Universidad de las Ciencias Informáticas

Facultad 4

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

Portafolio de Desarrollo Personal para la centralización de las evidencias de trabajo

Autores:

Pavel Alejandro Rodríguez Acosta

Eduardo Alejandro López Urquiaga

Tutores:

Ing. Mairelis Gari Maribona

Ing. Yerandy Manso Guerra

La Habana, Cuba.
Junio de 2014

Pensamiento

"Todos y cada uno de nosotros paga puntualmente su cuota de sacrificio consciente de recibir el premio en la satisfacción del deber cumplido, conscientes de avanzar con todos hacia el Hombre Nuevo que se vislumbra en el horizonte."

Ernesto Che Guevara

Declaración de Autoría

Declaramos ser los únicos autores de este trabajo y concedemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____
Pavel A. Rodríguez Acosta

_____
Eduardo A. López Urquiaga

_____
Ing. Mairelis Gari Maribona

_____
Ing. Yerandy Manso Guerra

Agradecimientos

De Eduardo

Agradezco a mis padres y mi hermana por todo el apoyo y cariño que me han brindado.

A Nadia por acompañarme durante estos cinco años.

A mi compañero de tesis, Pavel.

A mis amigos y a todas las personas que conocí en la universidad, porque de todos aprendí algo.

De Pavel

Agradezco a mis padres por haberme orientado en todos los momentos de mi vida, cuando tuve que tomar decisiones difíciles, por estar conmigo en las buenas y en las malas, y por sobre todo por ser los guías de mi futuro.

A mi hermano por tantas cosas que hemos hecho juntos y en tantos momentos y lugares que hemos compartido y que nunca podrán ser olvidados.

A una persona que me hizo ver la vida de una forma diferente y que por cosas del destino no podemos compartir las bondades que da la vida.

A todos mis abuelos, tíos y primos que a pesar de la distancia fueron un gran apoyo en todos estos años.

A Yami por ser mi fuente de inspiración en estos últimos momentos de la tesis.

A todos los compañeros de grupo, a los que pudieron llegar al final del camino y a los que ya no están y a todas las personas que de una forma u otra contribuyeron a mi formación como mejor persona.

Dedicatoria

De Eduardo

A mis padres.

De Pavel

A mis padres, por haberme educado tan bien y hacerme la persona que soy hoy.

Resumen

Desde su creación, los portafolios electrónicos han servido de apoyo a los distintos procesos sociales. Estas herramientas presentan las mismas características que los portafolios tradicionales, pero enfocado en el uso de las tecnologías digitales, con el fin de proporcionar un espacio para el desarrollo personal. La presente investigación está enfocada en la implementación de un portafolio de desarrollo personal, que pueda interactuar en ambientes de forma independiente o integrado a otros sistemas informáticos de distintos propósitos, con el objetivo de brindar una herramienta donde las personas puedan almacenar sus evidencias personales generadas a lo largo de su vida, permitiéndoles realizar una autorreflexión de su desarrollo personal. Para la implementación de la herramienta, se realizó un estudio basado en la situación actual de este tipo de sistemas, tanto en el ámbito nacional como internacional, con el objetivo de definir las principales características y funcionalidades presentes en las mismas. Para lograr la interoperabilidad de los datos almacenados entre los sistemas de este tipo, se utilizó el estándar Leap2A. Para la implementación de la herramienta se utilizó el lenguaje PHP en su versión 5.4.16, donde se integraron los *framework* jQuery 1.10, Bootstrap 3.1.1 y Symfony 2.3.7. Se utilizó como entorno de desarrollo Netbeans 7.4, como herramienta de modelado Visual Paradigm for UML 8.0 y la metodología XP para el proceso de desarrollo de software. Finalmente, se logró una herramienta capaz de integrarse con otros sistemas informáticos de distintos propósitos mediante la utilización de servicios web.

Palabras claves: portafolio electrónico, evidencias personales, interoperabilidad, integración.

Índice

Índice de ilustraciones

Índice de tablas

Introducción

El creciente uso de las Tecnologías de la Informática y las Comunicaciones (TIC) en la actualidad, ha propiciado un avance vertiginoso en distintas ramas de la sociedad y principalmente en la rama de la educación. A través del empleo de estas tecnologías, el proceso de enseñanza – aprendizaje ha podido desarrollarse de forma exitosa y la presentación de los trabajos personales, como parte fundamental de estos procesos, no se ha visto alejada de este increíble impacto en el que las TIC constituyen un medio para marcar la diferencia en cuanto a los métodos tradicionales de evaluación, basados en ambientes presenciales.

La nueva concepción del aprendizaje, basada en el uso de estas tecnologías, requiere que la presentación del trabajo personal sea uno de sus pilares fundamentales. De igual forma, se hace necesario la recopilación de información, que sirva para medir características del estudiante en el proceso educativo tales como: desempeño, evolución y comportamiento (Díaz Barriga, Romero y Heredia, 2011). En tal sentido, se requiere de una herramienta que sea capaz de brindar la evolución que se va observando en los estudiantes y docentes, con el fin de ofrecer un entorno colaborativo para poder encaminar acciones en pos de mantener un ascenso vertiginoso en el proceso evolutivo del desarrollo personal.

Por tanto, uno de los elementos más importantes es tener acceso a los resultados de las actividades personales de cada individuo, o sea, tener acceso a las evidencias[1] personales que se generan a lo largo del proceso, permitiendo obtener información de los logros y conocimientos alcanzados por los mismos. Ante la necesidad de englobar estos logros y conocimientos, procesar la información de los mismos e intercambiarlos entre distintos sistemas, es que surgen los portafolios electrónicos o e - Portafolio[2]. Con la utilización de este tipo de herramienta se pueden determinar avances en el aprendizaje, destrezas y habilidades de cada individuo y establecer o no el desarrollo de determinadas competencias.

Con el objetivo de utilizar las herramientas y servicios que brinda la red de redes, y en aras de agilizar y consolidar los procesos sociales, es que han tomado un elevado auge los e - Portafolio. Estos comenzaron con vagas expectativas, sobre si podían cumplir o no con el rol de sustituir a los portafolios tradicionales. Es por ello que se ha comenzado a desarrollar herramientas de este tipo, manteniendo los mismos objetivos de los antiguos portafolios, pero ahora desde un enfoque

[1] Nomenclatura de los elementos que se generan de la actividad de los usuarios en un sistema.
[2] Término que se utilizará a partir de este momento para referirse a los portafolios electrónicos.

más moderno y adaptado a las nuevas tecnologías de la información y las comunicaciones, viéndose beneficiados por las ventajas que proveen estas, lo cual les aporta accesibilidad a un mayor número de personas, presentación y contenido más interactivo, así como una capacidad de almacenamiento de información superior (Vargas y Carrillo, 2008). Estas acciones han constituido una revolución dentro de la rama educativa principalmente, aunque son utilizados para otros fines, dígase con fines comerciales y colaborativos; y son más cada día los que se suman a esta labor, tanto empresas privadas como comunidades de desarrollo.

Con el surgimiento de los e – Portafolio, surgieron cuestiones relacionadas con la interoperabilidad[3] y usabilidad de los mismos. Para ello se han desarrollado un conjunto de herramientas, especificaciones y estándares. Entre las especificaciones más utilizadas para estos fines se encuentran el IMS[4] e-Portafolio y el IMS Leap2A.

Los portafolios se muestran actualmente de dos formas esenciales, como componentes que pueden ser integrados a Sistemas de Administración del Aprendizaje (LMS, de sus siglas en inglés *Learning Management System*) o simplemente como herramientas individuales, que permiten al usuario crear su propio portafolio personal.

El uso de los portafolios en el mundo se ha desplegado hacia las distintas enseñanzas de la formación educativa, y la cifra de instituciones que utilizan esta técnica de aprendizaje es muy elevada, dando margen a la amplia utilización de este tipo de tecnología. En la región europea se manifiesta un uso muy difundido de los portafolios en la educación y países como España, Italia, Francia e Inglaterra, fungen como los que llevan la vanguardia en la utilización de estas tecnologías. Algunas de las universidades, de prestigioso currículum, que usan el e – Portafolio entre otras, se encuentran (Ramírez y Mendoza, 2005): Universidad de Stanford (USA[5]), Universidad de Nottingham (UK[6]), Universidad de Oeste Australia (AU[7]), Universidad de Virginia (USA), Universidad de Texas (USA), Universidad de Loughborough (UK).

La región latinoamericana y caribeña no se ha quedado rezagada en la utilización de las nuevas tecnologías educativas. Países como Argentina, México y Brasil despuntan como los que más han

[3] Es la capacidad que tiene un producto o un sistema, cuyas interfaces son totalmente conocidas, para funcionar con otros productos o sistemas existentes o futuros y eso sin restricción de acceso o de implementación.
[4] El consorcio *IMS Global Learning* y el conjunto de especificaciones.
[5] En sus siglas en inglés Estados Unidos de América.
[6] En sus siglas en inglés Reino Unido.
[7] En sus siglas en inglés Australia.

difundido y aplicado el uso de los e - Portafolio dentro de sus sistemas de enseñanza. Son varios los trabajos de diferente índole, tanto investigativos como prácticos, que se han presentado a la comunidad internacional, con el objetivo de mejorar y llegar a la idoneidad de las herramientas e – Portafolio.

Existen herramientas para la creación de portafolios electrónicos, así como componentes y aplicaciones informáticas que de una u otra forma incorporan características similares a las de un e – Portafolio, pero estas, en su gran mayoría, carecen de espacios donde el usuario pueda realizar la autorreflexión de sus resultados personales y académicos. No presentan opciones de personalización para todos los tipos de usuarios que interactúan con el sistema, por lo que no cumplen con las expectativas de los mismos. Dentro de las características de estas herramientas se hace notar que no presentan un intercambio de información de forma bidireccional cuando interactúan con otros sistemas informáticos.

La Universidad de las Ciencias Informáticas (UCI), es una institución creada para la formación de profesionales en la rama de la informática y las tecnologías, pero a su vez se le ha encomendado la misión de informatizar los procesos que intervienen en la sociedad cubana. Para ello se ha creado un modelo de formación que vincula el estudio con la producción y la investigación. Cuenta con varios centros de desarrollo de software para acometer dicha misión, enfocados en diferentes áreas sociales, y dentro de ellos, se encuentra el Centro de Tecnologías para la Formación (FORTES[8]), donde se desarrollan herramientas y sistemas informáticos para la rama educativa.

Como parte de la definición de la arquitectura de referencia para el desarrollo de aplicaciones en el Centro FORTES, se ha iniciado la formalización de marcos de trabajo en cada Línea de Productos de Software (LPS), con el propósito de disminuir la diversidad tecnológica de las soluciones. Dicho Centro cuenta con varios proyectos productivos, y uno de ellos es el proyecto Alfaomega, donde se ejecuta la implementación de la Plataforma Educativa Zera. Dicha plataforma cuenta con varios módulos, que incorporan funcionalidades incluidas en plataformas mundialmente conocidas como Sakai, Moodle y Dokeos. Uno de sus módulos es el Portafolio, cuya función es reunir las evidencias de aprendizaje que se generan a través de todas las actividades que tienen lugar en la plataforma por parte de los estudiantes.

El módulo Portafolio tuvo su concepción en el año 2011(Gallego y Guerrero, 2011), donde se realizó una propuesta de portafolio de evidencias, cuya idea fundamental se basaba en el

[8] Término que se utilizará a partir de este momento para referirse al Centro de Tecnologías para la Formación.

establecimiento de una estrecha relación entre las calificaciones y las evidencias subidas a la plataforma por parte de los usuarios. En este paso de la evolución del módulo Portafolio, se identificaron los instrumentos de evaluación que serían utilizados para desarrollar el proceso educativo dentro de la Plataforma Educativa Zera, y se implementaron las funcionalidades que el cliente estimaba conveniente en aquel momento.

Con el avance de la concepción del aprendizaje en la educación, los e – Portafolio se convirtieron en herramientas esenciales para llevar a cabo este proceso. Es por ello que en el año 2012 se logran integrar los instrumentos de evaluación, que brindaba la Plataforma Educativa Zera, con el módulo Portafolio, logrando la interoperabilidad dentro de la misma, y con una arquitectura basada en los estándares más utilizados para ello (Calderín y Pérez, 2012). Aunque se lograron los objetivos que se trazaron en ese momento: lograr que se almacenaran todas las evidencias de aprendizaje alcanzadas por el estudiante, permitir consultar sus datos personales y estar implementado bajo las especificaciones de los estándares anteriormente mencionados; dicho módulo se muestra, actualmente, totalmente dependiente de la Plataforma Educativa Zera, debido a que la arquitectura y el modelo de datos que lo componen no hacen posible su integración con otros sistemas informáticos similares. Además este módulo no incorpora un portafolio para los restantes roles que intervienen en la plataforma, y con ello lograr que estos también preserven evidencias de desempeño, habilidades y logros alcanzados. Está exento de funcionalidades que permitan una mayor personalización de su espacio de trabajo. Además la intención con la que se creó el anterior módulo es completamente educativa.

En el Centro no existe una herramienta capaz de centralizar las experiencias y resultados del trabajo personal de cada individuo. Las herramientas de autor, Creando Objetos de Aprendizaje (CRODA) y Repositorio de Objetos de Aprendizaje (RHODA), proveen a los docentes de funcionalidades para la creación y almacenamiento de sus Objetos de Aprendizaje (OA), pero solo pueden almacenar en estas herramientas aquellos recursos que sean validados para su publicación, por lo que si el docente desea almacenar o publicar otro trabajo que esté realizando, y que no sea validado por estas herramientas, pues no tiene un espacio donde ejecutar estas acciones.

Una vez analizadas las necesidades existentes, se plantea como **problema de investigación:** ¿Cómo facilitar el almacenamiento y análisis de evidencias del trabajo personal entre sistemas informáticos de distintos propósitos?

El **objeto de estudio** se enmarca en los sistemas informáticos para la recopilación y análisis de evidencias del trabajo personal y el **campo de acción** se enfoca en los portafolios de desarrollo personal.

Por todo lo planteado, para darle solución al problema de investigación, se plantea como **objetivo general:** desarrollar un portafolio de desarrollo personal que permita el almacenamiento y análisis de evidencias de trabajo y que pueda integrarse, a su vez, con otros sistemas informáticos.

Para dar cumplimiento al objetivo general, se plantean como **objetivos específicos:**

- ✓ Construir los referentes teóricos relacionando los aspectos fundamentales que sustentan la investigación, mediante los cuales se consulta, extrae y recopila la información relevante sobre el problema a investigar.
- ✓ Desarrollar funcionalidades y componentes que contribuyan a la obtención de un portafolio de desarrollo personal, que pueda integrarse con sistemas informáticos desarrollados en el Centro, a partir del diseño realizado y de los requisitos definidos.
- ✓ Comprobar el funcionamiento del sistema mediante pruebas de software.

Para desarrollar la investigación, se emplearán los siguientes **métodos investigativos:**

Métodos teóricos

- **Análisis histórico – lógico:** el uso de este método permite conocer la evolución que han tenido los portafolios de desarrollo personal desde su concepción, como una forma más de presentar el desempeño y los logros alcanzados por los individuos. Además brinda la posibilidad de investigar acerca de los estándares y tendencias tecnológicas actuales en el campo de los portafolios.
- **Analítico – sintético:** a través de este método se hace posible el procesamiento de la teoría y la documentación para conformar el marco teórico y la definición de los elementos fundamentales relacionados a los portafolios de desarrollo personal.
- **Inductivo – deductivo:** posibilita determinar las características de los sistemas encargados del análisis y almacenamiento de evidencias de trabajo, que puedan ser aplicables a un portafolio de desarrollo personal, y además permitirá la descripción e implementación de las funcionalidades identificadas.

- **Modelación:** hace posible desarrollar la modelación de los principales procesos que intervienen en el sistema, a través de la abstracción y la posterior representación de la realidad.

Métodos empíricos

Se utiliza la técnica de la entrevista, para identificar lo que se quiere lograr con el desarrollo del portafolio.

- **Entrevista:** se aplica este método para identificar los requisitos funcionales que requiera el cliente para el desarrollo de la herramienta portafolio.

Resultados esperados:

Con la realización del presente trabajo se pretende crear las bases para fortalecer las herramientas de gestión del aprendizaje desarrolladas en el Centro FORTES. Se espera obtener un portafolio de desarrollo personal que pueda integrarse con otros sistemas informáticos, permitiendo la interoperabilidad de los datos almacenados y que permita realizar un análisis sobre las evidencias personales almacenadas.

El presente trabajo está estructurado en tres capítulos, desarrollados a partir del estudio realizado.

Capítulo I: Fundamentación teórica.

En este capítulo se hace referencia a los elementos teóricos que soportan la investigación. Además se realiza un estudio del arte de la evolución y situación actual de las herramientas cuyo objetivo se centra en el almacenamiento y análisis de evidencias del trabajo personal. Se describen soluciones similares existentes y se realizan comparaciones para llegar a la propuesta de solución que sustenta la investigación. Se presentan los lenguajes de programación y modelado, así como los estándares y tecnologías que se utilizarán para desarrollar la propuesta planteada, fundamentando su selección en base al estudio realizado.

Capítulo II: Presentación de la propuesta de solución.

En este capítulo se realiza una descripción de las características del sistema a implementar. Se especifican los requisitos funcionales y no funcionales que debe cumplir la aplicación. También se muestra el modelo conceptual y la descripción de los principales conceptos que intervienen en la construcción del sistema. Se brinda una propuesta de las secciones por las que estará compuesto

el sistema una vez concluida el proceso de implementación.

Capítulo III: Implementación y prueba.

Este capítulo presenta lo relacionado con el proceso de implementación de las funcionalidades identificadas, sobre la base del lenguaje de programación y metodología seleccionadas. Se especifica la arquitectura seleccionada, así como los patrones de diseño e implementación aplicados. Se definen los tipos de pruebas que se aplicarán al sistema y los resultados que arrojan las mismas luego de su aplicación y se especifica el uso de los servicios por los que estará compuesta la aplicación.

Capítulo 1: Fundamentación Teórica

El primer paso para comenzar una investigación es realizar el estudio del arte. En el caso de la presente investigación, se hace imprescindible conocer el estado y uso de los portafolios electrónicos y específicamente, los portafolios de desarrollo personal, tanto fuera como dentro de nuestro ámbito nacional. Para ello se debe planificar un sistema de búsqueda y análisis de información, mediante el cual pueda nutrirse la investigación y crear una base teórica sobre los elementos que aporte esta estrategia. Se necesita establecer las características y principales rasgos que presentan este tipo de herramientas, para lograr dar una solución al problema de investigación planteado, que cumpla con los requisitos esenciales para su puesta en práctica. Para determinar cómo realizar la investigación, se hace necesario además hacer un estudio de las herramientas, tecnologías y estándares que son utilizados en el desarrollo de este tipo de aplicación. Se debe definir la metodología de desarrollo, así como los lenguajes de desarrollo y herramientas para implementar la solución de dicho estudio.

1.1. Portafolio electrónico

El portafolio electrónico, también definido como portafolio digital, e – Portafolio o Webfolio[9], aunque contiene el mismo material que un portafolio tradicional, los recursos pueden ser capturados, organizados, guardados y presentados digitalmente. Los portafolios son instrumentos usados para diferentes fines en una variedad de contextos de aprendizaje, la evaluación, valoración y promoción (Vargas y Carrillo, 2008).

Enfocándose en el uso educativo de los e – Portafolio se puede decir que es un método de enseñanza y aprendizaje, que consiste en el aporte de producciones de diferente índole por parte del estudiante a través de las cuales se pueden juzgar sus capacidades en el marco de una disciplina o materia de estudio. Estas producciones informan del proceso personal seguido por el estudiante, permitiéndole a él y los demás ver sus esfuerzos y logros, en relación a los objetivos de aprendizaje y criterios de evaluación establecidos previamente (Hernández, 2006).

Otros autores expresan que el e-Portafolio es una herramienta didáctica fundamental para posibilitar procesos de auto-reflexión sobre fortalezas, debilidades, dificultades, progresos y éxitos de cada estudiante (Gallego, 2011), lo que facilita una buena conjugación con la evaluación por competencias al considerarse que esta “debe servir a una finalidad última en el proceso de aprendizaje: hacer más conscientes a los estudiantes de cuál es su nivel de competencias, de cómo resuelven las tareas y de qué puntos fuertes deben potenciar y qué puntos débiles deben corregir para enfrentar situaciones de aprendizaje futuras” (Cano, 2011).

[9] “Enseñanza basada en Web”. Pero luego de la discusión de algunos autores se resolvió mediante el uso común del término *e – learning* para englobar a ambos.

Como resultado de las anteriores definiciones, se puede afirmar que el portafolio electrónico se muestra como una herramienta capaz de ser usada en distintos contextos, como herramienta de promoción de información personal y comercial, además de ser utilizada para fines educativos con el objetivo de apoyar los procesos de aprendizaje a distintos niveles de escolaridad y para la presentación de contenidos de forma digital.

1.1.1. Tipos de portafolio implementados bajo tecnología web

Algunos autores señalan que para determinar el tipo de portafolios con el que se trabaja es muy importante establecer su propósito (Wray, 2008). En correspondencia con lo anterior, otros autores han planteado que la forma y el sentido del portafolio depende del objetivo y la finalidad que se busque (Barragán, 2005). La mayoría de la bibliografía consultada coincide en que existen portafolios académicos, civiles, profesionales y vitales. A continuación se especifica el tipo de portafolio el cual dará solución al problema planteado en la presente investigación.

Portafolio de Desarrollo Personal

Según la JISC-CETIS[10], existe otro tipo de portafolio electrónico, definido con el nombre de portafolio de desarrollo personal (Calderín y Pérez, 2012). Estos autores alegan además que un portafolio para la planificación del desarrollo personal contiene registros de aprendizaje, el desempeño, los logros y el resultado de esa reflexión, incluidos los planes para el desarrollo futuro. Este podría incluir un portafolio de aprendizaje, pero va más allá de eso, ya que suele estar relacionada con el desarrollo profesional y el empleo, por lo que también, probablemente, es utilizada como portafolio de presentación. Por las características que debe presentar la solución informática para la presente investigación, puede nutrirse de funcionalidades presentes en portafolios de tipo presentación y vitrina. El primero brinda la posibilidad de mostrar los mejores trabajos del alumno, escogidos bajo su criterio, que debe justificar su elección. De este modo el alumno aprende a ser crítico con su trabajo y a autoevaluarse. Al mostrar sus mejores trabajos, en una presentación o en la web, el alumno mejora su autoestima y aumenta su motivación (Gregori, 2008). Mientras que el segundo se basa principalmente en la organización de las evidencias e información personal en espacios denominados vitrinas, dando la posibilidad al usuario de una mejor presentación de estos elementos.

De forma general se puede afirmar que el portafolio de desarrollo personal se muestra como un interesante instrumento para la organización y promoción de la información personal y una guía para el individuo, permitiéndole la autoevaluación y autorreflexión personal, para cumplir metas y objetivos en su desarrollo personal. Además permite la presentación de los trabajos y resultados personales.

[10] Organización encargada de los estándares para la educación en el Reino Unido.

1.1.2. Enfoques de un portafolio de desarrollo personal

Un portafolio de desarrollo personal se divide en espacios o secciones, enfocados en tres tipos fundamentales (Calvo, 2014):

- **Espacio personal:** es donde el individuo plasma libremente todo lo que quiera acerca de su persona, sus intereses, gustos, entre otros; la idea es que en este espacio se pueda reflejar parte de su personalidad.
- **Espacio profesional:** es donde el individuo plasma sus actividades extracurriculares, tiene un espacio para hablar de las prácticas profesionales que realizó, y de igual forma, si está desarrollando un trabajo o ponencia, puede incluir la sinopsis de estos en el espacio. Además puede quedar plasmado los planes para el desarrollo futuro y lo que desea lograr en su actividad profesional.
- **Espacio académico:** este es el lugar que va más relacionado con las materias que se haya cursado; la información que se encuentra plasmada es la que tiene que ver con las asignaturas. Se encuentra más orientado a los procesos de enseñanza – aprendizaje y los recursos que de estos se derivan.

1.2. Estudio de sistemas similares

Actualmente en el mundo se encuentra muy difundido el uso de portafolios electrónicos, con el objetivo de apoyar y gestionar, de alguna forma, los procesos de enseñanza – aprendizaje que se aplican en las distintas enseñanzas de la ciencia educativa, teniendo en cuenta que la mayor cantidad de estas herramientas presentan, dentro de sus propósitos, estar enfocadas para fines educativos. Para ello se han implementado herramientas y aplicaciones informáticas que ayuden a satisfacer el objetivo antes mencionado. Dichas herramientas se manifiestan de diferentes maneras, de acuerdo al uso que se le da en un determinado contexto social. La mayoría no cumplen con las especificaciones para llegar a ser catalogadas como portafolios electrónicos, pero presentan características similares a estos. Se pueden encontrar como editores web o herramientas web de uso general.

1.2.1. Análisis de soluciones existentes en Cuba

Módulo Portafolio - Plataforma Educativa Zera

Este sistema fue desarrollado en el marco del proyecto productivo Alfaomega, perteneciente al Centro FORTES de la UCI. Su funcionamiento está basado en la integración de instrumentos de evaluación[11] presentes en la Plataforma Educativa Zera (Calderín y Pérez, 2012), donde se

[11] Recursos que se emplean para recolectar y registrar información sobre un proceso.

obtiene la información necesaria de cada uno de ellos para ser utilizada como evidencia de aprendizaje y desempeño del estudiante.

Dicho módulo permite exportar los datos del portafolio para cada usuario, bajo las especificaciones del estándar Leap2A. Conforma los expedientes de estudiantes, incluyendo no solo su perfil, sino también su desempeño y comportamiento ante el estudio.

Brinda la posibilidad de realizar un análisis general del avance del aprendizaje, apoyándose en las competencias que debe vencer el estudiante. Se presenta como una herramienta de aval, ya que muestra al estudiante tal y como es, y los logros que ha alcanzado. El módulo presentado anteriormente, permite el almacenamiento y análisis de evidencias y la presentación de las mismas.

1.2.2. Análisis de soluciones existentes en el mundo

Editores web

Los editores web son herramientas que proveen una serie de funcionalidades, que van desde generación de código fuente hasta la presentación de simples plantillas, donde pueden crearse espacios de trabajo y sitios personales con un muy buen nivel funcional (Comunidades Virtuales, 2013). Las herramientas para la creación de e – Portafolio que a continuación se presentan, fungen dentro del grupo de editores web ya que presentan características similares a las descritas anteriormente.

Edu – portfolio

Es un portafolio de tipo vitrina, donde la información se organiza en los diferentes espacios denominados vitrinas. Se presenta como una herramienta portafolio de forma independiente y accesible vía web. Está diseñado para distintos niveles de escolaridad, haciéndolo flexible para su interacción de acuerdo al tipo de usuario que lo utilice. En cada vitrina de presentación, el usuario tiene la posibilidad de incluir los resultados y logros de sus cursos, así como los archivos y elementos que considere necesario para su desarrollo personal. (*Edu-portfolio.org: Su portafolio electrónico,* 2013)

Dicha herramienta facilita la organización, el intercambio y la búsqueda de información. Ofrece también diferentes niveles de protección (el contenido puede ser público, protegido o archivado), así como una interfaz para los tutores. Integra un motor de búsqueda que el propietario puede activar o no, facilitando la búsqueda de información en el interior del portafolio.

El mencionado portafolio, a pesar de no presentarse como un portafolio de desarrollo personal en su totalidad, presenta características y funcionalidades a través de las cuales se puede nutrir la solución final propuesta para la presente investigación.

A continuación se muestra una tabla comparativa, donde se exponen algunas herramientas e – Portafolio, presentando características inherentes a portafolios electrónicos como lo son la recolección, reflexión, presentación y compartición de evidencias, con el objetivo de establecer las diferencias entre cada una de ellas (Muñoz, Sánchez, Sahagún y Bria, 2008).

Tabla 1. Características de las herramientas e – Portafolio.

Herramientas e – Portafolio	Recolección	Reflexión	Organización y Presentación	Compartición de evidencias	Funcionamiento
Mahara	- archivos (*Files*). - permite blogs de usuario con post incluidos. - formularios guiados	- admite la descripción genérica del portafolio y la del resto de los elementos.	- ofrece la posibilidad de múltiples portafolios. - permite la clasificación de las evidencias.	- se puede compartir el portafolio en su totalidad, estableciendo a quien compartir los recursos.	- portafolio independiente. - integrado al LMS Moodle vía SSO[12].
MyStuff	- formularios guiados. - notas con archivos adjuntos.	- admite la descripción genérica del portafolio y la del resto de los elementos.	- ofrece la posibilidad de múltiples portafolios. - permite la clasificación de las evidencias.	- puede compartirse tanto el portafolio como evidencias individuales	- se integra al LMS Moodle en forma de bloque.
SPDC Portafolio[13]	- archivos (*Files*). - texto en línea.	- permite comentar el portafolio, así como las evidencias.	- admite la creación de múltiples portafolios. - no permite la organización de la presentación de evidencias.	- ofrece compartir entre usuarios el portafolio en su conjunto como el resto de sus recursos, a distintos niveles de compartición.	- se integra al LMS Moodle en forma de bloque.
Exabis	- archivos (*Files*). - textos en línea. - urls.	- los vínculos y los archivos admiten comentarios.	- organiza las evidencias en un único portafolio. - no presenta	- permite compartir evidencias entre usuarios,	- se integra al LMS Moodle en forma de bloque.

[12] *Single Sign On*, procedimiento de autenticación que habilita al usuario para acceder a varios sistemas con una sola instancia de identificación.
[13] Herramienta integrada a la plataforma Moodle. Permite la activación de los bloques "*Portafolio Keeper*" (portafolio en sentido estricto) y "*File Cabinet*" (repositorio de archivos).

			control en cómo se muestran las evidencias.	permitiendo hacer comentarios.	

