Universidad de las Ciencias Informáticas

Facultad 4

Aplicación ofimática para la creación de presentaciones electrónicas en línea

---

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

Autor:

Raúl Noa Pedroso

Tutores:

Ing. Yunior Orosa Velázquez

Ing. Yordanis Rodríguez Rodríguez

Co-tutores:

Ing. Jorge Martínez Padrón

MSc. Roberto López Dosagues

La Habana, Junio 2014

"Año 56 de la Revolución"

Debemos ser parte del cambio que queremos ver en el mundo...

Ghandi

Declaración de autoría

Declaro que soy el autor de este trabajo y autorizo a la Facultad 4 de la Universidad de las Ciencias Informáticas, así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste se firma la presente a los _____ días del mes de _____ del año 2014.

_____

Autor

Raúl Noa Pedroso

_____

Tutor

Ing. Yunior Orosa Velázquez

_____

Tutor

Ing. Yordanis Rodríguez Rodríguez

_____

Co-tutor

Ing. Jorge Martínez Padrón

_____

Co-tutor

MSc. Roberto López Dosagues

Agradecimientos

De Raúl Noa Pedroso:

Agradezco:

A mis padres que sin ellos no hubiera llegado hasta aquí, que siempre quisieron verme aquí y que este también era su sueño;
A mis tutores que hicieron más de lo que podían para que pudiéramos llegar;

A mis compañeros de grupo y todos aquellos que compartieron conmigo estos 5 años.

Resumen

Un elemento importante en la actualidad es lograr transmitir con claridad una idea, de forma tal que el espectador comprenda aquello que se quiere expresar. La utilización de herramientas que permitan formular diferentes conceptos a través del uso de imágenes, diagramas, videos y otros medios, tienen cada vez más aceptación; en este contexto se encuentran las presentaciones electrónicas. En los últimos años la Web de la Universidad de las Ciencias Informáticas (UCI) ha ido creciendo aceleradamente, condicionando la necesidad de mejorar la calidad de sus contenidos con propósitos comunicativos y formativos para la comunidad universitaria. El presente trabajo tiene como objetivo desarrollar una aplicación ofimática que permita crear y visualizar presentaciones electrónicas en línea para ampliar las posibilidades de divulgación de la información digital en la Universidad de las Ciencias Informáticas. Para guiar el proceso de desarrollo se utilizó la metodología *Extreme Programming* (XP) que además permitió generar los artefactos necesarios y lograr su correcta documentación. La implementación fue realizada utilizando los lenguajes HTML5, CSS3 y JavaScript utilizando como Entorno de Desarrollo Integrado a Netbeans. El desarrollo guiado por las pruebas unitarias y de aceptación permitió comprobar la correcta implementación de las funcionalidades de la solución propuesta.

Palabras clave: aplicación ofimática, aplicación web, presentación electrónica.

Índice de contenidos

Índice de contenidos

Índice de tablas

Índice de tablas

Índice de tablas

Índice de tablas

Índice de figuras

Introducción

En la actual era de la información, el conocimiento ha venido a desempeñar un papel fundamental en los procesos que se realizan a diario. La Web es hoy un gran espacio de intercambio universal, una vitrina de acceso fácil e inmediato a una cantidad extensa y diversa de recursos en línea. Internet tiene en la actualidad un impacto profundo a nivel mundial en el área del trabajo, el entretenimiento, el conocimiento, la educación, entre otras esferas.

En la economía de la información, el profesional debe ser capaz de adaptarse a nuevas tareas, procesos y fuentes, a medida que la demanda, la tecnología y las políticas institucionales aceleran su ritmo de cambio y actualización constante (Gutiérrez, 2008). En estos tiempos, apenas hay parcela del saber y del conocimiento que no quede rápidamente obsoleta. La Web, comparada a las enciclopedias y a las bibliotecas tradicionales, ha permitido una descentralización repentina y extrema de los datos.

Estudiantes, investigadores, científicos, escritores, periodistas y aficionados a los temas más diversos e inverosímiles, tienen en Internet un lugar donde buscar información, un dato, encontrar una novedad y sobre todo intercambiar conocimiento.

Desde los primeros años de la humanidad, el hombre ha buscado e innovado disímiles métodos y vías para transmitir ideas, órdenes, experiencias y conocimiento; con la utilización de la Red de Redes el abanico de posibilidades es mucho mayor.

Un elemento importante en la actualidad es lograr transmitir con claridad una idea, de forma tal que el espectador comprenda aquello que se quiere expresar. La utilización de herramientas que permitan expresar diferentes conceptos a través del uso de imágenes, diagramas, videos y otros medios, tienen cada vez más aceptación; en este contexto se encuentran las presentaciones electrónicas.

Las presentaciones electrónicas no solo se utilizan en exposiciones de conferencias, eventos, foros y congresos; también constituyen un medio auxiliar de apoyo al proceso de enseñanza y aprendizaje. Estas mejoran notablemente la atención, la comprensión y el aprendizaje, ayudan a mantener el interés sobre el tema que se está abordando, así como incrementan la retención de la información presentada (Gómez, 2007). Con el uso de la Web, se amplían las posibilidades para difundir estas presentaciones con una mayor repercusión.

Introducción

Internet ha creado un escenario en el que las posibilidades de interacción que las nuevas herramientas ofrecen junto al acceso de un nuevo público objetivo, con interés en compartir, expresar y comunicar han configurado un nuevo modelo caracterizado por el dinamismo, la transcendencia de los contenidos y las comunidades de usuarios (Pérez, 2014). En este sentido, es importante la labor que realizan diferentes redes sociales (como Facebook[1]) y aplicaciones o servicios web centrados en la publicación de diversos contenidos que se apoyan en nuevas tecnologías como la computación en la nube.

Para el usuario es importante tener acceso a contenidos de calidad y relevancia, utilizando para esto los medios contemporáneos para consultar información. Una web que ofrezca contenidos relevantes para los usuarios puede obtener como beneficio un aumento en cuanto a número de visitas, mejorar la promoción del sitio y el posicionamiento web[2]. Sin embargo, en determinadas ocasiones no existe la posibilidad de adaptar todo el contenido que se necesita mostrar debido a cuestiones como la extensión o ubicación del mismo; haciéndose necesario buscar nuevas vías para insertar el contenido en la web. En esta situación es importante buscar métodos que ayuden a enriquecer el contenido a presentar a partir de la utilización de materiales de apoyo que estimulen e incentiven la aprehensión de la idea a exponer.

Como parte de las directrices trazadas por la Revolución cubana, impulsar el quehacer docente e investigativo en el país ha tomado un papel fundamental en el desarrollo de la sociedad. Cuba ha logrado formar una fuerza profesional, que demuestra su madurez con las investigaciones y trabajos que se exponen en diferentes eventos y espacios, con el fin de fomentar el desarrollo científico de la Isla.

La Universidad de las Ciencias Informáticas (UCI) fundada por el Comandante en Jefe Fidel Castro Ruz, constituye un centro estratégico para el desarrollo científico y profesional del país. "*La investigación científica en la UCI contribuye significativamente a la superación profesional y formación académica de su claustro, mejora su competencia como profesores y favorece la calidad del proceso de aprendizaje de los estudiantes, mediante su incorporación al trabajo científico vinculado al proceso de desarrollo e innovación*" (Dirección de Investigaciones, 2012). Entre sus misiones fundamentales asume la de convertir aplicaciones y servicios informáticos en un espacio de difusión de sus resultados, enlace y comunicación

[1] Facebook es una red social creada por Mark Zuckerberg, fue fundada en el 2004 y se estima que en la actualidad cuenta con más de 500 millones de usuarios que intercambian textos, videos, fotografías y cualquier otro tipo de archivo digital (Kirkpatrick, 2011).

[2] El posicionamiento web es el proceso de mejorar la visibilidad de un sitio web en los resultados de los diferentes buscadores.

permanente entre docente, investigadores, desarrolladores y las comunidades universitarias en general (UCI, 2012).

La Web de la UCI está compuesta por aproximadamente 159 sitios que contienen más de 700 000 páginas (Mondelo y Amat, 2010). Estos sitios responden a diferentes temáticas y objetivos, habiendo diversidad en cuanto al tamaño de los sitios (dígase los sitios con mayor cantidad de páginas) y el contenido que presentan. En los últimos años la Web de la UCI ha ido creciendo aceleradamente, condicionando la necesidad de mejorar la calidad de sus contenidos con propósitos comunicativos y formativos para la comunidad universitaria; mejorarlos, posibilita además, obtener un aumento en cuanto a las visitas y popularidad de los sitios.

Aun cuando se emplean diversos recursos que amplían los contenidos expuestos en la Red de la UCI a partir de la utilización de imágenes, audios, videos, documentos adjuntos[3], entre otros; **es insuficiente la divulgación de la información digital**. El contenido publicado referente a convocatorias, eventos, programas de actividades, conferencias científicas, entre otros; se apoya de documentos adjuntos para complementar y detallar información. Por cuestiones de tiempo o la dependencia a las funcionalidades de otras aplicaciones que no sean las del propio navegador, no todos los usuarios descargan y visualizan dichos adjuntos, contribuyendo a que la comunidad universitaria no este lo suficientemente informada. Lo anterior determina que sea importante utilizar algún medio que contribuya a solventar este problema y que a la vez favorezca la creación de nuevos contenidos acordes con el modelo actual en Internet.

Específicamente, el uso de un recurso como las presentaciones electrónicas facilita el desarrollo de las actividades formativas y de comunicación, contribuyendo a la divulgación de la información. Pero existen un conjunto de problemas que afectan su correcta utilización y que se describen a continuación:

- **Dependencia de la ubicación y de una estación de trabajo** (entiéndase como computadora, portátil u otro dispositivo utilizado para trabajar) para la creación de las presentaciones electrónicas. Es usual que los usuarios realicen presentaciones electrónicas desde una estación de trabajo específica. Al trasladarse del lugar desde dónde se realizó el trabajo, puede ser que no puedan usar las presentaciones debido a que no existan condiciones de compatibilidad.

[3] Un documento adjunto es un archivo (entiéndase un documento de texto, una presentación electrónica entre otros) que por lo general se envía junto a un mensaje de correo electrónico, pero que también puede ser anexado para su descarga en una página web.

Introducción

- La **heterogeneidad de sistemas operativos**, condiciona la multiplicidad de aplicaciones ofimáticas. Lo anterior conlleva a problemas de compatibilidad debido a la falta de soporte entre las versiones ofimáticas, y la inexistencia de un estándar que unifique la variedad de formatos y extensiones de archivo existentes, tales como *.ppt, .pptx, .odp, .key* entre otras.
- La **forma en que se visualizan las presentaciones electrónicas no es siempre la correcta**, consecuencia de la heterogeneidad de las aplicaciones ofimáticas, debido a problemas como el cambio de fuentes y estilos.
- **Imposibilidad de insertar una presentación electrónica en la Web** sin la dependencia en la utilización de *plugins* u otro medio para su visualización. En la actualidad para poder visualizar una presentación electrónica en un sitio web, es necesario el uso de algún *plugin* (como Adobe Flash[4]) que convierta la presentación en imágenes o, depender de alguna de las plataformas existentes (SlideShare[5], Slideboom[6] entre otras) cuyo uso no es recomendable en la UCI por limitaciones tecnológicas.

Es por ello que el **problema a resolver** de la presente investigación queda formulado de la siguiente forma:

¿Cómo ampliar las posibilidades de divulgación de la información digital en la Universidad de las Ciencias Informáticas?

Objeto de estudio:

Las aplicaciones ofimáticas para presentaciones electrónicas.

Campo de acción:

El desarrollo de aplicaciones ofimáticas para la creación y visualización de presentaciones electrónicas en línea.

[4] El *plugin* Adobe Flash es utilizado en las páginas web para la visualización de videos, juegos, animaciones y otros contenidos. Su utilización ha decaído en los últimos años, producto de los problemas que presenta para su correcto uso y de la tendencia actual en la Web de no depender de *plugins* para visualizar contenidos.
[5] Disponible en: http://www.slideshare.net
[6] Disponible en: http://www.slideboom.com

Introducción

Objetivo general:

Desarrollar una aplicación ofimática que permita crear y visualizar presentaciones electrónicas en línea para ampliar las posibilidades de divulgación de la información digital en la Universidad de las Ciencias Informáticas.

Objetivos específicos:

- Elaborar el marco teórico conceptual a partir del estudio del estado del arte sobre las aplicaciones ofimáticas para presentaciones electrónicas.
- Identificar los requerimientos con que debe cumplir la solución propuesta para la definición de las funciones básicas que el sistema debe proporcionar y sus propiedades.
- Implementar los requerimientos definidos para la obtención de la solución propuesta.
- Realizar pruebas de software a la solución desarrollada.

A partir de los objetivos trazados se plantea la siguiente **idea a defender:** El desarrollo de una aplicación ofimática que permita crear y visualizar presentaciones electrónicas en línea ampliaría las posibilidades de divulgación de la información digital en la Universidad de las Ciencias Informáticas.

Tareas de investigación:

- Revisión bibliográfica de los principales referentes teóricos en el desarrollo de aplicaciones ofimáticas en línea.
- Análisis de las soluciones similares que permitan cumplir el objetivo general.
- Realización de entrevistas a los administradores de diferentes sitios web de la UCI para la obtención de los requerimientos con que debe contar la solución propuesta.
- Selección de la metodología de desarrollo para guiar el proceso de elaboración de la solución propuesta.
- Selección de las herramientas y tecnologías a utilizar en la implementación de la solución propuesta.
- Descripción de los artefactos a generar durante la implementación de la solución propuesta.

- Implementación de la solución propuesta.
- Realización de pruebas a las funcionalidades implementadas.

Para dar cumplimiento a los objetivos propuestos se recurrió a la utilización de **métodos científicos de investigación** con el objetivo de crear las bases necesarias que tributen al éxito de la investigación.

Métodos teóricos

- Histórico – lógico: con el propósito de analizar la evolución histórica de las soluciones similares y en especial de sus características, tanto a nivel internacional como nacional.
- Analítico – sintético: para el análisis de la documentación relacionada con el objeto de estudio, determinando los elementos necesarios para el cumplimiento del objetivo general.
- Inductivo – deductivo: para estudiar las soluciones similares existentes tanto a nivel internacional como nacional atendiendo a sus funcionalidades y características con el propósito de determinar algunos de los requerimientos con que debe contar la solución propuesta.

Métodos empíricos:

- Entrevista semiestructurada: con la finalidad de obtener información acerca de las necesidades de una aplicación web para la creación de presentaciones electrónicas en línea, los requerimientos que debería poseer así como las recomendaciones que contribuyan al cumplimiento del objetivo general.

El presente documento consta de tres capítulos, desarrollados a partir del estudio realizado. La descripción de los mismos se muestra a continuación:

Capítulo I: Fundamentación teórica:

En este capítulo se hace referencia a los elementos teóricos que soportan la investigación, se presentan los lenguajes de desarrollo, así como las tecnologías y herramientas utilizadas en el desarrollo de la solución. Se describen las soluciones similares existentes, y se hace hincapié en sus ventajas y desventajas como vía para el desarrollo de la propuesta.

Introducción

Capítulo II: Descripción de la solución propuesta:

En este capítulo se realiza una descripción de las características del sistema a implementar mediante las historias de usuarios. Se realiza la planificación de las iteraciones a desarrollar y el plan de entrega. Se conforma la arquitectura de la solución, en conjunto con los patrones arquitectónicos y de diseño seleccionados.

Capítulo III: Implementación y pruebas:

Abarca todo lo relacionado con la implementación del sistema, utilizando los lenguajes de desarrollo definidos y la metodología seleccionada. Se definen los tipos de pruebas que se le realizaron al sistema, con el objetivo de asegurar la eficiencia de la solución y se muestran los resultados obtenidos.

Capítulo I: Fundamentación teórica

1.1 Introducción

En la actualidad, donde el Internet presenta un gran auge y avance cubriendo las necesidades de las personas y ayudándolas a llevar a cabo sus tareas diarias; surgen las aplicaciones web. Estas le ofrecen al usuario la oportunidad de acceder a la información de manera más rápida, realizar tareas o investigaciones de una manera más eficiente y acceder a servicios para crear documentos o presentaciones; todo esto sin necesidad de instalar algún software en su computadora.

A continuación se exponen un conjunto de conceptos y fundamentos teóricos con el objetivo de desarrollar el marco teórico relacionado con los aspectos definidos en el objeto de estudio y el campo de acción, a fin de crear las bases teóricas que propicien la generación de la solución propuesta. En el caso de la presente investigación es muy importante conocer la tendencia en el uso de las aplicaciones ofimáticas, sobre todo enfocado a su uso desde la web. Se fundamenta la selección de las herramientas, metodología y tecnologías a utilizar en el desarrollo de la investigación.

1.2 Conceptos asociados al dominio del problema

Aplicación web

Es una aplicación hipermedia diseñada para la web, que permite confeccionar páginas dinámicas. Esto permite una constante actualización e incluso personalización, capaces de adaptarse a los tipos de usuarios y en casos avanzados, a cada usuario en particular (Mateu, 2004). Una aplicación web intenta portar las clásicas aplicaciones de escritorio hacia entornos web, que son utilizadas a través de los distintos navegadores, agregando portabilidad y capacidad de acceder desde diferentes dispositivos. Además de la inmediatez de acceso a la aplicación, otras de las características más notables que presenta una aplicación web, y que han logrado un incremento en su utilización, son el hecho de ser la mayoría gratuitas o a precios módicos e incitar la colaboración entre los usuarios. A la par del auge en el uso de aplicaciones web como medio para facilitar la realización de diferentes actividades, se encuentra el surgimiento del término de computación en la nube.

Capítulo I

Computación en la nube

La computación en la nube o *cloud computing*, constituye un nuevo paradigma de la época actual, y es definida por el Instituto Nacional de Estándares y Tecnologías (NIST en inglés) como "*un modelo para habilitar un acceso conveniente, por demanda, a un conjunto de recursos computacionales configurables, tales como redes, servidores, almacenamiento, aplicaciones y servicios, los que pueden ser rápidamente aprovisionados y liberados con un esfuerzo mínimo de administración o de interacción con el proveedor de servicios*" (eLAC, 2011). Para Joyanes (2009), consiste en un conjunto de tecnologías de computación que están configurando un nuevo orden mundial en las Tecnologías de la Información y las Comunicaciones (TIC) que parte, esencialmente, de las expectativas creadas por la web 2.0 entre los usuarios personales y corporativos. Esta ofrece servicios de la web social, que promueven el establecimiento de relaciones abiertas y la construcción de conocimiento bajo un esquema de dar y recibir, según Torres (2008), de la web ontológica y de la web ubicua a través de sus modelos de despliegue, de entrega y de facturación. En la actualidad este enfoque está siendo ampliamente utilizado en la realización de diferentes aplicaciones, debido a ventajas como un rendimiento optimizado y la seguridad.

Aplicación ofimática

Es el conjunto de técnicas, aplicaciones y herramientas informáticas que sirven para la organización, presentación y manipulación de la información. Son ampliamente utilizadas en funciones de oficina para optimizar, automatizar y mejorar los procedimientos o tareas relacionados (Muñoz y González, 2011). En la actualidad, debido a los grandes volúmenes de información que se deben procesar, la tendencia en el uso de estas herramientas se incrementa gradualmente. Este incremento no se limita solo a soluciones hechas para ser utilizadas como aplicaciones de escritorio, sino que también se puede observar en diferentes opciones que se están desarrollando para la web. Estas herramientas son empleadas en la realización de hojas de cálculo, documentos de texto, diseños de publicaciones, boletines, presentaciones electrónicas, entre otros.

Presentación electrónica

Es un producto informático que se basa en imágenes elaboradas en las computadoras y que se muestran mediante un proyector. Se realiza en programas que permiten crear de una manera rápida, llamativa y profesional láminas o diapositivas digitales donde se pueden insertar textos, imágenes y elementos

multimedia como video, audio y animación (Paéz, 2010). Tiene como objetivo realizar exposiciones visuales ante un público numeroso y mejorar notablemente la atención, comprensión y el aprendizaje. En la escuela cubana, la presentación electrónica es tradicionalmente el método de apoyo a las clases presenciales que se auxilian de las TIC (Duquesne, 2007). Estas presentaciones se realizan con programas que por lo general forman parte de paquetes ofimáticos, donde se pueden encontrar otras herramientas para la confección de distintos documentos. En los últimos tiempos, con el auge de Internet, su desarrollo también se ha vuelto inmerso utilizando aplicaciones alojadas en la web.

Presentación web

Conjunto de diapositivas que contienen texto, imágenes, gráficos, animaciones, sonidos y videos como apoyo a una exposición; dichas presentaciones se realizan, editan y publican por medio de Intranet/Internet (Leyva y Hidalgo, 2011). Estas cada vez tienen mayor aceptación por parte de los usuarios, debido a la facilidad de uso propio del entorno en dónde se gestionan.

1.3 Análisis de soluciones existentes

Un programa de presentaciones es un software utilizado para mostrar información normalmente esquematizada en una o más diapositivas (Comunidad de Madrid, 2013). Entre sus características principales está la posibilidad de permitir colocar texto, gráficos, películas y otros objetos en páginas individuales o diapositivas. Típicamente incluye tres funciones principales: un editor que permite insertar un texto y darle formato, un método para insertar y manipular imágenes y gráficos y un sistema para mostrar el contenido en forma continua (Yebes y Romero, 2003).

Son de gran utilidad en presentaciones orales siendo estos utilizados para generar documentos que sirven de apoyo visual al presentador. Hay muchos tipos de presentaciones, para educación, o para comunicar noticias en general. Los programas de presentación pueden servir de ayuda o reemplazar a las formas tradicionales de dar una presentación, como por ejemplo panfletos, pizarras o transparencias. Existen varias soluciones que permiten la creación de presentaciones electrónicas en línea, cada una con sus características particulares. A continuación se muestra un análisis realizado de las soluciones similares existentes.

Prezi

Es una aplicación multimedia para la creación de presentaciones de una manera dinámica desde internet. Ofrece la posibilidad de organizar la información en forma de un esquema y exponerlo con libertad sin la

secuencia de diapositivas. A diferencia del modelo lineal con que se desarrollan las presentaciones tradicionales donde se mueven desde un concepto a otro, Prezi estimula a los estudiantes a que identifiquen patrones, comparaciones, relaciones y diferencias entre la información; representado la información como una especie de mapa conceptual (Conboy *et al*., 2012). La aplicación permite trabajar en línea con todas sus herramientas de forma gratuita, pero tiene la limitante de que entonces el autor no tiene la posibilidad de definir la privacidad de su presentación; esta opción solo está definida mediante modalidades de pago que además ofrecen una mayor capacidad de almacenamiento, entre otras ventajas.

Google Slides

Slides es parte de la suite de aplicaciones en la nube de la empresa Google, que funcionan a través de Google Drive para permitir la visualización y edición de documentos, presentaciones, hojas de cálculo y otros tipos de formato de trabajo. Siendo una alternativa viable a Microsoft PowerPoint o Keynote, Slides cuenta con varias funcionalidades que la hacen atractiva e idónea para la realización de presentaciones (AETecno, 2013).

Entre las funcionalidades que más destacan se encuentra la posibilidad de utilizar diferentes plantillas, modificarlas y crear nuevas; además de dar la posibilidad de compartirlas con otros usuarios a través de Google+. Además cuenta con la posibilidad de ver las presentaciones a pantalla completa, lo cual mejora la visualización de estas y ofrece la posibilidad de trabajar con la aplicación sin tener conexión y una vez restablecida la conexión, los cambios automáticamente se actualizan y se guardan en la nube. Esta última funcionalidad todavía se encuentra restringida al navegador Chrome, perteneciente a la propia empresa. Para la utilización de la aplicación es necesario crearse una cuenta en Google+ y una vez autenticado el usuario puede compartir su presentación para que otras personas la visualicen o trabajen juntos en su confección. Para la creación de la cuenta existe la limitante de que el usuario debe introducir un número de teléfono celular válido para que esta pueda activarse, lo que dificulta su utilización en países subdesarrollados como Cuba.

