciones
1- Funcionalidades que poseen las aplicaciones existentes. ✓ Mostrar el contenido del Objeto de Aprendizaje (OA). ✓ Mostrar la estructura del OA en forma de árbol. 2- Características de la interfaz. ✓ Organización del contenido.
Resultados
1- Se realizó un estudio de los diferentes visores que reproducían OA, y para mostrar el contenido del mismo se tomó como referencia el visor Reload Scorm Player, así como para mostrar la estructura. 2- En la interfaz, se organizó el contenido de acuerdo a las necesidades del cliente y ninguno de los visores que se estudiaron cumplían con las mismas.