Del anterior estudio se puede concluir que las mencionadas herramientas e – Portafolio presentan características comunes tales como la recolección y reflexión sobre las evidencias. Exabis, SPDC Portafolio y MyStuff presentan la característica de poder integrarse en forma de bloque o módulo al LMS Moodle, pero no dan la posibilidad de ser utilizados como herramientas independientes. Mahara sí presenta ambas posibilidades, puede funcionar como herramienta e - Portafolio independiente y además puede integrarse vía SSO con el mencionado LMS. Este tipo de integración no es la que se requiere para emplear en la solución informática de la presente investigación, puesto que requiere de una configuración previa de parámetros para establecer la integración entre ambas aplicaciones (Mahara y LMS Moodle), lo que no satisface el objetivo que se persigue con la presente investigación, brindar la posibilidad de integrarse con otras herramientas permitiendo la retroalimentación bidireccional de información a través de servicios web activos.

Herramientas web de uso general

Dentro de este tipo de herramientas se encuentran el conjunto de aplicaciones gratuitas en la web, *Google Drive* y *Google Sites*. El primero permite la creación de archivos en línea, dígase presentaciones, hojas de cálculo, carpetas, formularios y dibujos; mientras que el segundo permite la creación de páginas web. Ambos recursos representan una excelente opción para generar el e – Portafolio.

Las *Wikis* son un tipo de servicio web cuya principal característica es la construcción de contenidos de manera colaborativa, organizada mediante una estructura hipertextual de páginas. Esta es una buena opción si se requiere de la conformación grupal del e – Portafolio. Además este recurso permite incluir diversos formatos para la entrega de evidencias.

Dentro de este ramal de herramientas están presentes sitios web como *Weebly*[14], *Yola*[15] y *Webnode*[16], sitios que permiten la creación de páginas web en las que pueden incluirse todo tipo de evidencias y en el espacio del blog, la realimentación y/o discusión que pueda surgir de la actividad. Suelen utilizarse más como e – Portafolio tipo vitrina, que se caracterizan por contener las mejores evidencias obtenidas al final del proceso.

De forma general, las herramientas antes descritas presentan características que pueden ser

[14] Acceso en la internet: http://www.weebly.com/
[15] Acceso en la internet: https://www.yola.com/es
[16] Acceso en la internet: http://www.webnode.mx/

utilizadas para la conformación del portafolio de desarrollo personal, tales como la forma en que organizan y permiten crear evidencias de todo tipo, dígase documentos y archivos multimedia. Además, la manera en que presentan la información, se muestra como un recurso a tener en cuenta para su utilización en la implementación.

1.3. Análisis de los LMS que incluyen e – Portafolio.

Como fundamento al objetivo general de la presente investigación, a continuación se hará un estudio de las plataformas educativas que tengan incorporado un portafolio electrónico. Todo esto persigue como objetivo realizar un análisis de cuáles son las principales características que presentan los portafolios electrónicos integrados a LMS, y cómo exportan los datos de los estudiantes, para arribar a una conclusión y determinar cuál de los estándares se debe utilizar en la implementación del portafolio de desarrollo personal.

Para analizar las características que presentan los e - Portafolio que se encuentran integrados a plataformas educativas, se realizó un estudio a través de diversas fuentes (Romero y Troyano, 2014), (FORTES, 2011), (Prendes, 2009), (S.L, 2011), (Gonzáles, 2008), (Remón, 2013), y se identificaron las siguientes plataformas: Blackboard / WebCT, Moodle, Sakai, A Tutor, Docebo, Dokeos, Claroline y Plataforma Educativa Zera. Luego de haber seleccionado las plataformas se presenta el siguiente cuadro resumen:

Tabla 2. Plataformas seleccionadas para el análisis del uso de los e - Portafolio.

Plataformas seleccionadas	Incluye portafolio	Tipo de portafolio
Blackboard / WebCT	Sí	Académico y Evaluación.
Moodle	Sí, integrado con Mahara Portafolio.	Académico (aunque puede utilizarse como evaluación).
Sakai	Cuenta con una herramienta para su creación.	Evaluación y Académico.
Dokeos	No	-
A Tutor	Utiliza la herramienta MyStuff para su creación.	Académico
Docebo	Sí	Presentación
Zera	Sí	Presentación y Académico
Claroline	No	-

De la tabla comparativa se seleccionaron las siguientes plataformas: Blackboard / WebCT, Moodle, A Tutor, Docebo, Sakai y Zera, por tener incluido un portafolio electrónico o alguna herramienta que permita su creación. Se descartó la posibilidad de análisis de las plataformas Dokeos y Claroline, por no incluir componentes con función de e - Portafolio.

Después de haber analizado las características de los e - Portafolio incluidos en las plataformas

seleccionadas, se evidencia que tanto en WebCT como en Moodle el alumno cuenta con una página personal donde puede incluir sus trabajos reflexivos y datos personales.

Por su parte, la plataforma educativa Sakai, cuenta con la herramienta Portafolio de Código Abierto (OSP, en sus siglas en inglés *Open Source Portafolio*)[17], la cual permite al alumno crear una extensa página con contenido multimedia de su trabajo y progresión, que puede ser accedida tanto desde dentro de la plataforma como fuera del ámbito de la misma. Además dispone de numerosas plantillas que ayudan al estudiante a crear y divulgar su progresión y los estudios realizados.

1.4. Selección del estándar a implementar para el portafolio electrónico

La elaboración de un portafolio es importante como parte de la presentación de los resultados de trabajos personales, pero el producto final no debería ser algo sobre lo que el individuo pierda el control una vez que no haga uso de esta herramienta. Los usuarios deberían tener la posibilidad de "transportar" su portafolio de un contexto a otro, o de una herramienta a otra, de forma que tenga independencia para continuar con su elaboración. Por lo tanto, un e – Portafolio ideal debería permitir algún tipo de exportación de sus contenidos y estructura (Muñoz, Sánchez, Sahagún y Bria, 2008).

Para realizar la exportación de contenidos dentro de los portafolios electrónicos, debe cumplirse con un estándar establecido para ello. Luego de seleccionadas las herramientas para la creación de e - Portafolio y Plataformas Educativas en las secciones anteriores, se realizó un estudio con el objetivo de elegir el estándar mediante el cual se exportan los contenidos dentro de este tipo de herramienta. Además se tuvo en cuenta otros sistemas que presentan funcionalidades que permiten la creación de espacios similares a portafolios como son: PebblePad[18], Myprogressfile[19] y MyKnowledge Map[20]. Para el estudio que se presenta a continuación, se escogieron los estándares más utilizados para ello, de acuerdo con la organización británica antes mencionada, JISC - CETIS, el cual se encuentra fundamentado a través del siguiente cuadro comparativo:

Tabla 3. Comparativa para inferir que estándar implementar dentro del portafolio.

	Exporta a					
	SCORM	HTML	IMS LIP	IMS e – Portafolio	IMS Leap2A	Aspectos de interés
Blackboard / WebCT	X					

[17] Herramienta incluida en la plataforma educativa Sakai para la creación de portafolios personales.
[18] http://www.pebblepad.co.uk/.
[19] http://www.myprogressfile.com/
[20] http://www.eportfolio.eu/organisations/my-knowledge-map.

Moodle						Utiliza la herramienta Exabis para exportar el contenido.
A Tutor						Utiliza la herramienta MyStuff.
Docebo						No brinda la opción de exportar.
Sakai		X				
Zera						Utiliza el módulo Portafolio
Herramientas						
Mahara		X			X	
Myprogressfile					X	
MyKnowledge Map					X	
PebblePad					X	
Exabis	X					
Portafolio Zera					X	
SPDC Portafolio						No exporta.
MyStuff		X				Puede exportar las compilaciones en formato RTF[21].
Portafolio					X	

Después de realizado el análisis, se reafirma el estándar Leap2A como el más utilizado entre Plataformas Educativas y herramientas para la creación de e – Portafolio. Luego de seleccionado el estándar de exportación se especifican las características del mismo.

1.4.1. Especificación del estándar seleccionado

La especificación Leap2A ha sido desarrollada para dar soporte a la interoperabilidad entre las herramientas e - Portafolio y sistemas similares. Para desarrollar esta especificación se ha usado el formato Atom[22]. Los elementos principales especificados con Leap2A son: habilidad, logro,

[21] En sus siglas en inglés *Rich Text Format* (Formato de texto enriquecido).

[22] Atom es un formato de documento basado en XML que describe la lista relacionada a la información conocida como "feeds". Los *feeds* se compone de una serie de elementos, conocidos como "entradas", cada uno con un conjunto ampliable de adjuntos metadatos. El caso de uso principal que se ocupa de Atom es la sindicación de Web contenidos como *weblogs* y titulares de noticias a los sitios Web, así como directamente a los agentes de usuario.

actividad, reunión, organización, persona, plan, recurso y selección (Fabregat, 2010).

Esta familia de especificaciones se destina a cubrir la representación de varios tipos de información, centrada en los individuos, que recogen, crean, reflexionan y utilizan su propia información para el aprendizaje, el desarrollo, la auto – presentación, o con fines relacionados.

Leap2A ofrece un modelo para representar la información y los recursos creados, controlados o recogidos por las personas, descripciones de lo que han logrado, creado, hecho o experimentado; información sobre sí mismos, sus habilidades y cualidades; cualquier tipo de información de apoyo o pruebas de cualquier fuente, sus pensamientos o reflexiones sobre lo pasado, presente o futuro, sus planes y la presentación de las selecciones de esta información a otras personas (Fabregat, 2010).

Modelo de datos del Leap2A

Este estándar posee un modelo de datos en el cual está presente la flexibilidad de sus elementos, pues un atributo puede relacionarse con los restantes. Para profundizar en los datos que maneja el mencionado estándar consultar **Anexo 1: Modelo de datos del Leap2A.**

Licencia: **Anexo 2: Dictamen emitido por la Universidad de las Ciencias Informáticas sobre la licencia del estándar IMS Leap2A.**

1.5. Ambiente de desarrollo

Obtener un software con la calidad requerida y que cumpla con las expectativas del cliente final, implica la utilización de metodologías, tecnologías y herramientas que contribuyan a conformar el área de desarrollo, mediante la cual se elabora el producto final. Como parte de la definición de la arquitectura de referencia para el desarrollo de aplicaciones en el Centro FORTES, se ha iniciado la formalización de marcos de trabajo en cada Línea de Productos de Software (LPS), con el propósito de disminuir la diversidad tecnológica de las soluciones. Es por ello que la selección de las tecnologías, metodologías y herramientas para el desarrollo de la solución final, debe estar enmarcado dentro de las especificaciones que plantea el marco de trabajo del Ambiente Integrado de Aprendizaje (AIA), aunque lo planteado no significa que no se puedan utilizar otras tecnologías que no se encuentren definidas dentro del mencionado marco de referencia siendo fundamentada su selección, en cualquier caso, por un estudio previo; por lo que ello constituiría el ambiente de desarrollo para el presente trabajo. A continuación se detallan las características del mismo.

1.5.1. Metodología de desarrollo de software

Todo producto de software ha sido diseñado e implementado siguiendo una metodología de desarrollo de software. Se han definido diversas metodologías, pero las tradicionales y ágiles fungen como las que más se utilizan para desarrollar sistemas informáticos.

Para seleccionar qué metodología es la más adecuada en el desarrollo del proceso de construcción de software en la presente investigación, se realizó un estudio entre las metodologías ágiles y tradicionales (Joskowics, 2008) y (Letelier y Penadés, 2006).

De forma general las metodologías tradicionales, aunque también denominadas pesadas, clásicas o robustas, no resultan ser el enfoque más adecuado para muchos proyectos de software actuales, donde el entorno del sistema es muy cambiante, y en donde se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad. Sin embargo las metodologías ágiles cumplen muy bien el rol de ser muy adaptables a los cambios que puedan sucederse a lo largo del proceso de producción de software, en lugar de ser predictivos, ya que se encuentran especialmente orientadas a proyectos pequeños y constituyen una solución a 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. Las robustas se caracterizan por ser muy rígidas, algunos autores las catalogan como "camisas de fuerza", y dirigidas por la documentación que se genera en cada una de las actividades desarrolladas; mientras que las ágiles intentan ser procesos que se adaptan y crecen con el cambio. Estas últimas se encuentran, en gran medida, orientadas a las personas y no tanto al proceso; y por su parte, las tradicionales, tienden a intentar planear una gran parte del proceso del software en gran detalle para un plazo largo de tiempo, con el objetivo de definir un proceso que funcionará bien independientemente de quien lo utilice.

Luego de realizado el anterior estudio, los autores del presente trabajo seleccionaron las metodologías ágiles como las que más se adaptan para el desarrollo de la solución de la investigación, puesto que el ambiente bajo el cual se trabajará se adecua a las características de las mencionadas metodologías. A continuación se selecciona, dentro de las metodologías ágiles, cual fluctúa como la que mejor se adapta para su utilización en el presente trabajo. Para ello se muestra un estudio comparativo entre las principales metodologías de este tipo (Letelier y Penadés, 2006) y (Khramtchenko, 2004):

SCRUM: indicada para proyectos con un rápido cambio de requisitos. El desarrollo de software se realiza mediante iteraciones, con una duración de 30 días. El resultado de cada iteración es un incremento ejecutable que se muestra al cliente. Consta de reuniones a lo largo del proyecto, donde el equipo de desarrollo coordina e integra todo el proceso productivo.

***Crystal Methodologies*:** conjunto de metodologías para el desarrollo del software caracterizadas por estar centradas en las personas que componen el equipo y la reducción al máximo del número de artefactos producidos. El equipo de desarrollo es un factor clave, por lo que se deben invertir esfuerzos en mejorar sus habilidades y destrezas, así como tener políticas de trabajo en equipo bien definidas.

Adaptative Software Development (ASD): consta de un proceso iterativo, orientado a los componentes del software más que a las tareas, y se presenta muy tolerante a los cambios. El ciclo de vida que propone presenta tres fases esenciales: especulación, colaboración y aprendizaje. La revisión de los componentes sirve para aprender de los errores y volver a iniciar el ciclo de desarrollo.

Feature – Driven Development (FDD): define un proceso iterativo que consta de 5 pasos. Las iteraciones son cortas, las cuales pueden ser de hasta 2 semanas. Se centra en las fases de diseño e implementación del sistema partiendo de una lista de características que debe incluir el software.

Extreme Programming (XP): centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo y preocupándose por el aprendizaje de los desarrolladores. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo. Presenta características como la comunicación fluida entre todos los participantes y simplicidad en las soluciones implementadas. Se define adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

Además de enunciar las principales características de las metodologías ágiles antes mencionadas, se tomó como basamento para su selección la tabla disponible en el **Anexo #3: Agilidad de cada metodología,** tomado de (Highsmith, 2002), donde se muestran indicadores en correspondencia con el nivel de agilidad que presentan cada una de las metodologías estudiadas. A cada metodología se le asigna un valor entre 1 y 5 de acuerdo al nivel de satisfacción para cada indicador, donde el valor 5 funge como la máxima calificación.

Del anterior estudio se evidencia que existen dos metodologías, ASD y XP, que presentan similares niveles de agilidad para sus indicadores por lo que se decide seleccionar de entre ambas a XP para desarrollar la solución al presente problema de investigación, debido a que las características que define se pueden aplicar en concordancia al ambiente en que se implementará el producto final.

XP cuenta con un ciclo de vida muy bien estructurado, resumiéndose en cuatro fases fundamentales según (Joskowics, 2008). La primera es la **fase de exploración**, en la que el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Además se elaboran las "historias de usuario" que no es más que las especificaciones del cliente de lo que el sistema debe realizar. La segunda es la **fase de planificación**, donde se aprueban las historias de usuario y se acuerda el alcance del proyecto. Los desarrolladores estiman el esfuerzo a dedicar a cada historia y se define el cronograma de ejecución. Se precisan las iteraciones de desarrollo y las pruebas para cada iteración, por parte

del cliente. La tercera es la **fase de producción**, mediante la cual se ejecutan las iteraciones elaboradas en la fase anterior, presentando al cliente avances del desarrollo de forma constante, logrando la retroalimentación entre cliente y desarrolladores. Una de las condiciones para ejecutar esta fase es la disponibilidad total del cliente. La cuarta es la **fase de mantenimiento**, la cual requiere un mayor esfuerzo para satisfacer las tareas del cliente por parte de los desarrolladores. Esta fase puede requerir cambios en la arquitectura del sistema.

1.5.2. Lenguaje de modelado

La metodología seleccionada, XP, no plantea una gran cantidad de artefactos referentes a modelos y diseños para representar los sistemas, simplemente se deben generar los necesarios para que se entienda el completo funcionamiento del mismo. El lenguaje de modelado que se utilizará para representar los procesos a través de modelos y diagramas en la presente investigación será el Lenguaje Unificado de Modelado (UML, de sus siglas en inglés *Unified Modeling Language*) en su versión V2.4.1.

Lenguaje de Modelado Unificado

UML, de sus siglas en inglés *Unified Modeling Language,* es un lenguaje de modelado de sistemas de software y está respaldado por el Grupo de Gerencia de Objetos (OMG, de sus siglas en inglés *Object Management Group*). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Ofrece un estándar para describir una visión del sistema, incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables (Magaña Orúe, 2009).

El objetivo de UML es proporcionar una arquitectura de sistema, ingeniería del software, y desarrollo de software con herramientas de análisis, diseño e implementación basada en los sistemas de software tanto para modelado de procesos de negocio, como para procesos similares (Magaña Orúe, 2009).

1.5.3. Herramientas para el modelado

Además de la aproximación metodológica y el lenguaje, uno de los factores determinantes en el éxito de un proyecto de software, lo constituye la acertada selección de las herramientas de trabajo que se utilizarán durante su desarrollo, es por ello que en la actualidad la administración de proyectos atiende cuidadosamente la designación de herramientas como una tarea de vital importancia (Bernardo, Anaya de Páez, Marín y Bilbao López, 2005).

Las herramientas de Ingeniería de Software Asistida por Computador (CASE, de sus siglas en inglés *Computer Aided Software Engineering*), están tomando cada vez más relevancia en la

planeación y ejecución de proyectos que involucren sistemas de información, pues suelen inducir a sus usuarios a la correcta utilización de metodologías que le ayudan a llegar con facilidad a los productos de software construidos (Bernardo, Anaya de Páez, Marín, Bilbao López, 2005).

Visual Paradigm for UML

Visual Paradigm for UML es una herramienta CASE que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, implementación y pruebas. Ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite construir diagramas de diversos tipos, código inverso, generar código desde diagramas y generar documentación (Cabrera González, González y Torres, 2012).

Teniendo en cuenta las características y los beneficios que brinda para la construcción del software, especialmente referente al modelado, se decidió utilizar *Visual Paradigm for UML* como herramienta de modelado para el desarrollo del portafolio de desarrollo personal en su versión v8.0. Además de lo planteado, se tuvo en cuenta que la mencionada herramienta es la que se utiliza en la UCI para el modelado de procesos de desarrollo de software. Presenta una gama de funcionalidades que lo hacen adaptable y usable para cualquier metodología de desarrollo de software que se utilice. En este caso la metodología seleccionada XP, establece una serie de artefactos, los cuales pueden ser modelados y representados a través de la mencionada herramienta.

1.5.4. Lenguajes de desarrollo de software

En este apartado se analizan las tecnologías y definiciones arquitectónicas que plantea el marco de referencia definido por el Centro FORTES (Salvador Broche, 2013), debido a que la investigación y producción de la solución informática final se encuentran encaminadas sobre la base de la utilización de los mencionados recursos.

Lenguajes del lado del cliente

Lenguaje de Marcado de Hipertexto

Lenguaje de Marcado de Hipertexto (HTML, de sus siglas en inglés *HyperText Markup Language*), es el lenguaje básico para la elaboración de páginas web. Define comandos, marcas o etiquetas que permiten delimitar la estructura lógica de un documento web (Mir, 2012). Para su selección se basó en características como un lenguaje sencillo que no necesita de grandes conocimientos cuando se cuenta con un editor web y permite describir hipertexto. El texto es estructurado de forma lógica, debido al glosario de etiquetas que provee para la presentación del contenido.

Se utilizará el lenguaje HTML en su versión v5.0.

Hojas de Estilos en Cascada

Hojas de Estilos en Cascada (CSS, de sus siglas en inglés *Cascading Style Sheets*), es un lenguaje que se utiliza para definir el estilo de presentación que tendrá un documento HTML o XML y por extensión XHTML. Estas hojas de estilo se utilizan para separar la estructura del documento de la presentación del mismo (Mir, 2012).

Según la W3C[23] , "(…) es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos (…)".

Se utilizará CSS en su versión v3.0.

JavaScript

Lenguaje de programación que se utiliza, principalmente, para crear páginas web dinámicas. Permite a los desarrolladores la programación de una interfaz de usuario mejorada, aumentando entre otras cosas la experiencia de los usuarios en la web. Es un lenguaje interpretado por lo que no es necesario compilar los programas para poder ejecutarlos, es decir, los programas escritos con *JavaScript* se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios (Benigni, Antonelli y Vázquez, 2009).

Lenguaje de Marcado Extensible

Lenguaje de Etiquetado Extensible (XML, de sus siglas en inglés *Extensible Marquee Language*) no es más que un conjunto de reglas para definir etiquetas semánticas que nos organizan un documento en diferentes partes. XML es un metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados. Este lenguaje es abierto, optimizado para su uso en la Web, y que va a permitirnos describir el sentido o la semántica de los datos.

Este lenguaje será utilizado en su versión v1.1, para almacenar la información generada por el estándar a implementar.

Tecnologías del lado del cliente

Ajax

JavaScript Asíncrono y XML (Ajax, de sus siglas en inglés *Asynchronous JavaScript and XML*), no es una tecnología en sí mismo. En realidad se trata de varias tecnologías independientes que se unen de forma nueva y sorprendente (Eguiluz Pérez, 2008). Las tecnologías que forman Ajax son:

- XHTML y CSS, para crear una presentación basada en estándares.

[23] *World Wide Web Consortium*.

- DOM[24], para la interacción y manipulación dinámica de la presentación.
- XML, XSLT[25] y JSON[26], para el intercambio y la manipulación de la información.
- *XMLHttpRequest*, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.

La importancia de esta tecnología está en que al utilizarla, la misma se ejecuta en el lado del cliente, es decir el navegador web ofrece mantener una comunicación asíncrona con el servidor (Sánchez Menéndez, Menéndez y Morales, 2012).

Con la propiedad mencionada anteriormente, cuando se desee hacer algún cambio en el desarrollo web, no será necesario que la página se recargue nuevamente; lo que se traduce en aplicaciones mucho más interactivas, rápidas y eficientes.

Con Ajax la interacción que tiene el usuario con alguna aplicación se mejora porque no tendrá que saltar de página en página para hacer alguna tarea específica, así también se mejora que el usuario no se detenga cuando cierta aplicación necesite algo del servidor (Danny, 2008).

En el presente trabajo se utilizará Ajax para la comunicación asincrónica de los datos[27].

Lenguajes del lado del servidor

Procesador de Hipertexto

Procesador de Hipertexto (PHP, de sus siglas en inglés *Hypertext Preprocessor*), es un lenguaje de programación que se caracteriza por la fluidez y rapidez de sus *scripts*. Diseñado originalmente para el desarrollo web de contenido dinámico, es gratuito, multiplataforma, cuenta con una gran biblioteca de funciones y detallada documentación. Ofrece integración con numerosas bibliotecas externas y es compatible con varios Gestores de Bases de Datos (SGBD) como: MySQL y PostgreSQL (Baukes, 2013).

Se utilizará PHP en su versión v5.4.16.

1.5.5. Framework

El término *framework* (marco de trabajo) se refiere a una estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras un framework se puede considerar como una aplicación genérica incompleta y configurable a la que se le puede añadir las últimas piezas para construir una aplicación concreta.

[24] *Document Object Model* (Modelo de Objetos del Documento o Modelo en Objetos para la Representación de Documentos).
[25] Siglas de *Extensible Stylesheet Language Transformations*, lenguaje de hojas extensibles de transformación.
[26] Acrónimo de *JavaScript Object Notation*, formato ligero para el intercambio de datos.
[27] Asincrónica: los datos adicionales se solicitan al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página.

Los objetivos principales que persigue este tipo de tecnología son: acelerar el proceso de desarrollo, reutilizar el código ya existente y promover buenas prácticas de desarrollo con el uso de patrones (Calderín y Pérez, 2012).

Capa de Presentación

jQuery

jQuery es una biblioteca de *javascript* rápida y concisa que simplifica el recorrido de un documento HTML, de manejo de eventos, animaciones e interacciones Ajax para el desarrollo web rápido (jQuery Community, 2013). Estas características posibilitan que los contenidos se puedan mostrar y se interactúe con los mismos de forma dinámica, brindando la posibilidad de usar este *framework* para apoyar el cumplimiento de las funcionalidades necesarias.

Se utilizará el framework jQuery en su versión v1.10.2.

Bootstrap

El *framework* Bootstrap surgió de la idea de hacer interfaces de aplicaciones web más amigables con el usuario. Logra definir una serie de etiquetas que posibilitan un maquetado de las interfaces de usuario a una mayor precisión. Entre las características que apoyaron su selección se encuentran (Vermilion, 2013) que se presenta extensible para disímiles navegadores web, todos los estilos se encuentran en un único archivo y ha sido optimizado para lograr un buen rendimiento de las aplicaciones web, las interfaces de usuario incluyen listado de grupos y paneles, posee un grupo de componentes que lo hacen personalizable a cualquier ambiente de desarrollo de software y presenta una facilidad de uso por parte de los diseñadores de interfaces de usuario, permitiendo un perfecto maquetado del contenido.

Se propone utilizar el framework Bootstrap en su versión v3.1.1, debido a que los cambios realizados de una versión a otra no son relevantes.

Capa de Lógica de Negocio

Symfony

Symfony2 es un *framework* de desarrollo web PHP, pero también es una filosofía de trabajo y una comunidad próspera. Symfony2 facilita a los desarrolladores una forma de trabajo estructurada que añade valor a su trabajo, dando la posibilidad de definirse en un perfil de movilidad más amplia (Martínez, 2013).

El *framework* Symfony2 presenta las siguientes características (Potencier, 2011), las que nos conllevaron a utilizarlo como marco de trabajo de nuestra solución: implementa el patrón Modelo Vista Controlador (MVC, de sus siglas en inglés *Model View Controller*), proporciona una estructura al código fuente, forzando al desarrollador a crear código más legible para un futuro

mantenimiento, encapsula operaciones complejas en instrucciones sencillas, es un *framework* multiplataforma, usa Programación Orientada a Objetos (POO), posibilita soportar varios gestores de base de datos como MySQL, Oracle y PostgreSQL, siendo importante por si en el futuro se decide cambiar el gestor y soporta integración con el ORM Doctrine y con los *framework* de la capa de presentación jQuery y Bootstrap.

Por todas las características antes mencionadas se selecciona Symfony en su versión v2.3.7 para la implementación del portafolio de desarrollo personal.

Capa de Acceso a Datos

Mapeador de Objetos Relacional

Un Mapeador de Objetos Relacionales (ORM, de sus siglas en inglés *Object Relational Mapper*), es una técnica de programación que permite convertir datos entre el sistema de datos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional, es decir, las tablas de la base de datos pasan a ser clases y los registros se pueden manejar con facilidad (Romero y Torres, 2013).

Dentro de las ventajas que nos condujo a seleccionarlo para desarrollar nuestra solución se encuentran (Rosales Morales, Morales, Clark y Oliva, 2013): rapidez en el desarrollo de las aplicaciones, creación automática de un modelo adecuado a partir del esquema de tablas y relaciones, evitando que se cometan errores, los códigos generados por herramientas ORM tienen un excelente diseño, ya que se basan en el uso de patrones y buenas prácticas y se caracterizan por abstraer el motor de base de datos, lo que significa que se pueda cambiar fácilmente de un sistema gestor a otro sin afectar al resto de la aplicación.

Doctrine

Doctrine es un ORM para PHP que proporciona persistencia transparente de los objetos desde el lenguaje PHP. Utiliza el patrón Data Mapper[28] en el núcleo del proyecto, logrando separar la lógica de dominio / negocio de la persistencia en un sistema de bases de datos relacional (Rosales Morales, Morales, Clark y Oliva, 2013).

Presenta una serie de características que lo hace ineludiblemente muy utilizable en proyectos de desarrollo de software y fomenta su elección para el presente trabajo (Rosales Morales, Morales, Clark y Oliva, 2013): soporte para las operaciones de Crear, Obtener, Actualizar y Borrar (CRUD, de sus siglas en inglés *Create, Retrieve, Update and Delete*) habituales, desde la creación de nuevos registros a la actualización de los antiguos, permite crear manualmente y automáticamente

[28] Data Mapper: Es una capa del software que separa los objetos en memoria de la base de datos. Su función es transferir los datos entre estos dos y también para aislarlos unos de otros.

el modelo de base de datos a implementar y presenta soporte para varios motores de bases de datos (como MySQL, PostgreSQL y Microsoft SQL).

El empleo de Doctrine posibilitará un trabajo más rápido y eficaz. Posee una amplia documentación lo que facilita la retroalimentación continua y presenta soporte para migraciones. Se decide utilizar la versión v2.3 de dicho ORM.

1.5.6. Sistema Gestor de Base de Datos

PostgreSQL

Es un SGBD Relacional de código abierto. Presenta un conjunto de características y funcionalidades que lo hacen más profundo en comparación con su archirrival MySQL. (*PostgreSQL: Dossier de Prensa de PostgreSQL 9.3,* 2014)

Algunas de las características que avalan su selección como SGBD son: presencia de funciones de indexación, incluyendo exploraciones funcionales en índices parcial, funcional y de múltiple indexado, establece confirmaciones asincrónicas para todo el sistema, o por usuario o transacción, los conectores de datos foráneos están habilitados para escritura, lo que permite el intercambio bidireccional de datos entre sistemas, presenta métodos constructores y extractores para JSON y contiene vistas actualizables automáticas.