Keynote

Es una aplicación web de presentación que forma parte de la suite ofimática de aplicaciones creada por Apple llamada iWork. Entre sus principales características sobresale la posibilidad de utilizar temas ya definidos y de incluir en la presentación transiciones y diapositivas 3D. Permite exportar las

presentaciones a diferentes extensiones, soporta la inclusión de varios formatos de archivos multimedia y permite la integración con los dispositivos móviles de Apple (Wood, 2012). Para su utilización es necesario contar con una cuenta de Apple, ya que no existe una versión libre para su uso, lo cual limita su utilización a solo los clientes de dicha empresa. Para poder ser considerado cliente de la empresa se debe haber comprado alguno de los dispositivos que ofrece, los que tienen precios elevados, o adquirir una cuenta, también a un alto precio.

Microsoft Office 365

Constituye la suite ofimática de la empresa Microsoft en la nube, lanzada en el 2011, incluye a la mayor parte de las herramientas de su paquete ofimático para escritorio. Entre sus características más llamativas se encuentran la integración con dispositivos móviles, la posibilidad de compartir documentos con otros usuarios y observar en tiempo real como son editados por varios usuarios como si de una red social se tratara, el usuario podrá acceder al servicio con su cuenta desde el navegador y podrá almacenar sus documentos e información mediante el servicio SkyDrive de Microsoft (Katzer y Crawford, 2013). Como se mencionó, para poder hacer uso del servicio es necesario poseer una cuenta que se obtiene previa compra y suscripción con las diferentes ofertas existentes pues es un producto privativo que no tiene ninguna versión gratuita.

Módulo Presentaciones Web para Moodle

El Módulo Presentaciones Web está destinado para la plataforma de teleformación Moodle en su versión 2.3.x. Creado en la UCI para utilizarse en la plataforma Moodle, permite insertar diferentes elementos como texto e imágenes y descargar la presentación en formato PDF. Pese hacer una solución orientada a al sector educativo, presenta varias desventajas como es el hecho de estar confeccionada utilizando la tecnología Adobe Flash la cual es dependiente de *plugins* externos para poder funcionar en un navegador web. También está diseñada para la plataforma Moodle, por lo que su uso se restringe a tener acceso a dicha plataforma por el usuario que necesite utilizarla y sus funcionalidades son limitadas como es el hecho de no poder insertar formas.

Análisis de las soluciones

Después de haber analizado las soluciones similares para la creación de presentaciones electrónicas en línea, se pudo llegar a la conclusión de que estas son en su mayoría aplicaciones privativas, con licencias costosas. Además su utilización se ve restringida a sus plataformas privadas, por lo que no es

recomendable su utilización teniendo en cuenta que Cuba se encuentra abogando por la soberanía tecnológica y también por los problemas de acceso existentes producto de las restricciones del bloqueo económico[7]. También el análisis de sus funcionalidades arrojó que no permiten exportar una presentación electrónica para que pueda ser visualizada en un sitio web sin dependencia alguna de la plataforma en la que se confeccionó.

Por estas razones las soluciones estudiadas, serán utilizadas solo como referencia para el desarrollo de la solución propuesta. Mediante su análisis se pudieron definir las principales tendencias en cuanto al diseño y las funcionalidades básicas con que debe contar la solución. Además se llegó a la conclusión de que la solución a desarrollar debe ser multiplataforma, de código abierto y licencia libre, apoyando de esta forma la soberanía tecnológica en el país.

1.4 Ambiente de desarrollo

Obtener un software con la calidad requerida, implica la utilización de metodologías, tecnologías y herramientas que permitan conformar el área de trabajo. Para comenzar el desarrollo de la herramienta para la creación de presentaciones electrónicas en línea se realiza un estudio de las tecnologías y tendencias de metodologías, lenguajes y herramientas de modelado, *framework*, servidores web y lenguajes de desarrollo con el objetivo de definir el ambiente de desarrollo de la misma.

1.4.1 Metodologías de desarrollo de software

"*Una metodología es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información. Una metodología está formada por fases, cada una de las cuales se puede dividir en sub-fases, que guiarán a los desarrolladores de sistemas a elegir las técnicas más apropiadas en cada momento del proyecto y también a planificarlo, gestionarlo, controlarlo y evaluarlo*" (Avison y Fitzgerald, 1995).

Básicamente, las metodologías de desarrollo de software se pueden dividir en dos grupos: metodologías pesadas (tradicionales) y metodologías ágiles. El esquema tradicional para el desarrollo de software ha demostrado ser efectivo y necesario en proyectos de gran tamaño (respecto a tiempo y recursos), donde

[7] El bloqueo económico es el cerco comercial, económico y financiero impuesto por Estados Unidos a Cuba desde el 7 de febrero de 1962. Fue convertido en ley en 1992 y 1995. Es uno de los más duraderos de la historia, condenado quince veces por las Naciones Unidas (Cubadebate, 20014).

por lo general se exige un alto grado de ceremonia en el proceso. Sin embargo, este enfoque no resulta ser el más adecuado para muchos de los proyectos actuales en los que el entorno del sistema es muy cambiante, y en donde se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad (Canos, Letelier y Penadés, 2003). Por lo anteriormente mencionado, las características propias de la solución y el ambiente de desarrollo (dígase equipo de desarrollo, tiempo de desarrollo, entre otros aspectos), se decide optar por la aplicación de una metodología ágil.

Metodologías ágiles

Estas metodologías están orientadas a proyectos pequeños y constituyen una solución a la medida para ese entorno, aportando una elevada simplificación que a pesar de ello no renuncia a las prácticas esenciales para asegurar la calidad del producto. Además de que permiten a los equipos de desarrollo centrarse en el software mismo, en vez de en su diseño y documentación (Sommerville, 2005).

Dentro de las metodologías ágiles se incluyen: *Extreme Programming* (XP), SCRUM, *Crystal Methodologies*, XBreed, *Adaptive Software Development* (ASD), *Dynamic Systems Development Method* (DSDM), entre otras (Canós, Letelier y Penadés, 2003).

Para el desarrollo de la presente investigación se analizaron las metodologías ágiles SCRUM y XP, debido a que ambas promueven el trabajo en equipo, fomentan la interacción sistemática entre el cliente y equipo de desarrollo y están suficientemente documentadas.

SCRUM

La metodología ágil SCRUM desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle, se basa fundamentalmente en entregas iterativas con ciclos de corta duración (30 días). A estos ciclos la metodología SCRUM los denomina *sprint* y en ellos se desarrollan las mejoras de los productos que se hayan planificado. La metodología establece una reunión al inicio de cada *sprint* para determinar el trabajo que se va a realizar, otra al final para evaluar el resultado, y revisiones diarias que realiza el equipo en su auto-gestión (Palacio, 2008).

SCRUM es una metodología ágil principalmente indicada para proyectos con un rápido cambio de requisitos, dispone de herramientas para la gestión de cada una de sus fases, y es ideal para pequeños equipos de diez o menos miembros; sin embargo no indica ni provee ninguna práctica concreta para el desarrollo de software. Según Carvajal (2008), en una comparativa entre las metodologías ágiles SCRUM

y XP, indica que la metodología XP está más enfocada a presentar diferentes prácticas y SCRUM a la gestión de proyectos. Además, a pesar de las ventajas que pudiera suponer la aplicación de la metodología ágil SCRUM, es necesario señalar que los roles que esta propone (propietario del producto, *SCRUM Master*, equipo de desarrollo, cliente y gestor) no están visiblemente representados en el presente proyecto (Palacio, 2007).

Extreme Programming (XP)

La Programación Extrema o *Extreme Programming* (XP) es una metodología de desarrollo de software inscrita en el contexto ágil y una de las más agresivas en cuanto a velocidad se refiere. Como proceso de software diferente al convencional, nace de la mano de Kent Beck en el año 1996 como una nueva manera de encarar proyectos de software proponiendo una metodología basada esencialmente en la simplicidad y agilidad (Beck y Andres,2004).

XP resalta una serie de valores y principios que se deben de tener en cuenta para ser llevados a la práctica durante el desarrollo del proyecto. Más que una metodología XP se considera una disciplina sostenida por valores y principios propios de las metodologías ágiles. Existen cinco valores puntuales en los que se basa XP (Shore y Warden, 2008):

Coraje: el equipo de desarrollo debe estar preparado para tomar las decisiones correctas, no importa cuán difíciles sean estas, y para transmitir a las partes interesadas cualquier inquietud o problema que encuentre durante la realización del proyecto.

Comunicación: es muy importante que exista un ambiente de colaboración y comunicación en el interior del equipo de desarrollo, así como en la interacción de este con el cliente. En XP la interacción con el cliente es tan estrecha, que es considerado parte del equipo de desarrollo.

Simplicidad: se simplifica el diseño para agilizar el desarrollo, facilitar el mantenimiento y descartar las ideas que realmente no se necesiten; solo se desarrolla lo que el cliente demanda, de la forma más sencilla.

Retroalimentación: se presenta desde el comienzo del proyecto, ayuda a encaminarlo y darle forma. Esta se presenta en los dos sentidos, por parte del equipo de trabajo hacia el cliente, con el fin de brindarle información sobre la evolución del sistema, y desde el cliente hacia el equipo en los aportes a la construcción del proyecto.

Respeto: los miembros del equipo de desarrollo deben tratarse entre ellos y los demás con dignidad, reconociendo la experiencia de todos y su deseo de éxito.

El ciclo de vida de la metodología XP está compuesto de seis fases según Carvajal (2008):

Exploración: los clientes describen en las historias de usuario (HU) las características que el sistema debe poseer y que serán incluidas en la primera versión.

Planificación: se establece la prioridad de las HU y se acuerda el contenido de la primera entrega del proyecto, así como su estimación temporal.

Iteraciones: se decide qué historias se realizan en cada iteración, así como las pruebas funcionales ejecutadas al final de cada iteración.

Producción: se llevan a cabo un conjunto de pruebas extras, de rendimiento y funcionamiento que son necesarias antes de entregar el producto.

Mantenimiento: una vez sea liberada la primera versión a los usuarios, el sistema se debe mantener en el entorno de producción siempre y cuando aún hayan iteraciones en fase de producción.

Cierre del proyecto: ya no hay más HU que deban ser implementadas y las necesidades del cliente han sido satisfechas.

En XP, las iteraciones son relativamente cortas ya que entre más rápido se entregue un resultado al cliente, más retroalimentación se obtiene y esto va a representar una mejor calidad del producto a largo plazo. De manera general, esta característica de la metodología XP es relevante y resulta conveniente para el desarrollo de la presente solución propuesta.

Selección de la metodología de desarrollo

Luego del análisis anterior de las metodologías ágiles para el desarrollo de software y teniendo en cuenta que el presente proyecto es pequeño, con un equipo de desarrollo reducido y el corto tiempo con que se cuenta para el desarrollo de la solución, se decide elegir a la metodología XP. Al igual que SCRUM el desarrollo en XP es iterativo e incremental, sin embargo XP no regula las iteraciones en *sprint* de un tiempo fijo como SCRUM permitiendo mayor flexibilidad al equipo.

Además la utilización de XP permite seguir algunas de las buenas prácticas que esta metodología propone, como es el hecho de que el cliente esté siempre disponible para asesorar a los programadores,

ser consultado y observar los posibles cambios; aspecto que garantiza gran parte del éxito del proyecto. También se busca hacer frente al ambiente cambiante que se presenta en los requerimientos de la aplicación, el empleo de un estándar de codificación y un horario de trabajo de un máximo de 40 horas por semana sin trabajar horas extras. Estas prácticas se ajustan perfectamente a las condiciones reales del presente proyecto. Es importante reflejar también el hecho de que el desarrollador cuenta con experiencia en la utilización de esta metodología, permitiendo la aplicación de buenas prácticas adquiridas durante su uso que contribuirán a una mejor planificación y estimación del proyecto.

Teniendo en cuenta el ciclo de vida de la metodología XP propuesto por Beck (1999) y las fases definidas por Echeverry y Delgado (2007) se utilizan las siguientes fases:

Planificación: se comienza a interactuar con el cliente y el resto del grupo de desarrollo para definir los requerimientos de la solución. Serán definidas en esta fase las HU, la estimación de esfuerzo por HU, el plan de iteraciones y el plan de entregas.

Diseño: esta tarea es permanente durante la vida del proyecto partiendo de un diseño inicial que va siendo corregido y mejorado en el transcurso del proyecto. Se definirá en esta fase el prototipo no funcional de interfaz de usuario, la arquitectura de software con los patrones utilizados y las tarjetas CRC (Clase, Responsabilidad y Colaboración).

Implementación: la implementación es un proceso que se realiza en forma paralela con el diseño. En esta fase se definirá el estándar de codificación a emplear, las tareas de ingeniería generadas para cada iteración y un diagrama de componentes.

Pruebas: el buen uso de las pruebas es un elemento importante en XP. Sólo se deberá liberar una nueva versión si esta ha pasado satisfactoriamente por la totalidad de las pruebas. En caso contrario se empleará el resultado obtenido para identificar el error y solucionarlo. En esta fase se describen las pruebas unitarias y de aceptación realizadas y los resultados arrojados.

1.4.2 Lenguaje de modelado

Desde los inicios de la informática se han estado utilizando distintas formas de representar los diseños de una forma personal o con algún modelo gráfico. La falta de estandarización en la manera de representar gráficamente un modelo impedía que los diseños gráficos realizados se pudieran compartir fácilmente entre distintos diseñadores (Rumbaugh, Jacobson y Booch, 2007). Se necesitaba por tanto un lenguaje no

sólo para comunicar las ideas a otros desarrolladores sino también para servir de apoyo en los procesos de análisis de un problema. Con este objetivo se creó el Lenguaje Unificado de Modelado (UML: *Unified Modeling Language*), que se ha convertido en ese estándar tan ansiado para representar y modelar la información con la que se trabaja en las fases de análisis y, especialmente, de diseño.

Lenguaje Unificado de Modelado (UML)

El lenguaje UML tiene una notación gráfica muy expresiva que permite representar en mayor o menor medida todas las fases de un proyecto informático: desde el análisis con los casos de uso, el diseño con los diagramas de clases, objetos, entre otros, hasta la implementación y configuración con los diagramas de despliegue.

UML ayuda al usuario a entender la realidad de la tecnología y la posibilidad de que reflexione antes de invertir y gastar grandes cantidades en proyectos que no estén seguros en su desarrollo, reduciendo el coste y el tiempo empleado en la construcción de las piezas que constituirán el modelo.

Este lenguaje incluye conceptos semánticos, notación y principios generales. Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo (Rumbaugh, Jacobson y Booch, 2006), esto es favorable ya que la metodología XP posee esta característica.

En sentido general mediante este lenguaje se podrá especificar todas las decisiones tomadas en el análisis, diseño e implementación de la solución, construyéndose de esta forma modelos completos y permitirá documentar todos los artefactos necesarios del proceso de desarrollo. Es importante aclarar que pese a que la metodología XP no define la generación de ningún diagrama en específico, se genera este tipo de artefacto para lograr un mejor entendimiento de la arquitectura de la solución. Se seleccionó la versión UML 2.0 como lenguaje de modelado debido a que está ampliamente documentado, facilitando su utilización.

1.4.3 Herramientas CASE para el modelado UML

Con el auge que ha adquirido UML (Lenguaje Unificado de Modelado - *Unified Modeling Languaje*) en los últimos años, se han creado numerosas herramientas de modelado de diagramas UML, donde seleccionar

la herramienta CASE (Ingeniería de Software Asistida por Computadora – *Computer Aided Software Engineering*).

Se puede definir a las herramientas CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software (Davis y Mata, 1992).

Visual Paradigm

Es una herramienta CASE que utiliza como lenguaje de modelado UML y que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, implementación, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Se integra con varios IDE como Eclipse y NetBeans lo que agiliza el desarrollo (Cabrera y Pompa, 2012). Se utiliza Visual Paradigm en su versión 5 como herramienta CASE, además de las razones anteriormente mencionadas, por el hecho de ser un herramienta multiplataforma y fácil de usar.

1.4.4 Lenguajes utilizados para el desarrollo

La Web es, básicamente, un entorno cliente/servidor con tres componentes: el lado del cliente, el lado del servidor y la red (Powell, 2001). Existen en la actualidad lenguajes de programación ampliamente usados para el desarrollo de aplicaciones web, los que han ido surgiendo y evolucionando según las necesidades. El tiempo de desarrollo en la actualidad es crítico, tanto por razones de marketing como por límites en el presupuesto y los recursos, de ahí la importancia de seleccionar las tecnologías adecuadas para la solución a desarrollar (Cáceres y Marcos, 2007).

Los lenguajes de programación para el desarrollo de aplicaciones web pueden clasificarse en dos grupos:

Lenguajes del lado del cliente: se refiere a los que se ejecutan en el cliente web, o sea, en el navegador web del usuario. Permiten confeccionar la presentación de la aplicación web, a la que se le pueden acoplar controles de validación de datos, para evitar la validación de los mismos en el servidor (Luján, 2002).

Lenguajes del lado del servidor: son aquellos que se ejecutan en el servidor, y envían la información a mostrar al cliente. Permiten crear sitios web dinámicos con acceso a bases de datos, susceptibles de personalización y dotados de información en tiempo real (Welling y Thomson, 2009).

Por las características propias de la solución no existe una distinción entre los lenguajes del lado del cliente y los del lado del servidor, puesto que el lenguaje utilizado para el trabajo en estos grupos es el mismo, JavaScript, considerado un lenguaje del cliente y utilizado también como código del lado del servidor.

HTML5

HTML5 es una nueva versión del antiguo lenguaje de etiquetas HTML (*Hypertext Markup Language*), constituyendo un nuevo concepto para la construcción de sitios web y aplicaciones en una era que combina dispositivos móviles, computación en la nube y trabajos en red.

Provee básicamente tres características: estructura, estilo y funcionalidad. La estructura es parte esencial de un documento. La misma proporciona los elementos necesarios para ubicar contenido estático o dinámico, y es también una plataforma básica para aplicaciones (Gauchat, 2012). Esta nueva versión de HTML incorpora etiquetas que facilitan la creación de aplicaciones web, entre las que sobresale la etiqueta <canvas>; muy utilizada para el trabajo con imágenes y animaciones.

El elemento *Canvas* es parte de HTML5 y permite la generación dinámica de formas 2D[8], imágenes de mapa de bits y animaciones en una página web mediante JS; lenguaje dominado por el desarrollador. Una vez creado el gráfico se pueden programar acciones para que el usuario interactúe con él, permitiendo crear animaciones, aplicaciones y juegos (Fulton y Fulton, 2011).

Se trata de un modelo de procedimiento de nivel bajo, el que actualiza un mapa de bits y no tiene una gráfica de escena integrada. Es decir, *Canvas* no crea objetos vectoriales al estilo de otros entornos como SVG (*Scalable Vector Graphics*), sino mapas de bits como una imagen fotográfica. *Canvas* presenta otra característica que lo convierte en la opción elegida, como es el hecho de poseer un mejor comportamiento a medida que la cantidad de objetos en la pantalla aumentan, requisito indispensable a tener en cuenta en la presente solución (Microsoft, 2012). Además como el desarrollador tiene experiencia con el uso de esta

[8] Se refiere al proceso de poder programar dinámicamente elementos en dos dimensiones que son generados, como imágenes.

tecnología y existen una serie de recursos como *framework*, que facilitan el trabajo y poseen una amplia comunidad de respaldo; se decide utilizar el elemento *Canvas*.

CSS3

CSS que significa *Cascading Style Sheets* u Hojas de Estilo en Cascada, es el lenguaje utilizado para darle forma y aplicar el diseño al contenido HTML y XML que forman una página o aplicación web. Constituye la mejor forma de separar contenidos de su presentación y es imprescindible para la creación de páginas web complejas, ya que se puede modificar el diseño sin tener que transformar el código HTML o XML de manera constante, agilizando considerablemente así el proceso de cambios (Schmitt *et al*., 2005).

La utilización de CSS en su versión 3 (CSS3) permite una mejora en cuanto al diseño y el trabajo con los componentes que conforman la solución, obteniéndose una interfaz atractiva para el usuario. Se gana en el uso de animaciones, organización de los elementos y simplificación del código, cumpliendo con las tendencias actuales en el desarrollo de la web (McFarland, 2013).

JavaScript 1.8.5

Es un lenguaje de programación ligero e interpretado, orientado a objetos, muy utilizado en la creación de páginas web dinámicas (Issi, 2002). El manejo de objetos facilita la programación de páginas interactivas, al responder a eventos en tiempo real, permitiendo ganar en interactividad dinámica con el usuario.

Técnicamente, JavaScript (JS) es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. Eso significa que los programas escritos con JS se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios (Eguíluz, 2009). En el desarrollo de la solución es ampliamente utilizado para la manipulación del área de mapa de bits de la etiqueta <canvas> de HTML5.

JS se ha utilizado ampliamente en los niveles del lado del cliente de aplicaciones web (es decir, en el código que se ejecuta en el navegador del usuario) durante varios años, con el objetivo de proveer una mayor experiencia de usuario. Pero en los últimos años, ha habido un aumento en el interés de JS no solo para el lado del cliente sino también como código del lado del servidor. En la actualidad existen servidores web (por ejemplo, Node.js) que utilizan como lenguaje a JS, debido a que es adecuado para la programación basada en eventos (Means, 2012).

1.4.5 Framework y bibliotecas

Teniendo en cuenta los lenguajes a utilizar para el desarrollo se definen una serie de *framework* y bibliotecas que agilizan el desarrollo de la solución propuesta. El término *framework* se refiere a una estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. Enfocado en el desarrollo web se puede decir que constituye un conjunto de clases que cooperan orientadas a un diseño reutilizable, formando una infraestructura que agiliza y facilita el desarrollo de aplicaciones web (Frederick, Ramsay y Blades, 2008).

Los objetivos principales que persigue un *framework* son: acelerar el proceso de desarrollo, reutilizar el código ya existente y promover buenas prácticas de desarrollo con el uso de patrones.

Una biblioteca (en inglés *library*) en términos informáticos, constituye un conjunto de rutinas almacenadas en un archivo. Cada conjunto de instrucciones dentro de una biblioteca tiene un nombre, y cada uno realiza una tarea diferente, a menudo muy específica. Este tipo de conjunto de instrucciones simplifica el trabajo y evita duplicar el esfuerzo cada vez que sea necesario realizar una tarea específica (Castledine y Sharkie, 2010).

jQuery 2.0

En la capa de presentación se utilizará la biblioteca jQuery en su versión 2.0. jQuery es una biblioteca de JS rápida y concisa que simplifica el trabajo con un documento HTML, el manejo de eventos, animaciones e interacciones Ajax para un desarrollo web rápido (Bibeault y Katz, 2008). Estas características posibilitan que los contenidos se puedan mostrar y se interactúe con estos de forma dinámica, pudiéndose usar este *framework* para apoyar el cumplimiento de las funcionalidades necesarias.

Modernizr 2.0.6

Con la utilización de lenguajes como HTML5 y CSS3, los desarrolladores necesitan conocer la compatibilidad de los elementos que incorporan estos lenguajes con el soporte de los navegadores en dónde se despliegan. Modernizr es una pequeña biblioteca JS que detecta la capacidad de los navegadores de hacer uso de esta nueva generación de tecnologías web (Watson, 2012). Detectando el soporte del navegador en el que se está desplegando, le permite al desarrollador tomar medidas si el elemento a utilizar no es soportado. Entre las medidas que se pueden tomar está la de poner como alternativa una especificación que sea soportada por el navegador en cuestión o de incorporar el soporte a

ese elemento mediante *polyfills*[9]. Por las necesidades de que la solución se pueda visualizar correctamente en diferentes versiones de navegadores se escoge esta biblioteca.

JSColor 1.4.2

JSColor es una biblioteca simple y fácil de usar como selector de color. Ha sido utilizado en proyectos como WordPress y entre sus ventajas se encuentra el hecho de poder instalarse de forma rápida y fácil, tener una apariencia adaptable según las necesidades del desarrollador, no necesitar de ningún *framework*, no utilizar ventas emergentes y poseer una amplia compatibilidad entre los diferentes navegadores (JSColor, 2013). Por estas características, además del hecho de poseer una licencia LGPL es utilizada en la solución.

Node.js 0.10.18

Node.js es un *framework* y entorno de ejecución, con entrada/salida de datos en una arquitectura orientada a eventos y por lo tanto asíncrona que se ejecuta sobre el intérprete de JS creado por Google, V8. Permite construir aplicaciones web rápidas y altamente escalables, es muy utilizado en el trabajo con datos en tiempo real (Teixeira, 2013). Similar a servidores web como Apache, que se utiliza para el trabajo con PHP, Node.js surge como una nueva forma de aprovechar la experiencia con JS, esta vez, desde el lado del servidor, permitiendo generar software de manera sencilla con recursos asíncronos y orientados a eventos, creando un solo hilo de procesos para todos los clientes, lo que hace que el servidor soporte muchas más conexiones. Producto de su arquitectura orientada a eventos, el uso de memoria es bajo, el rendimiento es alto y el modelo de programación es más simple.

Este *framework* incorpora una serie de módulos predefinidos y además es posible incluirle otros desarrollados por terceros que incorporan las más variadas funcionalidades, facilitando el desarrollo de aplicaciones web. Entre los *framework* que Node.js incorpora como módulos y que fueron seleccionados se encuentran:

- **Express 4.0:** es un *framework* de desarrollo de aplicaciones web minimalista y flexible para Node.js, que provee una serie de características que permiten construir una aplicación web basada en el patrón Modelo-Vista-Controlador (Muñoz, 2013). Entre sus características cabe destacar el robusto sistema de enrutamiento de peticiones que posee y el soporte para generar páginas HTML dinámicas a partir de plantillas con capacidad de utilizar varios motores de renderizado de vistas.

[9] En el desarrollo web, un *polyfill* (o *polyfiller*) es código descargable que proporciona servicios que no están integrados en un navegador web.

Bien documentado y con una gran popularidad, cuenta con una amplia comunidad que lo utiliza y mejora constantemente.

- **Jade 1.3.1:** es un motor de plantilla que simplifica la sintaxis de HTML y acelera el proceso de desarrollo con Node.js, además de ser utilizado por defecto por el *framework* Express (Means, 2012). Utiliza un lenguaje en dónde se define cómo utilizar las etiquetas HTML entre otros parámetros.

Por estas ventajas, el hecho de utilizar una licencia MIT (*Massachusetts Institute of Technology*) y funcionar además con el lenguaje seleccionado JS, con el que se encuentra familiarizado el desarrollador, se selecciona a Node.js para el desarrollo de la solución propuesta.

Selección del framework para el trabajo con el elemento *Canvas*

La utilización del elemento *Canvas* en los últimos años se ha incrementado, debido a su uso para la realización de animaciones y juegos para la Web. Pero trabajar con este elemento puede ser complejo si no se posee la experiencia suficiente, por lo que se han creado una serie de recursos que facilitan su manejo como EaselJS, Paper.js, Fabric.js y Processing.js entre los más destacados. Debido a la amplia documentación existente y comunidad de desarrollo con que cuentan, a continuación se analizan solamente a EaselJS y Paper.js.

EaselJS es un *framework*, bajo licencia MIT, que proporciona un conjunto de soluciones para trabajar con gráficos e interactuar con el elemento *Canvas*. Entre sus características se encuentra el hecho de que provee una API (del inglés *Application Programming Interface*) que le resulta familiar a los desarrolladores de Adobe Flash, pero que además abarca las propiedades de JS. Provee una lista jerárquica, igual que Adobe Flash, para poder controlar los elementos que se dibujan. Además de un conjunto de clases de ayuda que hacen el trabajo con el elemento *Canvas* mucho más fácil (CreateJS, 2014).

Paper.js es un *framework* de código libre para la generación de gráficos vectoriales, creado por Jürg Lehni y Jonathan Puckey, que explota las potencialidades del elemento *Canvas* de HTML5. Fácil de aprender por principiantes y con muchas opciones para explotar por parte de usuarios intermedios y avanzados; cuenta con una amplia comunidad activa de programadores, una buena documentación y es distribuido bajo la licencia MIT. Entre sus ventajas sobresale la posibilidad de utilizar diferentes funcionalidades para crear y trabajar con gráficos vectoriales y curvas de Bézier; en una interfaz de programación coherente. La

manipulación y el dibujo de los elementos gráficos es automática y optimizada, lo que permite construir y modificar sus elementos y estilos de forma fácil, y delegar la tarea de dibujo al *framework* (Puckey, 2013).

EaselJS y Paper.js permiten la creación (dibujo), animación y edición básica (trasladar, escalar y rotar) de cualquier tipo de elemento, así como la interacción con estos. En el caso de Paper.js se brinda un mayor volumen de información acerca de los elementos creados, permitiendo un mejor nivel de detalle para su edición. Mientras que EaselJS se centra más en la creación y animación de los elementos. Además Paper.js brinda la capacidad de poder exportar los elementos creados en formato SVG y JSON (*JavaScript Object Notation*), facilitando la tarea de salvar el trabajo realizado, lo cual es de interés para el desarrollador.

Debido a las características mencionadas de Paper.js se decide emplearlo por el desarrollador en su versión 0.9.16 para el trabajo con elemento *Canvas*. Además, el desarrollador tiene experiencia con el trabajo mediante este *framework*, lo que minimiza la curva de aprendizaje y acelera el proceso de desarrollo.

1.4.6 Entorno de desarrollo integrado

Un Entorno de Desarrollo Integrado o IDE (en inglés *Integrated Development Environment*) es un programa informático que agrupa diversas herramientas de programación para facilitar la tarea al programador y lograr mayor rapidez a la hora de desarrollar. (Burd, 2005).

NetBeans

NetBeans en su versión 8.0, como IDE seleccionado por la presente investigación, es un entorno de desarrollo integrado libre, sin restricciones de uso, con una base sólida para desarrollar aplicaciones complejas basadas en un enfoque modular (Böck, 2011). Pensado para el lenguaje de programación Java fundamentalmente, puede ser utilizado para el trabajo con otros lenguajes, existiendo además un número importante de módulos para extender el IDE NetBeans.

Entre las características que presenta esta herramienta se encuentran:

- Provee un esqueleto para organizar el código fuente, el editor conjuntamente integra los lenguajes como HTML, JS y CSS.

- Ofrece la línea de comandos de depuración: La salida del programa PHP aparece en una pantalla de línea de comandos en el IDE de sí mismo y se puede inspeccionar el código HTML generado sin tener que cambiar a un navegador.
- Posee un sistema para examinar todo los directorios de cada proyecto, haciendo reconocimiento y carga de clases, métodos y objetos, para acelerar la programación.
- Permite realizar aplicaciones utilizando frameworks, haciendo el trabajo más ágil.
- Su editor, sobre todo el de PHP, es mucho más ágil y a la vez robusto, contiene más ayuda en línea, reconocimiento de sintaxis y todo lo que provee las versiones de PHP.
- Integra muy bien la utilización Xdebug, permitiendo inspeccionar y examinar cada variable local, establecer puntos de interrupción y evaluar el código.

Además de las características anteriormente mencionadas se tuvieron en cuenta otras que facilitan el desarrollo de la solución propuesta; como es el caso de permitir autocompletamiento para el lenguaje de programación JS, el remarcado de sintaxis soporta los lenguajes utilizados, se integra correctamente con la herramienta CASE Visual Paradigm y el desarrollador cuenta con experiencia en su uso.

1.4.7 Herramienta para el prototipado

Los prototipos constituyen una herramienta muy útil para hacer participar a los usuarios y evaluar el sistema desde las primeras fases. Pencil es un *plugin* del navegador Mozilla Firefox que permite crear un prototipo no funcional de una aplicación y que puede ser usado tanto por clientes como por desarrolladores (Pencil Project, 2012). Permite acordar con el cliente aspectos claves de la solución a desarrollar, como la distribución general de los elementos, sus jerarquías y la navegación de los mismos. Debido a las características de Pencil que lo hacen fácil de utilizar, el hecho de ser libre y estar soportado por una comunidad que constantemente lo actualiza; es utilizado para hacer el prototipo de la solución propuesta.

1.5 Conclusiones parciales del capítulo

Como parte del estudio y análisis realizado del objeto de investigación, apoyado en los métodos de la investigación definidos, se pudo construir el marco teórico-conceptual que soporta la investigación. La bibliografía consultada permitió ganar en claridad acerca del objeto de estudio y el nivel de trabajos

existentes acerca del tema; pudiendo determinar el auge y la importancia que tienen las aplicaciones web en la actualidad. El análisis de las soluciones similares existentes arrojó que pese a que son herramientas profesionales para la creación de presentaciones electrónicas en línea, su utilización se ve limitada para la UCI debido a los altos precios de sus licencias privativas y a las restricciones comerciales existentes; además la dependencia a estas plataformas no responde a la soberanía tecnológica por la que está abogando el país. El análisis de estas soluciones permitió además definir las funcionalidades básicas con que debe contar la solución y se llegó a la conclusión de que debe ser multiplataforma, de código abierto y licencia libre, apoyando de esta forma la soberanía tecnológica en el país.

Se selecciona la metodología XP, pues define una guía y prácticas acordes con las condiciones en las que desarrolla la solución propuesta. Los lenguajes de programación a utilizar producto del estudio realizado fueron definidos, destacándose el uso de JS 1.8.5 para el trabajo con el elemento *Canvas*; a partir de esta selección se escogieron una serie de *framework* y bibliotecas que agilicen el trabajo como Paper.js. Para trabajar con estos lenguajes de programación es escogido el IDE NetBeans 8.0 y Visual Paradigm 5.0 para el modelado de los artefactos de ingeniería; sacando provecho de la integración entre estos programas.

Capítulo II: Descripción de la solución propuesta

2.1 Introducción

La metodología de desarrollo XP define un conjunto de prácticas y técnicas que le confieren al equipo de desarrollo una guía por la cual orientarse durante el tiempo de confección de la solución propuesta. El uso adecuado de estas técnicas es lo que permite lograr entre otras cosas una planificación, que de cumplirse, tiene como resultado la entrega del producto en tiempo.

Algunos de los principios, prácticas y técnicas que sirven de guía para el equipo de desarrollo y que se describen en este capítulo son: las historias de usuario (HU), el plan de iteraciones, plan de entrega y las tarjetas CRC. Se realiza el diseño de la arquitectura de software dónde quedan definidos los patrones arquitectónicos y de diseño. Además se describen las funcionalidades y se generarán los artefactos propuestos por la metodología seleccionada, permitiendo ganar en claridad y organización.

2.2 Descripción de la solución propuesta

Como propuesta de solución de esta investigación se plantea el desarrollo de una aplicación multiplataforma, de código abierto y licencia libre para la creación de presentaciones electrónicas en línea. Producto a los requerimientos obtenidos de las entrevistas realizadas a los administradores de diferentes sitios web de la UCI y del análisis de las soluciones similares se decidió que la solución propuesta fuera una aplicación web. Esta contaría con las ventajas de poder ser accedida desde cualquier lugar de la UCI y que las actualizaciones producto de versiones posteriores o la corrección de problemas, solo se tienen que realizar a la aplicación publicada para que todos los usuarios puedan beneficiarse al mismo tiempo.

La aplicación web contará con una interfaz visual sencilla en dónde estarán presentes las funcionalidades básicas necesarias para la creación de una presentación electrónica. El usuario podrá insertar diferentes contenidos como texto, formas o imágenes en las diapositivas de la presentación, pudiendo editar las propiedades de estos elementos. También podrá interactuar con las diapositivas tanto para crear nuevas, eliminarlas y hacer uso de otras opciones disponibles, como para editar sus propiedades. Las presentaciones electrónicas elaboradas pueden ser guardadas para posteriormente continuar trabajando con ellas o visualizarlas, y también pueden ser exportadas para ser incluidas en un sitio web para su visualización, sin dependencia alguna de recursos externos como *plugins*. La solución estará dividida en cinco áreas de trabajo (ver **Figura 1**) que a continuación se describen y las opciones que brindan.

Panel de herramientas

En el área superior estarán localizadas las opciones desde dónde el usuario podrá seleccionar los objetos a insertar en la diapositiva activa, como formas, texto o imágenes. Además se encuentran las herramientas para la transformación de los elementos insertados en la diapositiva activa y el manejo de estos elementos. Desde esta área el usuario podrá también visualizar la presentación en pantalla completa.

Barra de menús

En el área superior izquierda estarán las opciones para la gestión de las presentaciones electrónicas como crear una nueva presentación, guardarla, abrirla o exportarla. También están presentes las opciones de copiar, pegar y cortar un objeto insertado en la diapositiva activa.

Panel de propiedades

En el área derecha se encontrarán las propiedades relacionadas con el objeto seleccionado de la diapositiva activa o de la diapositiva seleccionada. El usuario podrá interactuar con las propiedades que se le presentan pudiendo apreciar en el mismo momento que se efectúa un cambio como queda este reflejado en el objeto o diapositiva seleccionada.

Área de trabajo

En el área del centro se encuentra ubicado el área de trabajo de la diapositiva activa, ahí el usuario podrá insertar diferentes objetos e interactuar con ellos desplazándolos o editando sus propiedades.

Panel de las diapositivas

En el área izquierda se encuentran las vistas previas desde donde el usuario podrá gestionar las diapositivas de la presentación electrónica creando, eliminando, duplicando o desplazando una diapositiva.

Barra de estado

En el área inferior estará localizada la posibilidad de insertar una nota a la diapositiva activa y de visualizar la presentación en pantalla completa.

2.3 Usuarios relacionados con el sistema

Se define como usuarios relacionados con el sistema a aquellos que interactúan de una u otra forma con la aplicación web para la creación y visualización de presentaciones electrónicas en línea y obtienen un resultado de todos los procesos que se ejecuten en ella. La aplicación web no presenta restricciones de

acceso para su uso, todos los usuarios tienen los mismos privilegios sobre las funcionalidades del sistema.

Tabla 1: Usuarios relacionados con el sistema

Usuario	Justificación
Usuario anónimo	Es cualquier usuario que hace uso de la aplicación web para la creación y visualización de presentaciones electrónicas en línea.

2.4 Planificación

Esta fase es la encargada de comenzar el ciclo de vida de XP. En esta se escriben las HU que serán incluidas en la primera versión del producto, describiendo las funcionalidades con que contará la solución. El equipo de desarrollo durante este tiempo se dedica a familiarizarse con las tecnologías y herramientas que utilizará a lo largo del proyecto, probando las herramientas y construyendo un prototipo simple para probar las posibilidades de la arquitectura (Carvajal, 2008). El período de tiempo de esta fase puede variar desde unas pocas semanas hasta unos pocos meses, dependiendo de la familiaridad del equipo con las tecnologías a emplear.

También en esta fase el cliente establece la prioridad de cada HU y los programadores realizan una estimación del esfuerzo necesario para desarrollar cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses (Joskowicz, 2008).

La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas HU se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de HU.

2.4.1 Aspectos no funcionales del sistema

Los aspectos no funcionales son propiedades o cualidades que el producto debe tener; son características que lo hacen atractivo, usable, rápido y confiable (Sommerville, 2005).

- **Diseño e implementación**
  - La solución debe ser implementada utilizando los lenguajes CSS3, HTML5 y JS 1.8.5.
  - Hacer uso del *framework* Paper.js para el trabajo con el elemento *Canvas*.

- Emplear el IDE NetBeans en su versión 8.0.
- Utilizar un estándar de codificación en la implementación de la solución.

➢ **Apariencia**

- El sistema contará con una interfaz amigable para el usuario.
- El tamaño de los textos debe facilitar la lectura.
- Los íconos deben ser elementos que representen las funcionalidades de la solución.
- Debe poder visualizarse en diferentes resoluciones de pantalla.

➢ **Aspectos legales**

- Usar herramientas de software libre bajo las licencias GNU/GPL.
- La solución una vez desarrollada será de código abierto y licencia GPL (*General Public License*).

➢ **Portabilidad**

- La solución deberá funcionar en cualquier navegador, garantizando la compatibilidad con las tecnologías utilizadas.

➢ **Usabilidad**

- La solución podrá ser usada por cualquier usuario con conocimientos mínimos en la utilización de una aplicación ofimática.

➢ **Hardware**

- Procesador Dual Core a 2.5 GHz o mayor.
- 1 GB de RAM o superior.
- Dispositivo de red de al menos 100 Mbits.
- Espacio en disco duro de 100 Mb como mínimo.

2.4.2 Historias de usuarios

Las HU son la técnica utilizada en XP para especificar los requisitos del sistema. En estas el cliente describe brevemente las características que el sistema debe poseer, escritas generalmente en un lenguaje natural y sin mucha terminología técnica.

Capítulo II

Teniendo en cuenta las entrevistas realizadas a los administradores de diferentes sitios web de la UCI y el análisis de las soluciones similares, se determinaron los requerimientos básicos con que debe contar la solución propuesta. La información de una HU puede variar y ajustarse a las características específicas del proyecto. Teniendo en cuenta diferentes HU analizadas se define la siguiente estructura a utilizar para representar las funcionalidades que serán implementadas.

Tabla 2: Descripción de la estructura de las HU

<table>
<tr><th colspan="3">Historia de usuario</th></tr>
<tr><td>No: Posee el número asignado a la HU.</td><td colspan="2">Nombre: Atributo que contiene el nombre de la HU.</td></tr>
<tr><td colspan="3">Usuario: El usuario del sistema que utiliza o protagoniza a la HU.</td></tr>
<tr><td>Prioridad en negocio: Evidencia el nivel de prioridad de la HU en el negocio.</td><td colspan="2">Riesgo en desarrollo: Evidencia el nivel de riesgo en caso de no realizarse la HU.</td></tr>
<tr><td>Puntos estimados: Atributo que contiene la estimación hecha por el equipo de desarrollo del tiempo de duración de la HU. Cuando el valor es 1 equivale a una semana ideal de trabajo. En la metodología XP está definida una semana ideal como 5 días hábiles trabajando 40 horas, es decir, 8 horas diarias. Por lo que cuando el valor de dicho atributo es de 0.5 equivale a 2 días y medio de trabajo, lo que se traduce en 20 horas.</td><td colspan="2">Iteración asignada: Precisa la iteración en la que será desarrollada la HU.</td></tr>
<tr><td colspan="3">Descripción: Posee una breve descripción de lo que realizará la HU.</td></tr>
<tr><td colspan="3">Observaciones: Brinda información extra que sea conveniente agregar para hacer más comprensible la HU.</td></tr>
</table>

El trabajo con las HU es bastante flexible y dinámico, debido a que estas pueden reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas en el transcurso del proyecto. Cada HU debe ser comprensible y delimitada para que pueda ser implementada en un período de tiempo relativamente corto (Penades y Letelier, 2006).

Capítulo II

Teniendo en cuenta que las HU, como se mencionó anteriormente, pueden remplazarse por otras más generales, se llegó a un acuerdo previo consenso con el cliente de agrupar un conjunto de HU que están estrechamente relacionadas. Se definió también que se considerará como “objeto” a los elementos que el usuario puede insertar en la diapositiva (formas, texto e imágenes). A continuación se muestran las HU que representan a las funcionalidades críticas a implementar para la solución propuesta. En los anexos se encuentran las HU que representan a las restantes funcionalidades (ver **Anexo 1**).

Tabla 3: HU - Gestionar diapositiva

Historia de usuario	
No: 7	**Nombre:** Gestionar diapositiva
Usuario: Usuario anónimo	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada:** 1
Descripción: El usuario puede realizar diferentes operaciones con las diapositivas de la presentación como crear, eliminar, duplicar y desplazar diapositiva. También puede modificar algunas de las propiedades de la diapositiva como editar el color de fondo, para esto selecciona un color o aplica un degradado. Otra de las opciones que tiene es la posibilidad de una vez establecido un color de fondo en una diapositiva, aplicárselo a todas las restantes diapositivas de la presentación.	
Observaciones: El usuario para poder interactuar con una diapositiva, tiene que haberla seleccionado previamente.	

Tabla 4: HU - Gestionar presentación

Historia de usuario	
No: 8	**Nombre:** Gestionar presentación
Usuario: Usuario anónimo	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada:** 1

Descripción: El usuario para el trabajo con una presentación cuenta con varias opciones como la posibilidad de crear una nueva presentación, abrir una presentación que ya haya guardado, guardar una presentación con todos sus elementos y exportar una presentación para que pueda ser visualizada en cualquier sitio web sin necesidad de utilizar la solución.
Observaciones: Antes de cerrar la aplicación web, recargar el navegador o crear una nueva presentación el usuario tiene la opción de decidir si cancelar la acción o continuar.

2.4.3 Estimación de esfuerzos por historias de usuarios

Las estimaciones de esfuerzo asociado a la implementación de las HU la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. Las HU generalmente valen de uno a tres puntos.

En la metodología XP la estimación de las HU se realiza a partir de la experiencia del desarrollador sobre el tiempo que supone le tomará llevar a cabo la implementación. En caso de que el desarrollador no pueda estimar alguna HU, puede preguntarle al cliente para ganar en claridad sobre lo que hay que implementar o puede dividir la HU en otras más pequeñas (Beck, 1999).

Tabla 5: Estimación de esfuerzos por HU

No.	Historia de usuario	Estimación (semanas)
1	Dibujar formas	2
2	Editar formas	1
3	Insertar texto	0.5
4	Editar texto	1
5	Insertar imagen	0.5
6	Editar objeto	2
7	Gestionar diapositiva	2
8	Gestionar presentación	3

9	Visualizar presentación	2

2.4.4 Plan de iteraciones

Todo proyecto desarrollado con la metodología XP se debe dividir en iteraciones (Beck y Andres, 2004). 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 además la prioridad que el cliente le asigna a cada HU y el nivel de complejidad que estas poseen. En este plan se especifica detalladamente el orden de desarrollo de las HU dentro de cada iteración según su prioridad para el cliente así como la estimación total de dicha iteración.

Tabla 6: Plan de duración de las iteraciones

Iteración	Orden de las HU a implementar	Duración total
Iteración 1	Gestionar diapositiva Gestionar presentación	5 semanas
Iteración 2	Dibujar formas Editar formas Insertar texto Editar texto Insertar imagen Editar objeto	7 semanas
Iteración 3	Visualizar presentación	2 semanas

2.4.5 Plan de entregas

En el plan de entregas se realiza un cronograma de entregas donde el cliente establece que HU serán agrupadas para conformar una entrega y el orden de las mismas. Este cronograma será el resultado de una reunión entre todos los interesados del proyecto y se realiza sobre la base de las estimaciones de tiempos de desarrollo (Echeverry y Delgado, 2007).

Tabla 7: Plan de entregas

Historia de usuario	Primera iteración	Segunda iteración	Tercera iteración
Gestionar diapositiva	V 1.0	Finalizado	Finalizado
Gestionar presentación	V 1.1	Finalizado	Finalizado
Dibujar formas	-	V 1.2	Finalizado
Editar formas	-	V 1.3	Finalizado
Insertar texto	-	V 1.4	Finalizado
Editar texto	-	V 1.5	Finalizado
Insertar imagen	-	V 1.6	Finalizado
Editar objeto	-	V 1.7	Finalizado
Visualizar presentación	-	-	V 2.0

2.5 Diseño

El diseño en XP se realiza de forma iterativa, lo que le añade agilidad al proceso de desarrollo. XP hace énfasis en diseño simple, si alguna parte del sistema es de desarrollo complejo, se divide, un sistema simple se implementará con mayor rapidez que uno complejo (Beck y Andres, 2004). Esta metodología lleva a cabo un proceso de mejora continua del diseño, la refactorización, donde se perfecciona y modifica la estructura y codificación del código sin cambiar su funcionalidad. A continuación son seleccionados los patrones arquitectónicos y de diseño, se genera un prototipo no funcional de interfaz de usuario y las tarjetas CRC.