Para el desarrollo de la solución propuesta, se utilizará PostgreSQL en su versión v9.3.4.

1.5.7. Servidor web

Apache

En el ambiente de desarrollo que se especifica, será utilizado como servidor web Apache en su versión v2.4.6. El mencionado servidor web se caracteriza por ser de código abierto, extensible para plataformas Unix, Windows y Macintosh. Presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido.

1.5.8. Entorno de desarrollo

NetBeans

Este Entorno de Desarrollo Integrado (IDE, de sus siglas en inglés *Integrated Development Environment*), permite a los desarrolladores escribir, compilar, depurar y ejecutar programas informáticos. Es un IDE de código abierto escrito completamente en Java que permite crear aplicaciones de escritorio, web y aplicaciones para móviles utilizando los lenguajes Java, JavaFX, PHP, Java Script, Ruby y Ruby onRails, Groovy y Grails, y C/C++, además de presentar soporte para la tecnología Ajax. Está disponible para múltiples plataformas como son Windows, Mac, Linux y Solaris (Cabrera González, González y Torres, 2012).

Será utilizado el IDE NetBeans en su versión v7.4, debido a que, además de las prestaciones antes mencionadas, presenta un entorno de trabajo muy eficiente y que hacen posible trabajar con mayor fluidez y facilidad a la hora de implementar productos informáticos. Además permite la integración con el framework Symfony seleccionado para este trabajo, haciendo posible que algunas funcionalidades gestionadas a través de la consola de este marco de trabajo, se puedan ejecutar en el propio IDE.

1.5.9. Otras tecnologías

Para cumplir con una de las principales características que debe poseer la aplicación final, poder integrarse a otros sistemas informáticos, se hace necesario la utilización de tecnología relacionada con servicios web, aprovechando las ventajas que estos presentan para su utilización en aplicaciones informáticas.

Servicios web

El concepto de servicio web (SW) puede resultar confuso, ya que no existe una definición única aceptada sobre lo que son y lo que el concepto engloba. Los servicios web surgieron como un conjunto de protocolos y estándares, definidos por la W3C, para lograr la interoperabilidad en la interacción entre máquinas, sistemas software y aplicaciones a través de la red (Los Santos, 2009).

Para la definición e implementación de los SW que brindará la solución final, se hace necesario seleccionar la tecnología apropiada para desarrollar este proceso. Para ello se utilizará el estilo de arquitectura de *software* Transferencia de Estado Representacional (REST, de sus siglas en inglés *Representational State Transfer*).

En realidad, REST se refiere estrictamente a una colección de principios para el diseño de arquitecturas en red. Estos principios resumen como los recursos son definidos y diseccionados. El término frecuentemente es utilizado en el sentido de describir a cualquier interfaz que transmite datos específicos de un domino sobre HTTP sin una capa adicional (Navarro Marset, 2007).

Para identificar cuáles son las principales características que presenta la mencionada tecnología y que pueden ser adoptadas por el equipo de desarrollo para implementar los SW, se realizó un análisis de la información contenida en (Novoa y Álvarez, 2012), (Solano, 2012), (Marqués, 2012) y (Navarro Marset, 2007):

- Presenta un alto grado de escalabilidad en la interacción con los componentes, debido a la gran variedad de clientes que pueden acceder a través de la web: estaciones de trabajo, sistemas industriales, telefonía móvil.

- Se centra en el rendimiento a gran nivel para sistemas distribuidos hipermedia, por lo que el estado de una aplicación web debe ser capturada en varios documentos de hipertexto, residiendo tanto en el cliente como en el servidor.
- Incluye la generalidad de interfaces, por lo que el cliente puede interactuar con cualquier servidor HTTP sin ninguna configuración especial.
- La puesta en funcionamiento independiente resulta una realidad, ya que HTTP permite la extensibilidad mediante el uso de cabeceras, a través de las URIs[29], y mediante la habilidad para crear nuevos métodos y tipos de contenidos.
- Presenta alta compatibilidad con componentes intermedios, permitiendo reducir la latencia de interacción, reforzar la seguridad y encapsular otros sistemas.
- Permite mostrar cantidades de datos masivos, mediante el acceso con escaso consumo de recursos a sus operaciones debido al limitado número de operaciones que presenta.
- Mantiene una interacción dirigida por el usuario por medio de formularios, brindando la posibilidad de que cada objeto soporte las operaciones estándares (CRUD: *Create, Retrieve, Update, Delete*).
- Permite describir cómo nuestro servicio ha de ser invocado, mediante un documento WSDL[30] / WADL[31] o simplemente HTML.

1.5.10. Sistema de Control de Versiones

Una de las herramientas que se utiliza para el desarrollo del presente trabajo es el Sistema de Control de Versiones (CSV, de sus siglas en inglés *Control System Versions*) **Git,** puesta a disposición a la comunidad universitaria para su uso bajo el nombre de *GitLab***.** Esta es una herramienta que ayuda mucho a los programadores a tener control sobre su código, además de ser un sistema distribuido. Permite tener repositorios locales y remotos, brindando la posibilidad de ser accedidos por varios usuarios. Se presenta como aplicación *opensource*[32], y la administración de los repositorios se realiza mediante una interfaz web. Una de las características más notables de esta herramienta es que permite ver el código y editarlo en la propia interfaz web que provee. Además brinda la opción de poder descargar el código del proyecto, ya sea a través de descarga directa o a través de *commits*[33] mediante la consola (kotov, 2013).

Para el desarrollo de este trabajo se utiliza *GitLab* en su versión v4.1.0.

[29] *Uniform Resource Identifier* (Identificador de Recurso Uniforme).
[30] *Web Services Description Language* (Lenguaje de Descripción de Servicios Web).
[31] *Web Application Description Language* (Lenguaje de Descripción de Aplicación Web).
[32] Se refiere a herramientas de código abierto o licencia libre.
[33] Envío de datos que se ejecuta para actualizar información en un determinado contexto.

1.6. Conclusiones parciales del capítulo

Luego de la investigación realizada sobre los distintos criterios encontrados al analizar la utilización de los e – Portafolio que se encuentran a disposición de los usuarios y de las posibilidades que brindan este tipo de herramienta, se concluye que:

1. Atendiendo al contexto donde intervienen las herramientas e – Portafolio, se identificaron tres enfoques principales, mediante los cuales se puede estructurar este tipo de herramienta: Personal, Profesional y Académico.
2. Teniendo en cuenta las características de las herramientas e – Portafolio y las tecnologías asociadas a ellos, se seleccionó el estándar Leap2A para lograr la interoperabilidad de los datos almacenados.

Capítulo 2: Presentación de la propuesta de solución

Una vez concluido el estudio del estado del arte, donde se expresaron los fundamentos de la necesidad de realizar la presente investigación y se definió el ambiente de desarrollo mediante el cual se dará respuesta a la solución del problema; es necesario definir los principales procesos del negocio que estarán presentes en la aplicación final, así como definir las características del sistema a implementar. Se especificarán los requerimientos funcionales y no funcionales[34] que debe cumplir el sistema para satisfacer las necesidades del cliente. Además se desarrollarán las fases de Exploración y Planificación, las dos primeras de la metodología que se definió.

2.1. Modelo conceptual

En la presente investigación se decide realizar el modelo conceptual como una forma de representar los conceptos que intervienen en el flujo del sistema. A través de un estudio previo es que se definieron las instancias que intervienen en el flujo de negocio y las asociaciones presentes entre ellas.

En sentido general el modelo conceptual brinda la posibilidad a un analista de tener un medio mediante el cual se puede apoyar para comprender los conceptos que se manejan en un sistema y su relación con los requisitos que se desean implementar.

2.1.1. Conceptos identificados

Portafolio: es donde se guardan todas las evidencias generadas por los usuarios del sistema, así como sus datos personales, valoraciones de las evidencias o cualquier información relevante que propicie el desarrollo personal de los usuarios.

Sección: cada usuario puede crear sus propias secciones. En estas se guardan evidencias de acuerdo a la categoría que el usuario establezca, así como toda la información generada por cada evidencia guardada en la sección.

Usuario: es la persona que interactúa en el sistema y puede crear su portafolio personal. Es quien genera las evidencias y puede compartirlas a otros usuarios.

Perfil: espacio donde se muestran los datos personales de los usuarios.

Evidencia: es aquel recurso que genera el usuario del sistema, mediante el cual este puede realizar su autorreflexión y medir el nivel de preparación alcanzado.

Evidencia compartida: es aquella evidencia que ha sido compartida por un usuario, para mostrar su desarrollo personal. Esta puede ser comentada y valorada por otros usuarios.

[34] Algunos autores mencionan que en XP, a las capacidades y cualidades que debe cumplir el software se les denomina requerimientos funcionales y no funcionales, mientras otros expresan que se les denomina lista de reservas. En esta investigación se utilizarán el término lista de reservas para referirnos a los requerimientos funcionales del sistema.

Valoración: cada evidencia compartida puede ser valorada en un rango definido en el sistema.

Notificaciones: mensajes o alertas que emite el sistema al realizarse alguna acción.

Comentario: nota que puede emitir un usuario a una evidencia compartida.

Archivo: son los adjuntos que el usuario guarda en la aplicación para luego asignarlo a una evidencia. Estos pueden ser de cualquier formato, dígase archivos de audio, video, imágenes y documentos.

Etiqueta: representa los nombres con que se identifican las secciones que el usuario crea, esto facilita el sistema de búsqueda dentro de la aplicación.

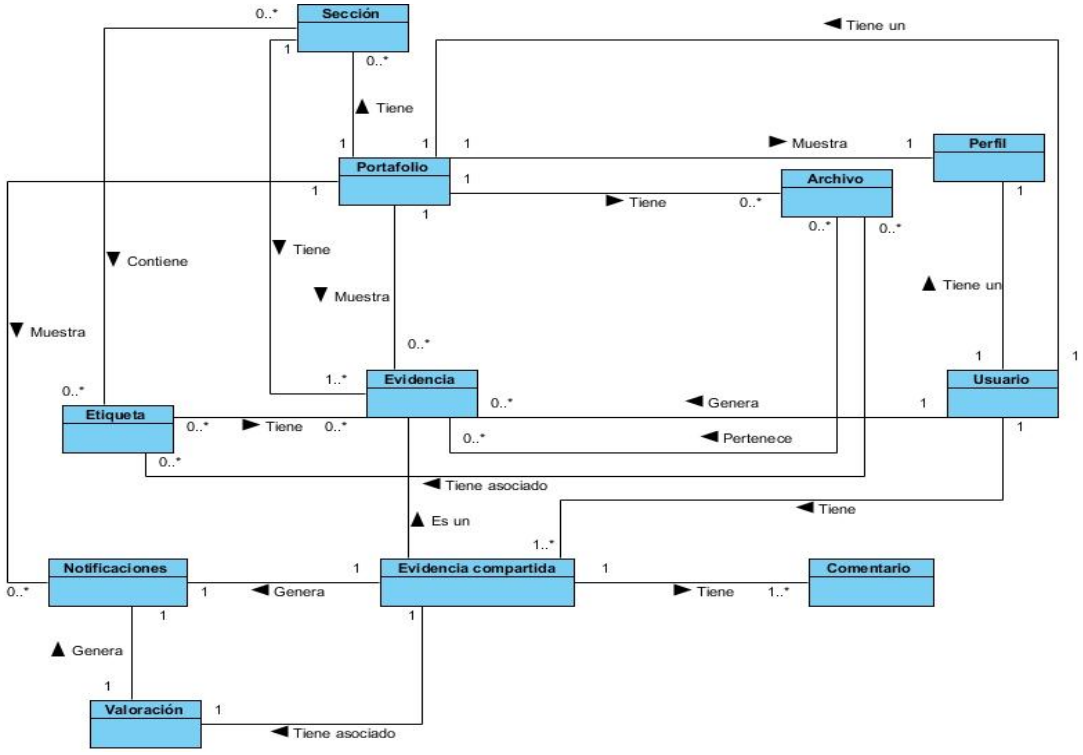


Ilustración 1. Representación del Modelo conceptual.

2.2. Características del sistema

El sistema propuesto hará función de componente, por lo que podrá ser utilizado dentro de cualquier sistema como módulo integrado, interactuando bajo el flujo de negocio del mismo; o simplemente como sistema independiente, brindando las mismas funcionalidades en ambos casos. Cuando esté funcionando como sistema independiente, estará disponible bajo un servidor web. En este ambiente el usuario interacciona directamente con el sistema y será quien realice todas las acciones que brinde la aplicación, a través del flujo: petición – respuesta.

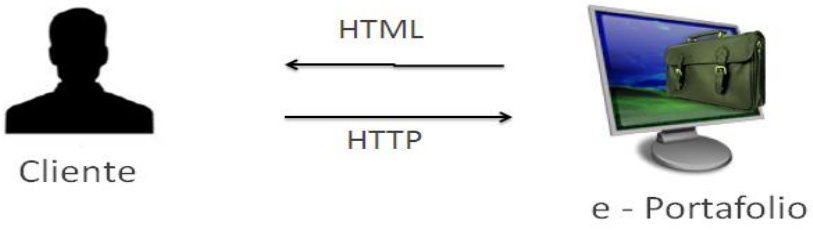


Ilustración 2. Representación del Funcionamiento del Portafolio como sistema independiente.

Cuando se encuentre integrado a otro sistema como aplicación independiente, esta conexión se realizará a través de la utilización de servicios web. La aplicación debe ser capaz de brindar información a los sistemas con los que se encuentre integrado y por consiguiente, debe nutrirse de los recursos que estos sean capaces de ofrecerle, es decir, mantener un flujo de información bidireccional en todo momento con los mencionados sistemas. Para ello la aplicación contará con un sistema de servicios web implementados, que darán soporte a las prestaciones antes mencionadas.

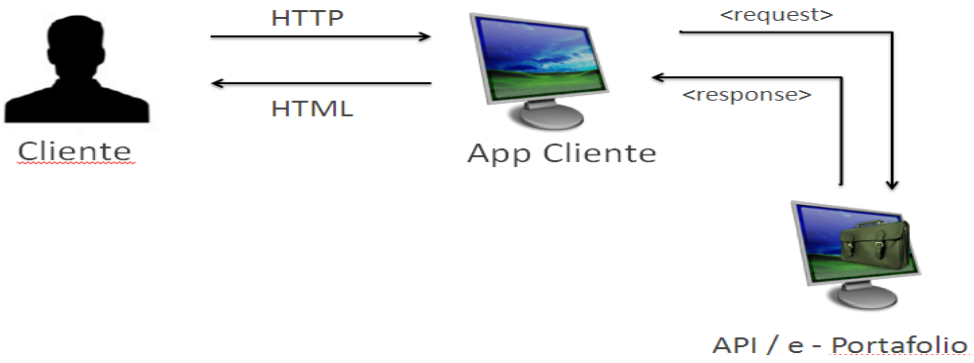


Ilustración 3. Representación del Funcionamiento del Portafolio como sistema integrado.

El funcionamiento de la solución propuesta se basa en el almacenamiento de evidencias de aprendizaje y la autorreflexión que son capaces de realizar los usuarios a través de estas. Estos tendrán la posibilidad de compartir sus evidencias personales con otros usuarios registrados en el sistema, permitiendo mostrar su desarrollo personal. El sistema deberá tener implementado un sistema de notificaciones que brinde al usuario una sección, donde pueda revisar los mensajes enviados, de acuerdo a las acciones que se realizan, ya sean sobre su portafolio personal o acciones que se lleven a cabo automáticamente por la aplicación.

2.2.1. Para el portafolio personal

En esta sección se describen los distintos espacios de trabajo por los que estará conformada la propuesta de solución, como una forma de conocer cuáles son las acciones que puede realizar un usuario dentro del sistema final.

Pantalla principal: el sistema presentará en este espacio de trabajo, los formularios de

autenticación y de registro, en caso de que el usuario no se encuentre autenticado en el sistema. Cuando el usuario se encuentre autenticado en el sistema, se muestra un nuevo espacio de trabajo personal, donde tiene la posibilidad de visualizar sus evidencias creadas, organizadas por las más recientes, además de consultar las notificaciones personales y las etiquetas que ha creado relacionadas con diferentes espacios del portafolio.

Registro de usuario: el usuario que desea hacer uso de la herramienta portafolio, debe registrarse en la misma. Una vez que se registra, automáticamente se crea el portafolio personal para el usuario registrado. El usuario debe ingresar sus datos personales tales como: nombre, apellidos, correo electrónico, seleccionar un idioma, seleccionar si desea recibir notificaciones por correo y una contraseña mediante la cual tendrá acceso a su espacio de trabajo.

Mi perfil: en esta sección el usuario tiene la posibilidad de visualizar los datos de su perfil personal y de editar aquellos parámetros que estime conveniente. Todo este proceso estará respaldado por una validación de los datos. Además en este espacio el usuario tiene la posibilidad de incluir una foto personal como identificación.

Secciones: el usuario puede organizar su portafolio personal en secciones. Es por ello que en este espacio de trabajo tiene la posibilidad de crear, editar, eliminar y visualizar las secciones, dentro de las cuales estarán almacenadas las evidencias personales. Por cada sección se puede crear una nueva evidencia, además de visualizar la cantidad de evidencias que contiene cada sección.

Notificaciones: en esta sección el usuario puede visualizar las notificaciones que les han sido generadas, ya sea por acciones realizadas de otros usuarios, tales como: valorar una evidencia compartida, compartir una evidencia o comentar una evidencia compartida. Tiene la posibilidad de eliminar aquellas notificaciones que estime conveniente y ver los detalles de cada una.

Archivos: en esta sección el usuario puede crear un espacio de trabajo donde almacene todos los archivos generados de su actividad personal. Estos pueden ser documentos, archivos multimedia o notas personales. Los recursos que se almacenen en esta sección pueden ser incluidos en las evidencias personales. Si el usuario desea puede eliminar cualquier archivo que haya almacenado con anterioridad, además de poder editar los datos de los mismos.

Mis evidencias: el usuario puede visualizar un listado de sus evidencias personales. Puede editar los parámetros de cada una de ellas incluyendo, si lo desea, otro archivo que se encuentre almacenado en el sistema. Además tiene la posibilidad de ver los detalles de cada evidencia y compartirlas con otros usuarios si estima conveniente. Puede eliminar las evidencias según desee.

Compartidas por mí: en esta sección el usuario tiene la posibilidad de visualizar aquellas

evidencias personales que ha compartido con otros usuarios registrados en el sistema. Puede editar los parámetros de la evidencia compartida, así como ver los detalles de la misma. Si lo estima conveniente puede eliminar aquellas que desee o puede compartirlas con otros usuarios del sistema.

Compartidas a mí: en esta sección el usuario puede visualizar las evidencias que le han sido compartidas por otros registrados en el sistema. Tiene la posibilidad de ver los detalles de cada una, valorar la evidencia y emitir un comentario acerca de esta.

Opciones del Portafolio: en esta sección el usuario tiene la opción de Exportar los datos de su portafolio, operación que se realiza bajo las especificaciones del estándar definido para ello: Leap2A.

FAQ: en este espacio el usuario tiene la posibilidad de conocer algunas cuestiones relacionadas con el uso de la herramienta e – Portafolio, y de suplir alguna duda que tenga con respecto a esta.

Contáctenos: en este espacio el usuario tiene la posibilidad de enviar un mensaje a los editores de la herramienta, con respecto a cualquier deficiencia detectada o inconformidad con la herramienta.

Acerca de: el usuario puede ver la información personal de los administradores de la herramienta, como forma de conocer dónde contactarlos.

2.3. Lista de reservas del producto

En esta sección se define la lista de reserva del producto, el equivalente a requerimientos funcionales del sistema de la metodología RUP[35]. Luego de previas consultas al cliente se definió un total de 41 aspectos funcionales, los cuales pueden ser cambiantes con el tiempo, en relación a la petición del cliente. De ahora en lo adelante se utilizará la abreviatura (RF) para identificar a cada funcionalidad de la aplicación. A continuación se precisan las mencionadas funcionalidades, agrupadas en relación a su propósito dentro del sistema.

Para la gestión de usuarios en el portafolio

RF 1. Gestionar usuarios del sistema.

- **RF 1.1.** *Registrar usuario en el sistema*: puede ingresar su nombre, apellidos, correo electrónico, si desea recibir notificaciones por correo, idioma personal y contraseña, y con el registro se crea su propio portafolio personal**.**
- **RF 1.2.** *Mostrar los datos del perfil del usuario:* muestra los datos del perfil del usuario autenticado.

[35] Proceso Unificado de *Rational,* el cual constituye un proceso de desarrollo de software.

- **RF 1.3.** *Editar perfil de usuario*: puede editar nombre, apellidos, correo electrónico, incluir o cambiar su foto personal, si desea recibir notificaciones por correo, idioma preferido y contraseña.

RF 2. *Autenticar usuario en el sistema*: debe autenticarse en el sistema para tener acceso a todos los espacios de trabajo que este brinda y puede salir del sistema una vez que lo desee.

Para la gestión de archivos personales en el portafolio

RF 3. Gestionar archivos personales.

- **RF 3.1.** *Visualizar archivos personales*: muestra el listado de los archivos almacenados en el sistema por el usuario.
- **RF 3.2.** *Crear un nuevo archivo personal:* el usuario puede almacenar sus archivos personales incluyéndolos en el sistema.
- **RF 3.3.** *Editar archivo personal*: el usuario puede editar los parámetros del archivo que seleccione: nombre, descripción etiquetas asociadas y archivo digital.
- **RF 3.4.** *Eliminar archivo personal*: el usuario puede eliminar el archivo personal que desee.
- **RF 3.5.** *Ver archivo personal:* el usuario puede abrir el archivo almacenado en el sistema.

Para la gestión de evidencias personales en el portafolio

RF 4. Gestionar evidencias personales.

- **RF 4.1.** *Visualizar evidencias personales*: muestra el listado de las evidencias creadas en el sistema por el usuario.
- **RF 4.2.** *Incluir nueva evidencia personal:* puede incluir una nueva evidencia personal, adjuntando un archivo si lo desea. Los datos a ingresar son: nombre, descripción, archivo(s) adjuntos y etiquetas asociadas.
- **RF 4.3.** *Editar parámetros de la evidencia:* puede editar los datos de la evidencia seleccionada. Los datos que se pueden editar son: nombre, descripción, archivo(s) adjuntos y etiquetas asociadas.
- **RF 4.4.** *Eliminar evidencia personal:* puede eliminar una evidencia personal y, a su vez, todos los datos creados con ella.
- **RF 4.5.** *Visualizar detalles de la evidencia:* se refiere a que puede visualizar los datos de la evidencia seleccionada.
- **RF 4.6.** *Ver archivo(s) personal(es) asociado(s) a la evidencia:* el usuario puede abrir el archivo asociado a la evidencia.

RF 5. Compartir evidencia personal.

- **RF 5.1.** *Mostrar un listado de los usuarios registrados en el sistema a los que se pueda compartir una evidencia*.
- **RF 5.2.** *Compartir evidencia personal*: una vez que se seleccione el (los) usuarios a los que se desea compartir una evidencia, se ejecuta esta acción.

RF 6. Consultar evidencias compartidas.

- **RF 6.1.** *Mostrar las evidencias compartidas por el usuario:* muestra un listado de todas las evidencias que el usuario ha compartido con los demás registrados en el sistema.
- **RF 6.2.** *Mostrar las evidencias que le han sido compartidas al usuario*: muestra un listado de las evidencias que le han sido compartida al usuario en cuestión.
- **RF 6.3.** *Valorar evidencia compartida*: los usuarios a los que se les compartió la evidencia pueden emitir una valoración de la misma, seleccionando un valor en un rango de 1 a 5.

Para la gestión de secciones personales

RF 7. Gestionar secciones.

- **RF 7.1.** *Visualizar secciones creadas:* muestra un listado de todas las secciones que el usuario ha creado en su portafolio.
- **RF 7.2.** *Incluir nueva sección:* se puede crear una nueva sección donde se concentren todas las evidencias creadas por el usuario. Se debe especificar título, descripción y las etiquetas asociadas en caso que lo desee el usuario.
- **RF 7.3.** *Editar datos de la sección:* se pueden editar los datos de una sección seleccionada por el usuario. Los campos a editar serán título, descripción y etiquetas asociadas.
- **RF 7.4.** *Visualizar datos de la sección*: muestra todos los datos de la sección seleccionada y a continuación un listado de todas las evidencias personales asociadas a la sección.
- **RF 7.5.** *Eliminar sección:* se pueden eliminar la(s) secciones que el usuario estime conveniente, eliminándose con esta acción toda la información contenida dentro de esta.

Para la gestión de notificaciones del sistema

RF 8. Consultar notificaciones personales: las operaciones que emitirán una notificación pueden ser al compartir una evidencia, al valorar una evidencia y al comentar una evidencia compartida.

- **RF 8.1.** *Mostrar un listado de notificaciones:* muestra el listado de las notificaciones recientes enviadas al usuario del sistema. El usuario puede seleccionar “ver todas las notificaciones” o seleccionar “ver las que no han sido leídas”.
- **RF 8.2.** *Eliminar notificaciones:* puede eliminar la notificación que desee una vez que la haya revisado.
- **RF 8.3.** *Ver detalles de la notificación:* puede visualizar los datos de la notificación seleccionada.

RF 9. *Exportar datos del portafolio*: se realiza bajo las especificaciones del estándar Leap2A.

Para la gestión de comentarios: los requisitos de incluir comentarios se encontrarán en la sección Evidencias compartidas del portafolio personal.

RF 10. Gestionar comentarios: los comentarios se realizan cuando una evidencia se encuentre compartida a otros usuarios.

- **RF 10.1.** *Incluir comentario en evidencia:* el usuario puede emitir comentarios a sus evidencias, como a las evidencias que le han sido compartidas.
- **RF 10.2.** *Ver cometarios de la evidencia:* se muestra un listado de todos los comentarios realizados a la evidencia compartida.

Para la integración mediante servicios web: la aplicación debe brindar servicios a otras aplicaciones informáticas de acuerdo a sus funcionalidades, además de poder obtener información que manejen las mismas.

RF 11. *Brindar servicios web:* el sistema brindará una serie de funcionalidades a través del uso de servicios web, con el objetivo de que otros sistemas puedan hacer uso de las mismas. Este requerimiento se especifica más detalladamente en la sección **Especificación de servicios web.**

Para el envío de mensajes del sistema

RF 12. Mensajes del sistema.

- **RF 12.1.** *Enviar Mensajes de contacto:* el sistema debe ser capaz de enviar mensajes de contacto a los administradores del sistema, de acuerdo a alguna inconformidad de los usuarios con el mismo.
- **RF 12.2.** *Enviar mensaje de notificación del portafolio:* si el usuario lo desea, el sistema le enviará un mensaje vía correo electrónico, con notificaciones y actividades que han tenido lugar en este durante su ausencia.

Para el sistema de búsqueda de la aplicación: el sistema debe ser capaz de tener implementado un sistema de búsqueda que propicie, en cualquier espacio de trabajo que se encuentre el usuario, poder visualizar los elementos que este requiera.

RF 13. Sistema de búsqueda del portafolio.

- **RF 13.1.** *Filtrar búsqueda de archivos:* cuando se va a crear una nueva evidencia, este espacio cuenta con un filtrado que permite buscar los archivos almacenados en el portafolio, de acuerdo a un patrón de búsqueda.
- **RF 13.2.** *Filtrar búsqueda de usuarios:* cuando se va a compartir una evidencia, este espacio cuenta con un filtrado que permite buscar los usuarios registrados en el sistema, de acuerdo a

un patrón de búsqueda.

- **RF 13.3.** *Búsqueda global:* el usuario tiene la posibilidad en todo momento de buscar aquellos elementos del portafolio que coincidan con su patrón de búsqueda.
- **RF 13.4.** *Búsqueda por etiquetas:* el usuario tiene la posibilidad de realizar búsqueda de elementos mediante las etiquetas asociadas a cada uno de ellos.

RF 14. *Internacionalización del idioma del portafolio:* el usuario tiene la posibilidad de cambiar el idioma del portafolio, en cualquier espacio o sección en que se encuentre trabajando. Los idiomas soportados serán inglés y español.

Para la administración de usuarios del sistema: en este espacio solamente podrán tener acceso los administradores del sistema, y a su vez acceder a las funcionalidades que este brinda.

RF 15. Administración de usuarios del sistema.

- **RF 15.1.** *Visualizar usuarios registrados en el sistema:* el administrador puede visualizar todos los datos de los usuarios registrados en el sistema.
- **RF 15.2.** *Eliminar usuarios del sistema:* el administrador puede eliminar un usuario del sistema.

2.4. Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable (Cruz Castro, 2011).

Los requerimientos no funcionales son cualidades con las que debe cumplir el *software*, las cuales determinan en muchos casos el nivel de calidad y de acabado del mismo (Hidalgo Urbino, 2010).

Para especificar los requerimientos del *software* se definieron una serie de categorías que engloban las capacidades que debe mantener un sistema para su correcto funcionamiento y una prestación adecuada de sus servicios.

- **Usabilidad**

RNF 1. Los elementos gráficos como los íconos deberán contar con un *tooltype* o mensaje flotante que señalen el tipo de recurso al que se refiere.

RNF 2. La aplicación deberá contar con una interfaz y navegación funcional, asequible a todo tipo de usuario. Se debe tener acceso al menú general desde cualquiera de las páginas.

RNF 3. El sistema debe mantener al usuario informado de las operaciones que este realiza a través de la interacción con el mismo.