2.5.1 Prototipo no funcional de interfaz de usuario

Un prototipo no funcional de interfaz de usuario es una representación parcial del diseño de un software y se utiliza para que el cliente pueda redefinir sus necesidades y comunicarlas al desarrollador (LSI, 2013). Se realiza con la finalidad de explorar los aspectos interactivos del sistema, incluyendo su usabilidad, accesibilidad y funcionalidad.

Capítulo II

En el prototipo no funcional que se muestra a continuación (ver **Figura 1**) se incluyen las funcionalidades definidas por el cliente.

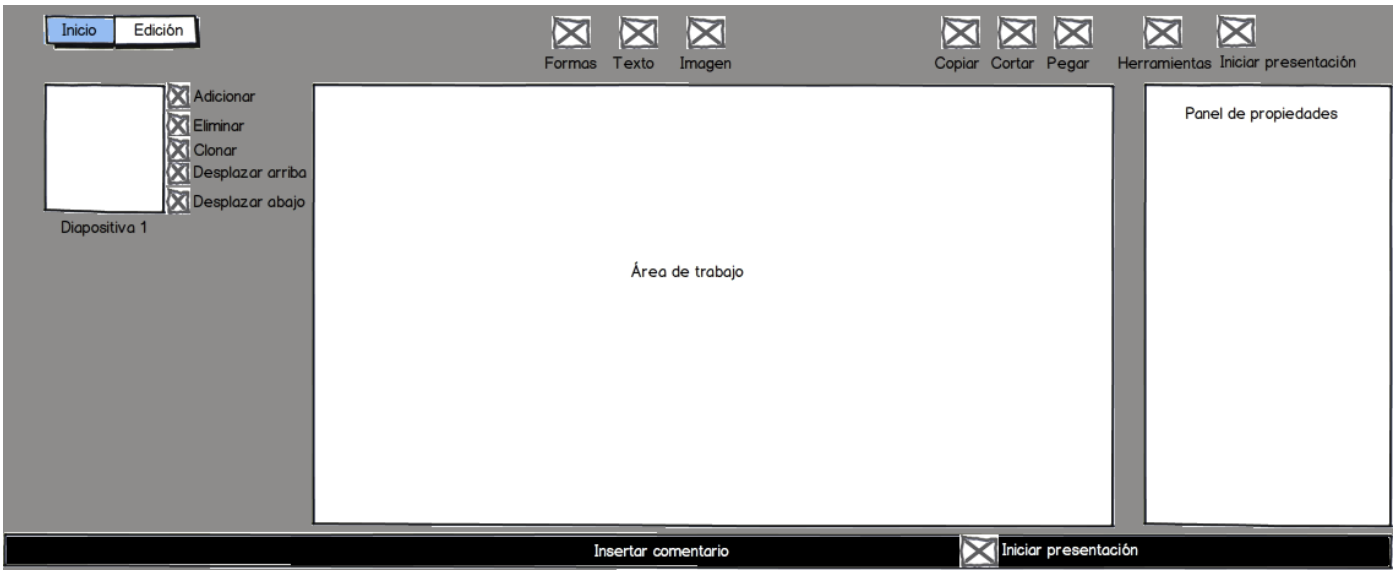

Figura 1: Prototipo no funcional de interfaz de usuario

2.5.2 Arquitectura de software

Según Rich Hilliard (2000) se entiende como arquitectura de software (AS) a "*la organización fundamental de un sistema encarnada en sus componentes, las relaciones de los componentes con cada uno de los otros y con el entorno, y los principios que orientan su diseño y evolución*". Una correcta AS es importante en un número considerable de escenarios evitando errores, reduciendo costos, encontrando fallas e implementando sistemas de importancia crítica. A consideración de Barry Boehm (1995) si un proyecto no ha logrado una arquitectura del sistema, incluyendo su justificación, el proyecto no debe empezar el desarrollo en gran escala; mientras que si se especifica la arquitectura como un elemento a entregar, se la puede usar a lo largo de los procesos de desarrollo y mantenimiento. Debido a la importancia que tiene la arquitectura de software, a continuación se definen los elementos que caracterizan la definida en esta aplicación web así como los patrones arquitectónicos y de diseño utilizados.

2.5.2.1 Patrones arquitectónicos

Mark Klein y Rick Kazman (1999) proponen una definición de los patrones o estilos arquitectónicos en la cual se especifica que son una descripción de los datos y la interacción de control entre los componentes. Estos artefactos de ingeniería son importantes porque expresan esquemas de organización estructural fundamentales para los sistemas de software; proporcionan un conjunto de subsistemas predefinidos,

especifican sus responsabilidades e incluyen guías y lineamientos para organizar las relaciones entre ellos. Existen una amplia diversidad de estilos y sub-estilos que se adaptan a las más variadas necesidades del software a desarrollar, y que tienen diferentes clasificaciones según el autor (Buschmann *et al*., 1996; Kircher y Jain, 2005). A continuación se realiza un análisis del patrón Modelo-Vista-Controlador (MVC) que constituye el patrón arquitectónico seleccionado.

Modelo-Vista-Controlador (MVC)

El patrón Modelo-Vista-Controlador (MVC) es un patrón arquitectónico muy utilizado en aplicaciones web, ya que facilita el funcionamiento, mantenimiento y escalabilidad del sistema, de forma simple y sencilla (Bahit, 2011). Este patrón se caracteriza por separar la parte gráfica de una aplicación, de los procesos lógicos y de los datos de la misma. El patrón MVC está compuesto por los siguientes elementos que conforman la arquitectura de la solución propuesta:

Vista: constituye la interfaz gráfica con que el usuario de la aplicación interactúa y es donde se presentan los resultados de los procesos al usuario. Esta capa se comunica con la capa inmediata inferior, la capa de la lógica de la aplicación. En la Vista está recogida la estructura de la aplicación a través de *index.jade* y *layout.jade* que se apoyan en el motor de plantilla Jade.js, y en la utilización de *style.css* para definir el estilo.

Controlador: es el encargado de modificar los valores de las variables, objetos y datos en general en el modelo; esto lo hace de acuerdo a lo solicitado por el usuario a través de la interfaz gráfica (Vista). La solución propuesta tiene como característica que la lógica ocurre en el lado del cliente, mediante clases JavaScript. Esta capa contiene a las clases controladoras *Slide.js, RunApp.js y MiniSlide.js* que se apoyan para su funcionamiento en los *framework* Paper.js y jQuery.js. También en esta capa se encuentran las clases que se ejecutan a través de Node.js para el control de las peticiones realizadas por el usuario en la Vista.

Modelo: es la representación específica de la información con la que se está operando en el instante de la ejecución de la aplicación, representa las variables, objetos y datos en general que se están modificando de acuerdo a lo que el usuario solicite. En esta capa se encuentra la clase *Save.js* encargada del almacenamiento de las presentaciones creadas y de exportar las presentaciones.

La utilización del patrón MVC disminuye el esfuerzo de los programadores, permite la reutilización de componentes y facilita el mantenimiento en caso de errores (López, 2009). Producto de estas ventajas y las características propias de la aplicación web a desarrollar, es seleccionado este patrón.

En la solución propuesta el patrón MVC se aplica a partir del uso del *framework* Express. Este permite definir una estructura del proyecto basada en dicho patrón que facilita el trabajo con Node.js. A continuación se muestra la estructura del proyecto generada por dicho *framework*.

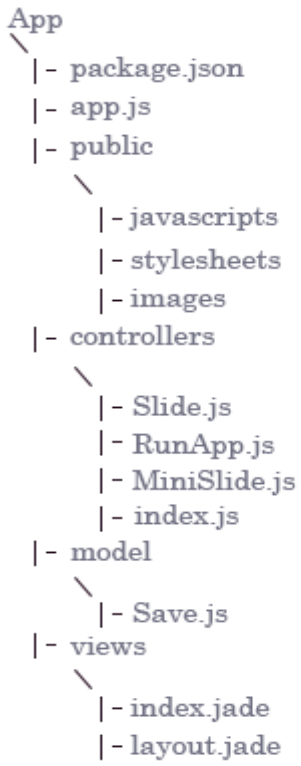

Figura 2: Estructura generada por el *framework* Express para la solución propuesta

2.5.2.2 Patrones de diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular (Gamma *et al*., 1995). Los patrones de diseño hacen más fácil reutilizar con éxito los diseños y arquitecturas, ayudan a elegir entre diseños alternativos, hacen a un sistema reutilizable y evitan alternativas que comprometen la reutilización. Su utilización facilita la localización de los objetos que formarán el sistema y especifican interfaces para las clases (Harmes y Diaz, 2008). Debido a la importancia que tienen para el desarrollo de la solución se seleccionaron un conjunto de patrones de diseño enfocados al desarrollo con JS.

Capítulo II

Creational Pattern (Patrón Creación)

El patrón Creación es la base de otros patrones de diseño muy utilizados, fácil de entender, trata con la idea de crear cosas nuevas, específicamente nuevos objetos (Osmani, 2012). Es utilizado en la solución a la hora de crear nuevas instancias del objeto *Slide* para guardar los valores de cada diapositiva creada en la presentación electrónica, las clases que lo utilizan son *MiniSlide.js* y *RunApp.js*.

Constructor Pattern (Patrón Constructor)

Los constructores se utilizan para crear tipos específicos de objetos, preparándolos además para su uso y para aceptar parámetros que el constructor utiliza para establecer valores a las variables propias. En JS, las funciones constructoras son generalmente una forma razonable para implementar instancias. Aunque JS no admite el concepto de clases, es compatible con funciones especiales de constructor (Osmani, 2012). Este patrón es utilizado para la construcción del objeto *Slide* con sus propiedades y métodos. A continuación se muestra un fragmento en la construcción de este objeto en la clase *Slides.js*.

```
1  function Slide(id, name, content, fondo)
2  {
3      this.id = id;
4      this.name = name;
5      this.content = content;
6      this.fondo= fondo;
7
8  }
```

Figura 3: Ejemplo de aplicación del patrón Constructor

Prototype Pattern (Patrón Prototipo)

Las funciones en JS tienen una propiedad llamada prototipo. El patrón constructor visto anteriormente presenta algunos problemas, uno es el hecho de que se dificulta la herencia y otro es el hecho de poder utilizar las funciones sin que se redefinan cada vez que se crea un nuevo objeto del constructor creado. Con el uso del patrón prototipo en conjunto con el constructor se soluciona este problema, logrando de esta manera que una única instancia de una función pueda ser compartida entre todos los objetos que se creen (Osmani, 2012). Esto es utilizado en la definición del constructor Slide, para así poder utilizar sus métodos por igual cada vez que se crea una instancia de este objeto. También este patrón establece una manera fácil de implementar la herencia. Este patrón es muy utilizado por el *framework* Paper.js que toma ventajas de su uso para su funcionamiento.

Facade Pattern (Patrón Fachada)

La intención de este patrón es proveer una interfaz unificada para un conjunto de interfaces de un subsistema, esto permite definir una interfaz de alto nivel que hace al subsistema fácil de usar. Dividir un sistema en subsistemas, ayuda a reducir la complejidad del mismo, un diseño común disminuye las dependencias y las comunicaciones entre subsistemas (Schmidt *et al*., 2013). Una forma de lograr lo anteriormente descrito es mediante el patrón fachada, que provee una única y simple interfaz para el subsistema. Este patrón es empleado por la librería jQuery, que aunque la implementación puede soportar métodos con un ancho rango de comportamientos, solo una limitada abstracción de estos métodos es presentada al público para su uso. En la solución también es utilizado por el *framework* Paper.js en el trabajo con el elemento C*anvas*.

2.5.3 Tarjetas CRC

La metodología XP representa las clases del sistema mediante tarjetas CRC. Cada tarjeta representa una clase con su nombre en la parte superior, en la sección inferior izquierda están descritas las responsabilidades y a la derecha las clases que le sirven de soporte.

La gran mayoría de los autores de metodologías de desarrollo orientadas a objetos, coinciden en que la identificación de un conjunto apropiado de clases y su correcta asignación de responsabilidades, son los pilares fundamentales de un diseño orientado a objetos (Bellin, 1997).

Tabla 8: Tarjeta CRC para la clase RunApp

Clase: RunApp	
Responsabilidades	**Colaboradores**
Dibujar formas Editar formas Insertar texto Editar texto Insertar imagen Editar objeto	Slide

Tabla 9: Tarjeta CRC para la clase MiniSlide

Clase: MiniSlide	
Responsabilidades	**Colaboradores**
Visualizar presentación Gestionar diapositiva Gestionar presentación	Slide Save

Tabla 10: Tarjeta CRC para la clase Slide

Clase: Slide	
Responsabilidades	**Colaboradores**
Contiene las propiedades de las diapositivas y su comportamiento	

Tabla 11: Tarjeta CRC para la clase Save

Clase: Save	
Responsabilidades	**Colaboradores**
Responsable de guardar y exportar una presentación creada por el usuario	

A continuación se describe el papel de cada clase en el funcionamiento de la solución propuesta:

- **Slide:** es la encargada de gestionar los datos de cada diapositiva, como el nombre y los objetos que contiene.
- **Save:** es la responsable de permitir al usuario guardar y abrir una presentación, para esto se apoya en Node.js.

- **RunApp:** es la encargada del trabajo con los objetos que el usuario desea insertar en la diapositiva. Se apoya en el framework Paper.js y en jQuery para realizar esta tarea, que va desde insertar a transformar un objeto.
- **MiniSlide:** controla las funcionalidades asociadas a la gestión de la presentación electrónica, como crear y guardar una presentación electrónica. Además se encarga de la gestión de las diapositivas, como crear, duplicar, eliminar y desplazar diapositiva.

2.6 Conclusiones parciales del capítulo

Haciendo uso de las prácticas y técnicas que define XP para mejorar el proceso de desarrollo, se definieron 3 iteraciones que comprenden un total de 9 HU (2 críticas y 7 no críticas), que describen los aspectos principales a tener en cuenta para el desarrollo de la solución, estableciendo una visión futura de las funcionalidades que debe poseer el sistema. Asociado a estas HU se construyó el plan de entregas que enmarcó el tiempo de desarrollo de las HU en 14 semanas, determinando un cronograma que especifica las entregas que deben hacerse y conjuntamente se elaboró el prototipo no funcional de interfaz de usuario. Esto permitió ganar en organización con respecto a qué HU era más importante realizar primero y tener una idea de cómo quedaría la solución.

Se conformó la arquitectura de la solución donde quedaron definidos los patrones arquitectónicos y de diseño a utilizar y 4 tarjetas CRC que representan las clases a implementar, definiendo las responsabilidades y los colaboradores. Lo anterior sirvió para organizar el proceso de desarrollo y conformar la estructura de la solución.

Capítulo III: Implementación y pruebas

3.1 Introducción

La implementación es la parte más importante en el proceso de la programación extrema. La metodología XP plantea que la implementación de un producto debe realizarse de forma iterativa, obteniendo al culminar cada iteración un producto funcional que debe ser probado y mostrado al cliente para incrementar la visión de los desarrolladores con la opinión de éste. Cada vez que se quiere implementar una parte de código, en XP, se tiene que escribir una prueba sencilla, y después escribir el código que logre pasar la prueba. En XP hay una máxima que dice “Todo el código que puede fallar tiene que tener una prueba”.

En el presente capítulo se describe el estándar de codificación utilizado para obtener un código eficiente y fácil de mantener. Se detallan las tres iteraciones realizadas, exponiéndose las tareas de ingeniería generadas para dar cumplimiento a cada HU con el fin de facilitar el trabajo al equipo de desarrollo. Se realizan las pruebas propuestas por la metodología seleccionada, documentando todos los resultados obtenidos.

3.2 Implementación

En esta fase se lleva a cabo la codificación de cada una de las HU por iteración. Estas HU se descomponen en tareas de ingeniería que son para el uso estricto de los programadores. Otro elemento importante en esta fase lo constituye la estandarización del código para facilitar la lectura y modificación del mismo por parte del desarrollador. A partir de la planificación realizada se llevaron a cabo tres iteraciones para lograr implementar los requerimientos identificados.

3.2.1 Estándar de codificación

Uno de los aspectos fundamentales que se debe tener en cuenta a la hora de implementar un software, es el uso de un estándar de codificación. El mejor método para asegurarse de que un equipo de programadores mantenga un código de calidad, dónde puedan trabajar de forma coordinada, es establecer un estándar de codificación sobre el que se efectuarán luego revisiones del código rutinarias. La utilización de técnicas de codificación sólidas y las buenas prácticas de programación son de gran importancia para la calidad del software y para obtener un buen rendimiento. Además, si se aplica de forma continuada un estándar de codificación bien definido, si se utilizan técnicas de programación apropiadas, y si se efectúan revisiones del código rutinarias, existen altas posibilidades de que el software se convierta en un sistema fácil de comprender y de mantener (Beck, 1999).

Capítulo III

Apoyado en las ventajas anteriores que acelerarán el proceso de construcción de la solución y facilitarán la mantenibilidad del código, se siguen las siguientes convenciones de codificación para el lenguaje JS apoyadas en las sugeridas por Zakas (2012); para lograr establecer un estilo de programación homogéneo y que cumpla con las mejores prácticas.

Indentación

- Se deben emplear cuatro espacios como unidad de indentación y no se deben utilizar tabuladores.
- Evitar las líneas de más de 80 caracteres. Si una línea va a ser más larga que 80 caracteres se debe romper después de un operador como la coma.

Espacio de paréntesis

- Cuando son usados paréntesis, estos no deben tener espacios en blanco inmediatamente después del paréntesis abierto o inmediatamente antes del paréntesis cerrado.

Comentarios

- Hacer un uso frecuente de comentarios para permitir a otros entender el código. Los comentarios de una sola línea se deben utilizar para la documentación de una sola línea de código o un grupo de líneas relacionadas. Los comentarios de múltiples líneas deben ser usados para documentar código que requiere más explicación. Cada comentario de múltiples líneas debe tener al menos tres líneas.

  Usar comentarios cuando:

  - El código es difícil de entender.
  - El código específico para el navegador es necesario pero no obvio.
  - La generación de documentación es necesaria para un objeto, método o propiedad.

Declaración de variables

- Todas las variables deben ser declaradas antes de ser utilizadas. Las declaraciones de las variables deben tener lugar al comienzo de la función utilizando una sola sentencia ***var*** con una variable por línea. Todas las líneas después de la primera variable declarada deben tener una indentación de un solo nivel por la cual los nombres de las variables se alineen. Las variables inicializadas deben estas primero, seguido de las variables sin inicializar.

Declaración de funciones

- Las funciones deben ser declaradas antes de ser utilizadas. No debe haber espacio entre el nombre de la función y el paréntesis abierto. Debe haber un espacio entre el paréntesis cerrado y llave abierta, y esta última debe estar en la misma línea que la palabra clave ***function***.

Asignación de nombres

- Se utilizarán descriptores en inglés que dejen claro el cometido de la variable, método o clase. Evitar en lo posible los nombre largos (menos de 15 letras sería lo ideal). Evitar nombres que difieran en una letra o en el uso de mayúsculas. Un nombre no debería constar de más de dos palabras. Cada tipo de elemento debe nombrarse debe nombrarse con una serie de reglas determinadas:
  - **Clases:** Nombres. La inicial en mayúscula.
  - **Métodos o funciones:** Deben ser verbos. La primera letra de la primera palabra en minúscula, el resto de las palabras empiezan por mayúscula.
  - **Variables:** Deben estar en minúscula. No deben empezar en ningún caso con el carácter "_".

La utilización de estos estándares de codificación por el desarrollador permitieron lograr una solución lo más legible posible, para que sea fácilmente entendible y se le pueda dar mantenibilidad cuando sea requerido.

3.2.2 Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas (Rumbaugh, Jacobson y Booch, 2007). Pueden ser simples archivos, paquetes o bibliotecas cargadas dinámicamente. Las relaciones de dependencia se utilizan en los diagramas de componentes para indicar que un componente utiliza los servicios ofrecidos por otro componente. Estos diagramas se utilizan para:

- Modelar la vista estática de un sistema.
- Mostrar la organización y las dependencias entre un conjunto de componentes.
- Mostrar qué componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema.

Capítulo III

Un componente es un empaquetamiento físico de los elementos de un modelo, como son las clases en el diseño. Algunos estereotipos estándar de componentes son los siguientes (Castillo, 2010):

- <<*executable*>> es un programa que puede ser ejecutado en nodo.
- <<*file*>> es un fichero que contiene código o datos.
- <<*library*>> es una biblioteca estática o dinámica.
- <<*table*>> es una tabla de base de datos.
- <<*document*>> es un documento.

Es válido aclarar que XP no propone concisamente los artefactos a utilizar en la implementación de una solución que utilice dicha metodología. Deja, en manos del equipo de desarrollo, dependiendo de sus capacidades de comunicación, la decisión de utilizar tantos tipos de diagramas de UML como crean posible, y así, facilitar el proceso de desarrollo.

El diagrama de componentes que se muestra a continuación, según la arquitectura definida basada en el patrón MVC, define la estructuración física del sistema mejorando el entendimiento del desarrollador sobre la aplicación de este patrón. Además es importante para poder ofrecer mantenimiento o la realización de versiones posteriores.

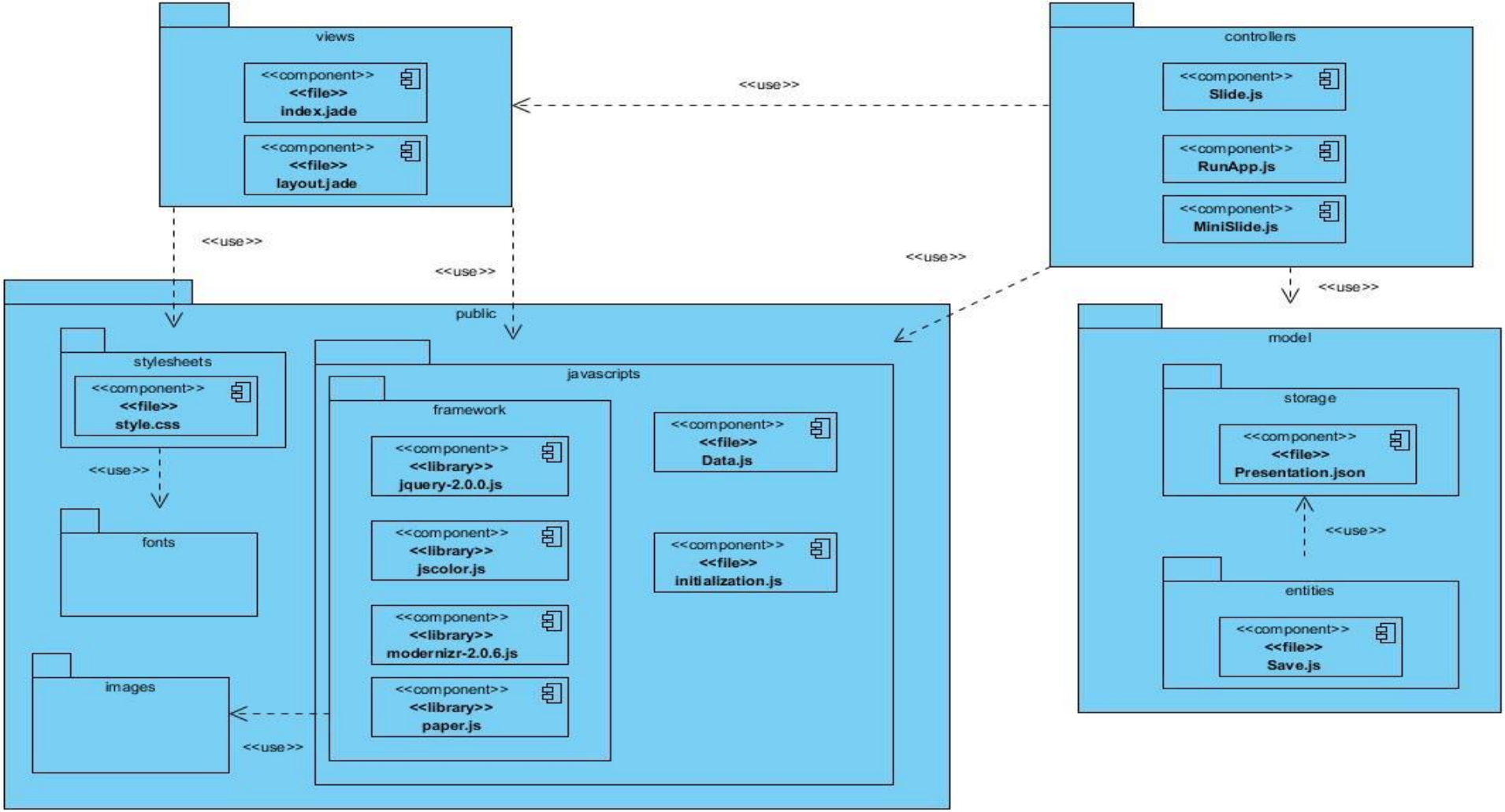

Figura 4: Diagrama de componentes de la solución

3.2.3 Tareas de ingeniería

Según plantea Kent Beck (1999) las HU se deben descomponer en tareas de programación o ingeniería (en inglés *task card*) y asignadas a los programadores para ser implementadas durante una iteración. Generalmente las tareas son más pequeñas que la HU completa, porque no se puede implementar toda una HU en algunos días; estas tareas deben tener una duración de entre 1 y 3 días aproximadamente. Las tareas no tienen que necesariamente ser entendidas por el cliente, pueden ser escritas en lenguaje técnico y son para el uso estricto de los programadores. A continuación se detallan cada una de las iteraciones y tareas de ingeniería asignadas a cada HU.

3.2.3.1 Iteración 1

En esta iteración se desarrollan las HU que presentan mayor prioridad, dando al sistema las primeras funcionalidades, estas son: gestionar diapositiva y gestionar presentación. Al concluir esta primera iteración se obtendrá una estructura básica del sistema. Para consultar las tareas de ingeniería implementadas en esta primera iteración (ver **Anexo 2)**.

Tabla 12: HU implementadas en la primera iteración

Historias de usuarios	Tiempo de implementación (semanas)	
	Estimación	Real
Gestionar diapositiva	2	2
Gestionar presentación	3	3

Para el desarrollo de la iteración se definieron las tareas de ingeniería que se enuncian a continuación.

Tabla 13: Tareas de ingeniería de la primera iteración

Historias de usuarios	Tareas de ingeniería
Gestionar diapositiva	Crear diapositiva Eliminar diapositiva Duplicar diapositiva Desplazar diapositiva

	Editar color de fondo de la diapositiva Aplicar color de fondo a todas las diapositivas
Gestionar presentación	Crear una nueva presentación Abrir una presentación Guardar una presentación Exportar presentación

3.2.3.2 Iteración 2

En esta iteración se desarrollan las HU: dibujar formas, editar formas, insertar texto, editar texto, insertar imagen y editar objeto. Estas funcionalidades permitirán insertar los objetos que el usuario seleccione en la diapositiva activa así como la posibilidad de editarlos. Para consultar las tareas de ingeniería implementadas en esta segunda iteración (ver **Anexo 3)**.

Tabla 14: HU implementadas en la segunda iteración

Historias de usuarios	**Tiempo de implementación (semanas)**	
	Estimación	**Real**
Dibujar formas	2	2
Editar formas	1	1
Insertar texto	0.5	0.5
Editar texto	1	1
Insertar imagen	0.5	0.5
Editar objeto	2	2

Para el desarrollo de la iteración se definieron las tareas de ingeniería que se enuncian a continuación.

Tabla 15: Tareas de ingeniería de la segunda iteración

Historias de usuarios	Tareas de ingeniería
Dibujar formas	Dibujar línea Dibujar rectángulo Dibujar circulo Dibujar elipse Dibujar estrella
Editar formas	Editar color de relleno de la forma Editar color de borde de la forma Editar ancho del borde de la forma
Insertar texto	Insertar texto
Editar texto	Modificar contenido del texto Editar fuente del texto Editar tamaño del texto Editar color del texto Establecer estilo predefinido al texto Texto en negrita Texto en cursiva Alineación del texto
Insertar imagen	Insertar imagen
Editar objeto	Girar objeto Eliminar objeto Trasladar objeto Modificar el tamaño de un objeto

	Copiar objeto Cortar objeto Pegar objeto

3.2.3.3 Iteración 3

En esta iteración se desarrolla la HU: visualizar presentación. Esta funcionalidad permitirá visualizar la presentación en pantalla completa y desplazarse entre las diapositivas. Para consultar las tareas de ingeniería implementadas en esta tercera iteración (ver **Anexo 4)**.

Tabla 16: HU implementadas en la tercera iteración

Historias de usuarios	**Tiempo de implementación (semanas)**	
	Estimación	**Real**
Visualizar presentación	2	2

Para el desarrollo de la iteración se definieron las tareas de ingeniería que se enuncian a continuación.

Tabla 17: Tareas de ingeniería de la tercera iteración

Historias de usuarios	**Tareas de ingeniería**
Visualizar presentación	Visualizar presentación en pantalla completa Desplazarse entre diapositivas

3.3 Pruebas

XP enfatiza en la realización de pruebas a lo largo del proyecto, con el fin de asegurar en todo momento la realización de lo planteado en el diseño. En este proceso no sólo participa el equipo de desarrollo, también es importante los aportes del cliente, sobre todo en las pruebas de aceptación. Cabe señalar que el diseño de pruebas se realiza para todas las partes del sistema como una práctica para garantizar su buen funcionamiento.

En este sentido XP plantea el uso de un desarrollo guiado por pruebas (en inglés *test-driven development* o TDD) que es una práctica de programación que involucra otras dos prácticas: Escribir las pruebas

primero (en inglés *Test First Development*) y refactorización (en inglés *Refactoring*). Para escribir las pruebas generalmente se utilizan las pruebas unitarias (en inglés *unit test*). En primer lugar, se escribe una prueba y se verifica que las pruebas fallan. A continuación, se implementa el código que hace que la prueba pase satisfactoriamente y seguidamente se refactoriza el código escrito, que es básicamente eliminar código duplicado. El propósito del TDD es lograr un código limpio que funcione (Blé, 2010). La idea es que los requisitos sean traducidos a pruebas, de este modo, cuando las pruebas pasen se garantizará que el software cumple con los requisitos que se han establecido.

3.3.1 Pruebas unitarias

Estas pruebas se aplican a todos los métodos no triviales de todas las clases del proyecto con la condición de que no se liberará ninguna clase que no tenga asociada su correspondiente paquete de pruebas. Uno de los elementos más importantes en estas es que idealmente deben ser construidas antes que los métodos mismos, permitiéndole al programador tener máxima claridad sobre lo que va a programar antes de hacerlo, así como conocer cada uno de los casos de prueba que deberá pasar; lo que optimizará su trabajo y su código será de mejor calidad (Koskela, 2008).

Las pruebas unitarias deben ser construidas por los programadores con el empleo de algún mecanismo que permita automatizarlas de modo tal que tanto su implementación y ejecución consuman el menor tiempo posible permitiendo sacarles un mayor provecho. El empleo de pruebas unitarias completas facilitan la liberación continua de versiones por cuanto al implementar algo nuevo y actualizar la última versión, solo es cuestión de ejecutar de forma automática las pruebas unitarias ya creadas para saber que la nueva versión no contiene errores.

Para la realización de las pruebas se utilizó Jasmine en su versión 2.0, que es un framework para el desarrollo dirigido por comportamiento que permite realizar pruebas al código JS. Este no depende de ningún otro framework JS y provee una sintaxis fácil de entender con la cual poder realizar las pruebas.

Se realizaron 3 pruebas unitarias a los métodos más importantes de las diferentes clases utilizadas, obteniéndose un código con calidad y optimizado; al mismo tiempo que se corrigieron los errores lógicos detectados. A continuación se muestra una de las pruebas realizadas, las demás se pueden consultar en el **Anexo 5**.

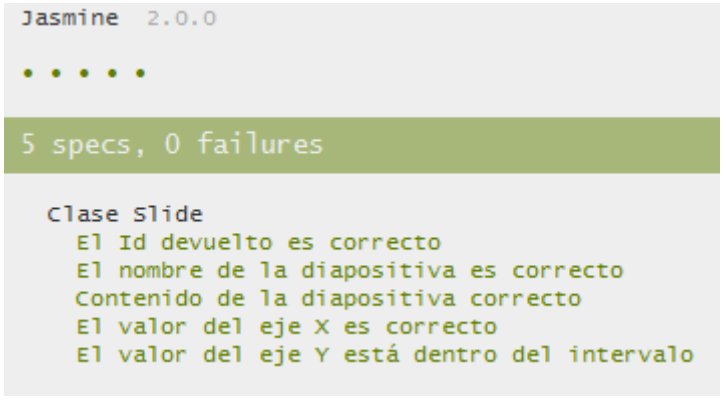

Figura 5: Resultado de la prueba unitaria realizada a la clase Slide

3.3.2 Pruebas de aceptación

Las pruebas de aceptación son supervisadas por el cliente basándose en los requerimientos descritos en las HU. En todas las iteraciones, cada una de las HU seleccionadas por el cliente deberán tener una o más pruebas de aceptación, de las cuales se describirá los casos de prueba y los errores identificados serán corregidos (Echeverry y Delgado, 2007).

Las pruebas de aceptación son pruebas de caja negra o funcional, que representan un resultado esperado de determinada transacción con el sistema. Para que una HU se considere aprobada, deberá pasar todas las pruebas de aceptación elaboradas para dicha historia. A continuación se define la siguiente estructura a utilizar para representar los casos de prueba de aceptación.

Tabla 18: Descripción de la estructura de los casos de prueba de aceptación

Caso de prueba de aceptación	
No: Posee el número asignado al caso de prueba.	**No. HU:** Posee el número asignado a la historia de usuario a realizar prueba.
Nombre: Nombre del caso de prueba.	
Descripción: Descripción de la prueba realizada.	
Condiciones de ejecución: Condiciones necesarias para poder realizar la prueba.	
Entrada / Pasos de ejecución: Serie de pasos necesarios para lograr la realización de la HU, y así realizar la prueba.	

Resultado esperado: Que cumpla con las restricciones del producto.
Evaluación: Se define si fue satisfactoria o no satisfactoria la prueba realizada.

A continuación se muestra una de las pruebas realizadas, las demás se pueden consultar en el **Anexo 6**.

Tabla 19: CP - Dibujar línea

Caso de prueba de aceptación	
No: 1	**No. HU:** 1
Nombre: Dibujar línea.	
Descripción: Prueba para la funcionalidad que permita dibujar una línea en la diapositiva activa.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción línea.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para dibujar una línea se tiene que seleccionar la opción. Se presiona el botón izquierdo del *mouse* en el lugar de la diapositiva dónde se va dibujar la línea y sin soltar el botón se traza la línea.	
Resultado esperado: La línea se traza correctamente en el lugar y con el tamaño deseado por el usuario.	
Evaluación: Prueba satisfactoria.	

Las pruebas a las funcionalidades se realizaron en 3 iteraciones obteniéndose como resultado:

- Para la 1era iteración 6 no conformidades significativas, 7 no conformidades no significativas y 8 recomendaciones.
- Para la 2da iteración 3 no conformidades significativas, 4 no conformidades no significativas y 5 recomendaciones.
- Para la 3era iteración 0 no conformidades significativas, 0 no conformidades no significativas y 2 recomendaciones.

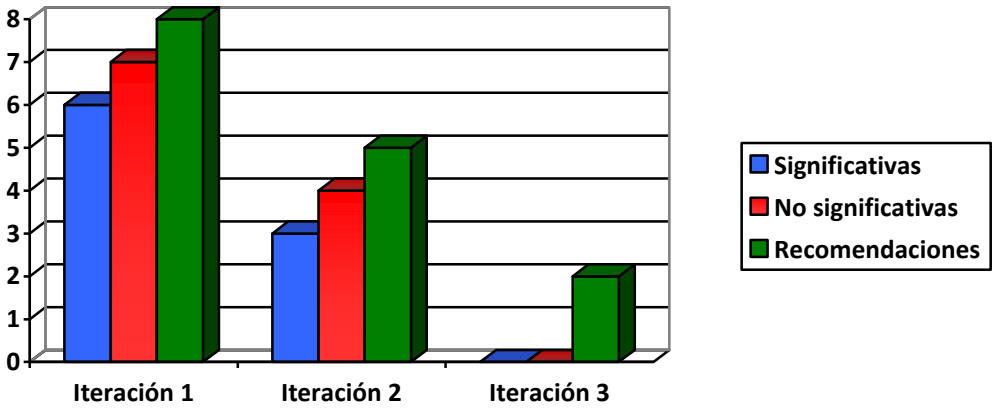

Figura 6: No conformidades significativas, no significativas y recomendaciones

Las no conformidades (NC) significativas se centraron en errores de validación y cambios en el diseño gráfico, y las NC no significativas en errores ortográficos. Se tomaron en cuenta las recomendaciones que surgieron como parte de las pruebas para mejorar las funcionalidades de la solución y fueron corregidas las NC detectadas a través de la validación, utilizando JS, y los arreglos realizados al diseño gráfico mediante CSS3. El uso de estas pruebas permitió realizar entregas de la solución propuesta al cliente que respondieran a sus requerimientos.

3.4 Conclusiones parciales del capítulo

En el presente capítulo se abordaron los temas referentes a la implementación y prueba de la solución. Con la definición de un estándar de codificación se pudo hacer un uso de técnicas de codificación sólidas y de buenas prácticas de programación que permitieron la obtención de un software con calidad y buen rendimiento. La implementación de las funcionalidades de cada HU se realizó a través de las tareas de ingeniería, logrando cumplir con los requerimientos definidos para el desarrollo de la solución.

La realización de un desarrollo dirigido por pruebas que se apoyó en las pruebas unitarias y de aceptación realizadas, agilizó el proceso de desarrollo y permitió obtener una solución con calidad acorde a los requerimientos del cliente.

Conclusiones

Después de desarrollar el presente trabajo y analizar los resultados obtenidos, las conclusiones generales a las que se arriban son:

- El estudio realizado permitió seleccionar como metodología de desarrollo a XP, sirviendo como guía para la organización de las actividades, la generación de los artefactos y la documentación necesaria.
- Las herramientas, lenguajes y tecnologías seleccionadas permitieron obtener una aplicación ofimática para la creación y visualización de presentaciones electrónicas en línea, a partir de las funcionalidades y características solicitadas por el cliente.
- La realización de las pruebas unitarias y de aceptación y el análisis de los resultados de las mismas, permitieron determinar y erradicar las deficiencias encontradas contribuyendo a la solidez del sistema.

De esta manera se concluye que se cumplió el objetivo de la investigación al desarrollar la presente solución informática cumpliendo las metas trazadas al comienzo de la investigación.

Recomendaciones

A partir de la investigación realizada se sugieren las siguientes recomendaciones con el objetivo de que muchas de las consideraciones dadas aquí sean objeto de revisión, de completamiento y de perfeccionamiento en futuros trabajos. Se recomienda:

- A la comunidad científica, tomar el presente trabajo como material de estudio en el desarrollo de propuestas similares.
- Incorporarle a la solución propuesta la capacidad de administración de usuarios y presentaciones electrónicas.
- Incluir otras funcionalidades a la solución como la posibilidad de insertar y editar tablas y gráficos.
- Explotar las capacidades de Node.js desarrollando la funcionalidad que permita la creación de una presentación electrónica por más de un usuario a la vez.
- Utilizar la solución en otros centros además de en la Universidad de las Ciencias Informáticas.

Referencias bibliográficas

1. AETECNO. Conoce las nuevas funciones de Google Slides. [En línea]. 2013. [Consultado el: 19 de febrero de 2014]. Disponible en: http://tecnoadmin.americaeconomia.com/tutoriales/conoce-las-nuevas-funciones-de-google-slides.
2. AVISON, D. y FITZGERALD, G. Information Systems Development - Methodologies, Techniques and Tools. London, McGraw-Hill, 1995. 505 p. ISBN: 0-07-709233-3.
3. BAHIT, E. El paradigma de la Programación Orientada a Objetos en PHP y el patrón de arquitectura de Software MVC. 2011. 66 p.
4. BECK, K. Extreme Programming Explained – Embrace Change. 1999. 224 p. ISBN: 0201616416.
5. BECK, K. y Andres, C. Extreme Programming Explained - Embrace Change. Massachusetts, Addison Wesley Professional, 2004. 224 p. ISBN: 0-321-27865-8.
6. BELLIN, D. The CRC Card Book. Addison – Wesley, 1997. 290 p. ISBN: 978-8-0201-8953-53.
7. BIBEAULT, B. y KATZ, Y. jQuery in action. Greenwich, Manning Publications, 2008. 339 p. ISBN: 1-933988-35-5.
8. BLÉ, C. Diseño Ágil con TDD. 2010. 305 p. ISBN: 978-1-4452-6471-4.
9. BÖCK, H. The definitive guide to Netbeans Platform 7. Apress, 2011. 592 p. ISBN: 978-1-4302-4101-0.
10. BOEHM, B. Engineering Context for Software Architecture. En: First International Workshop on Architecture for Software Systems. Seattle, 1995.
11. BURD, B. Beginning programming with java for dummies. Indiana, Wiley Publishing, 2005. 371 p. ISBN: 978-0-7645-8874-7.
12. BUSCHMANN, F., *et al*. Pattern-Oriented Software Architecture. John Wiley & Sons, 1996. 476 p. ISBN: 978-0471958697.
13. CABRERA, L. y POMPA, E. R. Extensión de Visual Paradigm for UML para el desarrollo dirigido por modelos de aplicaciones de gestión de información. Serie Científica de la Universidad de las Ciencias Informáticas, 2012, 5 (10): p. 1-11.

Referencias bibliográficas

14. CÁCERES, P. Y MARCOS, E. Procesos ágiles para el desarrollo de aplicaciones web. Universidad Rey Juan Carlos, 2007.
15. CANÓS, J. H., LETELIER, P. y PENADÉS, M. C. Metodologías Ágiles en el Desarrollo de Software. VII Jornadas de Ingeniería del Software y Bases de Datos. Taller Metodologías Ágiles en el Desarrollo de Software. Alicante: ISSI, 2003, p. 1-9.
16. CARVAJAL, J. C. Metodologías ágiles - Herramientas y modelo de desarrollo para aplicaciones Java EE como metodología empresarial. Máster en Tecnologías de la Información, UPC – Barcelona, 2008.
17. CASTILLO, V. Documentación del ciclo de vida de sistemas de información con UML. Universidad de Colima, 2010.
18. CASTLEDINE, E. y SHARKIE, C. jQuery – Novice to Ninja. SitePoint, 2010. 393 p. ISBN: 978-0-9805768-5-6.
19. COMUNIDAD DE MADRID. ¿Qué son los programas de presentaciones? [En línea].2013. [Consultado el: 18 de febrero de 2014] 1-2 p. Disponible en: http://www.madrid.org/cs/StaticFiles/Emprendedores/GuiaEmprendedor/tema7/F47_7.7_PRESENTACIONES.pdf.
20. CONBOY, C., *et al*. An Evaluation of the Potential Use and Impact of Prezi, the Zooming Editor Software, as a Tool to Facilitate Learning in Higher Education. Innovation in Practice, 2012, 7: p. 31-46.
21. CreateJS. EaselJS. [En línea]. 2014. [Consultado el: 10 de abril de 2014]. Disponible en: http://www.createjs.com/#!/EaselJS.
22. CUBADEBATE. Bloqueo contra Cuba. [En línea]. 2014. [Consultado el: 4 de abril de 2014]. Disponible en: http://www.cubadebate.cu/serie/bloqueo-contra-cuba/.
23. DAVIS, W. S. y MATA, A. Herramientas CASE - metodología estructurada para el desarrollo de los sistemas. Madrid, Thomson-Paraninfo, 1992. 687 p.
24. DIRECCIÓN DE INVESTIGACIONES. Política Científica de la Universidad de la Ciencias Informáticas [En línea]. Universidad de las Ciencias Informáticas, La Habana, 2012. [Consultado el: 25 de septiembre de 2013]. Disponible en: https://investigaciones.uci.cu/files/Descargas/Politica_Cientifica/Pol_cient.pdf.

Referencias bibliográficas

25. DUQUESNE, M. Herramientas para la producción de materiales didácticos para las modalidades de enseñanza semipresencial y a distancia. ACIMED, 2007, 16(2): p. 2-3. ISSN: 1024-9435.
26. ECHEVERRY, L. M. y DELGADO, L. E. Caso práctico de la metodología ágil XP al desarrollo de software. Proyecto de grado Ingeniero en Sistemas y Computación, Universidad Tecnológica de Pereira, 2007.
27. ELAC. Computación en nube: una alternativa en el cuidado del medio ambiente. Newsletter, 2011, No. 14: p. 10.12.
28. EGUÍLUZ, J. Introducción a JavaScript. Madrid, Autoedición, 2009. 183 p.
29. FREDERICK, S., RAMSAY, C. y BLADES, S. Learning ExtJS. Packt Publishing, 2008. 293 p. ISBN: 978-1-8-47195-14-2.
30. FULTON, S. y FULTON, J. HTML5 Canvas - Native Interactivity and Animation for the Web. Sebastopol, O' Reilly Media, 2011. 609 p. ISBN: 978-1-449-39390-8.
31. GAMMA, E., *et al*. Design patterns: elements of reusable object-oriented software. Boston, Addison-Wesley, 1995. 395 p. ISBN: 978-0-201-63361-0.
32. GAUCHAT, J. D. El gran libro de HTML5, CSS3 y JavaScript. Barcelona, Marcombo, 2012. 354 p. ISBN: 978-84-267-1782-5.
33. GÓMEZ, R. PowerPoint 2007. Madrid, Anaya Multimedia, 2007. 192 p. ISBN: 978-84-415-2203-9.
34. GUTIÉRREZ, C. Cómo funciona la Web. Centro de investigación de la Web, Universidad de Chile, 2008. 139 p. ISBN: 978-956-319-225-1.
35. HARMES, R. Y DIAZ, D. Pro JavaScript Design Patterns. Apress, 2008. 263 p. ISBN: 978-1-4302-0495-4.
36. HILLIARD, R. IEEE-std-1471-2000 recommended practice for architectural description of software-intensive systems [En línea]. IEEE, 2000. [Consultado el: 27 de febrero de 2014]. Disponible en: http://standards.ieee.org.
37. ISSI, L. JavaScript. Madrid, Anaya, 2002. 987 p. ISBN: 84-415-1384-8.
38. JOSKOWICZ, J. Reglas y prácticas en eXtreme Programming. Universidad de Vigo, 2008. 19 p.