- **Soporte**

RNF 4. La aplicación debe ser implementada bajo tecnología web, puesto que la misma será accedida a través de Internet cuando se encuentre prestando servicios como una aplicación independiente.

RNF 5. Ejecutarse sobre cualquier navegador, siendo como mínimo compatible con:

- Explorer 8.0 y superior.
- Mozilla Firefox 7.0 y superior.
- Opera 10.0.0 y superior.
- Chrome 7.0 y superior.

- **Hardware**

RNF 6. Los usuarios finales deberán contar como mínimo con:

- Procesador Pentium II o superior.
- 512 MB de RAM.
- 15 GB de HDD.
- Si no cuentan con un servidor local: conexión de banda ancha de 256Kbps como mínimo.

- **Restricciones de diseño e implementación**

RNF 7. Se aplicará la programación orientada a objetos.

RNF 8. El marco de trabajo de desarrollo que se utilizará es: Symfony v2.3.7.

RNF 9. Como IDE se empleará NetBeans v7.4.

RNF 10. Se empleará como SGBD PostgreSQL v9.3

RNF 11. Como servidor web se empleará Apache v2.4.7.

RNF 12. Se utilizará un estándar de codificación definido por el Centro FORTES.

- **Eficiencia**

RNF 13. Cuando una sesión permanece inactiva durante un tiempo especificado, el sistema debe cerrar dicha sesión de forma automática.

- **Apariencia o Interfaz externa**

RNF 14. Mantendrá en todo momento un diseño sencillo, con pocas imágenes y gráficos, con el objetivo de elevar el nivel de respuesta del sistema ante las peticiones del usuario.

2.5. Descripción de los roles del sistema

Los roles identificados son aquellas personas que van a interactuar con el sistema, en

dependencia del espacio de trabajo en que se encuentre.

Tabla 4. Descripción de los roles del sistema.

Rol	Descripción
Invitado	Rol que tiene acceso al sistema pero que aún no se ha registrado en el mismo. Tiene la posibilidad de registrarse y con ello crearse un portafolio personal, por lo que pasaría a fungir como rol de **Personal**; así como acceder a las secciones de "Acerca de" y "Contáctenos".
Personal	Este rol debe encontrarse registrado en el sistema y tiene los privilegios de hacer uso de los distintos espacios de trabajo que ofrece la herramienta, dígase "Archivos", "Secciones", "Evidencias", "Perfil", "Notificaciones", "Opciones de Exportar" y, además, las secciones "FAQ", "Acerca de" y "Contáctenos".
Administrador	Este rol debe encontrarse registrado en el sistema y tiene los privilegios de hacer uso de los distintos espacios de trabajo que ofrece la herramienta, dígase "Archivos", "Secciones", "Evidencias", "Perfil", "Notificaciones", "Opciones de Exportar" y, además, las secciones "FAQ", "Acerca de" y "Contáctenos". Además puede hacer uso de la sección de administración de la herramienta para la gestión y control de usuarios.

2.6. Exploración

Es la primera fase de la metodología XP, en la que se define el alcance general del proyecto, se exploran las diferentes formas de resolver problemas concretos de implementación que puedan presentarse y sobre todo se definen historias de usuario, que son la forma de definir los requisitos del sistema a implementar. Los programadores estiman los tiempos de desarrollo en base a esta información la cual podría variar cuando se analicen más en detalle en cada iteración. Debe quedar claro que las estimaciones realizadas en esta fase son primarias (ya que estarán basadas en datos de muy alto nivel), y podrían variar cuando se analicen más en detalle en cada iteración (Joskowics, 2008).

El propósito de la exploración es llegar a un entendimiento global entre desarrolladores y el cliente acerca de qué es lo que el futuro sistema debe hacer. Para ello se toman los requerimientos funcionales que se han definido y comienza el proceso de descripciones de las historias de usuario, los análisis en grupo y las tormentas de ideas. Se esclarecen las dudas sobre procesos que deben ser automatizados y que no han sido bien entendidos (Hidalgo Urbino, 2010).

2.6.1. Historias de usuario

Del inglés *User Stories* (en lo posterior HU), son la forma en que se especifican en XP los requisitos del sistema, las mismas no deben ser descritas en más de tres líneas e idealmente es el

cliente quien las redacta y prioriza. Por tanto serán descripciones cortas y escritas en el lenguaje del usuario, sin terminología técnica (Cruz Castro, 2011).

Las HU deben tener el detalle mínimo como para que los programadores puedan realizar una estimación poco riesgosa del tiempo que llevará su desarrollo. Cuando llegue el momento de la implementación, los desarrolladores dialogarán directamente con el cliente para obtener todos los detalles necesarios. Las HU deben poder ser programadas en un tiempo entre una y tres semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia (Joskowics, 2008).

Las HU pueden agrupar uno o varios requerimientos funcionales identificados. XP recomienda que las HU sean escritas por el propio cliente. Es por ello que debe existir una buena relación entre los clientes y el grupo de desarrollo, la falta de comunicación entre ambos factores influye directamente en el fracaso del proyecto. Las HU son priorizadas y colocadas de acuerdo a su prioridad en las iteraciones (Hidalgo Urbino, 2010), las cuales se verán más adelante. En esta sección se muestra una HU, las restantes pueden ser consultadas en el **Anexo 14: Especificación de HU.**

HU1 Gestionar usuarios del sistema

RF relacionados:

- **RF 1.1.** Registrar usuario en el sistema**.**
- **RF 1.2.** Mostrar los datos del perfil del usuario.
- **RF 1.3.** Editar perfil de usuario.

Tabla 5. HU Gestionar usuario del sistema.

Historia de usuario	
No.: 1 **Nombre:** Gestionar usuarios del sistema	
Usuario: Todos	
Prioridad en el negocio: Alta	**Nivel de complejidad:** Medio
Estimación: 5d	**Iteración asignada:** 1
Descripción: el usuario que desee interactuar con el sistema debe registrar los datos que se especifican y luego puede visualizar los datos personales de su perfil, además de poder editarlos.	
Información adicional (observaciones): da cumplimiento al *RF 1. Gestionar usuarios del sistema.*	

2.7. Planificación

La planificación es una fase corta, en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las HU, y asociadas a éstas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un Plan de Entregas, o *Release Plan* (Joskowics, 2008).

"Toda técnica de planificación de software debe tratar de crear visibilidad, de tal manera que todo individuo involucrado en el proyecto pueda realmente ver cuán largo es el proyecto. Esto significa que se necesitan metas claras, metas que no sean dudosas y que claramente representen progreso. Las metas deben además ser cosas que toda persona involucrada con el proyecto, incluyendo el cliente, pueda entender y aprender a confiar en éstas " (Beck y Fowler, 2001).

Esta fase de planificación se muestra como una de las más importantes dentro de la metodología XP, ya que se estiman los esfuerzos, por parte de los desarrolladores, para implementar las funcionalidades de la aplicación. Se necesita un constante intercambio con el cliente para suplir cualquier duda que surja durante el desarrollo del sistema y de existir algún cambio, que todas las partes involucradas sean informadas del mismo y de las consecuencias reales que arraigará tal evento. Además se hace necesario planificar la utilización de recursos humanos y materiales para su utilización en cualquier circunstancia del proceso.

En el **Anexo #4: Estimación de esfuerzos por HU**, se puede consultar este artefacto que genera la metodología seleccionada.

2.7.1. Plan de iteraciones

Después de haber conformado las HU y estimado el esfuerzo que será dedicado para su implementación, se procede a la etapa de planificación de implementación del sistema. En este plan se especifica el orden de prioridad con que serán implementadas las HU y las posibles fechas de liberación para cada una.

Los equipos de desarrollo deben decidir la longitud de las iteraciones y se recomienda que todas tengan la misma duración, pues permite que el equipo se acostumbre a seguir un ritmo. XP recomienda que deben ser cortas, de entre 2 y 3 semanas, pues los ciclos cortos permiten que el mismo cliente pruebe constantemente el software a partir de los casos de prueba. Alargar la longitud de las iteraciones en ciclos mayores de 1 mes permite que errores indeseados sean detectados de forma tardía (Hidalgo Urbino, 2010).

La implementación de las HU se planificó para que fueran realizadas, en primer orden, las de mayor interés para el negocio. Además se tuvo en cuenta que, aquellas funcionalidades que dependían de otra para su implementación, se colocaran en la misma iteración, permitiendo que el flujo de trabajo se realice de forma correcta y de acuerdo a lo establecido por el cliente. Para cada iteración se destinó una duración de 3 semanas, cada semana con 5 días laborables de 8 horas cada una, cumpliendo con las 40 horas semanales que plantea la metodología XP.

2.7.2. Plan de duración de iteraciones

Siguiendo el desarrollo de la metodología XP se crea el plan de duración de las iteraciones. En este plan se especifica más detalladamente el orden de desarrollo de las HU dentro de cada

iteración, así como la estimación completa de dicha iteración. El Plan de duración de las iteraciones se puede consultar en el **Anexo #5: Plan de duración de las iteraciones**.

2.7.3. Plan de entregas

En el **Anexo #6: Plan de entregas,** se puede consultar todo lo referente a los entregables por cada iteración de la implementación.

2.8. Tarjetas Clases – Responsabilidades – Colaboraciones

Las tarjetas CRC se definen para identificar las principales clases por las cuales está compuesta la aplicación informática y para definir, por cada una de las clases identificadas, la función que ejerce en el flujo de negocio y las relaciones que mantienen con otras clases del sistema. Este artefacto se especifica con el objetivo de obtener un diseño simple y evitar la implementación de funcionalidades innecesarias, además de proveer un refinamiento de las clases analizadas. Las tarjetas CRC pueden ser consultadas en el **Anexo #7: Especificación de las tarjetas CRC.**

2.9. Arquitectura de la información

La Arquitectura de la Información (AI), según (Reyes, 2012), es el conjunto de métodos y herramientas que permiten organizar los contenidos, para ser encontrados y utilizados por los usuarios, de manera simple y directa. La AI estará cumpliendo sus objetivos cuando un usuario entre por primera vez al sitio y pueda reconocer a quién pertenece el sitio web; lo pueda entender en forma rápida, sin esfuerzo y encontrar la información ofrecida fácilmente. Adicionalmente eso entregará el beneficio de que, quienes producen el sitio, podrán ubicar la nueva información sin tener que crear nuevas estructuras.

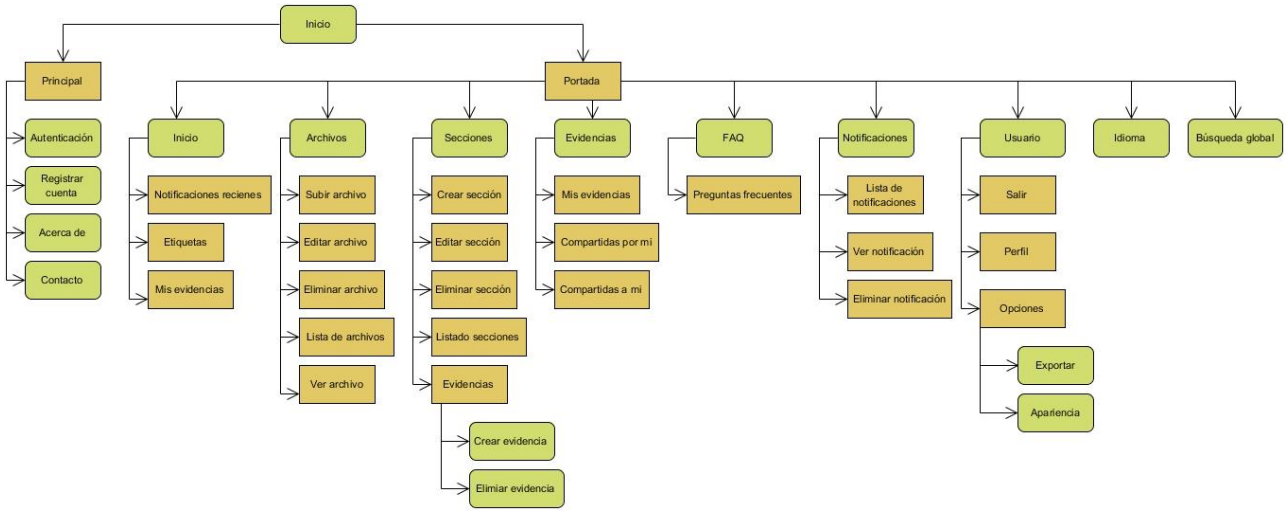


Ilustración 4. Representación de la Arquitectura de la Información del sistema.

2.10. Conclusiones parciales del capítulo

Luego de realizar un análisis de los principales conceptos que intervienen en el negocio, se logró obtener una propuesta de solución que dará respuesta a la problemática de la investigación. Por

ello se concluye que:

1. Teniendo en cuenta los diferentes contextos en los que estará involucrado el sistema final, se decidió que el mismo funcionará en dos contextos fundamentales: como herramienta independiente o brindando sus funcionalidades a otros sistemas informáticos, a través del uso de servicios web.
2. Atendiendo a los requerimientos que debe presentar el sistema, definidos por parte del cliente, se identificaron un total de 41 aspectos funcionales que darán soporte a todo el funcionamiento del mismo.
3. Prestando atención a lo planteado por la metodología seleccionada en relación a la distribución y planificación del trabajo de producción, se decidió establecer un total de 4 iteraciones con una duración de 3 semanas, cada semana con 40 horas laborales.

Capítulo 3: Implementación y pruebas

Luego de ser identificado y descrito la lista de reserva del producto a desarrollar, se hace necesario realizar un análisis y diseño de la misma, con el objetivo de obtener un sistema con una arquitectura robusta y capaz de sobrevivir a cambios. En el diseño se modela el sistema con el objetivo de dar una mejor comprensión del funcionamiento del mismo. Se definen los estilos arquitectónicos utilizados, así como los patrones de diseño que se manejaron en la implementación. Como parte de la implementación se identifican las tareas de ingeniería que darán cumplimiento a cada uno de los objetivos planteados para la investigación. Se especifican los tipos de pruebas que se aplicarán al sistema una vez concluido la fase de implementación, y se elaboran los casos de pruebas. Los servicios que brindará la herramienta se describen, teniendo en cuenta los parámetros que requiere cada uno y las respuestas para cada petición.

3.1. Modelo arquitectónico

En la construcción de software existen diversos componentes que proporcionan una filosofía de trabajo para los equipos de desarrollo, creando una visión que unifica la forma en que los miembros del equipo ven el sistema y además impone una transformación del mismo. Entre los mencionados componentes se encuentran los estilos arquitectónicos, patrones arquitectónicos y patrones de diseño, los cuales se encuentran estrechamente relacionados, conformando la base de la arquitectura de un software.

Existen diversas definiciones en lo que respecta a arquitectura de software, pero todas se ven reflejadas en las siguientes ideas fundamentales:

1. La arquitectura de software como un proceso particular dentro del ciclo de vida.
2. La arquitectura de software como la topología o la forma de articular distintos componentes en una solución.
3. La disciplina académica y profesional de la arquitectura de software.

Por ello la arquitectura de software se entiende como la organización fundamental de un sistema representada en sus componentes, las relaciones existentes entre ellos y con el entorno, y los principios que orientan su diseño y evolución (*IEEE Computer Society*, 2000).

3.1.1. Estilo arquitectónico

Un estilo arquitectónico es una transformación impuesta al diseño de todo un sistema, cumpliendo el objetivo de establecer una estructura para todos los componentes del mismo. Presentan un alcance global dentro de los sistemas ya que se encuentra enfocado en toda la arquitectura del

software (Pressman, 2005).

Por la forma en que se encuentra concebido el sistema, fueron aplicados dos estilos arquitectónicos: la arquitectura de llamada y retorno y la arquitectura orientada a objetos.

Arquitectura de llamada y retorno: este estilo fue aplicado por la necesidad existente de separar las funciones de la aplicación, principalmente las que se utilizan entre las interfaces y el modelo de datos, para lograr optimizar las peticiones que se realicen por parte de los usuarios. Además permite que una interfaz de usuario pueda mostrar múltiples vistas de los mismos datos simultáneamente.

Arquitectura orientada a objetos: los componentes del estilo se basan en principios Orientados a Objetos: encapsulamiento, herencia y polimorfismo. Son asimismo las unidades de modelado, diseño e implementación y los objetos y sus interacciones, el centro de las responsabilidades en el diseño de la arquitectura y en la estructura de la aplicación (Reynoso y Kicillof, 2004).

En sentido general se aplica esta arquitectura para encapsular aquellos atributos que pertenecen a un concepto o entidad específico, haciendo posible su reutilización a través de llamadas a métodos donde intervienen dentro de la lógica del negocio.

3.1.2. Patrón arquitectónico Modelo – Vista – Controlador

Una de las razones por las que se utiliza el patrón Modelo – Vista – Controlador (MVC) es por el nivel de organización que provee en las aplicaciones informáticas que, conjuntamente con la estructura interna que plantea el *framework* Symfony2, hace posible que la envergadura con que se desarrolla un sistema informático fluya de forma organizada. Este patrón separa la arquitectura de una aplicación en tres capas: modelo, vista y controlador. Cada una de estas capas tiene su funcionamiento específico, y una depende de otra para cumplir con sus responsabilidades dentro de la arquitectura.

El ***Modelo*** representa la información con la que trabaja la aplicación web, representando un intermediario entre la base de datos y el resto del sistema. Es el responsable del acceso directo a los datos, a través de funciones y restricciones que especifica la lógica del negocio. Esta capa hace uso del ORM Doctrine dentro de la arquitectura que plantea el *framework* Symfony2, el cual define funcionalidades y estructura para realizar las consultas y extracción de datos.

La ***Vista*** es la representación visual de los datos que provee el sistema, permitiendo que el usuario interactúe con el mismo de forma dinámica. Symfony2 plantea una estructura denominada

Tres Capas[36], para el sistema de interfaces de usuario, utilizando *Twig*, el cual se presenta como un lenguaje de plantillas moderno, seguro, rápido y con el que se puede crear plantillas concisas y muy fáciles de mantener (Eguiluz, 2012).

El ***Controlador*** es quien ejecuta las acciones dentro de la lógica del sistema. Funge como un intermediario entre las peticiones del usuario y lo que el sistema debe mostrar. Esta capa basa su funcionamiento en las solicitudes del usuario, obteniendo datos del modelo y visualizándolos nuevamente al usuario en una plantilla perteneciente a la vista.

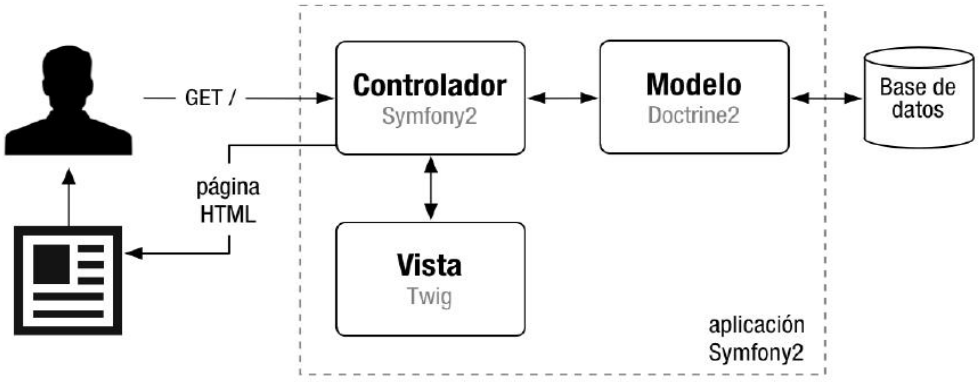


Ilustración 5. Esquema de la arquitectura interna de Symfony2 utilizando el patrón MVC.

3.1.3. Patrones de diseño

Una de las principales ventajas de utilizar *framework* de desarrollo es que están basados en patrones de diseño, característica que hace posible la gran usabilidad que tienen, casi siempre independientes del tipo de aplicación web que se desee desarrollar (Pérez, Ramírez, González y Vargas, 2011).

Los patrones de diseño brindan una buena solución a los problemas comunes de diseño en el desarrollo de sistemas informáticos y otras cuestiones relacionadas como diseño de interacción o interfaces de usuario. Para ello, Symfony2 define una serie de patrones de diseño, que se encuentran embebidos en su arquitectura, concebidos para que cualquier desarrollador los use en

[36] Representa una buena práctica en el trabajo con el *framework* Symfony2 y define que la aplicación debe estar estructurada en tres capas de plantillas fundamentales: plantilla para representar el maquetado de toda la aplicación, plantilla para representar la estructura general de las interfaces y las plantillas específicas para cada funcionalidad de la lógica del negocio.

pos de lograr un exitoso diseño en su sistema. Los patrones GRASP[37], incluidos dentro del propio *framework* Symfony2, describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones (Larman, 2004). En la siguiente sección se explican los patrones de diseño utilizados directamente en la solución.

Experto: dentro de la arquitectura que presenta el *framework* Symfony2, este patrón se muestra muy evidenciado en la capa perteneciente al modelo de datos. En la misma se encuentran, las clases encargadas de interactuar directamente con la base de datos a través del ORM Doctrine, las cuales generalmente se encuentran bajo el nombre de *name_entityRepository,* presentando la responsabilidad de realizar directamente las acciones de consultas debido a que contienen los atributos necesarios para realizar dichas operaciones.

Creador: este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento (Larman, 2004). El mismo se evidencia en la acción ***newAction()*** del archivo ***ArchivoController.php*** cuando se crea una instancia de la entidad ***Archivo*** para encapsular los datos que inserta el usuario e insertarlos en la base de datos.

```
public function newAction()
{
    $archivos = $this->getUser()->getPortafolio()->getArchivos();
    $paginator = $this->get('knp_paginator');

    try {
        $pagination = $paginator->paginate(
                $archivos, $this->getPage(), $this->container->getParameter('xalix.portafolio.archivos
        );
    } catch (QueryException $e) {
        throw $this->createNotFoundException('Page not found');
    }

    $entity = new Archivo();

    $form = $this->createCreateForm($entity);

    return $this->render('XalixPortafolioBundle:Archivo:new.html.twig', array(
                'form' => $form->createView(),
                'archivos' => $pagination,
    ));
}
```

Ilustración 6. Ejemplo de utilización del patrón Creador.

Bajo Acoplamiento: este patrón plantea que la dependencia entre las clases que conforman la arquitectura del sistema debe ser mínima, debido a que a la hora de realizar modificaciones en una clase, no afecte la estructura de las restantes. En la aplicación se concibió mantener

[37] Acrónimo que significa *General Responsability Asignment Software Patterns* (Patrones Generales de Software para Asignar Responsabilidades).

separadas las clases pertenecientes al modelo de datos, las vistas de usuario y las clases controladoras.

Alta Cohesión: este patrón plantea que las clases se deben apoyar del funcionamiento de otras, para lograr su objetivo; y no incluir en ella misma todo el proceso del negocio, ya que sería difícil de comprender, de reutilizar y le afectarían constantemente los cambios. Ejemplo de ello dentro de la aplicación, es a la hora de editar un archivo, la acción ***editAction()*** utiliza la colaboración de otras clases, tanto del modelo como de la vista, y además hace llamada a otra acción ***createEditForm()***, con el objetivo de delegar responsabilidades.

```
public function editAction($id)
{
    $em = $this->getDoctrine()->getManager();

    $entity = $em->getRepository('XalixPortafolioBundle:Archivo')->findArchivoTags($id);

    if (!$entity) {
        throw $this->createNotFoundException('File not found!');
    }

    $editForm = $this->createEditForm($entity);

    return $this->render('XalixPortafolioBundle:Archivo:edit.html.twig', array(
                'entity' => $entity,
                'edit_form' => $editForm->createView(),
    ));
}

private function createEditForm(Archivo $entity)
{
    $form = $this->createForm(new ArchivoType(), $entity, array(
        'action' => $this->generateUrl('archivo_update', array('id' => $entity->getId())),
        'method' => 'PUT',
    ));
```

Ilustración 7. Ejemplo de utilización del patrón Alta Cohesión.

Controlador: el *framework* Symfony2 incluye un *bundle*[38], denominado *FrameworkBundle,* el cual presenta una serie de clases controladoras, encargadas de las principales operaciones dentro del funcionamiento de Symfony2. Además se crean aquellas clases controladoras que darán solución a los principales requerimientos pertenecientes al flujo del negocio en cuestión, facilitando la centralización de las actividades.

Patrones conductuales

Strategy: este patrón plantea la utilización de varias estrategias para representar comportamientos distintos, utilizando una misma interfaz. Estos comportamientos pueden ser modificados e intercambiados en la aplicación, sin que afecte la lógica del sistema. En este trabajo es utilizado cuando se necesita visualizar una entidad **Evidencia.**

[38] Filosofía de desarrollo a partir de Symfony2 de denominar a la estructura de un componente.

```
public function showAction($seccion, $id)
{
    /*
    ...
     *          */
     if ($entity->getGranted()) {
         $form = $this->createForm(new ComentarioType(), new ComentarioModel
     } else {
         $form = $this->createEditForm($entity, $seccion);
     }

     /*...*/
}
```

Ilustración 8. Ejemplo de utilización del patrón *Strategy*.

Command: este patrón se puede observar en el sistema de *routing* de la aplicación, el cual parsea las direcciones URL[39] con el objetivo de precisar los parámetros de la misma y de esta forma informar qué acción debe ejecutarse para responder a la petición. Este comando brinda la posibilidad de delegar a los métodos ***Actions*** la comprobación de los parámetros de la petición.

Patrones estructurales

Decorator: este patrón plantea la utilización de una o varias plantillas globales, que guarden el código que es usual para todo el sistema, para no tener que repetirlo en cada interfaz. En este caso se representan en las plantillas **base.html.twig** y **frontend.html.twig,** el uso del patrón *Decorator*, las cuales establecen el maquetado para todas las plantillas del sistema.

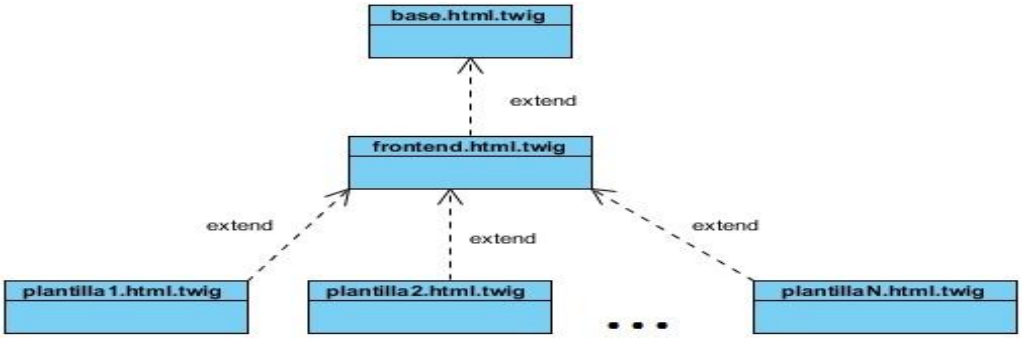


Ilustración 9. Utilización del patrón *Decorator*.

Proxy: este patrón es utilizado cuando se desea que cualquier sistema informático desee hacer uso de las funcionalidades de la herramienta e – Portafolio, y para ello se necesita crear una interfaz que sea capaz de proporcionar el acceso a los recursos que se encuentran en el portafolio, mediante la implementación de servicios web y de una REST API[40].

[39] Sigla en inglés de *Uniform Resource Locator* (Localizador de Recursos Uniformes).
[40] Interfaz de programación de aplicaciones basada en el estilo arquitectónico REST.

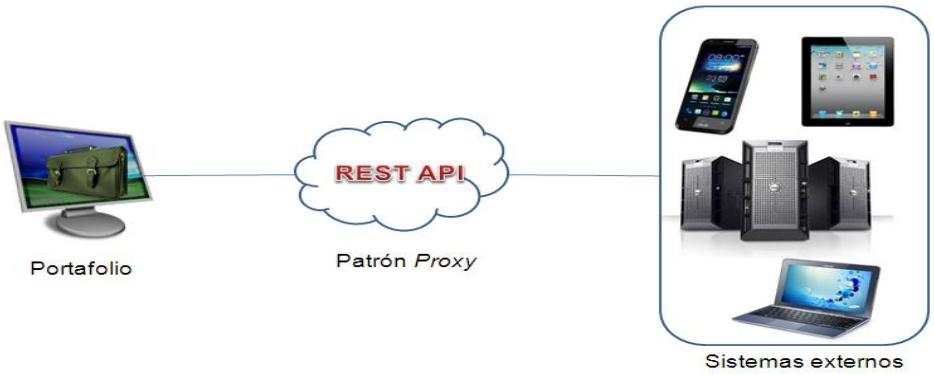


Ilustración 10. Utilización del patrón *Proxy*.

3.2. Diseño de la Base de Datos

En este acápite se presenta el modelo de datos a utilizar en la implementación del sistema, los cuales representan las tablas que conformarán la base de datos de la aplicación, donde serán almacenados los datos necesarios, para darle cumplimiento a los requerimientos del negocio. Las descripciones de cada una de las tablas se pueden encontrar en el **Anexo #8: Descripción de las tablas de la base de datos**.