Referencias bibliográficas

39. JOYANES, L. Computación en nube - el nuevo paradigma tecnológico. Revista cuatrimestral de las Facultades de Derecho y Ciencias Económicas y Empresariales, 2009, No. 76: p. 97-111.
40. JSCOLOR. [En línea]. 2013. [Consultado el: 26 de febrero de 2014]. Disponible en: http://jscolor.com/.
41. KATZER, M. y CRAWFORD, D. Office 365 - Migrating and Managing Your Business in the Cloud. Apress, 2013. 363 p. ISBN: 978-1-4302-6527-6.
42. KIRCHER, M. y JAIN, P. Pattern-Oriented Software Architecture – Patterns for Resource Management. John Wiley & Sons, 2005. 310 p. ISBN: 978-0470020890.
43. KIRKPATRICK, D. The Facebook Effect – The inside story company that is connecting the world. Simon and Schuster, 2011. 384 p. ISBN: 978-1-4391-0212-1.
44. KLEIN, M. H., *et al*. Attribute-based architecture styles. Springer US, 1999. 250 p. ISBN: 978-0-387-35563-4.
45. KOSKELA, L. Test Driven – Practical TDD and Acceptance TDD for Java Developers. Greenwich, Manning Publications, 2008. 487 p. ISBN: 1-932394-85-0.
46. LABRA, J. E. y FERNÁNDEZ, D. Una experiencia de aprendizaje basado en proyectos utilizando herramientas colaborativas de desarrollo de software libre. [En línea]. XII Jornadas de Enseñanza Universitaria de la Informática, 2006. [Consultado el: 27 de febrero de 2014]. Disponible en: http://www.di.uniovi.es/~labra/FTP/Papers/LabraJenui06.pdf.
47. LEYVA, D. y HIDALGO, J. Módulo para la realización de presentaciones web reusables sobre moodle. Revista Electrónica de Tecnología Educativa, 2011, No. 38: p. 1-15.
48. LÓPEZ, C. A. Cómo mantener el patrón modelo-vista-controlador en una aplicación orientada a la web. Inventum, 2009, No. 7: 72 – 78. ISSN: 1909-2520.
49. LSI. Interacción persona-ordenador – Ingeniería de la interfaz. [En línea]. Departamento de Lenguajes y Sistemas Informáticos, 2013. [Consultado el: 24 de abril de 2014]. Disponible en: https://www.lsi.us.es/docencia/get.php?id=4330.
50. LUJÁN, S. Programación de aplicaciones web - historia, principios básicos y clientes web. Alicante, Editorial Club Universitario, 2002. 354 p. ISBN: 84-8454-206-8.

Referencias bibliográficas

51. MATEU, C. Desarrollo de aplicaciones web. Barcelona, Universidad Oberta de Catalunya, 2004. 367 p. ISBN: 84-9788-118-4.

52. MCFARLAND, D. S. CSS3 - the missing manual. Sebastopol, O' Reilly, 2013. 607 p. ISBN: 978-1-449-32594-7.

53. MEANS, G. Node for Front-End Developers. O' Reilly Media, 2012. 45 p. ISBN: 978-1-449-31883-3.

54. MICROSOFT. SVG frente a Canvas: cómo elegir. [En línea]. Microsoft Developer Network, 2012. [Consultado el: 3 de abril de 2014]. Disponible en: http://msdn.microsoft.com/es-es/library/ie/gg193983%28v=vs.85%29.aspx.

55. MONDELO, Y. y AMAT, L. Tercer estudio webmétrico en la Universidad de las Ciencias Informáticas. Serie Científica de la Universidad de las Ciencias Informáticas, 2010, 3(11): p. 1-18.

56. MUÑOZ, A. Introducción a Node.JS a través de Koans. 2013. 217 p.

57. MUÑOZ, P. C. y GONZÁLEZ, M. Utilización de las herramientas ofimáticas en la enseñanza universitaria y necesidades formativas del profesorado. Revista de *Curriculum* y Formación de Profesorado, 2011, 15(1): p. 8-9. ISSN: 1989-639X.

58. OSMANI, A. Learning JavaScript Design Patterns. O' Reilly, 2012. 185 p. ISBN: 978-1-449-33181-8.

59. PAÉZ, R. M. Las presentaciones electrónicas, una opción para transmitir información científica. [En línea]. ULAN, 2010. [Consultado el: 27 de febrero de 2014] 4-5 p. Disponible en: http://ulan.ac/downloads/COMUNICACOES/8%20de%20oct/Rosa/JCientifica.pdf.

60. PALACIO, J. ScrumManager - Gestión de proyectos. Safe Creative, 2008. 93 p.

61. PALACIO, J. Flexibilidad con SCRUM. Safe Creative, 2007. 181 p.

62. PENADÉS, M. C y LETELIER, P. O. Metodologías ágiles para el desarrollo de software - eXtreme Programming (XP). Técnica administrativa, 2006, 5 (26): 3-8. ISSN: 1666-1680.

63. PENCIL PROJECT. Introduction to Pencil. [En línea]. 2012. [Consultado el: 12 de abril de 2014]. Disponible en: http://pencil.evolus.vn/wiki/devguide/Introduction.html.

64. PÉREZ, J. M. Aspectos legales a considerar para embeber contenidos en redes sociales. [En línea]. Zyncro, 2014. [Consultado el: 13 de abril de 2014]. Disponible en:

Referencias bibliográficas

https://blog.zyncro.com/2014/01/03/aspectos-legales-a-considerar-para-embeber-contenidos-en-redes-sociales/.

65. POWELL, T. A. Diseño de sitios web. Madrid, McGraw-Hill, 2001. 855 p. ISBN: 84-481-2905-9.
66. PUCKER, J. About Paper.js. [En línea]. Paper.js, 2013. [Consultado el: 10 de abril de 2014]. Disponible en: http://paperjs.org/about/.
67. ROBLEDILLO, I. Sistemas de información electrónica al servicio de la investigación. SCIRE, 1997, 3 (2): p. 107-113.
68. RUMBAUGH, J., JACOBSON, I. y BOOCH, G. El lenguaje unificado de modelado. Manual de referencia. California, Addison-Wesley, 2007. 688 p. ISBN: 9788478290871.
69. RUMBAUGH, J., JACOBSON, I. y BOOCH, G. El lenguaje unificado de modelado - Guía de usuario. California, Addison-Wesley, 2006. 552 p. ISBN: 9788478290765.
70. HERNÁNDEZ, R., FERNÁNDEZ, C. y BAPTISTA, P. Metodología de la investigación. McGraw-Hill, 2006. 839 p. ISBN: 970-10-5753-8.
71. SCHMITT, C., *et al*. Professional CSS - Cascading Style Sheets for Web Design. Indiana, Wiley Publishing, 2005. 411 p. ISBN: 978-0-7645-8833-4.
72. SCHMIDT, D. C., *et al*. Pattern-Oriented Software Architecture – Patterns for Concurrent and Networked Objects. John Wiley & Sons, 2013.
73. SOMMERVILLE, I. Ingeniería del software. Madrid, Pearson Addison Wesley, 2005. 712 p.
74. TEIXEIRA, P. Professional Node.js – Building JavaScript-Based Scalable Software. Indianápolis, John Wiley & Sons, 2013. 351 p. ISBN: 978-1-118-18546-9.
75. TORRES, J. Retos Educativos de la Web Social. Revista Cognición, 2008, No. 13. ISSN: 1850-1974.
76. UCI. Misión. [En línea]. 2012. [Consultado el: 5 de marzo de 2014]. Disponible en: http://www.uci.cu/?q=mision.
77. WATSON, A. Learning Modernizr. Packt Publishing Ltd, 2012. ISBN: 978-17-821-6023-6.
78. WELLING, L. y THOMSON, L. Desarrollo web con PHP y MySQL. Anaya Multimedia, 2009. 976 p. ISBN: 978-84-415-2553-5.

Referencias bibliográficas

79. WOOD, M. iWork – Keynote. Bookboom, 2012. 78 p. ISBN: 9788740300710.
80. YEBES, E. Y ROMERO, C. PowerPoint 2003. Anaya Multimedia – Anaya Interactiva, 2004. 192 p. ISBN: 978-84-415-1628-1.
81. ZAKAS, N. C. Maintainable JavaScript. Sebastopol, O' Reilly Media, 2012. 209 p. ISBN: 978-1-449-32768-2.

Glosario de términos

Adobe Flash: es una aplicación de creación y manipulación de gráficos vectoriales con posibilidades de manejo de código mediante un lenguaje de scripting llamado ActionScript.

Apache: es un servidor HTTP de código abierto para diferentes plataformas. Tiene amplia aceptación en la red siendo el servidor HTTP más utilizado en la actualidad.

API: Interfaz de programación de aplicaciones (IPA) o API (del inglés *Application Programming Interface*) es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Curvas de Bézier: es un sistema que se desarrolló para el trazado de dibujos técnicos y que debe su denominación en honor a Pierre Bézier. Es utilizado por el lenguaje PostScript para la generación de las curvas y los trazados.

Formas: constituyen objetos de dibujo que se crean a partir de líneas, curvas, rectángulos u otros objetos que pueden ser modificados, formateados, desplazados y mejorados (Gómez, 2007).

Google+: (pronunciado y a veces escrito Google Plus) es un servicio de red social operado por Google Inc.

Hipermedia: es el término con el que se designa al conjunto de métodos o procedimientos para escribir, diseñar o componer contenidos que integren soportes tales como: texto, imagen, video, audio, mapas y otros soportes de información emergentes.

Indentación: es un anglicismo (del inglés *indentation*) de uso común en informática, que significa mover un bloque de texto hacia la derecha insertando espacios o tabuladores, para así separarlo del margen izquierdo y distinguirlo mejor del texto adyacente.

Información digital: Es la información que puede ser manejada por ordenadores (Robledillo, 1997).

Java: es un lenguaje de programación de propósito general, concurrente, orientado a objetos y basado en clases.

Licencia GNU/GPL: la Licencia Pública General de GNU es la licencia más ampliamente usada en el mundo del software y garantiza a los usuarios finales la libertad de usar, estudiar, compartir y modificar el software.

Glosario de términos

Licencia LGPL: la Licencia Pública General Reducida de GNU es una licencia de software creada por la *Free Software Foundation* que pretende garantizar la libertad de compartir y modificar el software.

Licencia MIT: es una de tantas licencias de software que ha empleado el Instituto Tecnológico de Massachusetts (MIT por sus siglas en inglés) a lo largo de su historia. El texto de la licencia no tiene copyright, lo que permite la modificación del producto que la utilice, como reutilizar el software así licenciado tanto para ser software libre como para ser software no libre.

Mantenibilidad del código: es la facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores o mejorar el rendimiento.

Microsoft PowerPoint: es un programa de presentación desarrollado por la empresa Microsoft, ampliamente usado en distintos campos como la enseñanza, negocios, etc.

Mozilla Firefox: es un navegador web libre y de código abierto multiplataforma coordinado por la Corporación Mozilla y la Fundación Mozilla.

PDF: (sigla del inglés *portable document format*, formato de documento portátil) es un formato de almacenamiento de documentos digitales independiente de plataformas de software o hardware.

PHP: es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico.

Plugin: significa complemento, y no es más que una aplicación que se relaciona con otra para aportarle una nueva función y generalmente muy específica.

Prototipo: es un objeto diseñado para una demostración de cualquier tipo.

Refactorización: (del inglés *refactoring*) es una técnica de la ingeniería de software para reestructurar un código fuente, alterando su estructura interna sin cambiar su comportamiento externo.

SkyDrive: es un servicio de alojamiento de archivos de la empresa Microsoft.

Software: se conoce como software al equipamiento lógico de un sistema informático, que comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos que son llamados hardware.

SVG: Son una especificación para describir gráficos vectoriales bidimensionales, tanto estáticos como animados, en formato XML.

Glosario de términos

V8: es un motor de JavaScript creado por la empresa Google, constituye una máquina virtual rápida y de gran calidad.

WordPress: es un Sistema Administrador de Contenidos (CMS por sus siglas en inglés) especializado en la creación y gestión de un blog.

XML: siglas en inglés de *eXtensible Markup Language* (lenguaje de marcas extensible), es un lenguaje de marcas utilizado para almacenar datos en forma legible.

Anexos

Anexo 1 – Historias de usuario

Tabla 20: HU - Dibujar formas

Historia de usuario	
No: 1	**Nombre:** Dibujar formas
Usuario: Usuario anónimo	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada:** 2
Descripción: El usuario selecciona alguna de las formas disponibles en el panel de herramientas (línea, rectángulo, círculo, elipse y estrella) y pasa a trazarla en la diapositiva activa. Antes de seleccionar la forma el usuario puede modificar diferentes valores como el color del borde y del relleno, y el ancho de la línea; en caso contrario estos valores serán los predefinidos por la solución.	
Observaciones: Para poder trazar la forma seleccionada el usuario debe mantener el botón izquierdo del *mouse* presionado y establecer el tamaño que estime conveniente.	

Tabla 21: HU - Editar formas

Historia de usuario	
No: 2	**Nombre:** Editar formas
Usuario: Usuario anónimo	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 2
Descripción: El usuario selecciona alguna de las formas trazadas en la diapositiva activa y puede editar el color del relleno y el borde de la forma, así como el ancho de la línea de la forma seleccionada.	
Observaciones: Para poder editar las propiedades de una forma, el usuario tiene que haberla seleccionado	

previamente de la diapositiva activa.

Tabla 22: HU - Insertar texto

Historia de usuario	
No: 3	**Nombre:** Insertar texto
Usuario: Usuario anónimo	
Prioridad en el negocio: Media	**Riesgo de desarrollo:** Medio
Puntos estimados: 0.5	**Iteración asignada:** 2
Descripción: El usuario selecciona la opción de texto del panel de herramientas, escribe el texto y lo inserta en la diapositiva activa. Antes de insertar el texto el usuario puede modificar diferentes propiedades como el tamaño, la fuente, el color y la alineación; en caso contrario las propiedades serán las definidas por defecto por la solución.	
Observaciones: Para poder insertar texto en la diapositiva activa el usuario tiene que primero escribir el texto a insertar y después seleccionar la posición en que se insertará.	

Tabla 23: HU - Editar texto

Historia de usuario	
No: 4	**Nombre:** Editar texto
Usuario: Usuario anónimo	
Prioridad en el negocio: Media	**Riesgo de desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 2
Descripción: El usuario selecciona algún texto de la diapositiva activa, y puede modificar su contenido. También puede editar las propiedades del texto seleccionado como el tamaño, la fuente, el color, la alineación, poner el texto en negrita o en cursiva y elegir un estilo predefinido para el texto.	
Observaciones: Para poder editar un texto el usuario tiene que haberlo seleccionado previamente de la diapositiva activa.	

Tabla 24: HU – Insertar imagen

Historia de usuario	
No: 5	**Nombre:** Insertar imagen
Usuario: Usuario anónimo	
Prioridad en el negocio: Media	**Riesgo de desarrollo:** Alta
Puntos estimados: 0.5	**Iteración asignada:** 2
Descripción: Permite insertar una imagen en la diapositiva activa desde cualquier ubicación en la estación de trabajo del usuario. Para insertar una imagen el usuario selecciona la opción en el panel de herramientas, busca la ubicación de la imagen en la estación de trabajo y pasa a insertarla en la diapositiva activa.	
Observaciones: Para poder insertar una imagen en la diapositiva activa el usuario tiene que haber establecido la ruta de acceso hacia una imagen. Los formatos de imagen permitidos son *.gif, .png, .jpg y .jpeg*.	

Tabla 25: HU - Editar objeto

Historia de usuario	
No: 6	**Nombre:** Editar objeto
Usuario: Usuario anónimo	
Prioridad en el negocio: Media	**Riesgo de desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 2
Descripción: El usuario selecciona alguna de las opciones del panel de herramientas para editar un objeto como girar, eliminar, trasladar, escalar y transformar. También puede seleccionar la opción de copiar, cortar y pegar un objeto; o puede realizar estas operaciones a través de las teclas de acceso rápido. Después de seleccionada la opción a utilizar el usuario pasa a editar el objeto según lo necesite.	
Observaciones: Para poder editar un objeto el usuario tiene que haber seleccionado primero alguna opción del panel de herramientas.	

Tabla 26: HU - Visualizar presentación

Historia de usuario	
No: 9	**Nombre:** Visualizar presentación
Usuario: Usuario anónimo	
Prioridad en el negocio: Media	**Riesgo de desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 3
Descripción: El usuario puede visualizar la presentación realizada en pantalla completa y desplazarse entre las diapositivas a través del teclado o con la utilización del *mouse*.	
Observaciones: Para visualizar la presentación en pantalla completa el usuario debe seleccionar la opción en el panel de herramientas, en la parte inferior de la solución o mediante la opción que ofrece el propio navegador web que esté utilizando.	

Anexo 2 – Tareas de ingeniería para la iteración 1

Tabla 27: TI – Crear diapositiva

Tarea de ingeniería	
No: 1	**No. HU:** 7
Nombre: Crear diapositiva.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 15 de enero de 2014	**Fecha fin:** 17 de enero de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Dar la posibilidad de crear una nueva diapositiva. Tener en cuenta que la posición de la diapositiva creada será a partir de la diapositiva activa. Funciones: onMouseDown (event), add_Content (), new_Slide (), move_Option (), move_Slide (), show_Slide ().	

Clases: Slide, MiniSlide.

Tabla 28: TI - Eliminar diapositiva

Tarea de ingeniería	
No: 2	**No. HU:** 7
Nombre: Eliminar diapositiva.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 17 de enero de 2014	**Fecha fin:** 19 de enero de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Ofrecer la posibilidad de eliminar la diapositiva seleccionada. Una vez eliminada la diapositiva también se deben eliminar los objetos insertados en ella. Funciones: onMouseDown (event), set_Content (), move_Option (), move_Slide (), show_Slide (). Clases: Slide, MiniSlide.	

Tabla 29: TI - Desplazar diapositiva

Tarea de ingeniería	
No: 3	**No. HU:** 7
Nombre: Desplazar diapositiva.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 19 de enero de 2014	**Fecha fin:** 21 de enero de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Permitir poder mover la diapositiva de posición, un solo lugar al mismo tiempo, tanto hacia arriba como para debajo de otra diapositiva. Funciones: onMouseDown (event), move_Option (), move_Slide (), show_Slide ().	

Clases: Slide, MiniSlide.

Tabla 30: TI – Duplicar diapositiva

Tarea de ingeniería	
No: 4	**No. HU:** 7
Nombre: Duplicar diapositiva.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 21 de enero de 2014	**Fecha fin:** 23 de enero de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Permitir duplicar una diapositiva seleccionada, esto significa crear una diapositiva nueva con las mismas propiedades y objetos que la diapositiva seleccionada. Funciones: onMouseDown (event), new_Slide (), move_Option (), move_Slide (), show_Slide (). Clases: Slide, MiniSlide.	

Tabla 31: TI - Editar color de fondo de la diapositiva

Tarea de ingeniería	
No: 5	**No. HU:** 7
Nombre: Editar color de fondo de la diapositiva.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 30 de enero de 2014	**Fecha fin:** 31 de enero de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Permitir poder cambiar el color de fondo de la diapositiva seleccionada a través de la elección de un color o aplicando un degradado. La mini diapositiva también debe adoptar este mismo color de fondo. Funciones: onMouseDown (event), updateslide_Color ().	

Clases: Slide, MiniSlide.

Tabla 32: TI - Aplicar color de fondo a todas las diapositivas

Tarea de ingeniería	
No: 6	**No. HU:** 7
Nombre: Aplicar color de fondo a todas las diapositivas.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 5 de febrero de 2014	**Fecha fin:** 6 de febrero de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Aplicar el color de fondo de la diapositiva seleccionada a las restantes diapositivas de la presentación. Funciones: onMouseDown (event), updateslide_Color (). Clases: Slide, MiniSlide.	

Tabla 33: TI – Crear una nueva presentación

Tarea de ingeniería	
No: 7	**No. HU:** 8
Nombre: Crear una nueva presentación.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 12 de febrero de 2014	**Fecha fin:** 14 de febrero de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Permitir crear una nueva presentación. Funciones: new_Presentation (), new_Slide (), move_Option (), move_Slide (), show_Slide (). Clases: Slide, MiniSlide.	

Tabla 34: TI – Guardar una presentación

Tarea de ingeniería	
No: 8	**No. HU:** 8
Nombre: Guardar una presentación.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.8
Fecha inicio: 17 de febrero de 2014	**Fecha fin:** 21 de febrero de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Salvar la presentación electrónica creada con todas las diapositivas, elementos y configuraciones. Se debe generar un archivo con extensión json para que sea compatible con el *framework* Paper.js. Funciones: get_Data (), generate_Content (), download_File (). Clases: Slide, Save.	

Tabla 35: TI - Abrir una presentación

Tarea de ingeniería	
No: 9	**No. HU:** 8
Nombre: Abrir una presentación.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.8
Fecha inicio: 24 de febrero de 2014	**Fecha fin:** 28 de febrero de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Importar todos los elementos de la presentación incluidos en el archivo de extensión *.json*. Se deben adicionar las diapositivas con sus propiedades y objetos mediante la clase *Slide*. La visualización de la presentación estará controlada por la clase *MiniSlide*. Funciones: open_Presentation (), new_Slide (), move_Option (), move_Slide (), show_Slide (), load_figures (json).	

Clases: Slide, MiniSlide, RunApp.

Tabla 36: TI - Exportar presentación

Tarea de ingeniería	
No: 10	**No. HU:** 8
Nombre: Exportar presentación.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.6
Fecha inicio: 27 de febrero de 2014	**Fecha fin:** 03 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Desarrollar la funcionalidad *download* () en el fichero out.js de Node.js. Se debe generar un archivo con extensión *.js* que contenga la presentación a través de los datos obtenidos de los objetos creados con Paper.js. Funciones: dowload (), exp_Canvas (). Clases: Slide, Save.	

Tabla 37: TI – Alertar ante el cerrado de la aplicación al usuario

Tarea de ingeniería	
No: 11	**No. HU:** 8
Nombre: Alertar ante el cerrado de la aplicación al usuario.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 04 de marzo de 2014	**Fecha fin:** 06 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Detectar a través de JS el cierre o actualización de la aplicación web. Generar un mensaje de tipo *event* para que el usuario determine si continuar con la acción o cancelarla. Funciones: onbeforeunload (), new_Presentation ().	

Clases: MiniSlide.

Anexo 3 – Tareas de ingeniería para la iteración 2

Tabla 38: TI – Dibujar línea

Tarea de ingeniería	
No: 12	**No. HU:** 1
Nombre: Dibujar línea.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 10 de marzo de 2014	**Fecha fin:** 11 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Dibujar una línea desde un punto seleccionado por el usuario hasta otro, para esto se utilizará a Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), add_Content (). Clases: RunApp, Slide.	

Tabla 39: TI - Dibujar rectángulo

Tarea de ingeniería	
No: 13	**No. HU:** 1
Nombre: Dibujar rectángulo.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 11 de marzo de 2014	**Fecha fin:** 12 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Dibujar un rectángulo en el área seleccionada por el usuario, para esto se utilizará a Paper.js.	

Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), add_Content (). Clases: RunApp, Slide.

Tabla 40: TI - Dibujar círculo

Tarea de ingeniería	
No: 14	**No. HU:** 1
Nombre: Dibujar círculo.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 12 de marzo de 2014	**Fecha fin:** 13 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Dibujar un círculo en el área seleccionada por el usuario, para esto se utilizará a Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), add_Content (). Clases: RunApp, Slide.	

Tabla 41: TI - Dibujar elipse

Tarea de ingeniería	
No: 15	**No. HU:** 1
Nombre: Dibujar elipse.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 14 de marzo de 2014	**Fecha fin:** 15 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Dibujar un elipse en el área seleccionada por el usuario, para esto se utilizará a Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), add_Content ().	

Clases: RunApp, Slide.

Tabla 42: TI - Dibujar estrella

Tarea de ingeniería	
No: 16	**No. HU:** 1
Nombre: Dibujar estrella.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 15 de marzo de 2014	**Fecha fin:** 16 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Dibujar un elipse en el área seleccionada por el usuario, para esto se utilizará a Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), add_Content (). Clases: RunApp, Slide.	

Tabla 43: TI - Editar color de relleno de la forma

Tarea de ingeniería	
No: 17	**No. HU:** 2
Nombre: Editar color de relleno de la forma.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 17 de marzo de 2014	**Fecha fin:** 19 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Permitir editar el color de relleno de una forma seleccionada por el usuario. Cambiar la propiedad del relleno de la forma a través de Paper.js y con la utilización de jSColor.js para obtener el color RGB seleccionado por el usuario. Para detectar la selección de un color utilizar jQuery.js Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color ().	

Clases: RunApp.

Tabla 44: TI - Editar color de borde de la forma

Tarea de ingeniería	
No: 18	**No. HU:** 2
Nombre: Editar color de borde de la forma.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 20 de marzo de 2014	**Fecha fin:** 22 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Permitir editar el color de borde de una forma seleccionada por el usuario. Cambiar la propiedad del relleno de la forma a través de Paper.js y con la utilización de jSColor.js para obtener el color RGB seleccionado por el usuario. Para detectar la selección de un color utilizar jQuery.js Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color (). Clases: RunApp.	