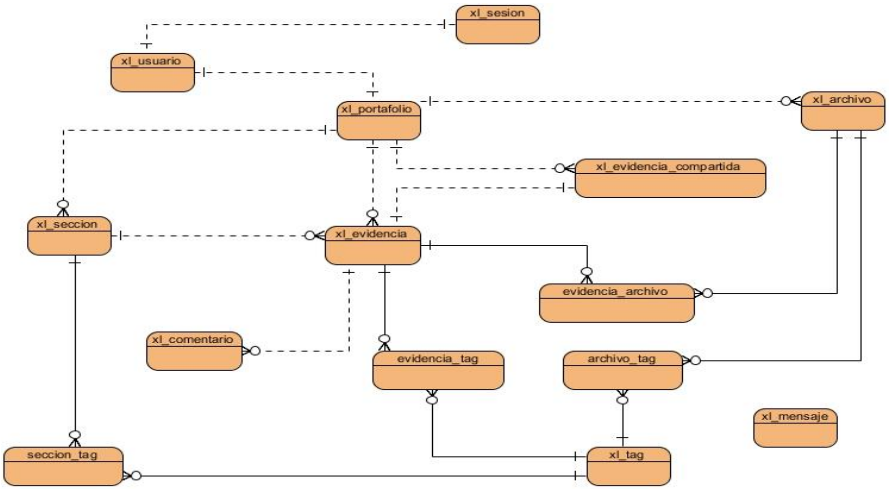


Ilustración 11. Diseño de la Base de Datos.

3.3. Diagrama de paquetes del sistema

El diagrama de paquetes ofrece un nivel de abstracción que permite apreciar cómo se divide el sistema en ciertas agrupaciones de elementos a las que se les denomina paquetes. Este diagrama puede ayudar a repartir entre los desarrolladores la implementación por paquetes (Hidalgo Urbino, 2010). En la presente investigación, se decidió separar la aplicación en cuatro paquetes fundamentales, cada uno de ellos presentando la estructura según el patrón MVC

explicado anteriormente, con el objetivo de organizar en cada uno de ellos, las reglas que presenta el negocio en cada caso.

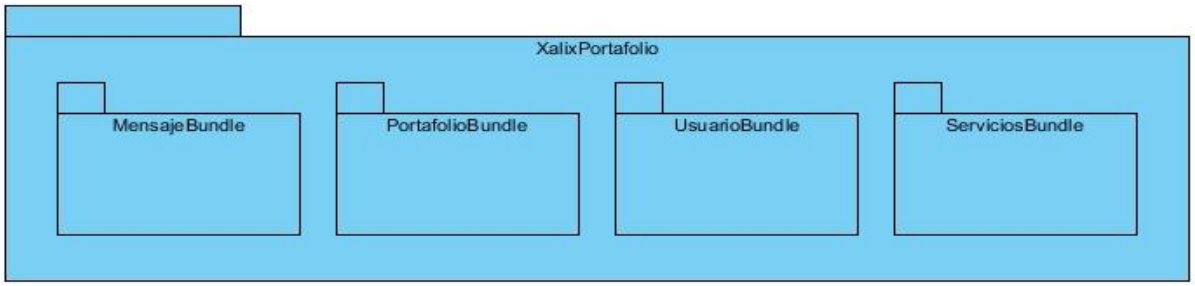


Ilustración 12. Diagrama de paquetes del sistema.

3.4. Diagrama de componentes

Los diagramas de componentes son otro de los artefactos generados en el proceso de desarrollo de software. Hacen posible visualizar la dependencia que existe entre los componentes que forman parte del sistema. Para el presente trabajo se generó un diagrama de componentes para cada paquete por separado, para lograr un mejor entendimiento de cómo se encuentra estructurado el sistema. Los diagramas pueden ser consultados en el **Anexo 12: Diagrama de componentes.**

3.5. Diagrama de Clases del sistema

Este tipo de diagrama se realiza con el objetivo de representar los atributos y operaciones que incluyen las clases que intervienen en el sistema, además se representan las relaciones que existen entre las mismas. Constituye una de las descripciones más detalladas del software, debido al nivel de detalles y especificación con que se realiza. Los diagramas de clases del sistema pueden ser consultados en el **Anexo 13: Diagrama de Clases del Sistema.**

3.6. Modelo de Despliegue

La metodología XP propone la representación de un modelo de despliegue, siempre y cuando no implique mayor esfuerzo en la implementación del mismo, para modelar el hardware utilizado en la implementación del sistema y la relación entre sus componentes. En este caso la aplicación se encuentra alojada en un servidor web y la misma mantiene una comunicación con el servidor de base de datos.

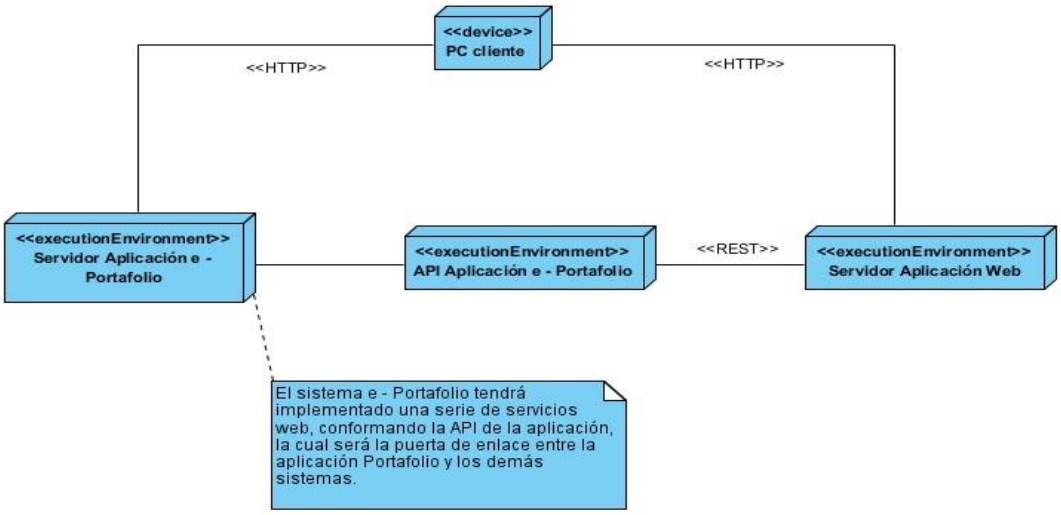


Ilustración 13. Diagrama de despliegue de la aplicación.

3.7. Implementación

En esta fase se implementan las HU definidas anteriormente, como parte del cumplimiento de los requerimientos del sistema. Al principio de esta actividad se realiza una revisión del plan de iteraciones y se modifica en caso de ser necesario. Mediante este plan se descomponen las HU en tareas de desarrollo y se asignan responsables de implementación a cada una de las tareas. Estas tareas son para el uso exclusivo de los programadores, definiéndose en lenguaje técnico y no necesariamente entendible por el cliente. La descripción de las tareas de ingeniería, correspondientes a cada iteración, pueden ser consultadas en el **Anexo #9: Tareas de Implementación.**

3.7.1. Estándares de implementación

Un estándar de codificación completo, comprende todos los aspectos de la generación de código. Al inicio de un desarrollo de software, es necesario establecer un estándar de codificación, para que todos los desarrolladores trabajen sobre un código similar. Si fuera necesario incorporar código fuente previo, o bien realizar mantenimiento a un sistema de software creado anteriormente, el estándar de codificación debería establecer cómo operar con el código ya existente (Pereda, 2013).

En el presente trabajo se tomó como referencia el estándar de codificación definido por el Centro FORTES para el marco de trabajo Xalix, el cual establece una serie de parámetros para el trabajo de codificación (Pereda, 2013).

Para el caso de las tablas de la base de datos, se estableció el prefijo **xl**_, por ejemplo, si una

tabla se llama **evidencia,** en la base de datos se identificaría como ***xl_evidencia*.**

3.8. Especificación de los servicios web

En esta sección se especifica la implementación relacionada con los servicios web que brindará la aplicación. Se muestran en detalles dichos servicios, así como las responsabilidades de cada uno. Esta especificación se realiza con el objetivo de brindar una visión de cómo funcionan cuando el sistema está haciendo uso de los mismos. La especificación de los servicios puede ser consultada en **Anexo 11: Especificación de los Servicios Web.** Para la implementación de los servicios que brindará la aplicación Portafolio, fue utilizado el componente *FOSRestBundle*[41], el cual provee una estructura capaz de implementar dichos servicios de una forma organizada, brindando la especificación de los requerimientos y parámetros para cada servicio web. En el ejemplo que se muestra a continuación, se utiliza la Librería de Cliente URL *(cURL,* de sus siglas en inglés *Client URL Library)*, para consumir los servicios web que brinda la aplicación. Para un mayor entendimiento de la mencionada librería, se muestra en la sección **Bibliografía Consultada** la referencia del manual de uso de la misma.

```php
<?php
//$url = "http://portafolio.local/app_dev.php/api/v1/registros";
$url = "http://localhost/testing/Better/web/app_dev.php/api/v1/accesos.json";
$body = '{"email":"ealopez@estudiantes.uci.cu","password":"barcelona"}' ;
$c = curl_init($url);
curl_setopt($c, CURLOPT_HEADER, 'Accept: application/json');
curl_setopt($c, CURLOPT_POST, true);

curl_setopt($c, CURLOPT_POSTFIELDS, $body);
curl_setopt($c, CURLOPT_RETURNTRANSFER, true);

$token = curl_exec($c);

curl_close($c);
echo $estado;
?>
```

Ilustración 14. Petición para consumir el servicio web “Credencial del usuario”.

Para poder hacer uso de los servicios web implementados en el sistema, deben obtenerse las credenciales del usuario logueado, obteniendo en este caso el *token* correspondiente a dicho usuario, mediante el cual se tendrá acceso a todas las funcionalidades brindadas como servicios web, garantizando de esta forma la seguridad al consumirlos.

[41] Paquete que provee varias herramientas para desarrollar REST API con el *framework* Symfony2.

```
//GET
<?php
$url = "http://localhost:8000/api/v1/secciones.json";
$curl = curl_init($url);
curl_setopt($curl, CURLOPT_HEADER, 'token: fbc7569f1b5324773b6fb7e60a0027e6');
curl_setopt($c, CURLOPT_HEADER, 'Accept: application/json');
curl_setopt($curl, CURLOPT_RETURNTRANSFER, TRUE);
$result = curl_exec($curl);
$error = curl_error($curl);
$http_status = curl_getinfo($curl, CURLINFO_HTTP_CODE);
curl_close($curl);
if( $http_status != "200" ) {
return $error;
}
return $result;
?>
```

Ilustración 15. Petición para consumir el servicio web “Obtener secciones” en formato JSON.

```
//GET
<?php
$url = "http://localhost:8000/api/v1/secciones";
$curl = curl_init($url);
curl_setopt($curl, CURLOPT_HEADER, 'token: fbc7569f1b5324773b6fb7e60a0027e6');
curl_setopt($c, CURLOPT_HEADER, 'Accept: text/html');
curl_setopt($curl, CURLOPT_RETURNTRANSFER, TRUE);
$result = curl_exec($curl);
$error = curl_error($curl);
$http_status = curl_getinfo($curl, CURLINFO_HTTP_CODE);
curl_close($curl);
if( $http_status != "200" ) {
return $error;
}
return $result;
?>
```

Ilustración 16. Petición para consumir el servicio web “Obtener secciones” en formato HTML.

```
//GET
<?php
$url = "http://localhost:8000/api/v1/secciones.xml";
$curl = curl_init($url);
curl_setopt($curl, CURLOPT_HEADER, 'token: fbc7569f1b5324773b6fb7e60a0027e6');
curl_setopt($c, CURLOPT_HEADER, 'Accept: text/xml');
curl_setopt($curl, CURLOPT_RETURNTRANSFER, TRUE);
$result = curl_exec($curl);
$error = curl_error($curl);
$http_status = curl_getinfo($curl, CURLINFO_HTTP_CODE);
curl_close($curl);
if( $http_status != "200" ) {
return $error;
}
return $result;
?>
```

Ilustración 17. Petición para consumir el servicio web “Obtener secciones” en formato XML.

Para las anteriores peticiones al servicio web “Obtener secciones”, el sistema provee las siguientes respuestas, en correspondencia al tipo de formato en que fue enviada la petición:

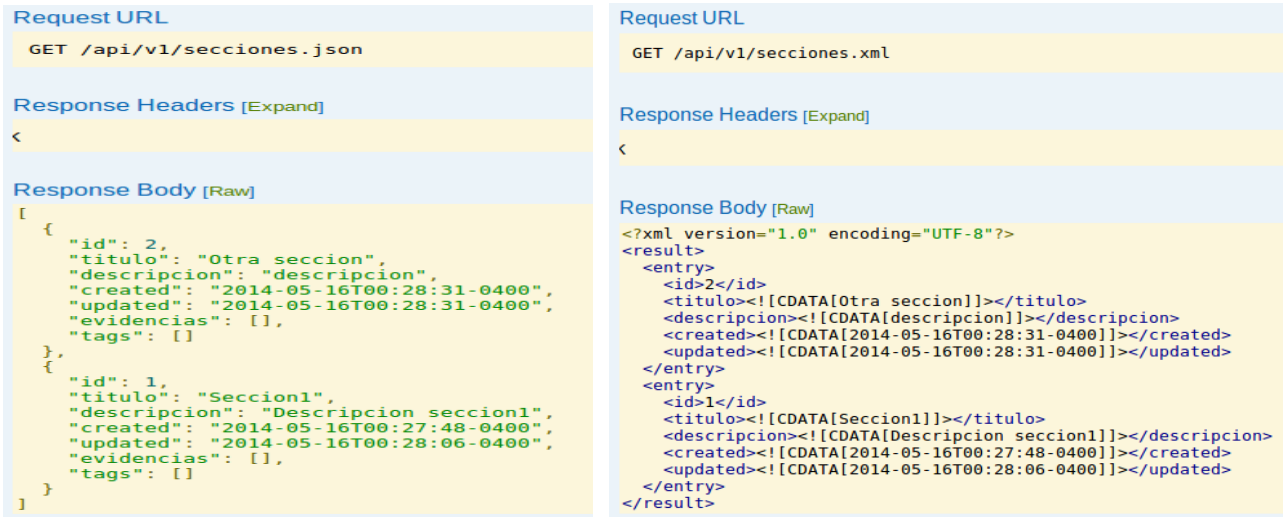


Ilustración 18. Respuesta a la petición para consumir el servicio web "Obtener secciones", en los formatos XML y JSON.

3.9. Pruebas de software

La prueba, son un conjunto de actividades que se planean con anticipación y se realizan de manera sistemática, para lo que se debe definir una plantilla para las pruebas de software (un conjunto de pasos en que se puedan incluir técnicas y métodos específicos del diseño de casos de prueba) (Pressman, 2005).

3.9.1. Pruebas realizadas

Para lograr que un software cumpla con todos los requerimientos propuestos por el cliente, debe someterse a una serie de pruebas o exámenes, con el objetivo de descubrir errores en el funcionamiento externo o interno del sistema. Se aplicaron, en general, tres tipos de pruebas para poder examinar todas las vulnerabilidades que pueda tener el software. A continuación se especifican los tipos de pruebas que se aplicaron.

Pruebas unitarias*:* este tipo de prueba constituye la piedra angular de la metodología XP, debido a que todos los módulos deben pasar las pruebas unitarias antes de ser liberados o publicados. Estas pruebas deben ser definidas antes de realizar el código, y se deben guardar junto a este en caso de futuras correcciones y actualizaciones. Cuando se encuentren errores, estos deben ser corregidos inmediatamente, y se deben definir nuevas pruebas para verificar que el error haya sido resuelto (Joskowics, 2008). Las mismas se aplicaron a las entidades **Mensaje, Comentario, Evidencia, Seccion, Tag** y **Usuario**, y a los servicios web ***ObtenerSeccionbyId*** y ***ObtenerAllSeccion***. Para realizar las pruebas unitarias se utilizó la librería *phpunit,* la cual busca y ejecuta las pruebas que se tienen definidas en el fichero que se le pasa como argumento. Esta

devuelve un informe sobre el estado de las pruebas y los errores que se han ido encontrando. En total se ejecutaron 22 pruebas unitarias, dentro de las cuales se realizaron 112 comprobaciones. Una muestra de cómo se implementaron las pruebas unitarias y los resultados que se obtuvieron luego de aplicarlas, pueden ser visualizados en el **Anexo #15: Pruebas unitarias.**

Pruebas de aceptación: el autor (Joskowics, 2008) plantea que estas pruebas son creadas en base a las HU definidas, y son el equivalente a las pruebas de caja negra definidas bajo la metodología RUP. El cliente debe definir diversos escenarios para comprobar que una HU ha sido correctamente implementada. Son los clientes los responsables de verificar que los resultados de estas pruebas sean correctos. Se hace necesario publicar los resultados de las pruebas de aceptación, para que todo el equipo de desarrollo esté al tanto de las mismas.

Pruebas de servicios web: para verificar que la Interfaz de Programación de Aplicaciones (API, de su siglas en inglés *Application Programming Interface*), encargada de la publicación de los servicios web de la aplicación Portafolio, funcione correctamente cuando se encuentre integrado a otra aplicación informática, se decide aplicar este tipo de prueba. Para ello se utilizó la interfaz que provee el componente mencionado en el epígrafe anterior (*FOSRestBundle*), que muestra una serie de parámetros y requerimientos que hacen posible probar cada servicio implementado. Este componente además permite ver la respuesta que genera luego de efectuada la petición.

3.9.2. Diseño de Casos de Prueba

En este epígrafe se especificaron los diseños de casos de prueba del sistema por cada iteración, correspondientes a cada HU, los cuales pueden ser consultados en el **Anexo #10: Diseño de Casos de Prueba**.

3.9.3. Resultados obtenidos

Las pruebas de aceptación se realizaron en cuatro iteraciones, con el objetivo de verificar que las funcionalidades creadas en cada entrega del *software,* arrojaran el resultado esperado. En las iteraciones se obtuvieron los siguientes resultados:

- **Primera iteración:** 17 No Conformidades (NC), de ellas 10 significativas, 5 no significativas y 2 recomendaciones.
- **Segunda iteración**: 8 NC, de ellas 5 significativas, 2 no significativas y 1 recomendación.
- **Tercera iteración**: 6 NC, de ellas 3 significativas, 1 no significativa y 2 recomendaciones.

- **Cuarta iteración**: 1 NC que no representó relevancia para el funcionamiento del sistema.

En sentido general se resolvieron todas las NC por parte del equipo de desarrollo, donde los resultados de dichas pruebas se representan en la siguiente gráfica:

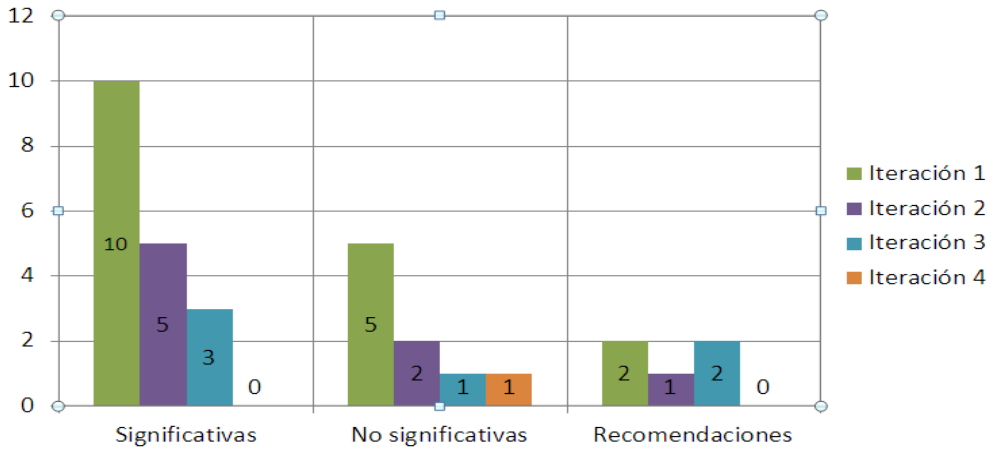


Ilustración 19. Resultados de las pruebas de aceptación.

3.9.4. Pruebas para la comprobación de las especificaciones del estándar Leap2A

Para comprobar el correcto funcionamiento del estándar implementado, se exportaron los datos del portafolio personal del estudiante y se importó en la herramienta Mahara[42]. Esta herramienta importó correctamente la información del portafolio del usuario, dígase su perfil, sus evidencias personales, las secciones donde estas se guardan y los archivos personales del mismo. De esta forma se logró cumplir con las especificaciones del estándar, logrando la interoperabilidad entre dos sistemas.

3.10. Conclusiones parciales del capítulo

Al finalizar el presente capítulo se obtuvo un sistema totalmente funcional, cumpliendo con los principales procesos del negocio que se plantearon para su desarrollo. Para lograr este objetivo se tuvieron en cuenta los siguientes aspectos:

1. Se definió la arquitectura por la que está compuesta el sistema, así como los patrones de diseño que se utilizaron para su implementación. La utilización de estos patrones logró establecer un uso adecuado de buenas prácticas en la implementación del sistema.

[42] Herramienta para hacer portafolios electrónicos.

2. Se modelaron los principales procesos que tienen lugar en el funcionamiento del sistema, y se especificaron las características del modelo de datos utilizado, con el objetivo de comprender la solución, y ayudó a determinar cómo se desarrolló el sistema a partir de las funcionalidades identificadas.
3. Se definieron las principales tareas de programación que dieron paso a la implementación del sistema, creando la base central de la aplicación y permitiendo describir cómo los elementos del diseño se implementan y organizan.
4. Una vez desarrolladas todas las funcionalidades previstas, se ejecutó un proceso de pruebas de software, donde se definieron los tipos de pruebas a aplicar en el sistema, con el objetivo de detectar y corregir fallas en el funcionamiento del sistema, aspecto determinante para obtener un producto de alta calidad.

Conclusiones generales

Con la culminación de la presente investigación se arribaron a las siguientes conclusiones:

1. A partir del análisis de la bibliografía consultada se pudo identificar que las herramientas e – Portafolio existentes no presentan un intercambio de información de forma bidireccional con otros sistemas informáticos y que el estándar más utilizado para lograr la interoperabilidad de los datos almacenados en estas herramientas es el Leap2A.
2. Se realizó el proceso de desarrollo de *software,* siguiendo como guía la metodología XP, y utilizando para ello los lenguajes PHP, JavaScript, XML y HTML, la tecnología Ajax para la comunicación asíncrona da datos, los *framework* Symfony2, jQuery y Bootstrap y el estilo de arquitectura REST para la implementación de los servicios web,
3. Se logró la implementación de un portafolio de desarrollo personal que permita el almacenamiento y análisis de evidencias personales entre sistemas informáticos de distintos propósitos, además que pueda interactuar en contextos de forma independiente, o integrado a otros sistemas utilizando para ellos servicios web.
4. La aplicación de pruebas al sistema, tales como pruebas unitarias, pruebas de aceptación, prueba a los servicios web y comprobación del estándar implementado, permitieron validar que la herramienta desarrollada cumple con los aspectos funcionales determinados por el cliente.

Recomendaciones

A continuación se enuncian las recomendaciones a tener en cuenta para futuros trabajos, tomando como referencia el actual:

1. Incluir las opciones de importar los paquetes en formato Leap2A que proveen otras herramientas utilizadas para la creación de portafolios.
2. Brindar la posibilidad al usuario de exportar aquellos recursos que seleccione, dígase evidencias personales, secciones, archivos o información personal.
3. Permitir una mayor personalización de los distintos espacios de trabajo que provee para la organización de la información personal.
4. Brindar soporte al protocolo SOAP para la implementación de servicios web mediante las funcionalidades identificadas para ello.

Glosario de términos

IMS: el nombre formal para el IMS es el *IMS Global Learning Consortium, Inc*. IMS se refiere a la interoperabilidad de los sistemas de aprendizaje y contenidos de aprendizaje y la integración de la empresa con estas capacidades.

Zera: significa semilla.

Leap2A: es una especificación abierta para la transferencia de información, propia del aprendizaje, entre diferentes sistemas.

Interoperabilidad: habilidad de dos o más sistemas o componentes para intercambiar información y utilizar la información intercambiada.

SCRUM: es un marco de trabajo para la gestión y desarrollo de software basada en un proceso iterativo e incremental utilizado comúnmente en entornos basados en el desarrollo ágil de *software*.

CASE: ***C****omputer* ***A****ided* ***S****oftware* ***E****ngineering*, Ingeniería de Software Asistida por Computadora) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software, reduciendo el costo de las mismas en términos de tiempo y de dinero.

DOM: *Document Object Model* (Modelo de Objetos del Documento), es esencialmente una interfaz de programación de aplicaciones (API) que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos.

XSLT: Transformaciones XSL es un estándar de la organización W3C que presenta una forma de transformar documentos XML en otros. Actualmente, XSLT es muy usado en la edición web, generando páginas HTML o XHTML. La unión de XML y XSLT permite separar contenido y presentación, aumentando así la productividad.

Scripts: se le conoce con este término a ciertos lenguajes interpretados.

Framework: es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de *software* concretos, con base a la cual otro proyecto de *software* puede ser más fácilmente organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

ORM: el Mapeo de Objeto Relacional es una técnica de programación para convertir datos entre el sistema, utilizado en un lenguaje de programación orientado a objetos y en una base de datos relacional, utilizando para ello un motor de persistencia. En la práctica esto crea una base de

datos orientada a objetos virtual, sobre la base de datos relacional.

Git: es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente.

GRASP: son patrones generales de software para asignación de responsabilidades. Aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software.

GoF: el libro *Design Patterns: Elements of Reusable Object-Oriented Software*, a menudo apodado *GoF* o *Gang of Four* (*la banda de los cuatro*, debido a sus cuatro autores), es un manual sobre ingeniería del software que describe soluciones a problemas habituales en el diseño de software.

Routing: término que se utiliza para identificar el sistema de rutas dentro de un *software* informático.

Token: también llamado componente léxico es una cadena de caracteres que tiene un significado coherente en cierto lenguaje de programación.

Hipertextual: la relación de un texto con otro anterior, de un texto B con otro texto A de una manera que no es un comentario." A ese texto anterior se le llamaría hipotexto y al posterior hipertexto.

Lenguaje interpretado: lenguaje de programación que está diseñado para ser ejecutado por medio de un intérprete, en contraste con los lenguajes compilados. Sin embargo, hay lenguajes que son diseñados para ser intrínsecamente interpretativos, por lo tanto un compilador causará una carencia de la eficacia.

URI: es una cadena de caracteres corta que identifica inequívocamente un recurso (servicio, página, documento, dirección de correo electrónico, enciclopedia). Normalmente estos recursos son accesibles en una red o sistema.

WSDL: lenguaje que describe la interfaz pública a los servicios Web. Está basado en XML y describe la forma de comunicación, es decir, los requisitos del protocolo y los formatos de los mensajes necesarios para interactuar con los servicios listados en su catálogo.

WADL: el lenguaje de descripción de aplicaciones web es una descripción XML entendible por aplicaciones web basadas en HTTP. Los modelos WADL son recursos proporcionados por los servicios web y las relaciones que se establecen entre ellos. WADL es el modo de simplificar la reutilización de servicios web que se basan en la arquitectura HTTP existente en la web.

API: Interfaz de programación de aplicaciones o API es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas.

Referencias bibliográficas

BARRAGÁN, R, 2005, **El portafolio, metodología de evaluación y aprendizaje de cara al nuevo Espacio Europeo, una experiencia práctica en la Universidad de Sevilla**. *Revista Latinoamericana de Tecnología Educativa*. 2005. Vol. 4, no. 1, p. 121 – 129.

BAUKES, Mike, 2013, ScriptRock. *MySQL vs Postgres* [online]. 9 August 2013. [Accessed 8 February 2014]. Available from: https://www.scriptrock.com/articles/postgres-vs-mysql/.

BECK, Kent and FOWLER, Martin, 2001, ***Planning Extreme Programming*** [online]. Addison-Wesley Professional. [Accessed 16 December 2014]. ISBN 9780201710915. Available from: http://books.google.es/books?hl=es&lr=&id=u13hVoYVZa8C&oi=fnd&pg=PR11&dq=Planning+Xtreme+Programming&ots=GK59ZQ9Tfh&sig=prR1mbpew9J5DoqFNyJaXo79PKQ#v=onepage&q=Planning%20Xtreme%20Programming&f=false

BENIGNI, Gladys, ANTONELLI, Octavio and VÁZQUEZ, Yonvanny, 2009, **Herramienta para reúso de código Javascript orientado a patrones de integración**. *Ciencias Básicas y Tecnología* [online]. 2009. Vol. 21, no. 1, p. 60 – 69. [Accessed 8 February 2014]. Available from: http:// www.ojs.udo.edu.ve/index.php/saber/article/download/104/40/.

BERNARDO, Juan, ANAYA DE PÁEZ, Raquel, MARÍN, Juan Carlos and BILBAO LÓPEZ, Alex, 2005, **Un estudio comparativo de herramientas para el modelado con UML**. *Revista Universidad EAFIT* [online]. March 2005. Vol. 41, no. 137, p. 60 – 76. [Accessed 8 February 2014]. Available from: http://publicaciones.eafit.edu.co/index.php/revista-universidad-eafit/article/view/838/737/.

CABRERA GONZÁLEZ, Lianet, GONZÁLEZ, Lianet Cabrera and TORRES, Enrique Roberto Pompa, 2012, **Extensión de Visual Paradigm for UML para el desarrollo dirigido por modelos de aplicaciones de gestión de información**. *Serie Científica* [online]. 15 October 2012. Vol. 5, no. 10. [Accessed 8 February 2014]. Available from: http://publicaciones.uci.cu/index.php/SC/article/view/1032/.

CALDERÍN, Dunet and PÉREZ, Leonardo, 2012, **Integración de los instrumentos de evaluación para la gestión de las evidencias en la plataforma educativa Zera**. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas. La Habana, Cuba : Universidad de las Ciencias Informáticas.

CALVO, Ginnette, 2014, **El Portafolio Digital como estrategia didáctica: la experiencia del curso Servicios de Información Automatizados**. *Revista Electrónica Semestral E – Ciencias de*

la Información [online]. 1 January 2014. Vol. 4, no. 1. [Accessed 18 December 2013]. Available from: http://www.revistas.ucr.ac.cr/index.php/eciencias/article/view/12865.

CANO, María E., 2011, **La evaluación por competencias en la educación superior**. *Revista de Curriculum y Formación del Profesorado* [online]. 15 April 2011. No. 1 - 16. [Accessed 3 February 2014]. Available from: http://www.ugr.es/local/recfpro/rev123COL1.pdf. COMUNIDADES VIRTUALES, 2013, **Módulos de Capacitación para el desarrollo web**. [online]. 2013. [Accessed 7 May 2014]. Available from: http://www.disenovisual.com/interficies/editores.htm

CRUZ CASTRO, Julio, 2011, **Sistema de Gestión de Calidad del Grupo de Calidad de FORTES. Módulo de Capacitación**. [online]. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas. La Habana, Cuba : Universidad de las Ciencias Informáticas. [Accessed 21 February 2014]. Available from: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_04629_11/1/TD_04629_11.pdf/.

DANNY, 2008, ***Techlosofy***. [online]. 2008. [Accessed 10 February 2014]. Available from: http://techlosofy.com/que-es-ajax.

DÍAZ BARRIGA, Frida, ROMERO, Eric and HEREDIA, Abraham, 2011, **El portafolio electrónico como instrumento para la reflexión sobre el desarrollo profesional y la formación en estudiantes de postgrado**. *Revista Observar*. 2011. No. 5, p. 20. Edu-portfolio.org : Su portafolio electrónico, 2013. [online], [Accessed 6 March 2014]. Available from: http://eduportfolio.org/.

EGUILUZ, Javier, 2012, **Desarrollo web ágil con Symfony2** [online]. [Accessed 21 March 2014]. Available from: http://symfony.es/libro.

EGUILUZ PÉREZ, Javier, 2008, **Introducción a Ajax** [online]. June 2008. [Accessed 28 May 2014]. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/introduccion_ajax_2caras.pdf

FABREGAT, Ramón, 2010, **Estándares para e-Learning adaptativo y accesible**. *RIED*. 2010. Vol. 13, no. 2, p. 45 – 71.
FORTES, 2011, **Estudio de la competencia en el mercado de herramientas *e – learning***. La Habana, Cuba : Centro de Tecnologías para la Formación (FORTES), Universidad de las Ciencias Informáticas.

GALLEGO, Domingo, 2011, **El e – portafolio como estrategia de enseñanza – aprendizaje**. *Revista Electrónica de Tecnología Educativa* [online]. 4 April 2011. No. 30. [Accessed 3 February 2014]. Available from: http://edutec.rediris.es/Revelec2/revelec30/edutec30_eportfolio_estrategia_ensenanza_aprendizaj

e.html

GALLEGO, Evelyn and GUERRERO, Eduar Smith, 2011, **Desarrollo del Portafolio Electrónico para la Plataforma Educativa Zera, e – Portafolio**. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas. La Habana, Cuba : Universidad de las Ciencias Informáticas.

GONZÁLES, Santiago, 2008, **Revisión de plataformas de Entorno de Aprendizaje**. Lima, Perú : Universidad Inca Garcilaso de la Vega, Facultad de Ingeniería de Sistemas, Cómputo y Telecomunicaciones.

GREGORI, Elena B., 2008, **El estilo e-portafolio.**

HERNÁNDEZ, Miguel, 2006, *Portafolio del estudiante*. 2006. Ficha Metodológica Universidad Valenciana.

HIDALGO URBINO, Rafael Jacobo, 2010, **Análisis, diseño e implementación de una herramienta que permita la centralización de la información gestionada por el módulo Resultados de la Colección Multisaber.** [online]. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas. La Habana, Cuba : Universidad de las Ciencias Informáticas. [Accessed 21 February 2014]. Available from: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_02857_10/1/TD_02857_10.pdf.

HIGHSMITH, James A., 2002, ***Agile software development ecosystems*** [online]. Addison-Wesley. [Accessed 16 December 2013]. ISBN 9780201760439. Available from: http://books.google.com.cu/books/about/Agile_software_development_ecosystems.html?id=N_yvRriESPcC&redir_esc=y

IEEE COMPUTER SOCIETY, 2000, ***IEEE Recommended Practice for Architectural Description of Software-Intensive Systems*** [online]. New York, NY 10016-5997, USA : The Institute of Electrical and Electronics Engineers, Inc. [Accessed 10 April 2014]. IEEE-SA Standards Board. Available from: http://cabibbo.dia.uniroma3.it/ids/altrui/ieee1471.pdf IEEE Std 1471-2000.

JOSKOWICS, José, 2008, **Reglas y Prácticas en eXtreme Programming**. España : Universidad de Vigo.

JQUERY COMMUNITY, 2013, ***jQuery 2.0 Released*** [online]. 18 April 2013. Available from: http://www.jquery.com.

KHRAMTCHENKO, Serguei, 2004, ***Comparing eXtreme Programming and Feature Driven Development in academic and regulated environments***. Final paper for CSCIE-275: Software Architecture and Engineering. Harvard University.

KOTOV, 2013, **GitLab, gestor de repositorios para Git. | RooTeando**. [online]. 3 February 2013.

[Accessed 10 April 2014]. Available from: http://rooteando.com/gitlab-administrador-web-de-repositorios-git.

LARMAN, Craig, 2004, **UML y patrones: introducción al análisis y diseño orientado a objetos** [online]. Félix Varela. [Accessed 9 April 2014]. Available from: http://books.google.com.cu/books?id=eNJjtwAACAAJ&dq=uml+y+patrones&hl=es&sa=X&ei=NNJVU47eDsTNsQTJ24D4BQ&redir_esc=y.

LETELIER, Patricio and PENADÉS, Mª Carmen, 2006, **Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP)**. [online]. 15 April 2006. Vol. 5, no. 26. [Accessed 16 December 2013]. Available from: http://www.cyta.com.ar.

LOS SANTOS, Alberto, 2009, **Revisión de los Servicios Web SOAP / REST: Características y rendimiento**. [online]. España : Universidad de Vigo. [Accessed 10 February 2014]. Available from: http://www.albertolsa.com/wp-content/uploads/2009/07/mdsw-revision-de-los-servicios-web-soap_rest-alberto-los-santos.pdf.

MAGAÑA ORÚE, Sara, 2009, **Estudio comparativo de Lenguajes de Modelado de procesos de negocio para su integración en procesos de desarrollo de software dirigido por modelos**. [online]. Proyecto de Fin de Carrera. Madrid, España : Universidad Carlos III de Madrid. [Accessed 8 February 2014]. Available from: http://e-archivo.uc3m.es/bitstream/handle/10016/9077/PFC_SARA_MAGANA_ORUE.pdf?sequence=1.

MARLON, 2009, **soapUI : Herramienta para pruebas de WebServices**. *Marlon J. Manrique* [online]. 12 May 2009. [Accessed 8 May 2014]. Available from: http://www.marlonj.com/blog/2009/05/soapui-herramienta-para-pruebas-de-webservices/

MARTÍNEZ, Vicen, 2013, **Desarrollo web y Marketing online. Introducción a Symfony2** [online]. 24 April 2013. [Accessed 8 February 2014]. Available from: http://vicenmartinez.wordpress.com/2013/04/24/introduccion-a-symfony2/.

MIR, Losep, 2012, **Estudio de los futuros estándares HTML5 y CSS3**. *Propuesta de actualización del sitio www.mpiua.net.* Lleida : Universidad de Lleida, Escuela Politécnica Superior, Ingeniería Técnica en Informática de Gestión.

MUÑOZ, Juan, SÁNCHEZ, Sonia, SAHAGÚN, Miguel and BRIA, Marc, 2008, **Moodle y los e-portfolio**. 23 October 2008. Universidad Autónoma de Barcelona.

NAVARRO MARSET, Rafael, 2007, *REST vs Web Services. Modelado, Diseño e Implementación de Servicios Web.* 2007.

PEREDA, Ernesto Vladimir, 2013, **Pautas de Codificación** [online]. 9 October 2013. Centro FORTES. [Accessed 10 February 2014]. Available from: http://gespro.fortes.prod.uci.cu/attachments/20023/Estandar_de_codificaci %C3%B3n_Xalix_.pdf Versión 1.0

PÉREZ, Irina Ivis Santiesteban, RAMÍREZ, Miguel Medina, GONZÁLEZ, Jorge Luis Piña and VARGAS, Yoennis Garrido, 2011, **Desarrollo de funcionalidades que faciliten al docente su preparación y el control del aprendizaje de los estudiantes en la plataforma educativa Zera. *Serie Científica*** [online]. 14 November 2011. Vol. 4, no. 11. [Accessed 10 April 2014]. Available from: http://publicaciones.uci.cu/index.php/SC/article/view/765.

PostgreSQL: Dossier de Prensa de PostgreSQL 9.3, 2014. [online], [Accessed 4 March 2014]. Available from: http://www.postgresql.org/about/press/presskit93/es/

POTENCIER, Fabien, 2011, ***What is Symfony2?*** [online]. 25 November 2011. [Accessed 8 February 2014]. Available from: http://www.fabien.potencier.org/article/49/what-is-symfony2.

PRENDES, Mª Paz, 2009, **Plataformas de Campus Virtual con Herramientas de Software Libre: Análisis comparativo de la situación actual en las universidades españolas**. [online]. Informe del Proyecto de la Secretaría de Estado de Universidades e Investigación. España : Secretaría de Estado de Universidades e Investigación. [Accessed 29 January 2014]. Available from: http://www.um.es/campusvirtuales/informe.html.

PRESSMAN, Roger S., 2005, **Ingeniería del Software. Un enfoque práctico**. [online]. Sexta Edición. McGraw-Hill Interamericana. [Accessed 3 April 2014]. ISBN 970-10-5473-3. Available from: http://eva.uci.cu/mod/resource/view.php?id=9302&subdir=/Pressman_6ta_Edicion

RAMÍREZ, Joaquín and MENDOZA, Marco A., 2005, **e – Portafolio un espacio de desarrollo en la web**. In : *Encuentro Internacional de Educación Superior: UNAM 2005* [online]. Palacio de Minería: Ciudad de México. 20 June 2005. [Accessed 3 February 2014]. Available from: http://www.reposital.evaed.unam.mx:8080/jspui/bitstream/123456789/1280/1/2005-04-01489e_portafolio.pdf.

REMÓN, Linet K., 2013, **Tecnología Portafolios Electrónicos**. Resumen Ejecutivo del Grupo de Inteligencia Empresarial. La Habana, Cuba : Centro de Tecnologías para la Formación (FORTES), Universidad de las Ciencias Informáticas.

REYES, Yunior Mesa, 2012, **Espacio de comunicación e intercambio para la Comunidad Técnica Cubana de PostgreSQL**. *Serie Científica* [online]. 11 January 2012. Vol. 5, no. 1. Available from: http://publicaciones.uci.cu/index.php/SC/article/view/586

REYNOSO, Carlos and KICILLOF, Nicolás, 2004, **Versión 1.0: Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.** [online]. Argentina : Universidad de Buenos Aires. [Accessed 4 April 2014]. Available from: http://carlosreynoso.com.ar/archivos/arquitectura/Estilos.PDF.

ROMERO, Iván and TORRES, Osmany, 2013, **Aspectos para la creación de módulos de gestión con Symfony 1.4.10 y Doctrine 1.2.** *Serie Científica de la Universidad de las Ciencias Informáticas* [online]. 13 April 2013. Vol. 5, no. 4. [Accessed 8 February 2014]. Available from: https://publicaciones.uci.cu/index.php/SC/article/view/856/527

ROMERO, Luisa Mª and TROYANO, José A., 2014, ***Análisis Comparativo entre las Plataformas de más Frecuente Implantación en los Sistemas Virtuales de Formación frente a un Modelo: Proyecto Sakai***. [online]. 3 February 2014. [Accessed 3 February 2014]. Available from: http://www.eatis.org/eatis2010/portal/paper/memoria/html/files/83.pdf.

ROSALES MORALES, Yanet, MORALES, Yanet Rosales, CLARK, Michael Eduardo Marrero and OLIVA, Adrián Trujillo, 2013, **Extensión de la herramienta Visual Paradigm para la generación de clases de acceso a datos con Doctrine 2.0**. *Serie Científica* [online]. 4 November 2013. Vol. 6, no. 10. [Accessed 8 February 2014]. Available from: http://publicaciones.uci.cu/index.php/SC/article/view/1271

S.L, elearning Solutions, 2011, ***Informe sobre Sakai 2.8, Moodle 2.0 y Bb Learn 9.1***.

SÁNCHEZ MENÉNDEZ, Milena, MENÉNDEZ, Milena Sánchez and MORALES, Anniel Arencibia, 2012, Módulo de tarjeta control para el sistema synta. *Serie Científica* [online]. 15 July 2012. Vol. 5, no. 7. [Accessed 10 February 2014]. Available from: http://publicaciones.uci.cu/index.php/SC/article/view/970.

SALVADOR BROCHE, Orlando Felipe, 2013, **Indicaciones para el trabajo en el marco de trabajo Xalix.** La Habana, Cuba : Universidad de las Ciencias Informáticas.

VARGAS, Edwin and CARRILLO, Alberto E., 2008, **Portafolio Virtual de Evidencias de Aprendizaje como Instrumento de Evaluación soportado en el Sistema de Gestión de Aprendizaje MOODLE.** Proyecto de Grado para optar al título de Ingeniería de Sistemas. Bucaramanga, España : Universidad Industrial de Santander, Facultad de Ingeniería Físico – Mecánicas.

VERMILION, 2013, ***Responsive CSS Framework Comparison. Bootstrap vs. Foundation vs. Skeleton*** [online]. 21 November 2013. [Accessed 10 February 2014]. Available from: http://responsive.vermilion.com/compare.php.

VILCHES PUPO, Jesús, 2012, **Servidor de integración continua y pruebas para código**

fuente. [online]. 2012. [Accessed 28 March 2014]. Available from: http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_05692_12/.

WRAY, S., 2008, ***Swimming Upstream: Shifting the Purpose of an Existing Teaching Portafolio Requirement***. *The professional educator*. 2008. Vol. 32, no. 1, p. 44 – 59.

Bibliografía consultada

Cabero Almenara, Julio, López Meneses, Eloy y Llorente Cejudo, Mª del Carmen. **E-Portafolio universitario como instrumento didáctico 2.0 para la reflexión, evaluación e investigación de la práctica educativa en el espacio europeo de educación superior.** 2012.

Ramírez Buentello, Joaquín y Mendoza Calderón, Marco A. **ePortafolio un espacio de desarrollo en la web**. 2005

Muñoz, Juan, Sánchez Sonia, Sahagún, Miguel y Bria Marc. **Moodle y los e – Portafolio.** Octubre de 2008.

Graf, Sabine y List, Beate. ***An Evaluation of Open Source E-Learning Platforms Stressing Adaptation Issues.*** 2009.

Wilson, Scott y Rees Jones, Peter. ***What is … IMS Learner Information Package?*** 10 de octubre de 2002.

Mesa Reyes, Yunior. **Espacio de comunicación e intercambio para la Comunidad Técnica de PostgreSQL.** Serie Científica, Universidad de las Ciencias Informáticas. 11 de enero de 2012. http://publicaciones.uci.cu/index.php/SC/article/view/586.

Casanova, Mutis y Osorio Rodríguez, Alain. **Herramienta generadora de ficheros de mapeo para la persistencia de esquemas de objetos relacionales.** Serie Científica, Universidad de las Ciencias Informáticas. 13 de junio de 2013. http://publicaciones.uci.cu/index.php/SC/article/view/1156.

Ramos, Juan Carlos y Depetris, Natalia. **Diseño de software basado en arquitecturas.** 2012.

García González, Roberto. **Patrones arquitectónicos de Aplicaciones Empresariales.** 2007. http://ocw.udl.cat/enginyeria-i-arquitectura/enginyeria-del-software-iii/Continguts/2%20-%20Patrones/Patrones%20Arquitectonicos%20de%20Aplicaciones%20Empresariales.pdf.

Edu-portfolio.org : Su portafolio electrónico. 2013. http://eduportfolio.org/.

Sánchez Santamaría, José. **El e – Portafolio en la docencia universitaria: Percepciones de los estudiantes y carga del trabajo.** *Revista Electrónica de Investigación y Docencia (REID), 7*, Enero, 2012, 31-55. ISSN: 1989-2446. http://www.revistareid.net/revista/n7/REID7art2.pdf.

Baldiris, Silvia, Hernández, Jorge y Fabregat, Ramón. **E – Portafolio soportado en estándares.** 7th Latin American and Caribbean Conference for Engineering and Technology. San Cristóbal, Venezuela. June 2-5, 2009.

Tuksinvarajarn, Ananya y Watson Todd, Richard. ***The E-pet: Enhancing Motivation in E-portfolios***. English Teaching Forum. 2009.

Sanchis Albelda, Rafa. **Análisis comparativo de LMS**. Proyecto Final de Carrera Ingenería Informática. 4 de Septiembre de 2013.

Rosales Morales, Yanet, Marrero Clark, Michael Eduardo y Trujillo Oliva, Adrián. **Extensión de la herramienta Visual Paradigm para la generación de clases de acceso a datos con Doctrine 2.0**. SC-UCI. 2013-11-04. http://publicaciones.uci.cu/index.php/SC/article/view/1271.

Martínez, Vicen. **Desarrollo web y Marketing online. Introducción a Symfony2**. 24 de abril de 2013. http://vicenmartinez.wordpress.com/2013/04/24/introduccion-a-symfony2/.

Bernardo Quintero, Juan, Anaya de Páez, Raquel, Marín, Juan Carlos y Bilbao López, Alex**. Un estudio comparativo de herramientas para el modelado con UML**. REVISTA Universidad EAFIT**.** Vol. 41. No. 137. 2005. pp. 60-76.

Medina Pasaje, Julio Luis**. Metodología y Herramientas UML para el Modelado y Análisis de Sistemas de Tiempo Real Orientados a Objetos.** Tesis Doctoral**.** Universidad de Cantabria Departamento de Electrónica y Computadores. Santander, junio de 2005.

Diego Gauchat, Juan**. El Gran libro de html5, css3 y Javascript.** Ediciones técnicas**.**

Mir Huguet, Josep. **Estudio de los futuros estándares html5 y css3. Propuesta de actualización del sitio www.mpiua.net.** Universidad de Lleida: Escuela Politécnica Superior Ingeniería Técnica en Informática de Gestión. Septiembre de 2012.

Benigni, Gladys, Antonelli, Octavio y Vázquez, Yonvanny. Herramienta para reúso de código Javascript orientado a patrones de integración. **Ciencias Básicas y Tecnología. Vol. 21, No. 1, pp. 60 – 69. 2009.** http:// www.ojs.udo.edu.ve/index.php/saber/article/download/104/40.

Sánchez Menéndez, Milena y Arencibia Morales, Anniel. **Módulo de tarjeta control para el sistema synta.** Vol. 5, No. 7. SC-UCI. 2012-07-15. http://publicaciones.uci.cu/index.php/SC/article/view/970.

Pérez A, Oiver Andrés. **Cuatro enfoques metodológicos para el desarrollo de software: RUP –**

MSF – XP – SCRUM. 10 de junio de 2012. http://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=9&sqi=2&ved=0CFkQFjAI&url=http%3A%2F%2Fwww.up4ved.org%2Fplantillas%2520caso%2520recorrido%2520virtual%2FRequisitos%2520Funcionales%2520y%2520No%2520Funcionales.doc&ei=tAgnU_y5NZK3kAeLxoCQBg&usg=AFQjCNGrwExF_tL9pu4sMvMfcsZaRvuFGw&bvm=bv.62922401,d.eW0&cad=rja.

Fernández Díaz, Milenis, Comas Pérez, Yassel y Espinosa Ramírez, José Gabriel**. Herramientas ORM para el desarrollo de aplicaciones en C++.** No. 12, Vol. 5, Año: 2012. SC-UCI. 15/12/2012. http://publicaciones.uci.cu/index.php/SC/article/download/123/65. ISSN: 2306-2495.

Prendes Espinosa, Mª Paz y Sánchez Vera, Mª del Mar. **Portafolio Electrónico: una posibilidad para los docentes.** Pixel-Bit. Revista de Medios y Educación. Nº 32 Marzo 2008 pp. 21- 34. Universidad de Murcia (España).

Hernández Silva, José y Lorandi Medina, Alberto Pedro. **El uso del portafolio de evidencias de aprendizaje como herramienta para la evaluación por competencias en una asignatura**. 2° Congreso Virtual sobre Tecnología, Educación y Sociedad. CTES. México. 22 al 25 de Enero 2013. Centro de Estudios e Investigaciones para el Desarrollo Docente. ISBN: 978-607-8254-60-6.

González Campos, Saúl y Fernández Martínez, Luis Felipe. **Programación Extrema: Prácticas, Aceptación y Controversia**. Mayo – Agosto, 2006. Año 3, No 14-15. Universidad Autónoma de Cd. Juárez.

Berrueta, Diego. **Programación extrema y software libre**. 22 de enero de 2006.

Del Valle Rojo, Silvana y Oliveros, Alejandro. **Requerimientos No funcionales para aplicaciones Web.** 13th Argentine Symposium on Software Engineering, ASSE. 2012.

Cataldi, Zulma y Lage, Fernando**. E-Portafolio. Una opción metodológica más auténtica para evaluación de aprendizajes autónomos en educación superior.** Facultad Regional Buenos Aires. Universidad Tecnológica Nacional Ciudad de Buenos Aires. Argentina.

The Leap2A specification for e-portfolio portability and interoperability. http://www.leapspecs.org/2A/core-specification.

Durand, William. ***REST APIs with Symfony2: The Right Way***. http://williamdurand.fr/2012/08/02/rest-apis-with-symfony2-the-right-way/.

Auto generate REST API docs from Symfony. http://stackoverflow.com/questions/8872276/auto-

generate-rest-api-docs-from-symfony.

Hamon, Hugo. ***Identifying Design Patterns in the Symfony Framework.*** Estambul, Turquía. 02 de mayo de 2014. ftp://ftp.prod.uci.cu/PHP/Documentacion/Symfony2/Design-Patterns-Symfony.pdf.

Novoa Proenza, Manuel A. y Álvarez Gómez, Yasmani Joaquin. **Capa de servicios web para el motor de búsqueda Orión.** Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas. Universidad de las Ciencias Informáticas. La Habana, junio de 2012.

Solano Farell, Yannier. **Solución para la integración de los principales servicios telemáticos de la Universidad de las Ciencias Informáticas con la Red Social Universitaria.** Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas. Universidad de las Ciencias Informáticas. La Habana, junio de 2012.

Navarro Marset, Rafael. ***REST vs Web Services.*** Modelado, Diseño e Implementación de Servicios Web. 2006 – 07.

Marqués, Asier. **Desarrollando un framework REST sobre los componentes de Symfony2.** Universidad de Castellón. España, 15 – 16 de junio de 2012.

Webber, Jim, Parastatidis, Savas y Robinson, Ian. ***REST in Practice. Hypermedia and Systems Architecture.*** Septiembre de 2010. ISBN: 978 – 0 – 596 – 80582 – 1.

De Donato, Giulio. ***Symfony2 REST API: the best way.*** *Welcome to the bundle.* http://www.welcometothebundle.com.

Durand, William. ***REST APIs with Symfony2: The Rigth Way.*** Zurich, Suiza. Agosto de 2012. http://www.williamdurand.fr/2012/08/02/rest-apis-with-symfony2-the-right-way/.

Anexos

Anexo 1: Modelo de datos del Leap2A

Tabla 6. Modelo de datos del Leap2A.

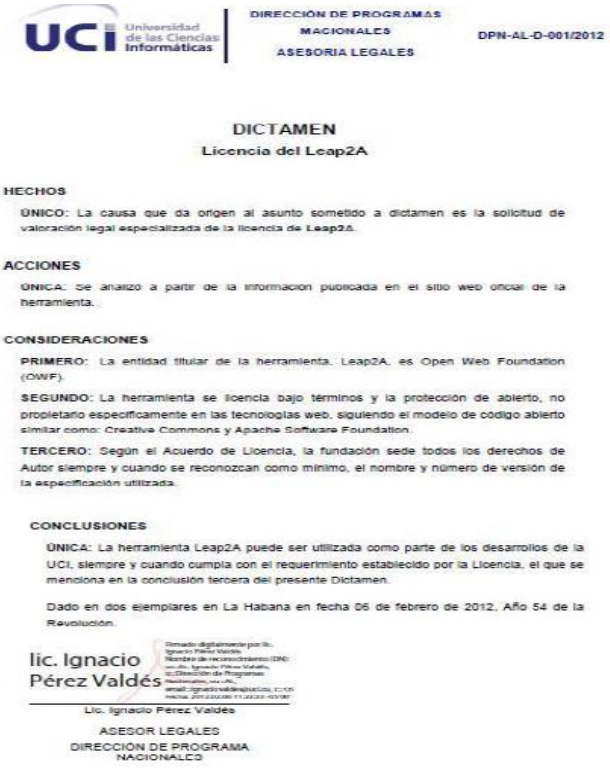
	Clases incluyendo los tipos de elementos	Propiedades		
		Relaciones de los elementos con otros elementos	Propiedades literales de Atom, Leap2A, etc	Propiedades y atributos mezclados
Tomado de Atom	**entry, feed, person construct**	**alternate, enclosure, in-reply-to, license, replies, related, self**	**content, id, name, published, rights, summary, title, updated**	**author, contributor, category, label, scheme, term**
Tomado de otros espacios			**Issued**	**contributor, creator, type**
Nativo del Leap2A	**ability, achievement, activity, affiliation, meeting, organization, person, plan, publication, resource, selection**	**attended by, attends, has_agenda, has_evidence, has_outcome, has_part, has reply, in_reply_to, is_agenda_of, is_evidence_of, is_outcome_of, is_part_of, reflected_on_by, reflects_on, relation, supported_by, supports**	**activetime, addressline, country, date, myrole, postcode, roleid, spatial, status, version**	**countrycode, display_order, field, label, point, service, stage, when_added**

Tabla 7. Atributos que almacenan la información personal y organizacional en el Leap2A.

Información Personal	Información de Organizaciones
full name, **legal family name,** **legal given name,**	**legal org name,** **preferred org name,** **country,**

preferred family name, **preferred given name,** **family name first,** **name prefix,** **name suffix,** **country,** **dob,** **gender,** **website,** **id,** **openid,** **email,** **homephone,** **workphone,** **mobile,** **minicome,** **fax,** **other**	**website,** **id,** **email,** **workphone,** **minicom,** **fax,** **other**

Anexo 2: Dictamen emitido por la Universidad de las Ciencias Informáticas sobre la licencia del estándar IMS Leap2A



UCI Universidad de las Ciencias Informáticas | DIRECCIÓN DE PROGRAMAS MACIONALES ASESORIA LEGALES DPN-AL-D-001/2012

DICTAMEN
Licencia del Leap2A

HECHOS

ÚNICO: La causa que da origen al asunto sometido a dictamen es la solicitud de valoración legal especializada de la licencia de Leap2A.

ACCIONES

ÚNICA: Se analizo a partir de la información publicada en el sitio web oficial de la herramienta.

CONSIDERACIONES

PRIMERO: La entidad titular de la herramienta, Leap2A, es Open Web Foundation (OWF).

SEGUNDO: La herramienta se licencia bajo términos y la protección de abierto, no propietario especificamente en las tecnologías web, siguiendo el modelo de código abierto similar como: Creative Commons y Apache Software Foundation.

TERCERO: Según el Acuerdo de Licencia, la fundación sede todos los derechos de Autor siempre y cuando se reconozcan como mínimo, el nombre y número de versión de la especificación utilizada.

CONCLUSIONES

ÚNICA: La herramienta Leap2A puede ser utilizada como parte de los desarrollos de la UCI, siempre y cuando cumpla con el requerimiento establecido por la Licencia, el que se menciona en la conclusión tercera del presente Dictamen.

Dado en dos ejemplares en La Habana en fecha 06 de febrero de 2012, Año 54 de la Revolución.

lic. Ignacio Pérez Valdés

Lic. Ignacio Pérez Valdés
ASESOR LEGALES
DIRECCIÓN DE PROGRAMA NACIONALES

Ilustración 20. Licencia emitida por la UCI sobre el Leap2A.

Anexo 3: Agilidad de cada metodología

Tabla 8. Ranking de "agilidad" para cada metodología.

Indicadores	SCRUM	Crystal	ASD	FDD	XP
Sistema como algo cambiante	5	4	5	3	5
Colaboración	5	5	5	4	5
Características de la metodología (CM)					
Resultados	5	5	5	4	5
Simplicidad	5	4	4	5	5
Adaptabilidad	4	5	5	3	3
Excelencia técnica	3	3	3	4	4
Prácticas de colaboración	4	5	5	3	5
Media CM	**4.2**	**4.4**	**4.4**	**3.8**	**4.4**

Media Total	**4.7**	**4.5**	**4.8**	**3.6**	**4.8**

Anexo 4: Estimación de esfuerzos por HU

Tabla 9. Estimación de esfuerzos por historias de usuario.

Historia de usuario	Estimación (días)
HU1. Gestionar usuarios del sistema	5
HU2. Autenticar usuario en el sistema	2
HU3. Gestionar archivos personales	3
HU4. Gestionar evidencias personales	5
HU5. Compartir evidencia personal	3
HU6. Consultar evidencias compartidas	4
HU7. Gestionar secciones	5
HU8. Consultar notificaciones personales	2
HU9. Exportar datos del portafolio	6
HU10. Gestionar comentarios	2
HU11. Brindar servicios web	13
HU12. Mensajes del sistema	4
HU13. Sistema de búsqueda del portafolio	3
HU14. Internacionalización del idioma	1
HU15. Administrar usuarios del sistema	2

Anexo 5: Plan de duración de las iteraciones

Tabla 10. Plan de duración de las iteraciones.