Tabla 45: TI - Editar ancho del borde de la forma

Tarea de ingeniería	
No: 19	**No. HU:** 2
Nombre: Editar ancho del borde de la forma.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 22 de marzo de 2014	**Fecha fin:** 23 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Cambiar el ancho del borde de la forma a través de la propiedad definida por Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color ().	

Clases: RunApp.

Tabla 46: TI - Insertar texto

Tarea de ingeniería	
No: 20	**No. HU:** 3
Nombre: Insertar texto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.5
Fecha inicio: 24 de marzo de 2014	**Fecha fin:** 27 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Insertar un texto en la posición que elija el usuario. Capturar la posición seleccionada por el usuario para pasarlo como parámetro en el momento de insertar el texto utilizando Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), add_Content (). Clases: RunApp, Slide.	

Tabla 47: TI - Modificar contenido del texto

Tarea de ingeniería	
No: 21	**No. HU:** 4
Nombre: Modificar contenido del texto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 27 de marzo de 2014	**Fecha fin:** 28 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Modificar el contenido de un texto seleccionado por el usuario. Se cambiará el contenido del texto a través de Paper.js para trabajar con el objeto *PointText* y jQuery.js para detectar lo que el usuario está escribiendo. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event).	

Clases: RunApp.

Tabla 48: TI - Editar fuente del texto

Tarea de ingeniería	
No: 22	**No. HU:** 4
Nombre: Editar fuente del texto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 29 de marzo de 2014	**Fecha fin:** 30 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Editar la fuente de un texto seleccionado por el usuario. Se cambiará la fuente del texto a través de Paper.js para trabajar con el objeto *PointText* y jQuery.js para detectar la selección de la fuente. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color (). Clases: RunApp.	

Tabla 49: TI - Editar tamaño del texto

Tarea de ingeniería	
No: 23	**No. HU:** 4
Nombre: Editar tamaño del texto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.1
Fecha inicio: 31 de marzo de 2014	**Fecha fin:** 31 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Editar el tamaño de un texto seleccionado por el usuario. Se cambiará el tamaño del texto a través de Paper.js para trabajar con el objeto *PointText* y jQuery.js para detectar la selección del tamaño de texto. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color ().	

Clases: RunApp.

Tabla 50: TI - Editar color del texto

Tarea de ingeniería	
No: 24	**No. HU:** 4
Nombre: Editar color del texto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.1
Fecha inicio: 31 de marzo de 2014	**Fecha fin:** 31 de marzo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Editar el color de un texto seleccionado por el usuario. Se cambiará el color del texto a través de Paper.js para trabajar con el objeto *PointText* y jQuery.js en conjunto con jSColor.js para detectar la selección del color. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color (). Clases: RunApp.	

Tabla 51: TI - Texto en negrita

Tarea de ingeniería	
No: 25	**No. HU:** 4
Nombre: Texto en negrita.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.1
Fecha inicio: 01 de abril de 2014	**Fecha fin:** 01 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Aplicar al texto seleccionado por el usuario el estilo Negrita. El estilo se aplicará través de Paper.js para trabajar con el objeto *PointText* y jQuery.js para detectar la selección de la opción.	

Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color ().
Clases: RunApp.

Tabla 52: TI - Texto en cursiva

Tarea de ingeniería	
No: 26	**No. HU:** 4
Nombre: Texto en negrita.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.1
Fecha inicio: 01 de abril de 2014	**Fecha fin:** 01 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Aplicar al texto seleccionado por el usuario el estilo Cursiva. El estilo se aplicará través de Paper.js para trabajar con el objeto *PointText* y jQuery.js para detectar la selección de la opción. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color (). Clases: RunApp.	

Tabla 53: TI – Establecer estilo predefinido al texto

Tarea de ingeniería	
No: 27	**No. HU:** 4
Nombre: Establecer estilo predefinido al texto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.1
Fecha inicio: 02 de abril de 2014	**Fecha fin:** 02 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Aplicar al texto seleccionado por el usuario un estilo de texto predefinido en la solución. El estilo se aplicará través de Paper.js para trabajar con el objeto *PointText* y jQuery.js para detectar el estilo de texto	

predefinido seleccionado. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color (). Clases: RunApp.

Tabla 54: TI - Alineación del texto

Tarea de ingeniería	
No: 28	**No. HU:** 4
Nombre: Alineación del texto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.1
Fecha inicio: 02 de abril de 2014	**Fecha fin:** 02 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Alinear el texto seleccionado por el usuario. La alineación al texto se aplicará través de Paper.js para trabajar con el objeto *PointText* y jQuery.js para detectar la opción seleccionada. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), update_Color (). Clases: RunApp.	

Tabla 55: TI - Insertar imagen

Tarea de ingeniería	
No: 29	**No. HU:** 5
Nombre: Insertar imagen.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.5
Fecha inicio: 03 de abril de 2014	**Fecha fin:** 06 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Insertar una imagen en la posición que elija el usuario. Capturar la posición seleccionada por el	

usuario para pasarlo como parámetro en el momento de insertar la imagen utilizando Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), add_Content (), upload_Image (). Clases: RunApp, Slide.

Tabla 56: TI - Girar objeto

Tarea de ingeniería	
No: 30	**No. HU:** 6
Nombre: Girar objeto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 06 de abril de 2014	**Fecha fin:** 08 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Girar el objeto seleccionado por el usuario. Se girará el objeto con respecto a su posición haciendo uso de Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event). Clases: RunApp.	

Tabla 57: TI - Trasladar objeto

Tarea de ingeniería	
No: 31	**No. HU:** 6
Nombre: Trasladar objeto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 08 de abril de 2014	**Fecha fin:** 09 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Trasladar un objeto seleccionado por el usuario. Se trasladará el objeto con respecto a su posición	

haciendo uso de Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event). Clases: RunApp.

Tabla 58: TI - Modificar el tamaño de un objeto

Tarea de ingeniería	
No: 32	**No. HU:** 6
Nombre: Modificar el tamaño de un objeto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 09 de abril de 2014	**Fecha fin:** 10 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Aplicar zoom sobre un objeto seleccionado por el usuario. Se aumentará o disminuirá el objeto con respecto a su tamaño haciendo uso de Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event). Clases: RunApp.	

Tabla 59: TI - Eliminar objeto

Tarea de ingeniería	
No: 33	**No. HU:** 6
Nombre: Eliminar objeto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio: 14 de abril de 2014	**Fecha fin:** 16 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Aplicar zoom sobre un objeto seleccionado por el usuario. Se aumentará o disminuirá el objeto con	

respecto a su tamaño haciendo uso de Paper.js. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), set_Content (). Clases: RunApp, Slide.

Tabla 60: TI – Copiar objeto

Tarea de ingeniería	
No: 34	**No. HU:** 6
Nombre: Copiar objeto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 16 de abril de 2014	**Fecha fin:** 17 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Copiar un objeto seleccionado por el usuario. Se copiara el objeto junto con todas sus propiedades definidas por Paper.js. Se utilizará jQuery.js para detectar el uso de las teclas de acceso rápido o la selección de la opción. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), get_Content (). Clases: RunApp, Slide.	

Tabla 61: TI - Pegar objeto

Tarea de ingeniería	
No: 35	**No. HU:** 6
Nombre: Pegar objeto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 17 de abril de 2014	**Fecha fin:** 18 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	

Descripción: Pegar un objeto seleccionado por el usuario. Se pegara un objeto, junto con todas sus propiedades definidas por Paper.js, tanto en la diapositiva activa como en otra diapositiva tantas veces lo requiera el usuario. Se utilizará jQuery.js para detectar el uso de las teclas de acceso rápido o la selección de la opción. Para poder llevar a cabo esta operación el usuario tiene que haber copiado algún objeto previamente. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), add_Content (). Clases: RunApp, Slide.

Tabla 62: TI - Cortar objeto

Tarea de ingeniería	
No: 36	**No. HU:** 6
Nombre: Pegar objeto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio: 18 de abril de 2014	**Fecha fin:** 19 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Cortar un objeto seleccionado por el usuario. Se cortará un objeto, junto con todas sus propiedades definidas por Paper.js. Se utilizará jQuery.js para detectar el uso de las teclas de acceso rápido o la selección de la opción. Funciones: onMouseDown (event), onMouseDrag (event), onMouseUp (event), set_Content (). Clases: RunApp, Slide.	

Anexo 4 – Tareas de ingeniería para la iteración 3

Tabla 63: TI - Visualizar presentación en pantalla completa

Tarea de ingeniería	
No: 37	**No. HU:** 9
Nombre: Visualizar presentación en pantalla completa.	

Tipo de tarea: Desarrollo.	**Puntos estimados:** 1
Fecha inicio: 22 de abril de 2014	**Fecha fin:** 27 de abril de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Visualizar la presentación electrónica elaborada por el usuario en pantalla completa. Detectar mediante JS el cambio en el tamaño de la aplicación si se activa la opción a través del navegador o con jQuery.js si se activa a través de la solución. Funciones: full_Screen (), reset (), resize_Canvas (), show_Slide (). Clases: MiniSlide, Slide.	

Tabla 64: TI - Desplazarse entre diapositivas

Tarea de ingeniería	
No: 38	**No. HU:** 9
Nombre: Desplazarse entre diapositivas.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 01
Fecha inicio: 27 de abril de 2014	**Fecha fin:** 02 de mayo de 2014
Programador responsable: Raúl Noa Pedroso.	
Descripción: Cambiar de diapositiva según las teclas presionadas por el usuario. Capturar que tecla presionó el usuario o si fueron los botones del *mouse* mediante jQuery.js. Funciones: full_Screen (), reset (), resize_Canvas (), show_Slide (). Clases: MiniSlide, Slide.	

Anexos

Anexo 5 – Pruebas unitarias

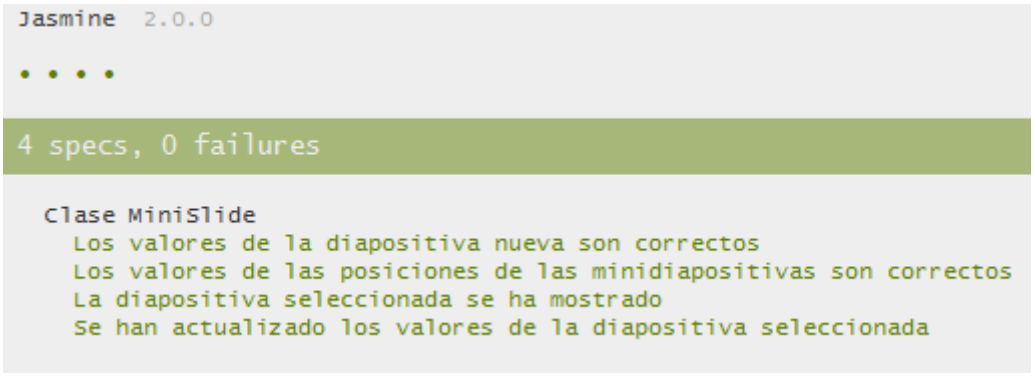

Figura 7: Resultado de la prueba unitaria realizada a la clase MiniSlide

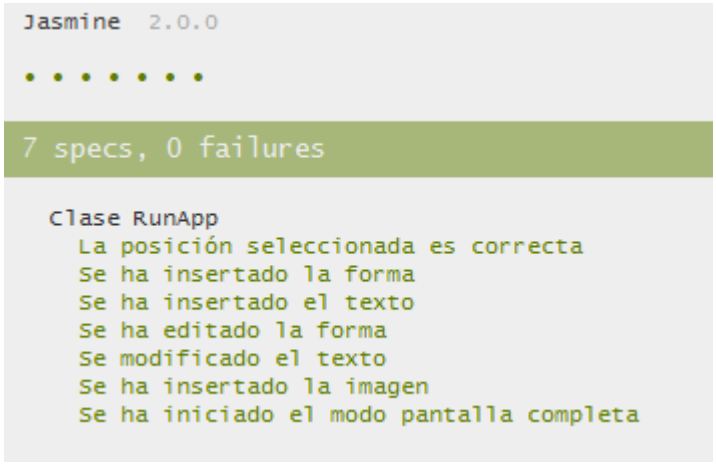

Figura 8: Resultado de la prueba unitaria realizada a la clase RunApp

Anexo 6 – Casos de prueba de aceptación

Tabla 65: CP - Dibujar rectángulo

Caso de prueba de aceptación	
No: 2	**No. HU:** 1
Nombre: Dibujar rectángulo.	
Descripción: Prueba para la funcionalidad que permita dibujar un rectángulo en la diapositiva activa.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción rectángulo.	

Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para dibujar un rectángulo se tiene que seleccionar la opción. Se presiona el botón izquierdo del *mouse* en el lugar de la diapositiva dónde se va dibujar el rectángulo y sin soltar el botón se traza el rectángulo.
Resultado esperado: El rectángulo se traza correctamente en el lugar y con el tamaño deseado por el usuario.
Evaluación: Prueba satisfactoria.

Tabla 66: CP - Dibujar círculo

Caso de prueba de aceptación	
No: 3	**No. HU:** 1
Nombre: Dibujar círculo.	
Descripción: Prueba para la funcionalidad que permita dibujar un círculo en la diapositiva activa.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción círculo.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para dibujar un círculo se tiene que seleccionar la opción. Se presiona el botón izquierdo del *mouse* en el lugar de la diapositiva dónde se va dibujar el círculo y sin soltar el botón se traza el círculo.	
Resultado esperado: El círculo se traza correctamente en el lugar y con el tamaño deseado por el usuario.	
Evaluación: Prueba satisfactoria.	

Tabla 67: CP - Dibujar elipse

Caso de prueba de aceptación	
No: 4	**No. HU:** 1
Nombre: Dibujar elipse.	
Descripción: Prueba para la funcionalidad que permita dibujar una elipse en la diapositiva activa.	

Condiciones de ejecución: El usuario tiene que haber seleccionado la opción elipse.
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para dibujar una elipse se tiene que seleccionar la opción. Se presiona el botón izquierdo del *mouse* en el lugar de la diapositiva dónde se va dibujar la elipse y sin soltar el botón se traza la elipse.
Resultado esperado: EL elipse se traza correctamente en el lugar y con el tamaño deseado por el usuario.
Evaluación: Prueba satisfactoria.

Tabla 68: CP - Dibujar estrella

Caso de prueba de aceptación	
No: 5	**No. HU:** 1
Nombre: Dibujar estrella.	
Descripción: Prueba para la funcionalidad que permita dibujar una estrella en la diapositiva activa.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción estrella.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para dibujar una estrella se tiene que seleccionar la opción. Se presiona el botón izquierdo del *mouse* en el lugar de la diapositiva dónde se va dibujar la estrella y sin soltar el botón se traza la estrella.	
Resultado esperado: La estrella se traza correctamente en el lugar y con el tamaño deseado por el usuario.	
Evaluación: Prueba satisfactoria.	

Tabla 69: CP - Editar color de relleno de la forma

Caso de prueba de aceptación	
No: 6	**No. HU:** 2
Nombre: Editar color de relleno de la forma.	

Descripción: Prueba para la funcionalidad que permita editar el color de relleno de la forma seleccionada según la opción seleccionada por el usuario.
Condiciones de ejecución: El usuario tiene que haber seleccionado una forma de la diapositiva activa.
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para editar el color de relleno de una forma, se selecciona la forma a través del botón izquierdo del *mouse*. Seguidamente aparece la opción de elegir un color de relleno. En el momento en que se escoja un color, este tiene que automáticamente ser el relleno de la forma seleccionada.
Resultado esperado: La forma seleccionada adopta el color elegido por el usuario.
Evaluación: Prueba satisfactoria.

Tabla 70: CP - Editar color de borde de la forma

Caso de prueba de aceptación	
No: 7	**No. HU:** 2
Nombre: Editar color de borde de la forma.	
Descripción: Prueba para la funcionalidad que permita editar el color de borde de la forma seleccionada según el color seleccionado por el usuario.	
Condiciones de ejecución: El usuario tiene que haber seleccionado una forma de la diapositiva activa.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para editar el color del borde de una forma, se selecciona la forma a través del botón izquierdo del *mouse*. Seguidamente aparece la opción de elegir un color de borde. En el momento en que se escoja un color para el borde, automáticamente este tiene que establecerse en la forma seleccionada.	
Resultado esperado: La forma seleccionada adopta el color para el borde elegido por el usuario.	
Evaluación: Prueba satisfactoria.	

Tabla 71: CP - Editar ancho del borde de la forma

Caso de prueba de aceptación	
No: 8	**No. HU:** 2
Nombre: Editar ancho del borde de la forma.	
Descripción: Prueba para la funcionalidad que permita editar el ancho del borde de la forma seleccionada según el valor establecido por el usuario.	
Condiciones de ejecución: El usuario tiene que haber seleccionado una forma de la diapositiva activa.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para editar el ancho del borde de una forma, se selecciona la forma a través del botón izquierdo del *mouse*. Seguidamente aparece la opción de establecer un valor para el ancho del borde. En el momento en que se establezca un valor para el borde, automáticamente se tiene que establecer ese ancho en la forma seleccionada.	
Resultado esperado: La forma seleccionada adopta como ancho del borde el valor establecido por el usuario.	
Evaluación: Prueba satisfactoria.	

Tabla 72: CP - Insertar texto

Caso de prueba de aceptación	
No: 9	**No. HU:** 3
Nombre: Insertar texto.	
Descripción: Prueba para la funcionalidad que permita insertar un texto en la diapositiva seleccionada.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción de insertar texto.	
Entrada / Pasos de ejecución: El texto a insertar por el usuario. Para insertar un texto en la diapositiva activa se selecciona la opción de insertar texto. Seguidamente se escribe el texto a insertar y después a través del botón izquierdo del *mouse* se selecciona la posición del texto en la diapositiva activa.	

Resultado esperado: El texto escrito por el usuario es insertado correctamente en el lugar de la diapositiva activa que el usuario definió.
Evaluación: Prueba satisfactoria.

Tabla 73: CP - Editar fuente del texto

Caso de prueba de aceptación	
No: 10	**No. HU:** 4
Nombre: Editar fuente del texto.	
Descripción: Prueba para la funcionalidad que permita editar la fuente del texto seleccionado.	
Condiciones de ejecución: El usuario tiene que haber seleccionado el texto.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para cambiar la fuente de un texto, se tiene que seleccionar este de la diapositiva. Seguidamente aparecen las distintas fuentes disponibles y se selecciona una fuente. En el momento en que el usuario selecciona una fuente esta se establece en el texto seleccionado.	
Resultado esperado: El texto seleccionado adopta la fuente elegida por el usuario.	
Evaluación: Prueba satisfactoria.	

Tabla 74: CP - Modificar contenido del texto

Caso de prueba de aceptación	
No: 11	**No. HU:** 4
Nombre: Modificar contenido del texto.	
Descripción: Prueba para la funcionalidad que permita modificar el contenido del texto seleccionado.	
Condiciones de ejecución: El usuario tiene que haber seleccionado el texto.	

Entrada / Pasos de ejecución: El texto que se quiere añadir. Para cambiar el contenido de un texto, se tiene que seleccionar este de la diapositiva. Seguidamente aparece una ventana con el contenido actual del texto y se puede cambiar este. En el momento en que el usuario cambia el contenido del texto tanto eliminando, agregando o modificando; este automáticamente cambia en la diapositiva activa.
Resultado esperado: El contenido del texto seleccionado se actualiza con las modificaciones del usuario.
Evaluación: Prueba satisfactoria.

Tabla 75: CP - Editar tamaño del texto

Caso de prueba de aceptación	
No: 12	**No. HU:** 4
Nombre: Editar tamaño del texto.	
Descripción: Prueba para la funcionalidad que permita editar el tamaño del texto seleccionado.	
Condiciones de ejecución: El usuario tiene que haber seleccionado el texto.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para cambiar el tamaño del texto, se tiene que seleccionar este de la diapositiva activa. Seguidamente aparece una opción con los tamaños de texto definidos y se puede seleccionar un valor. En el momento en que el usuario selecciona un valor, el tamaño del texto seleccionado adopta dicho valor.	
Resultado esperado: El tamaño del texto seleccionado adopta el valor seleccionado por el usuario.	
Evaluación: Prueba satisfactoria.	

Tabla 76: CP - Editar color del texto

Caso de prueba de aceptación	
No: 13	**No. HU:** 4
Nombre: Editar color del texto.	
Descripción: Prueba para la funcionalidad que permita editar el color del texto seleccionado.	
Condiciones de ejecución: El usuario tiene que haber seleccionado el texto.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para cambiar el color del texto, se tiene que seleccionar este de la diapositiva activa. Seguidamente aparece la opción para escoger un color por parte del usuario. En el momento en que el usuario selecciona un color, el color del texto seleccionado adopta dicho color.	
Resultado esperado: El color del texto seleccionado adopta el color seleccionado por el usuario.	
Evaluación: Prueba satisfactoria.	

Tabla 77: CP - Establecer estilo predefinido al texto

Caso de prueba de aceptación	
No: 14	**No. HU:** 4
Nombre: Establecer estilo predefinido al texto.	
Descripción: Prueba para la funcionalidad que permita establecer un estilo predefinido por la solución al texto seleccionado.	
Condiciones de ejecución: El usuario tiene que haber seleccionado el texto.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para establecer un estilo predefinido al texto, se tiene que seleccionar el texto de la diapositiva activa. Seguidamente aparece la opción para escoger un estilo de los definidos por la solución. En el momento en que el usuario selecciona un estilo predefinido, el texto seleccionado adopta dicho estilo.	

Resultado esperado: El texto seleccionado adopta el estilo predefinido seleccionado por el usuario.
Evaluación: Prueba satisfactoria.

Tabla 78: CP - Texto en negrita

Caso de prueba de aceptación	
No: 15	**No. HU:** 4
Nombre: Texto en negrita.	
Descripción: Prueba para la funcionalidad que permita establecer al texto seleccionado el estilo negrita.	
Condiciones de ejecución: El usuario tiene que haber seleccionado el texto.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para establecer el estilo negrita a un texto, se tiene que seleccionar este de la diapositiva. Seguidamente aparece la opción para establecer el estilo negrita. En el momento en que el usuario presiona la opción negrita, el texto seleccionado adopta este estilo. Si se desea remover este estilo se vuelve a realizar el paso anterior.	
Resultado esperado: El texto seleccionado adopta el estilo negrita.	
Evaluación: Prueba satisfactoria.	

Tabla 79: CP - Texto en cursiva

Caso de prueba de aceptación	
No: 16	**No. HU:** 4
Nombre: Texto en cursiva.	
Descripción: Prueba para la funcionalidad que permita establecer al texto seleccionado el estilo cursiva.	
Condiciones de ejecución: El usuario tiene que haber seleccionado el texto.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba.	

Para establecer el estilo cursiva a un texto, se tiene que seleccionar este de la diapositiva. Seguidamente aparece la opción para establecer el estilo cursiva. En el momento en que el usuario presiona la opción cursiva, el texto seleccionado adopta este estilo. Si se desea remover este estilo se vuelve a realizar el paso anterior.
Resultado esperado: El texto seleccionado adopta el estilo cursiva.
Evaluación: Prueba satisfactoria.

Tabla 80: CP – Alineación del texto

Caso de prueba de aceptación	
No: 17	**No. HU:** 4
Nombre: Alineación del texto.	
Descripción: Prueba para la funcionalidad que permita alinear el texto seleccionado.	
Condiciones de ejecución: El usuario tiene que haber seleccionado el texto.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Para alinear el texto, se tiene que seleccionar este de la diapositiva. Seguidamente aparecen las diferentes opciones para alinear un texto. En el momento en que el usuario escoge un tipo de alineación, el texto seleccionado adopta esta alineación.	
Resultado esperado: El texto seleccionado adopta la alineación escogida por el usuario.	
Evaluación: Prueba satisfactoria.	