Iteraciones	Orden de las HU a implementar	Prioridad	Días	Total de días de la iteración
Iteración 1	HU1. Gestionar usuarios del sistema	Alta	5	15
	HU2. Autenticar usuario en el sistema	Media	2	
	HU3. Gestionar archivos personales	Media	3	
	HU7. Gestionar secciones	Alta	5	
Iteración 2	HU4. Gestionar evidencias personales	Alta	5	15
	HU5. Compartir evidencia personal	Alta	3	
	HU6. Consultar evidencias compartidas	Alta	4	
	HU13. Sistema de búsqueda del portafolio	Media	3	
Iteración 3	HU11. Brindar servicios web	Alta	13	15
	HU15. Administración de usuarios del sistema	Media	2	
Iteración 4	HU8. Consultar notificaciones personales	Media	2	15
	HU9. Exportar datos del portafolio	Alta	6	
	HU10. Gestionar comentarios	Baja	2	
	HU12. Mensajes del sistema	Media	4	

	HU14. Internacionalización del idioma	Baja	1	

Anexo 6: Plan de entregas

Tabla 11. Plan de duración de las entregas.

Historia de usuario	Fin 1ra iteración 3ra semana de febrero	Fin 2da iteración 2da semana de marzo	Fin 3ra iteración 1ra semana de abril	Fin 4ta iteración 4ta semana de abril
HU1. Gestionar usuarios del sistema	*finalizado*	*finalizado*	*finalizado*	*finalizado*
HU2. Autenticar usuario en el sistema	*finalizado*	*finalizado*	*finalizado*	*finalizado*
HU3. Gestionar archivos personales	*V0.5*	*finalizado*	*finalizado*	*finalizado*
HU7. Gestionar secciones	*V0.3*	*finalizado*	*finalizado*	*finalizado*
HU4. Gestionar evidencias personales	-	*finalizado*	*finalizado*	*finalizado*
HU5. Compartir evidencia personal	-	*V0.7*	*finalizado*	*finalizado*
HU6. Consultar evidencias compartidas	-	*V0.8*	*finalizado*	*finalizado*
HU13. Sistema de búsqueda del portafolio	-	*V0.8*	*finalizado*	*finalizado*
HU11. Brindar servicios web	-	*V0.3*	*V0.5*	*finalizado*
HU15. Administración de usuarios del sistema	-	-	*V0.5*	*finalizado*
HU8. Consultar notificaciones personales	-	-	*V0.5*	*finalizado*
HU9. Exportar datos del portafolio	-	*V0.5*	*V0.8*	*finalizado*
HU10. Gestionar comentarios	-	*V0.4*	*V0.7*	*finalizado*
HU11. Sistema de búsqueda del portafolio				
HU12. Mensajes del sistema	*V0.3*	*V0.5*	*V0.8*	*finalizado*
HU14. Internacionalización del idioma	-	-	*V0.5*	*finalizado*

Anexo 7: Especificación de las tarjetas CRC

Módulo Usuario

Tarjeta CRC	
Número: 1	**Nombre de la clase:** DefaultController
Responsabilidades: - Registrar usuario en el sistema	

- Autenticar usuario en el sistema - Editar perfil de usuario - Visualizar perfil de usuario - Salir del sistema
Colaboraciones: Usuario, UsuarioPerfilType, UsuarioRegistroType, Portafolio
Tarjeta CRC
Número: 2 **Nombre de la clase:** Usuario
Responsabilidades: representa la entidad Usuario que define las características de los objetos de este tipo y será mapeada a tabla relacional en la base de datos de la aplicación.
Colaboraciones:
Tarjeta CRC
Número: 3 **Nombre de la clase:** UsuarioPerfilType
Responsabilidades: esta clase permite construir, en la vista, el formulario del perfil de usuario en las acciones de visualizar y editar perfil de usuario.
Colaboraciones:
Tarjeta CRC
Número: 4 **Nombre de la clase:** UsuarioRegistroType
Responsabilidades: esta clase permite construir, en la vista, el formulario de registro de usuario en la acción de registrar usuario.
Colaboraciones:

Módulo Portafolio

Tarjeta CRC
Número: 5 **Nombre de la clase:** ArchivoController
Responsabilidades: - Crear un nuevo archivo personal - Visualizar archivos personales - Editar archivo personal - Eliminar archivo personal
Colaboraciones: Archivo, ArchivoRepository, ArchivoType, Tag, TagRepository
Tarjeta CRC
Número: 6 **Nombre de la clase:** Archivo
Responsabilidades: representa la entidad Archivo que define las características de los objetos de este tipo y será mapeada a tabla relacional en la base de datos de la aplicación.
Colaboraciones:
Tarjeta CRC
Número: 7 **Nombre de la clase:** ArchivoRepository
Responsabilidades: esta clase permite consultar e insertar datos en la tabla relacional de la base de datos de la aplicación, perteneciente a la entidad Archivo.
Colaboraciones: Archivo
Tarjeta CRC
Número: 8 **Nombre de la clase:** ArchivoType
Responsabilidades: esta clase permite construir, en la vista, el formulario de gestión de archivos en las acciones de crear, editar, eliminar y visualizar los archivos personales.
Colaboraciones: TagType
Tarjeta CRC
Número: 9 **Nombre de la clase:** TagRepository
Responsabilidades: esta clase permite consultar e insertar datos en la tabla relacional de la base de datos de la aplicación, perteneciente a la entidad Tag.
Colaboraciones: Tag
Tarjeta CRC
Número: 10 **Nombre de la clase:** Tag
Responsabilidades: representa la entidad Tag que define las características de los objetos de

este tipo y será mapeada a tabla relacional en la base de datos de la aplicación.	
Colaboraciones:	
Tarjeta CRC	
Número: 11	**Nombre de la clase:** TagType
Responsabilidades: esta clase permite construir, en la vista, el formulario de tratamiento de etiquetas en las acciones de crear y eliminar etiquetas.	
Colaboraciones:	
Tarjeta CRC	
Número: 12	**Nombre de la clase:** SeccionController
Responsabilidades: - Crear nueva sección - Editar datos de la sección - Visualizar detalles de la sección - Eliminar sección	
Colaboraciones: Seccion, SeccionType, SeccionRepository	
Tarjeta CRC	
Número: 13	**Nombre de la clase:** SeccionRepository
Responsabilidades: esta clase permite consultar e insertar datos en la tabla relacional de la base de datos de la aplicación, perteneciente a la entidad Seccion.	
Colaboraciones: Seccion	
Tarjeta CRC	
Número: 14	**Nombre de la clase:** Seccion
Responsabilidades: representa la entidad Seccion que define las características de los objetos de este tipo y será mapeada a tabla relacional en la base de datos de la aplicación.	
Colaboraciones: Seccion	
Tarjeta CRC	
Número: 15	**Nombre de la clase:** SeccionType
Responsabilidades: esta clase permite construir, en la vista, el formulario de gestión de archivos personales en las acciones de crear, editar, eliminar y visualizar archivos personales.	
Colaboraciones: TagType	
Tarjeta CRC	
Número: 16	**Nombre de la clase:** EvidenciaController
Responsabilidades: crud y compartir - Crear evidencia personal - Visualizar detalles de la evidencia - Editar evidencia personal - Visualizar evidencias personales - Eliminar evidencia personal - Compartir evidencia personal	
Colaboraciones: Evidencia, EvidenciaType, EvidenciaRepository, Comentario, ComentarioType. Mensaje, MensajeRepository	
Tarjeta CRC	
Número: 17	**Nombre de la clase:** Evidencia
Responsabilidades: representa la entidad Evidencia que define las características de los objetos de este tipo y será mapeada a tabla relacional en la base de datos de la aplicación.	
Colaboraciones:	
Tarjeta CRC	
Número: 18	**Nombre de la clase:** EvidenciaRepository
Responsabilidades: esta clase permite consultar e insertar datos en la tabla relacional de la base de datos de la aplicación, perteneciente a la entidad Evidencia.	
Colaboraciones: Evidencia	
Tarjeta CRC	
Número: 19	**Nombre de la clase:** EvidenciaType
Responsabilidades: esta clase permite construir, en la vista, el formulario de gestión de evidencias personales en las acciones de crear, editar, eliminar, visualizar y compartir	

evidencias personales.	
Colaboraciones: TagType	
Tarjeta CRC	
Número: 20	**Nombre de la clase:** ComentarioController
Responsabilidades: - Crear comentario	
Colaboraciones: ComentarioType, ComentarioRepository, Comentario, Evidencia, EvidenciaRepository, Mensaje, MensajeRepository	
Tarjeta CRC	
Número: 21	**Nombre de la clase:** Comentario
Responsabilidades: representa la entidad Comentario que define las características de los objetos de este tipo y será mapeada a tabla relacional en la base de datos de la aplicación.	
Colaboraciones:	
Tarjeta CRC	
Número: 22	**Nombre de la clase:** ComentarioType
Responsabilidades: esta clase permite construir, en la vista, la sección de crear un nuevo comentario.	
Colaboraciones:	
Tarjeta CRC	
Número: 23	**Nombre de la clase:** ComentarioRepository
Responsabilidades: esta clase permite consultar e insertar datos en la tabla relacional de la base de datos de la aplicación, perteneciente a la entidad Comentario.	
Colaboraciones: Comentario	
Tarjeta CRC	
Número: 24	**Nombre de la clase:** CompartidaController
Responsabilidades: - Visualizar evidencia compartida por el usuario - Visualizar evidencias compartidas al usuario - Valorar evidencia - Visualizar detalles de la evidencia compartida - Emitir comentario	
Colaboraciones: Comentario, ComentarioType, Evidencia, EvidenciaRepository, Mensaje, MensajeRepository	
Tarjeta CRC	
Número: 25	**Nombre de la clase:** DefaultController
Responsabilidades: - Visualizar portada principal de la aplicación - Visualizar página de contacto - Visualizar archivo personal - Búsqueda por etiquetas	
Colaboraciones: Contacto, ContactType, Archivo, ArchivoRepository, Evidencia, EvidenciaRepository	
Tarjeta CRC	
Número: 26	**Nombre de la clase:** ContactType
Responsabilidades: esta clase permite construir, en la vista, la sección de contacto de la aplicación.	
Colaboraciones:	
Tarjeta CRC	
Número: 27	**Nombre de la clase:** Contacto
Responsabilidades: clase para definir un objeto de tipo Contacto para el envío de correos electrónicos e información de contacto.	
Colaboraciones:	
Tarjeta CRC	
Número: 28	**Nombre de la clase:** PortafolioRepository
Responsabilidades: esta clase permite consultar e insertar datos en la tabla relacional de la	

base de datos de la aplicación, perteneciente a la entidad Portafolio.	
Colaboraciones: Portafolio	
Tarjeta CRC	
Número: 29	**Nombre de la clase:** Portafolio
Responsabilidades: representa la entidad Portafolio que define las características de los objetos de este tipo y será mapeada a tabla relacional en la base de datos de la aplicación.	
Colaboraciones:	

Módulo Mensaje

Tarjeta CRC	
Número: 30	**Nombre de la clase:** DefaultController
Responsabilidades: - Visualizar notificaciones recientes. - Visualizar detalles de la notificación. - Visualizar todas mis notificaciones. - Eliminar notificación.	
Colaboraciones: Mensaje	
Tarjeta CRC	
Número: 31	**Nombre de la clase:** Mensaje
Responsabilidades: representa la entidad Mensaje que define las características de los objetos de este tipo y será mapeada a tabla relacional en la base de datos de la aplicación.	
Colaboraciones:	
Tarjeta CRC	
Número: 32	**Nombre de la clase:** MensajeRepository
Responsabilidades: esta clase permite consultar e insertar datos en la tabla relacional de la base de datos de la aplicación, perteneciente a la entidad Mensaje.	
Colaboraciones: Mensaje	

Módulo Servicios

Tarjeta CRC	
Número: 33	**Nombre de la clase:** ApiArchivoController
Responsabilidades: esta clase permite la visualización, creación, edición y eliminación de archivos a través de peticiones web, haciendo uso de los servicios web.	
Colaboraciones: Archivo, ArchivoRepository, ArchivoType, Usuario, UsuarioRepository	
Tarjeta CRC	
Número: 34	**Nombre de la clase:** ApiUsuarioController
Responsabilidades: esta clase permite el registro y edición del perfil de usuario, además de verificar el acceso de cada usuario en el sistema, mediante peticiones web, haciendo uso de los servicios web.	
Colaboraciones: Sesion, SesionRepository, UsuarioApiType, UsuarioPerfilType, Usuario, UsuarioRepository. Portafolio, PortafolioRepository	
Tarjeta CRC	
Número: 35	**Nombre de la clase:** ApiEvidenciaController
Responsabilidades: esta clase permite la visualización, creación, edición y eliminación de evidencia a través de peticiones web, haciendo uso de los servicios web.	
Colaboraciones: Evidencia, EvidenciaRepository, EvidenciaType, Comentario, ComentarioRepository, ComentarioType, Usuario, UsuarioRepository, Sesion, SesionRepository, Usuario, UsuarioRepository	
Tarjeta CRC	
Número: 36	**Nombre de la clase:** UsuarioApiType
Responsabilidades: esta clase permite construir un formulario con los datos del usuario, obviando algunos parámetros que no son necesarios enviar a través de las peticiones.	
Colaboraciones:	

Anexo 8: Descripciones de las tablas de la base de datos

Nombre de la tabla: xl_usuario		
Descripción: almacena los datos de los usuarios en el sistema		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador del usuario.
nombre	string (255)	Nombre del usuario.
apellidos	string (255)	Apellidos de los usuarios.
email	string (255)	Correo electrónico de los usuarios.
salt	string (255)	Valor aleatorio utilizado para codificar la contraseña del usuario.
password	string (255)	Contraseña del usuario.
visitas	integer	Cantidad de visitas que se realiza al portafolio del usuario.
foto_path	string (255)	Foto de perfil del usuario.
created	datetime	Momento de registro del usuario.
updated	datetime	Momento en que se actualizan los datos del usuario.
estado	integer	
lastlogin	datetime	Momento en el que estuvo autenticado el usuario.
lastlogout	datetime	Momento en que sale del sistema el usuario.
permite_email	boolean	Especifica si el usuario desea obtener notificaciones por correo.
language	string	Lenguaje seleccionado por el usuario.
theme	string	Tema seleccionado por el usuario.
Nombre de la tabla: xl_tag		
Descripción: etiquetas para facilitar la búsqueda de recursos.		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador de la etiqueta.
nombre	string (255)	Nombre de la etiqueta.
Nombre de la tabla: xl_seccion		
Descripción: almacena la información de las secciones del portafolio.		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador de la sección.
portafolio_id	integer	Identificador del portafolio al cual pertenece la sección.
titulo	string (20)	Título de la sección.

descripcion	text (500)	Descripción de la sección.
created	datetime	Momento que se crea la sección.
updated	datetime	Momento en que se actualizan los datos de la sección.
Nombre de la tabla: xl_portafolio		
Descripción: almacena la información referente a los portafolios personales.		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador del portafolio personal.
usuario_id	integer	Identificador del usuario creador del portafolio.
created	datetime	Momento en que se crea el portafolio.
Nombre de la tabla: xl_mensaje		
Descripción: almacena la información referente a los mensajes de los usuarios.		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador del mensaje.
created	datetime	Momento en que se crea el mensaje.
updated	datetime	Momento en que se actualizan los parámetros del mensaje.
remitente	string	Remitente del mensaje.
destinatario	integer	Destinatario del mensaje.
contenido	text	Contenido en el cuerpo del mensaje.
asunto	string	Asunto del mensaje.
leido	boolean	Especifica si el mensaje ha sido leído o no.
Nombre de la tabla: xl_evidencia_compartida		
Descripción: almacena la información de cuando una evidencia es compartida.		
Atributo	**Tipo**	**Descripción**
evidencia_id	integer	Identificador de la evidencia que se comparte.
portafolio_id	integer	Identificador del portafolio al cual pertenece la evidencia compartida.
Nombre de la tabla: xl_evidencia		
Descripción: almacena los datos referentes a las evidencias personales.		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador de la evidencia.
portafolio_id	integer	Identificador del portafolio al cual pertenece la evidencia.
sección_id	integer	Identificador de la sección a la cual pertenece la evidencia.

nombre	string	Nombre de la evidencia.
descripción	text	Descripción de la evidencia.
created	datetime	Momento en que se crea la evidencia.
updated	datetime	Momento en que se actualizan los datos de la evidencia.
valoración	decimal (2)	Valoración de una evidencia.
totalvaloraciones	integer	Cantidad de veces que ha sido valorada la evidencia.
Nombre de la tabla: xl_comentario		
Descripción: almacena los comentarios realizados a las evidencias compartidas.		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador del comentario.
evidencia_id	integer	Identificador de la evidencia compartida que contiene el comentario.
nombre	string (255)	Nombre del comentario.
contenido	text	Contenido del comentario.
created	datetime	Momento en que se crea el comentario.
Nombre de la tabla: xl_archivo		
Descripción: almacena los datos referentes a los archivos personales.		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador del archivo.
portafolio_id	integer	Identificador del portafolio al cual pertenece el archivo.
nombre	string	Nombre del archivo.
size	decimal (2)	Tamaño que puede presentar el archivo.
created	datetime	Momento en que se crea el archivo.
tipo_mime	string	Almacena el tipo de archivo que se crea.
descripción	text (500)	Descripción del archivo personal.
adjunto	file (10M)	Recurso adjunto que contiene el archivo, que puede tener un tamaño máximo de hasta 10MegaBytes.
created	datetime	Momento que se crea el archivo.
updated	datetime	Momento en que se actualizan los datos del archivo.
Nombre de la tabla: seccion_tag		
Descripción: almacena los datos relacionados con la sección y etiquetas asociadas.		
Atributo	**Tipo**	**Descripción**
seccion_id	integer	Identificador de la sección.
tag_id	integer	Identificador de la etiqueta.

Nombre de la tabla: evidencia_tag		
Descripción: almacena los datos relacionados con las evidencias y etiquetas asociadas.		
Atributo	**Tipo**	**Descripción**
evidencia_id	integer	Identificador de la evidencia.
tag_id	integer	Identificador de la etiqueta.
Nombre de la tabla: evidencia_archivo		
Descripción: almacena los datos relacionados con las evidencias y archivos asociados.		
Atributo	**Tipo**	**Descripción**
evidencia_id	integer	Identificador de la evidencia.
archivo_id	integer	Identificador del archivo.
Nombre de la tabla: archivo_tag		
Descripción: almacena los datos relacionados con los archivos y etiquetas asociadas.		
Atributo	**Tipo**	**Descripción**
archivo_id	integer	Identificador del archivo.
tag_id	integer	Identificador de la etiqueta.
Nombre de la tabla: xl_sesion		
Descripción: almacena los datos relacionados a la sesión del usuario del sistema.		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador de la sesión.
usuario	integer	Identificador del usuario de la sesión.
token	string	Credencial perteneciente a la sesión del usuario.
expira	datetime	Momento que expira la sesión del usuario.

Anexo 9: Tareas de implementación

Iteración 1.

Tarea	
Número de Tarea: 1	
Nombre de la Tarea: Selección de los estándares de codificación	
Tipo de tarea: Investigación	**Estimación:** 2
Fecha inicio: 20/01/2014	**Fecha fin:** 21/01/2014
Responsable: Pavel A. Rodríguez Acosta y Eduardo A. López Urquiaga	
Descripción: selección de los estándares de codificación bajo los cuales se implementará el sistema.	
Tarea	
Número de Tarea: 2	
Nombre de la Tarea: Definir estándar de exportación de datos	

Tipo de tarea: Investigación	**Estimación:** 3
Fecha inicio: 22/01/2014	**Fecha fin:** 24/01/2014
Responsable: Pavel A. Rodríguez Acosta y Eduardo A. López Urquiaga	
Descripción: se realiza un estudio de los estándares más utilizados en herramientas e – Portafolio para la exportación de los datos de los usuarios y se selecciona el más indicado para esta situación.	
Tarea	
Número de Tarea: 3	
Nombre de la Tarea: Creación de la estructura de la aplicación	
Tipo de tarea: Desarrollo	**Estimación:** 2
Fecha inicio: 27/01/2014	**Fecha fin:** 28/01/2014
Responsable: Pavel A. Rodríguez Acosta	
Descripción: se implementa la estructura que tendrá la aplicación teniendo en cuenta la arquitectura que plantea el framework Symfony2. Se crearán tres módulos principales: **PortafolioBundle, UsuarioBundle** y **MensajeBundle.**	
Tarea	
Número de Tarea: 4	
Nombre de la Tarea: Definición del modelo de datos	
Tipo de tarea: Desarrollo	**Estimación:** 4
Fecha inicio: 29/01/2014	**Fecha fin:** 03/02/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se define el modelo de datos que dará soporte al flujo de información del sistema. Además se generan las entidades: *Archivo, Comentario, Evidencia, Portafolio, Sección y Tag*, pertenecientes a **PortafolioBundle,** la entidad *Usuario* perteneciente a **UsuarioBundle** y la entidad *Mensaje* perteneciente a **MensajeBundle**, las cuales serán mapeadas posteriormente como tablas relacionales de la base de datos. Además se crean las clases repositorios para cada entidad generada.	
Tarea	
Número de Tarea: 5	
Nombre de la Tarea: Diseño de las interfaces	
Tipo de tarea: Diseño y Desarrollo	**Estimación:** 2
Fecha inicio: 04/02/2014	**Fecha fin:** 05/02/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se implementan las plantillas **base.html.twig** y **frontend.html.twig** como las principales de la aplicación**,** de las cuales extenderán las restantes plantillas. Además se crea la plantilla **about.html.twig,** que muestra información acerca de la aplicación.	
Tarea	
Número de Tarea: 6	**Número de HU:** *HU1*
Nombre de la Tarea: Registro de usuarios en el sistema	
Tipo de tarea: Desarrollo	**Estimación:** 2

Fecha inicio: 06/02/2014	**Fecha fin:** 07/02/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se crea la clase controladora **DefaultController.php** que se encarga del registro de usuario a través de la acción **registroAction(),** la cual recoge los datos entrados por el usuario y los valida con procedimientos específicos para ello**.** Además se crean las plantillas **registro.html.twig,** que muestra el formulario de registro al usuario**.** Se crea el archivo **UsuarioRegistroType.php,** el cual construye el formulario de registro para el usuario.	
Tarea	
Número de Tarea: 7	**Número de HU:** *HU1*
Nombre de la Tarea: Perfil de usuario	
Tipo de tarea: Desarrollo	**Estimación:** 3
Fecha inicio: 10/02/2014	**Fecha fin:** 12/02/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se agregan a la clase controladora **DefaultController.php** las acciones **perfilAction(),** la cual muestra el formulario con toda la información del perfil del usuario autenticado, permite modificar la información y guarda los cambios en la base de datos; y **showAction(),** la cual muestra el perfil del usuario autenticado solamente**.** Además se crean las plantillas **show.html.twig** y **perfil.html.twig,** que contiene el formulario de edición de perfil**.** Se crea el archivo **UsuarioPerfilType.php,** el cual construye el formulario de edición de perfil de usuario.	
Tarea	
Número de Tarea: 8	**Número de HU:** *HU2*
Nombre de la Tarea: Autenticación de usuarios	
Tipo de tarea: Desarrollo	**Estimación:** 2
Fecha inicio: 13/02/2014	**Fecha fin:** 14/02/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se agregan a la clase controladora **DefaultController.php** las acciones **loginAction(),** la cual autentica a un usuario en el sistema y **desconectaAction(),** que finaliza la sesión del usuario, verificando primeramente que el usuario se encuentra autenticado**.** Además se crea la plantilla **login.html.twig** que muestra el formulario de autenticación**.**	
Tarea	
Número de Tarea: 9	**Número de HU:** *HU3*
Nombre de la Tarea: Gestión de archivos en el sistema	
Tipo de tarea: Desarrollo	**Estimación:** 3
Fecha inicio: 17/02/2014	**Fecha fin:** 19/02/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se crea la clase controladora **ArchivoController.php** que que presenta las acciones **createAction(), editAction() y updateAction(), removeAction() y deleteAction(), indexAction(),** que se encargan de la creación, edición, eliminación y visualización de archivos en la aplicación**.** Además se crean las plantillas **archivos.html.twig, edit.html.twig, fields.html.twig y edit.html.twig,** que se encargan de mostrar toda la información referente a los archivos personales**.** Además se crea el archivo **ArchivoType.php,** que se encarga de	

construir todos los formularios que intervienen en las acciones anteriores.

Tarea	
Número de Tarea: 10	**Número de HU:** *HU7*
Nombre de la Tarea: Gestión de secciones en el sistema	
Tipo de tarea: Desarrollo	**Estimación:** 5
Fecha inicio: 20/02/2014	**Fecha fin:** 26/02/2014
Responsable: Pavel A. Rodríguez Acosta	
Descripción: se crea la clase controladora **SeccionController.php** que presenta las acciones **showAction(), createAction() y newAction(), editAction() y updateAction(), removeAction(), indexAction(),** que se encargan de la creación, edición, eliminación y visualización de secciones personales en la aplicación**.** Además se crean las plantillas **edit.html.twig, index.html.twig, new.html.twig, secciones.html.twig, norequired.html.twig, modalSeccion.html.twig y show.html.twig,** que se encargan de mostrar toda la información referente a los archivos personales**.** Además se crea el archivo **SeccionType.php,** que se encarga de construir todos los formularios que intervienen en las acciones anteriores. Se crea la plantilla **paginador.html.twig,** quien se encarga de visualizar un listado de recursos con un paginado incluido. Esta última plantilla será utilizada en las funcionalidades en donde es necesario mostrar un listado de objetos.	

Iteración 2.

Tarea	
Número de Tarea: 11	**Número de HU:** *HU4*
Nombre de la Tarea: Gestión de evidencias personales en el sistema	
Tipo de tarea: Desarrollo	**Estimación:** 5
Fecha inicio: 27/02/2014	**Fecha fin:** 05/03/2014
Responsable: Pavel A. Rodríguez Acosta	
Descripción: se crea la clase controladora **EvidenciaController.php** que presenta las acciones **showAction(), createAction() y newAction(), editAction() y updateAction(), removeAction(),** que se encargan de la creación, edición, eliminación y visualización de las evidencias personales en la aplicación**.** Además se crean las plantillas **edit.html.twig, new.html.twig y show.html.twig,** que se encargan de mostrar toda la información referente a los archivos personales**.** Además se crea el archivo **EvidenciaType.php** y **TagType.php,** que se encarga de construir todos los formularios que intervienen en las acciones anteriores.	
Tarea	
Número de Tarea: 12	**Número de HU:** *HU5*
Nombre de la Tarea: Compartir evidencia personal	
Tipo de tarea: Desarrollo	**Estimación:** 3
Fecha inicio: 06/03/2014	**Fecha fin:** 10/03/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se agregan a la clase controladora **EvidenciaController.php** las acciones **shareAction(),** que se encargan de compartir una evidencia**.** Además se crean las plantillas **share.html.twig y shareModal.html.twig,** que se encargan de mostrar el listado de los	

usuarios de la aplicación a los que se les puede compartir una evidencia**.**	
Tarea	
Número de Tarea: 13	**Número de HU:** *HU6*
Nombre de la Tarea: Consultar evidencias compartidas	
Tipo de tarea: Desarrollo	**Estimación:** 4
Fecha inicio: 11/03/2014	**Fecha fin:** 14/03/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se crea la clase controladora **CompartidaController.php** que contiene las acciones **bymeAction(), withmeAction(), showAction(), valuateAction(),** que se encargan de visualizar las evidencias compartidas por el usuario y las que le han compartido, además de poder evaluar una evidencia compartida**.** Se crean las plantillas **sharedbyme.html.twig, sharedwithme.html.twig,** que se encargan de mostrar el listado de los usuarios de la aplicación a los que se les puede compartir una evidencia**.**	
Tarea	
Número de Tarea: 14	**Número de HU:** *HU13*
Nombre de la Tarea: Sistema de búsqueda del portafolio	
Tipo de tarea: Desarrollo	**Estimación:** 3
Fecha inicio: 17/03/2014	**Fecha fin:** 19/03/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se agregan a la clase controladora **EvidenciaController.php** la acción **indexAction(),** que se encargan de mostrar un listado de las evidencias personales del usuario**.** Además se crea la plantilla **index.html.twig,** que se encargan de mostrar el listado de las evidencias personales, conjuntamente con su información**.**	

Iteración 3.

Tarea	
Número de Tarea: 15	**Número de HU:** *HU11*
Nombre de la Tarea: Brindar servicios web	
Tipo de tarea: Desarrollo	**Estimación:** 13
Fecha inicio: 20/03/2014	**Fecha fin:** 07/04/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se crean las clases controladoras **ApiArchivoController.php, ApiEvidenciaController.php, ApiSeccionController.php, ApiUsuarioController.php,** las cuales serán las encargadas de generar las funcionalidades del sistema como servicios web. Además se crean plantillas con el objetivo de visualizar la información que se enviará a través de las peticiones que genere el sistema. Se implementa la clase **servicios.php** con el objetivo de brindar los servicios que se pueden consumir en la aplicación y los métodos para hacerlo.	
Tarea	
Número de Tarea: 16	**Número de HU:** *HU15*
Nombre de la Tarea: Administración de usuarios del sistema	
Tipo de tarea: Desarrollo	**Estimación:** 2

Fecha inicio: 08/04/2014	**Fecha fin:** 09/04/2014
Responsable: Pavel A. Rodríguez Acosta	
Descripción: se utiliza el componente ***SonataAdminBundle*** para realizar todas las tareas de administración referente a los usuarios.	

Iteración 4.