Tabla 81: CP - Insertar imagen

Caso de prueba de aceptación	
No: 18	**No. HU:** 5
Nombre: Insertar imagen.	
Descripción: Prueba para la funcionalidad que permita insertar una imagen en la diapositiva activa.	

Condiciones de ejecución: El usuario tiene que haber seleccionado la opción insertar imagen.
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Se selecciona la opción insertar imagen. Seguidamente aparece la opción para buscar el lugar en dónde se encuentra la imagen a insertar. Con el botón izquierdo del *mouse* se presiona sobre el botón "Examinar", se selecciona la imagen y se presiona el botón "aceptar". Entonces se selecciona el lugar en la diapositiva activa para insertar la imagen.
Resultado esperado: La imagen seleccionada se inserta en el lugar de la diapositiva elegido por el usuario.
Evaluación: Prueba satisfactoria.

Tabla 82: CP - Girar objeto

Caso de prueba de aceptación	
No: 19	**No. HU:** 6
Nombre: Girar objeto.	
Descripción: Prueba para la funcionalidad que permita girar un objeto seleccionado de la diapositiva activa. El objeto puede ser una forma, un texto o una imagen.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción girar objeto.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Se selecciona la opción girar objeto. Seguidamente con el botón izquierdo del *mouse* se presiona sobre el objeto a girar y manteniendo el botón apretado se gira en el sentido de las manecillas del reloj.	
Resultado esperado: El objeto escogido tiene que haber girado en el sentido de las manecillas del reloj.	
Evaluación: Prueba satisfactoria.	

Tabla 83: CP - Trasladar objeto

Caso de prueba de aceptación	
No: 20	**No. HU:** 6
Nombre: Trasladar objeto.	
Descripción: Prueba para la funcionalidad que permita mover un objeto seleccionado de la diapositiva activa. El objeto puede ser una forma, un texto o una imagen.	
Condiciones de ejecución: El usuario tiene que seleccionar el objeto a trasladar.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Se selecciona la opción trasladar objeto. Seguidamente con el botón izquierdo del *mouse* se presiona sobre el objeto a trasladar y sin soltar el botón, desplazar el objeto por la diapositiva activa hacia su nueva posición.	
Resultado esperado: El objeto seleccionado se traslada a su nueva posición.	
Evaluación: Prueba satisfactoria.	

Tabla 84: CP - Modificar el tamaño de un objeto

Caso de prueba de aceptación	
No: 21	**No. HU:** 6
Nombre: Modificar el tamaño de un objeto.	
Descripción: Prueba para la funcionalidad que permita cambiar el tamaño de un objeto seleccionado.	
Condiciones de ejecución: El usuario tiene que seleccionar la opción escalar.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Se selecciona la opción zoom. Seguidamente se presiona con el botón izquierdo del *mouse* sobre el objeto y sin soltar el botón se mueve el puntero del *mouse* hacia arriba y hacia abajo para agrandar o disminuir el tamaño del objeto.	

Resultado esperado: El objeto seleccionado cambia su tamaño.
Evaluación: Prueba satisfactoria.

Tabla 85: CP - Eliminar objeto

Caso de prueba de aceptación	
No: 22	**No. HU:** 6
Nombre: Eliminar objeto.	
Descripción: Prueba para la funcionalidad que permita eliminar el objeto seleccionado.	
Condiciones de ejecución: El usuario tiene que seleccionar la opción eliminar.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Se selecciona la opción eliminar objeto. Seguidamente se presiona con el botón izquierdo del *mouse* sobre el objeto a eliminar y este es eliminado de la diapositiva.	
Resultado esperado: El objeto seleccionado es eliminado de la diapositiva.	
Evaluación: Prueba satisfactoria.	

Tabla 86: CP - Copiar objeto

Caso de prueba de aceptación	
No: 23	**No. HU:** 6
Nombre: Copiar objeto.	
Descripción: Prueba para la funcionalidad que permita copiar un objeto seleccionado.	
Condiciones de ejecución: El usuario tiene que seleccionar el objeto de la diapositiva.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Se selecciona el objeto de la diapositiva activa. Seguidamente se habilita la opción de copiar y el usuario puede	

realizar esta operación presionando sobre la opción con el botón izquierdo del *mouse* o con las teclas de acceso rápido.
Resultado esperado: El objeto seleccionado es copiado.
Evaluación: Prueba satisfactoria.

Tabla 87: CP - Cortar objeto

Caso de prueba de aceptación	
No: 24	**No. HU:** 6
Nombre: Cortar objeto.	
Descripción: Prueba para la funcionalidad que permita cortar un objeto seleccionado.	
Condiciones de ejecución: El usuario tiene que seleccionar el objeto de la diapositiva.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Se selecciona el objeto de la diapositiva activa. Seguidamente se habilita la opción de cortar y el usuario puede realizar esta operación presionando sobre la opción con el botón izquierdo del *mouse* o con las teclas de acceso rápido. Una vez cortado el objeto debe desaparecer de la diapositiva activa.	
Resultado esperado: El objeto seleccionado es cortado.	
Evaluación: Prueba satisfactoria.	

Tabla 88: CP - Pegar objeto

Caso de prueba de aceptación	
No: 25	**No. HU:** 6
Nombre: Pegar objeto.	
Descripción: Prueba para la funcionalidad que permita pegar un objeto.	
Condiciones de ejecución: El usuario tiene que haber copiado o cortado algún objeto.	

Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Si el usuario ha copiado o cortado algún objeto se habilita la opción de pegar y el usuario puede realizar esta operación presionando sobre la opción con el botón izquierdo del *mouse* o con las teclas de acceso rápido. El objeto puede ser pegado el número de veces que estime el usuario y en las diapositivas que el elija.
Resultado esperado: El objeto seleccionado es pegado.
Evaluación: Prueba satisfactoria.

Tabla 89: CP - Crear diapositiva

Caso de prueba de aceptación	
No: 26	**No. HU:** 7
Nombre: Crear diapositiva.	
Descripción: Prueba para la funcionalidad que permita crear una diapositiva.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona la opción crear diapositiva a través del botón izquierdo del *mouse*.	
Resultado esperado: Se crea una nueva diapositiva.	
Evaluación: Prueba satisfactoria.	

Tabla 90: CP - Eliminar diapositiva

Caso de prueba de aceptación	
No: 27	**No. HU:** 7
Nombre: Eliminar diapositiva.	
Descripción: Prueba para la funcionalidad que permita eliminar una diapositiva.	

Condiciones de ejecución: El usuario tiene que haber seleccionado la opción.
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona la opción eliminar diapositiva a través del botón izquierdo del *mouse*.
Resultado esperado: Se elimina la diapositiva junto con todos sus objetos.
Evaluación: Prueba satisfactoria.

Tabla 91: CP - Duplicar diapositiva

Caso de prueba de aceptación	
No: 28	**No. HU:** 7
Nombre: Duplicar diapositiva.	
Descripción: Prueba para la funcionalidad que permita duplicar una diapositiva.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona la opción duplicar diapositiva a través del botón izquierdo del *mouse*.	
Resultado esperado: Se duplica la diapositiva seleccionada junto con todos sus objetos y propiedades.	
Evaluación: Prueba satisfactoria.	

Tabla 92: CP - Desplazar diapositiva

Caso de prueba de aceptación	
No: 29	**No. HU:** 7
Nombre: Desplazar diapositiva.	
Descripción: Prueba para la funcionalidad que permita desplazar una diapositiva.	

Condiciones de ejecución: El usuario tiene que haber seleccionado la opción.
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona la opción desplazar diapositiva a través del botón izquierdo del *mouse*. Puede desplazarla hacia arriba o hacia abajo según su posición.
Resultado esperado: Se desplaza la diapositiva seleccionada junto con todos sus objetos y propiedades.
Evaluación: Prueba satisfactoria.

Tabla 93: CP - Editar color de fondo de la diapositiva

Caso de prueba de aceptación	
No: 30	**No. HU:** 7
Nombre: Editar color de fondo de la diapositiva.	
Descripción: Prueba para la funcionalidad que permita editar el color de fondo de la diapositiva seleccionada.	
Condiciones de ejecución: El usuario tiene que haber seleccionado una diapositiva.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona una diapositiva a través del botón izquierdo del *mouse*. Seguidamente aparecen las opciones para elegir un color de fondo o aplicar un degradado. En el momento que el usuario selecciona una de las opciones anteriores la diapositiva seleccionada adopta dicho color o degradado como fondo.	
Resultado esperado: La diapositiva seleccionada adopta el color o degradado elegido por el usuario.	
Evaluación: Prueba satisfactoria.	

Tabla 94: CP - Aplicar color de fondo a todas las diapositivas

Caso de prueba de aceptación	
No: 31	**No. HU:** 7
Nombre: Aplicar color de fondo a todas las diapositivas.	

Descripción: Prueba para la funcionalidad que permita aplicar un color de fondo a todas las diapositivas.
Condiciones de ejecución: El usuario tiene que haber seleccionado una diapositiva.
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona una diapositiva a través del botón izquierdo del *mouse*. Seguidamente aparece la opción para aplicar el color de fondo de dicha diapositiva a las demás restantes. Se selecciona esta opción a través del botón izquierdo del *mouse*.
Resultado esperado: La diapositiva seleccionada adopta el color o degradado elegido por el usuario.
Evaluación: Prueba satisfactoria.

Tabla 95: CP - Crear una nueva presentación

Caso de prueba de aceptación	
No: 32	**No. HU:** 8
Nombre: Crear una nueva presentación.	
Descripción: Prueba para la funcionalidad que permita crear una nueva presentación.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona la opción crear una nueva presentación a través del botón izquierdo del *mouse*.	
Resultado esperado: Se crea una nueva presentación.	
Evaluación: Prueba satisfactoria.	

Tabla 96: CP - Abrir una presentación

Caso de prueba de aceptación	
No: 33	**No. HU:** 8
Nombre: Abrir una presentación.	
Descripción: Prueba para la funcionalidad que permita abrir una presentación.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona la opción abrir una presentación a través del botón izquierdo del *mouse*. Seguidamente aparece una ventana para que busque la localización de la presentación guardada.	
Resultado esperado: Se abre la presentación guardada con todos sus elementos.	
Evaluación: Prueba satisfactoria.	

Tabla 97: CP - Guardar una presentación

Caso de prueba de aceptación	
No: 34	**No. HU:** 8
Nombre: Guardar una presentación.	
Descripción: Prueba para la funcionalidad que permita guardar una presentación.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona la opción guardar una presentación a través del botón izquierdo del *mouse*. Seguidamente aparece una ventana para que acepte guardar la presentación.	
Resultado esperado: Se guarda la presentación junto con todos sus elementos.	

Evaluación: Prueba satisfactoria.

Tabla 98: CP - Exportar presentación

Caso de prueba de aceptación	
No: 35	**No. HU:** 8
Nombre: Exportar presentación.	
Descripción: Prueba para la funcionalidad que permita exportar una presentación.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario selecciona la opción exportar presentación a través del botón izquierdo del *mouse*. Seguidamente aparece una ventana para que acepte exportar la presentación.	
Resultado esperado: Se exporta la presentación junto con todos sus elementos.	
Evaluación: Prueba satisfactoria.	

Tabla 99: CP - Alertar ante el cerrado de la aplicación al usuario

Caso de prueba de aceptación	
No: 36	**No. HU:** 8
Nombre: Alertar ante el cerrado de la aplicación al usuario.	
Descripción: Prueba para la funcionalidad que permita alertar al usuario ante el cerrado de la aplicación.	
Condiciones de ejecución: El usuario tiene que haber elegido la opción recargar el navegador o cerrar la aplicación.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. El usuario elige la opción recargar navegador o cerrar la aplicación a través del botón izquierdo del *mouse* o con la tecla *F5*. Seguidamente aparece una ventana para que acepte cerrar la aplicación o cancelar la operación.	

Resultado esperado: Aparece una ventana alertando al usuario para que acepte la operación o cancele la operación.
Evaluación: Prueba satisfactoria.

Tabla 100: CP - Visualizar presentación en pantalla completa

Caso de prueba de aceptación	
No: 37	**No. HU:** 9
Nombre: Visualizar presentación en pantalla completa.	
Descripción: Prueba para la funcionalidad que permita visualizar la presentación en pantalla completa.	
Condiciones de ejecución: El usuario tiene que seleccionar la opción de visualizar en pantalla completa de la aplicación o utilizar la del navegador web.	
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Se habilita la opción de pantalla completa, está se puede habilitar tanto desde la aplicación como desde la opción del navegador web. Para salir de esta opción se sale con la funcionalidad del propio navegador o presionando la tecla *Escape*.	
Resultado esperado: La presentación es visualizada en pantalla completa.	
Evaluación: Prueba satisfactoria.	

Tabla 101: CP - Desplazarse entre diapositivas

Caso de prueba de aceptación	
No: 38	**No. HU:** 9
Nombre: Desplazarse entre diapositivas.	
Descripción: Prueba para la funcionalidad que permita desplazarse entre las diapositivas de la presentación.	
Condiciones de ejecución: El usuario tiene que haber seleccionado la opción de visualizar la presentación en	

pantalla completa.
Entrada / Pasos de ejecución: No existen datos de entrada para esta prueba. Se tiene que haber habilitado la opción de visualizar la presentación en pantalla completa. Para desplazarse entre las diapositivas de la presentación el usuario lo puede realizar a través de las teclas del teclado *Espacio* y las flechas *hacia atrás* y *hacia delante* o mediante los botones del *mouse*.
Resultado esperado: El usuario de desplaza por las diapositivas de la presentación.
Evaluación: Prueba satisfactoria.

Anexo 7 – Entrevista semiestructurada

"*Las entrevistas semiestructuradas se basan en una guía de asuntos o preguntas y el entrevistador tiene la libertad de introducir preguntas adicionales para precisar conceptos u obtener mayor información sobre los temas deseados (es decir, no todas las preguntas están predeterminadas)*" (Hernández, Fernández y Baptista, 2006). A continuación se define la entrevista realizada a los administradores de diferentes sitios web de la Universidad y el resumen de los resultados obtenidos.

Guía de la entrevista aplicada

Lugar: Universidad de las Ciencias Informáticas

Entrevistador: Raúl Noa Pedroso

Introducción

Se realiza una exposición por parte del autor de la investigación sobre la propuesta a desarrollar.

Preguntas

1. ¿Está familiarizado con el uso de aplicaciones ofimáticas para la creación de presentaciones electrónicas?
2. ¿Qué aplicaciones ofimáticas conoce?
3. ¿Considera usted que existe algún problema producto de la utilización de diferentes aplicaciones ofimáticas para la creación de una presentación electrónica?
4. ¿Considera importante la posibilidad de poder visualizar presentaciones electrónicas desde un sitio web?

5. Suponga que cuenta con una aplicación para poder crear una presentación electrónica e insertarla en su sitio web sin la necesidad de depender de *plugins* u otros medios para su visualización ¿Considera importante la existencia de una aplicación para lograr este objetivo?
6. ¿Qué requerimientos básicos considera que debería poseer una aplicación de este tipo?

Resumen de las respuestas obtenidas

Los entrevistados estuvieron de acuerdo con la existencia de diferentes problemas para la creación de presentaciones electrónicas, debido a la variedad de aplicaciones ofimáticas para lograr este objetivo. En la mayoría de los casos estuvieron de acuerdo con la importancia de poder visualizar un recurso como las presentaciones electrónicas desde la Web. Manifestaron también que pese a que les gustaría poder dar la opción a los usuarios de acceder a diferentes contenidos mediante la visualización directa de este recurso desde sus sitios web, no conocían cómo o se les hacía engorroso utilizar las formas que conocían. Dieron como criterio positivo la posibilidad de contar con una aplicación que les permitiera crear una presentación electrónica que luego se pudiera visualizar en sus sitios web sin ninguna dependencia externa. Los entrevistados dieron su opinión acerca de los requerimientos básicos que debería tener una aplicación de este tipo, los cuales fueron tenidos en cuenta en la implementación de la solución propuesta.

Observaciones

Al final de la entrevista se le preguntó a cada entrevistado si tenía algo que agregar o alguna duda, se explicó lo que se iba a hacer con los datos recolectados y se insistió en la posibilidad de su participación durante el desarrollo de la solución propuesta.

Anexo 8 – Imágenes de la solución

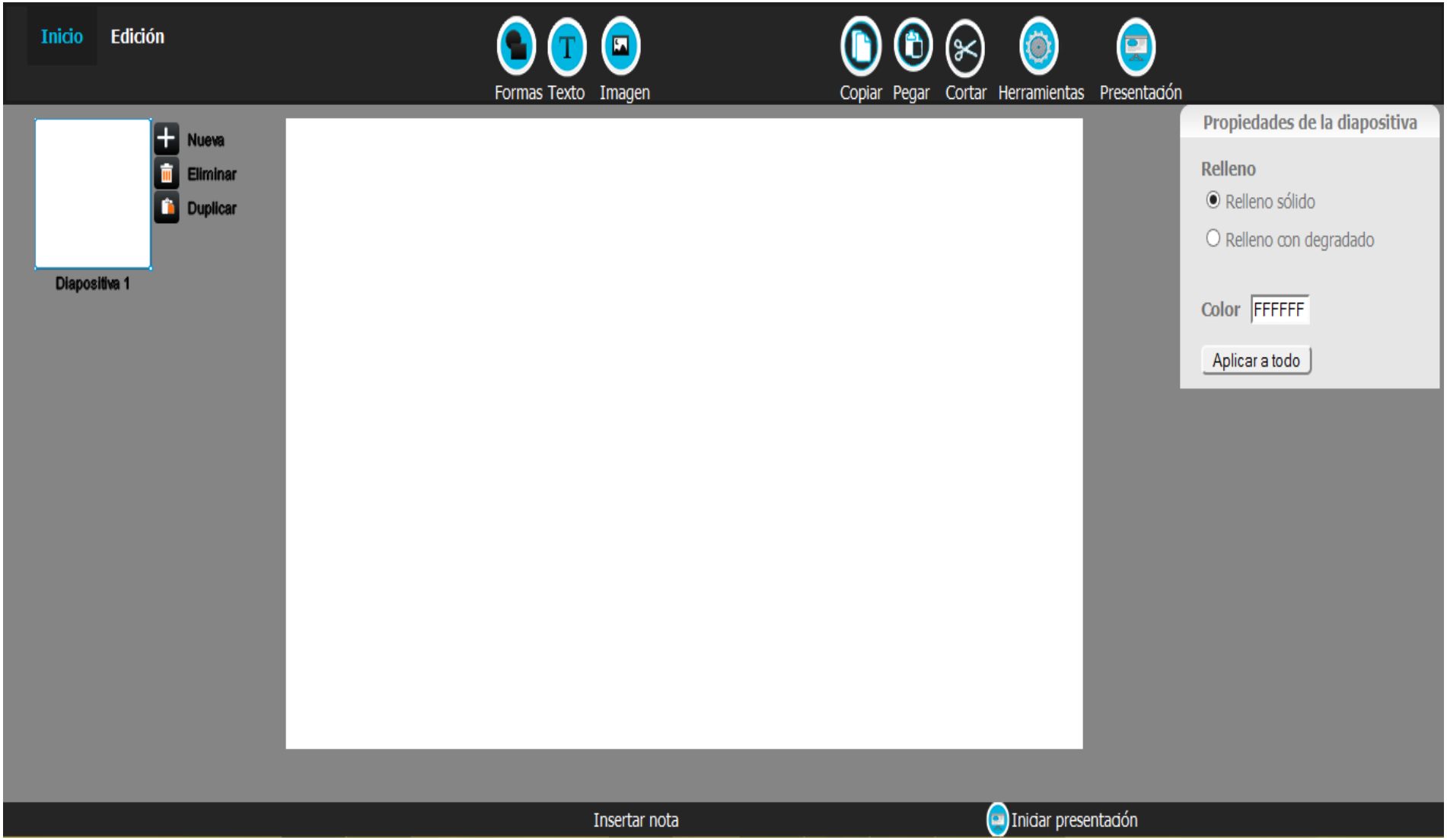

Figura 9: Interfaz principal de la solución

Figura 10: Ejemplo de utilización de las funcionalidades de la solución

Anexos

Anexo 9 – Avales

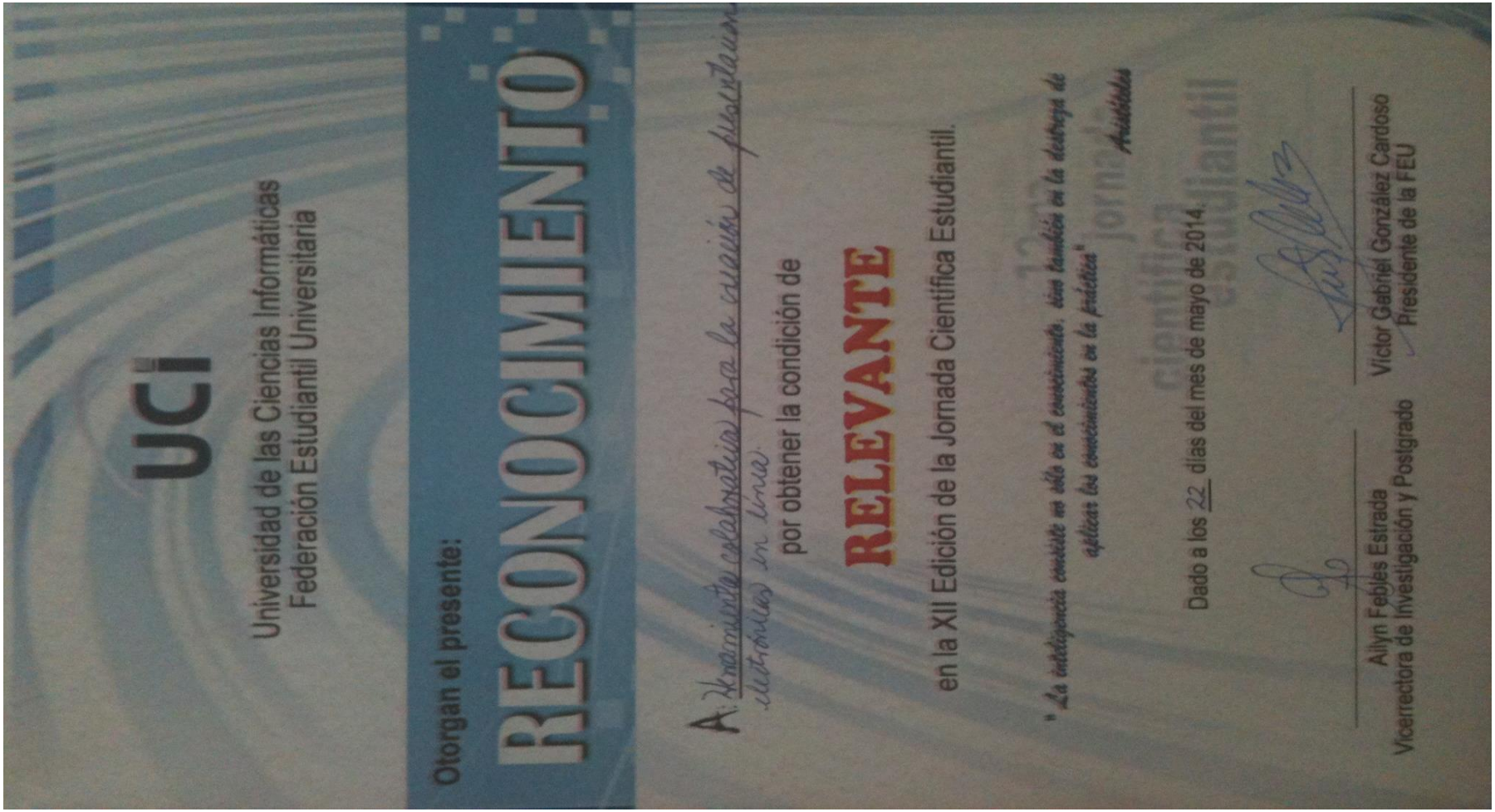

Figura 11: Reconocimiento obtenido en la XI Jornada Científica Estudiantil