Tarea	
Número de Tarea: 17	**Número de HU:** *HU8*
Nombre de la Tarea: Consultar notificaciones personales	
Tipo de tarea: Desarrollo	**Estimación:** 2
Fecha inicio: 10/04/2014	**Fecha fin:** 11/04/2014
Responsable: Pavel A. Rodríguez Acosta	
Descripción: se crea la clase controladora **DefaultController.php** que contiene la acción **portadaAction(),** que se encarga de mostrar la pantalla de bienvenida cuando un usuario se autentica**.** Además se crean las plantillas **index.html.twig, faq.html.twig** y **flashes.html.twig,** que se encargan de mostrar un listado con las evidencias personales y las notificaciones más recientes del usuario.	
Tarea	
Número de Tarea: 18	**Número de HU:** *HU9*
Nombre de la Tarea: Exportar datos del portafolio	
Tipo de tarea: Desarrollo	**Estimación:** 6
Fecha inicio: 14/04/2014	**Fecha fin:** 21/04/2014
Responsable: Pavel A. Rodríguez Acosta	
Descripción: se crea la clase controladora **PortafolioController.php** que presenta la acción **exportAction,** que se encarga de exportar los datos del portafolio a formato **leap2A**.	
Tarea	
Número de Tarea: 19	**Número de HU:** *HU10*
Nombre de la Tarea: Comentarios de las evidencias	
Tipo de tarea: Desarrollo	**Estimación:** 2
Fecha inicio: 22/04/2014	**Fecha fin:** 23/04/2014
Responsable: Pavel A. Rodríguez Acosta	
Descripción: se agregan a las clases controladoras **CompartidaController.php** y **EvidenciaController.php** la acción **createComment(),** y se crea la clase controladora **ComentarioController.php** presentando la acción **createAction(),** la cual crea un nuevo comentario para la evidencia personal. Se crean las plantillas **comentarios.html.twig** y **form.html.twig,** para visualizar los formularios correspondientes a los comentarios de las evidencias. Además se crean las clases **ComentarioHandler.php, ComentarioModel.php** y **ComentarioType.php.**	
Tarea	
Número de Tarea: 20	**Número de HU:** *HU12*

Nombre de la Tarea: Mensajes del sistema	
Tipo de tarea: Desarrollo	**Estimación:** 4
Fecha inicio: 24/04/2014	**Fecha fin:** 29/04/2014
Responsable: Eduardo A. López Urquiaga	
Descripción: se crea la clase controladora **DefaultController.php,** perteneciente a **PortafolioBundle,** que contiene la acción **contactAction(),** la cual crea un objeto de tipo **Contacto()** y envía un mensaje al correo electrónico del usuario solicitado. Además se crean las plantillas **contact.txt.twig** y **contact.html.twig,** que muestran el formulario de envío de mensaje.	
Tarea	
Número de Tarea: 21	**Número de HU:** *HU14*
Nombre de la Tarea: Internacionalización del idioma	
Tipo de tarea: Desarrollo	**Estimación:** 1
Fecha inicio: 30/04/2014	**Fecha fin:** 30/04/2014
Responsable: Pavel A. Rodríguez Acosta	
Descripción: se crean los archivos **messages.en.xliff** y **messages.es.xliff**, con todas las traducciones de los distintos recursos que intervienen en el sistema, de acuerdo al lenguaje seleccionado.	

Anexo 10: Diseño de casos de prueba

Iteración 1.

Caso de prueba de aceptación	
Código: HU1_P1	**HU1**
Nombre: Registrar usuario en el sistema.	
Descripción: el flujo se inicia cuando un usuario desea registrarse en el sistema, y con ello crearse un portafolio personal.	
Condiciones de ejecución: el usuario no puede estar registrado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar la opción Registrarse. 2. Llenar los campos del formulario de registro.	
Resultado esperado: el usuario es registrado satisfactoriamente en el sistema y se crea un portafolio personal.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU1_P2	**HU1**
Nombre: Mostrar datos del perfil de usuario	
Descripción: el flujo se inicia cuando el usuario desea visualizar los datos de su perfil personal.	

Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Perfil”.	
Resultado esperado: el sistema debe mostrar una vista con todos los datos del perfil del usuario autenticado.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU1_P3	**HU1**
Nombre: Editar perfil de usuario	
Descripción: el flujo se inicia cuando el usuario desea editar los datos de su perfil personal.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar la opción Perfil. 2. Selecciona “Editar datos del perfil”.	
Resultado esperado: el sistema debe mostrar un formulario con todos los campos de edición de los datos personales del usuario autenticado.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU2_P1	**HU2**
Nombre: Autenticar usuario en el sistema	
Descripción: el flujo se inicia cuando el usuario desea ingresar en el sistema.	
Condiciones de ejecución: el usuario no debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Llenar los datos del formulario de autenticación. 2. Seleccionar “Autenticar”.	
Resultado esperado: el sistema debe mostrar el espacio de trabajo principal del portafolio personal para el usuario autenticado.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU3_P1	**HU3**
Nombre: Visualizar archivos personales	
Descripción: el flujo se inicia cuando el usuario desea visualizar los archivos que ha almacenado en el sistema.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Archivos”.	
Resultado esperado: el sistema debe mostrar un formulario donde el usuario pueda incluir un	

nuevo archivo, y seguidamente un listado de todos los archivos que ha almacenado en el sistema. En caso de que la cantidad de archivos excedan de 3, el sistema mostrará un paginado con los archivos restantes.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU3_P2	**HU3**
Nombre: Crear un nuevo archivo personal	
Descripción: el flujo se inicia cuando el usuario desea incluir un nuevo archivo al sistema.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Archivos”. 2. Seleccionar “Seleccione un archivo”. 3. Seleccionar “Subir archivo”.	
Resultado esperado: el sistema debe mostrar el nuevo archivo incluido en el listado y además mostrar una notificación de confirmación.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU3_P3	**HU3**
Nombre: Editar archivo sin almacenar	
Descripción: el flujo se inicia cuando el usuario desea incluir un archivo al sistema y el archivo lo quiere reemplazar por otro.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Archivos”. 2. Seleccionar “Seleccione un archivo”. 3. Seleccionar “Editar archivo sin almacenar”.	
Resultado esperado: el sistema debe permitirle al usuario seleccionar otro archivo para reemplazarlo.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU3_P4	**HU3**
Nombre: Eliminar archivo sin almacenar	
Descripción: el flujo se inicia cuando el usuario desea incluir un archivo al sistema y el archivo que seleccionó lo desea eliminar.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Archivos”.	

2. Seleccionar “Seleccione un archivo”. 3. Seleccionar “Eliminar archivo sin almacenar”.
Resultado esperado: el sistema debe eliminar el archivo que el usuario seleccionó.
Evaluación de la prueba: Satisfactoria
Caso de prueba de aceptación
Código: HU3_P5 — **HU3**
Nombre: Editar archivo personal
Descripción: el flujo se inicia cuando el usuario desea editar los datos de un archivo personal.
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.
Entrada/Pasos de ejecución: 1. Seleccionar “Archivos”. 2. Seleccionar la opción “Editar archivo”. 3. Llenar los campos del formulario de edición. 4. Seleccionar “Actualizar”.
Resultado esperado: el sistema debe mostrar el archivo editado con sus nuevos datos y además el sistema emitirá una notificación de confirmación.
Evaluación de la prueba:
Caso de prueba de aceptación
Código: HU3_P6 — **HU3**
Nombre: Eliminar archivo personal
Descripción: el flujo se inicia cuando el usuario desea eliminar un archivo personal.
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.
Entrada/Pasos de ejecución: 1. Seleccionar “Archivos”. 2. Seleccionar la opción “Eliminar archivo”.
Resultado esperado: el sistema debe mostrar un formulario con todos los datos del usuario autenticado.
Evaluación de la prueba: Satisfactoria.
Caso de prueba de aceptación
Código: HU3_P7 — **HU3**
Nombre: Ver archivo personal
Descripción: el flujo se inicia cuando el usuario desea visualizar un archivo personal.
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.
Entrada/Pasos de ejecución: 1. Seleccionar “Archivos”. 2. Selecciona el archivo que desea visualizar.
Resultado esperado: el sistema debe mostrar el archivo al usuario, en dependencia de qué

tipo de archivo sea.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU7_P1	**HU7**
Nombre: Visualizar secciones creadas	
Descripción: el flujo se inicia cuando el usuario desea visualizar las secciones creadas.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Secciones”.	
Resultado esperado: el sistema debe mostrar un listado con todas las secciones creadas por el usuario. En caso de que la cantidad de secciones excedan de 5, el sistema mostrará un paginado con las secciones restantes.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU7_P2	**HU7**
Nombre: Incluir nueva sección	
Descripción: el flujo se inicia cuando el usuario desea crear una nueva sección.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Secciones”. 2. Seleccionar “Crear Sección”. 3. Llenar los datos del formulario. 4. Seleccionar “Aceptar”.	
Resultado esperado: el sistema debe mostrar un listado donde aparezca la nueva sección creada. Además emitirá una notificación de confirmación.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU7_P3	**HU7**
Nombre: Editar datos de la sección	
Descripción: el flujo se inicia cuando el usuario desea editar los datos de una sección.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Secciones”. 2. Seleccionar “Editar Sección”. 3. Llenar los datos del formulario. 4. Seleccionar “Actualizar”.	1. Seleccionar “Secciones”. 2. Seleccionar una sección. 3. Seleccionar “Acciones – Editar sección”. 4. Llenar los datos del formulario. 5. Seleccionar “Actualizar”.

Resultado esperado: en ambos casos el sistema debe mostrar la nueva sección con sus datos actualizados. Además emitirá una notificación de confirmación.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU7_P4	**HU7**
Nombre: Visualizar datos de la sección	
Descripción: el flujo se inicia cuando el usuario desea editar los datos de una sección.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Secciones”. 2. Seleccionar una sección.	
Resultado esperado: el sistema debe mostrar los datos de la sección seleccionada y un listado de las evidencias asociadas a esa sección.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU7_P5	**HU7**
Nombre: Eliminar sección	
Descripción: el flujo se inicia cuando el usuario desea eliminar una sección.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Secciones”. 2. Seleccionar “Eliminar sección”. 3. Seleccionar “Aceptar” o “Cancelar”.	
Resultado esperado: el sistema debe mostrar un mensaje de confirmación de eliminación. Si el usuario selecciona “Cancelar”, el sistema no realiza ninguna acción. Si selecciona “Aceptar” se elimina la sección y el sistema muestra una notificación de confirmación.	
Evaluación de la prueba: Satisfactoria	

Iteración 2.

Caso de prueba de aceptación	
Código: HU5_P1	**HU5**
Nombre: Mostrar un listado de los usuarios registrados en el sistema a los que se pueda compartir una evidencia.	
Descripción: el flujo se inicia cuando el usuario desea compartir una evidencia personal.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Evidencias”.	1. Seleccionar “Secciones”.

2. Seleccionar “Mis evidencias”. 3. Seleccionar evidencia. 4. Seleccionar “Compartir”.	2. Seleccionar sección. 3. Seleccionar evidencia. 4. Seleccionar “Compartir”.
Resultado esperado: en ambos casos el sistema debe mostrar un listado de los usuarios registrados en el sistema a los cuales se les puede compartir una evidencia.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU5_P2	**HU5**
Nombre: Compartir evidencia personal.	
Descripción: el flujo se inicia cuando el usuario desea compartir una evidencia personal.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución:	
1. Seleccionar “Evidencias”. 2. Seleccionar “Mis evidencias”. 3. Seleccionar evidencia. 4. Seleccionar “Compartir”. 5. Seleccionar usuario(s) 6. Seleccionar “Actualizar”.	1. Seleccionar “Secciones”. 2. Seleccionar sección. 3. Seleccionar evidencia. 4. Seleccionar “Compartir”. 5. Seleccionar usuario(s) 6. Seleccionar “Actualizar”.
Resultado esperado: en ambos casos el sistema debe mostrar los detalles de la evidencia compartida y además una notificación de confirmación.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU4_P1	**HU4**
Nombre: Visualizar evidencias personales	
Descripción: el flujo se inicia cuando el usuario desea visualizar sus evidencias personales.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Evidencias”. 2. Seleccionar “Mis evidencias”.	
Resultado esperado: el sistema debe mostrar un listado con todas las evidencias creadas por el usuario. En caso de que la cantidad de evidencias excedan de 5, el sistema mostrará un paginado con las evidencias restantes.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU4_P2	**HU4**
Nombre: Incluir nueva evidencia personal	

Descripción: el flujo se inicia cuando el usuario desea crear una evidencia personal.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Secciones”. 2. Seleccionar “Nueva evidencia”. 3. Llenar los datos del formulario. 4. Seleccionar “Aceptar”.	1. Seleccionar “Secciones”. 2. Seleccionar sección. 3. Seleccionar “Acciones”. 4. Seleccionar “Nueva evidencia”. 5. Llenar los datos del formulario. 6. Seleccionar “Aceptar”.
Resultado esperado: en ambos casos el sistema debe mostrar la sección a la cual pertenece la evidencia creada y seguidamente un listado de las evidencias asociadas a esa sección. Además el sistema mostrará una notificación de confirmación.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU4_P3	**HU4**
Nombre: Editar parámetros de la evidencia	
Descripción: el flujo se inicia cuando el usuario desea editar los parámetros de una evidencia personal.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Evidencias”. 2. Seleccionar “Mis evidencias”. 3. Seleccionar evidencia. 4. Seleccionar “Editar evidencia”. 5. Llenar los datos del formulario. 6. Seleccionar “Actualizar”.	1. Seleccionar “Secciones”. 2. Seleccionar sección. 3. Seleccionar evidencia. 4. Seleccionar “Editar evidencia”. 5. Llenar los datos del formulario. 6. Seleccionar “Actualizar”.
Resultado esperado: en ambos casos el sistema debe mostrar la sección a la cual pertenece la evidencia editada y seguidamente un listado de las evidencias asociadas a esa sección. Además el sistema mostrará una notificación de confirmación.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU4_P4	**HU4**
Nombre: Eliminar evidencia personal	
Descripción: el flujo se inicia cuando el usuario desea editar los parámetros de una evidencia personal.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución:	

1. Seleccionar “Evidencias”. 2. Seleccionar “Mis evidencias”. 3. Seleccionar evidencia. 4. Seleccionar “Eliminar evidencia”.	1. Seleccionar “Secciones”. 2. Seleccionar sección. 3. Seleccionar evidencia. 4. Seleccionar “Eliminar evidencia”.
Resultado esperado: en ambos casos el sistema debe eliminar la evidencia seleccionada y seguidamente una notificación de confirmación.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU4_P5	**HU4**
Nombre: Visualizar detalles de la evidencia	
Descripción: el flujo se inicia cuando el usuario desea ver los detalles de una evidencia personal.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución:	
1. Seleccionar “Evidencias”. 2. Seleccionar “Mis evidencias”. 3. Seleccionar evidencia.	1. Seleccionar “Secciones”. 2. Seleccionar sección. 3. Seleccionar evidencia.
Resultado esperado: en ambos casos el sistema debe mostrar los detalles de la evidencia seleccionada.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU4_P6	**HU4**
Nombre: Ver archivos personales asociados a la evidencia	
Descripción: el flujo se inicia cuando el usuario desea visualizar el(los) archivo(s) personales que se encuentren asociados a una evidencia.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución:	
1. Seleccionar “Evidencias”. 2. Seleccionar “Mis evidencias”. 3. Seleccionar evidencia. 4. Seleccionar “Archivos”.	1. Seleccionar “Secciones”. 2. Seleccionar sección. 3. Seleccionar evidencia. 4. Seleccionar “Archivos”.
Resultado esperado: en ambos casos el sistema debe abrir el(los) archivo(s) que se encuentren asociados a la evidencia.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU6_P1	**HU6**
Nombre: Mostrar evidencias compartidas por el usuario	

Descripción: el flujo se inicia cuando el usuario desea visualizar las evidencias que ha compartido.
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.
Entrada/Pasos de ejecución: 1. Seleccionar “Evidencias”. 2. Seleccionar “Compartidas por mí”.
Resultado esperado: el sistema debe mostrar un listado de las evidencias que han sido compartidas por el usuario.
Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código: HU6_P2	**HU6**
Nombre: Mostrar las evidencias que le han sido compartidas al usuario.	
Descripción: el flujo se inicia cuando el usuario desea visualizar las evidencias que le han sido compartidas.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Evidencias”. 2. Seleccionar “Compartidas a mí”.	
Resultado esperado: el sistema debe mostrar un listado de las evidencias que le han sido compartidas al usuario.	
Evaluación de la prueba: Satisfactoria	

Caso de prueba de aceptación	
Código: HU6_P3	**HU6**
Nombre: Valorar evidencia compartida	
Descripción: el flujo se inicia cuando el usuario desea valorar una evidencia que le han compartido.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Evidencias”. 2. Seleccionar “Compartidas a mí”. 3. Seleccionar evidencia. 4. Emitir “Valoración”.	
Resultado esperado: el sistema debe mostrar los detalles de la evidencia valorada y además un texto de confirmación.	
Evaluación de la prueba: Satisfactoria	

Caso de prueba de aceptación	
Código: HU13_P1	**HU13**

Nombre: Filtrar búsqueda de archivos	
Descripción: el flujo se inicia cuando el usuario desea visualizar las evidencias que le han sido compartidas.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Secciones”. 2. Seleccionar sección. 3. Seleccionar “Acciones”. 4. Seleccionar “Nueva evidencia”. 5. Filtrar por archivos	1. Seleccionar “Secciones”. 2. Seleccionar “Nueva evidencia”. 3. Filtrar por archivos.
Resultado esperado: el sistema debe mostrar, a medida que el usuario escribe caracteres, un listado de los archivos que coinciden con los patrones de búsqueda.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU13_P2	**HU13**
Nombre: Filtrar búsqueda de usuarios	
Descripción: el flujo se inicia cuando el usuario desea compartir una evidencia.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Secciones”. 2. Seleccionar “Mostrar evidencias”. 3. Seleccionar evidencia. 4. Seleccionar “Compartir”. 5. Filtrar por usuarios.	1. Seleccionar “Evidencias”. 2. Seleccionar “Mis evidencias”. 3. Seleccionar evidencia. 4. Seleccionar “Compartir”. 6. Filtrar por usuarios.
Resultado esperado: el sistema debe mostrar, a medida que el usuario escribe caracteres, un listado de los usuarios que coinciden con los patrones de búsqueda.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU13_P3	**HU13**
Nombre: Búsqueda global	
Descripción: el flujo se inicia cuando el usuario desea buscar algún elemento en el portafolio.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Insertar patrones de búsqueda. 2. Seleccionar “Buscar”.	
Resultado esperado: el sistema debe mostrar, un listado de los elementos que coinciden con los patrones de búsqueda. Esta búsqueda se realizará en base al nombre y descripción de los	

elementos.
Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código: HU13_P4	**HU13**
Nombre: Búsqueda por etiquetas	
Descripción: el flujo se inicia cuando el usuario desea buscar un elemento a través de sus etiquetas asociadas.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar una etiqueta en el panel "Etiqueta".	
Resultado esperado: el sistema debe mostrar un listado con todos los elementos asociados a la etiqueta seleccionada.	
Evaluación de la prueba: Satisfactoria	

Iteración 4.

Caso de prueba de aceptación	
Código: HU8_P1	**HU8**
Nombre: Mostrar listado de notificaciones	
Descripción: el flujo se inicia cuando el usuario desea visualizar las notificaciones que le han sido emitidas. Puede seleccionar ver todas las notificaciones o ver las que no han sido leídas.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar "Notificaciones"	
Resultado esperado: el sistema debe mostrar los un listado de todas las notificaciones que le han sido emitidas al usuario. Además cuando se autentique, debe visualizarse un panel con las notificaciones más recientes.	
Evaluación de la prueba: Satisfactoria	

Caso de prueba de aceptación	
Código: HU8_P2	**HU8**
Nombre: Eliminar notificaciones	
Descripción: el flujo se inicia cuando el usuario desea eliminar las notificaciones que le han sido emitidas.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar "Notificaciones". 2. Seleccionar notificación. 3. Seleccionar "Eliminar".	

Resultado esperado: el sistema debe mostrar el listado de las notificaciones donde haya sido eliminada la seleccionada por el usuario. Además el sistema emite una notificación de confirmación.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU8_P3	**HU8**
Nombre: Ver detalles de la notificación	
Descripción: el flujo se inicia cuando el usuario desea ver los detalles de las notificaciones que le han sido emitidas.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Notificaciones”. 2. Seleccionar notificación. 3. Seleccionar “Ver detalles”.	
Resultado esperado: el sistema debe mostrar los detalles de la notificación seleccionada por el usuario.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU9_P1	**HU9**
Nombre: Exportar datos del portafolio	
Descripción: el flujo se inicia cuando el usuario desea exportar los datos de su portafolio.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Opciones”. 2. Seleccionar “Exportar”.	
Resultado esperado: el sistema debe generar un archivo con la estructura **leap2a.zip** y debe permitirle al usuario descargarlo hacia una dirección local de su PC.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU10_P1	**HU10**
Nombre: Incluir comentario en evidencia	
Descripción: el flujo se inicia cuando el usuario desea incluir un comentario en una evidencia creada por él o una evidencia compartida.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Evidencias”. 2. Seleccionar “Mis evidencias”.	1. Seleccionar “Evidencias”. 2. Seleccionar “Compartidas a mí”.

3. Seleccionar evidencia. 4. Incluir comentario. 5. Seleccionar “Comentario”.	3. Seleccionar evidencia. 4. Incluir comentario. 5. Seleccionar “Comentario”.
Resultado esperado: el sistema debe mostrar los detalles de la evidencia y un listado donde aparezca el nuevo comentario.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU10_P2	**HU10**
Nombre: Ver comentarios de la evidencia	
Descripción: el flujo se inicia cuando el usuario desea visualizar los comentarios emitidos a una evidencia creada por él o una evidencia compartida.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución:	
1. Seleccionar “Evidencias”. 2. Seleccionar “Mis evidencias”. 3. Seleccionar evidencia.	1. Seleccionar “Evidencias”. 2. Seleccionar “Compartidas a mí”. 3. Seleccionar evidencia.
Resultado esperado: el sistema debe mostrar los detalles de la evidencia y un listado donde aparezca los comentarios emitidos a esa evidencia.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU12_P1	**HU12**
Nombre: Enviar mensajes de contacto	
Descripción: el flujo se inicia cuando el usuario desea enviar un mensaje de contacto a los administradores del sistema.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Contáctenos”. 2. Llenar los datos del formulario. 3. Seleccionar “Enviar”.	
Resultado esperado: el sistema debe enviar un mensaje vía correo electrónico a la dirección especificada en el formulario. Además el sistema muestra una notificación de confirmación de mensaje enviado.	
Evaluación de la prueba: Satisfactoria	
Caso de prueba de aceptación	
Código: HU12_P2	**HU12**
Nombre: Enviar mensaje del portafolio	
Descripción: el flujo se inicia cuando el usuario desea recibir notificaciones del sistema por	

correo electrónico.
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.
Entrada/Pasos de ejecución: 1. Seleccionar “Perfil”. 2. Seleccionar “Editar datos del perfil”. 3. Seleccionar “Recibir notificaciones por correo”. 4. Seleccionar “Actualizar”.
Resultado esperado: el sistema enviará un mensaje con las últimas actividades que han tenido lugar en el sistema desde la última vez que el usuario accedió a este.
Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código: HU14_P1	**HU14**
Nombre: Internacionalización del idioma	
Descripción: el flujo se inicia cuando el usuario desea cambiar el idioma en el que se muestran los contenidos del portafolio.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Seleccionar “Perfil”. 2. Seleccionar “Editar datos del perfil”. 3. Seleccionar “Idioma predeterminado”. 4. Seleccionar “Actualizar”.	1. Seleccionar “Idioma”.
Resultado esperado: en ambos casos el sistema debe mostrar el contenido del portafolio en el idioma seleccionado por el usuario.	
Evaluación de la prueba: Satisfactoria	

Caso de prueba de aceptación	
Código: HU14_P2	**HU14**
Nombre: Internacionalización del idioma	
Descripción: el flujo se inicia cuando el usuario desea accede al sistema con el idioma predeterminado que seleccionó.	
Condiciones de ejecución: el usuario no debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: 1. Llenar formulario de autenticación. 2. Seleccionar “Autenticar”.	
Resultado esperado: el sistema debe mostrar el espacio de trabajo con los elementos traducidos al idioma predeterminado del usuario.	
Evaluación de la prueba: Satisfactoria	

Anexo 11: Especificación de los Servicios Web

Nombre del servicio: Listar secciones			**Método de envío de información:** GET
Nombre de la ruta: /api/v1/secciones.{_format}			
Requerimientos			
Nombre	**Requerimiento**	**Tipo**	**Descripción**
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
Responsabilidad: permite obtener el listado de todas las secciones del portafolio.			
Parámetros de salida: • 200 – Acción realizada con éxito. • 400 – La petición generó un error de ejecución. • 401 – Retorna cuando no está autorizado.			

Nombre del servicio: Crear sección			**Método de envío de información:** POST
Nombre de la ruta: /api/v1/secciones.{_format}			
Requerimientos			
Nombre	**Requerimiento**	**Tipo**	**Descripción**
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
Parámetros de entrada			
Nombre del parámetro		**Tipo de dato**	**Requerido**
xalix_portafoliobundle_seccion [titulo]		string	true
xalix_portafoliobundle_seccion [descripcion]		string	false
Responsabilidad: crea una nueva sección con los parámetros especificados.			
Parámetros de salida: • 200 – Acción realizada con éxito. • 400 – Retorna cuando el formulario de envío tuvo errores. • 401 – Retorna cuando no está autorizado.			

Nombre del servicio: Formulario crear sección			**Método de envío de información:** GET
Nombre de la ruta: /api/v1/secciones/new.{_format}			
Requerimientos			
Nombre	**Requerimiento**	**Tipo**	**Descripción**
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
Responsabilidad: presenta el formulario a usar para la creación de una nueva sección.			
Parámetros de salida: • 200 – Acción realizada con éxito. • 401 – Retorna cuando no está autorizado.			

Nombre del servicio: Obtener sección por id	**Método de envío de información:** GET

Nombre de la ruta: /api/v1/secciones/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la sección que se desea obtener.

Responsabilidad: obtiene una sección dado el identificador.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – La petición generó un error de ejecución.
- 404 – Retorna cuando la sección no fue encontrada.

Nombre del servicio: Editar sección parcialmente — **Método de envío de información:** PATCH

Nombre de la ruta: /api/v1/secciones/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la sección que se desea obtener.

Parámetros de entrada

Nombre del parámetro	Tipo de dato	Requerido
xalix_portafoliobundle_seccion [titulo]	string	true
xalix_portafoliobundle_descripcion [descripcion]	string	false

Responsabilidad: permite editar parcialmente los datos de una sección, en relación a los parámetros que se especifiquen.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando el formulario de envío tuvo errores.
- 401 – Retorna cuando no está autorizado.
- 404 – Retorna cuando la sección no fue encontrada.

Nombre del servicio: Editar sección — **Método de envío de información:** PUT

Nombre de la ruta: /api/v1/secciones/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la sección que se desea obtener.

Parámetros de entrada

Nombre del parámetro	Tipo de dato	Requerido
xalix_portafoliobundle_seccion [titulo]	string	true
xalix_portafoliobundle_descripcion [descripcion]	string	false

Responsabilidad: permite editar la sección cuyo identificador es especificado.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando el formulario de envío tuvo errores.
- 401 – Retorna cuando no está autorizado.
- 404 – Retorna cuando la sección no fue encontrada.

Nombre del servicio: Eliminar sección | **Método de envío de información:** DELETE

Nombre de la ruta: /api/v1/secciones/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la sección que se desea obtener.

Responsabilidad: permite eliminar una sección cuyo identificador es especificado.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 401 – Retorna cuando no está autorizado.
- 404 – Retorna cuando la sección no fue encontrada.

Nombre del servicio: Registrar usuario | **Método de envío de información:** POST

Nombre de la ruta: /api/v1/usuarios.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Parámetros de entrada

Nombre del parámetro	Tipo de dato	Requerido
xalix_serviciosbundle_usuario [nombre]	string	true
xalix_serviciosbundle_usuario [apellidos]	string	true
xalix_serviciosbundle_usuario [email]	string	true
xalix_serviciosbundle_usuario [password] [first]	string	true
xalix_serviciosbundle_usuario [password] [second]	string	true
xalix_serviciosbundle_usuario [permite_email]	boolean	false

Responsabilidad: registra un usuario en el sistema.

Parámetros de salida:

- 200 – Acción realizada con éxito.

- 226 – Retorna cuando el email ya está en uso.
- 400 – Retorna cuando los parámetros fueron mal enviados.

Nombre del servicio: Usuario logueado | **Método de envío de información:** GET

Nombre de la ruta: /api/v1/usuario.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Responsabilidad: obtener datos del usuario logueado.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 401 – Retorna cuando no está autorizado.
- 400 – La petición generó un error de ejecución.

Nombre del servicio: Actualizar usuario parcialmente | **Método de envío de información:** PATCH

Nombre de la ruta: /api/v1/usuario.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Parámetros de entrada

Nombre del parámetro	Tipo de dato	Requerido
xalix_usuariobundle_registro [nombre]	string	false
xalix_usuariobundle_registro [apellidos]	string	false
xalix_usuariobundle_registro [email]	string	false
xalix_usuariobundle_registro [password] [first]	string	false
xalix_usuariobundle_registro [password] [second]	string	false
xalix_usuariobundle_registro [permite_email]	boolean	false
xalix_usuariobundle_registro [foto]	file	false

Responsabilidad: actualiza parcialmente los datos del usuario.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando el formulario de envío tuvo errores.
- 401 – Retorna cuando no está autorizado.

Nombre del servicio: Actualizar datos de usuario | **Método de envío de información:** PUT

Nombre de la ruta: /api/v1/usuario.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Parámetros de entrada		
Nombre del parámetro	**Tipo de dato**	**Requerido**
xalix_usuariobundle_registro [nombre]	string	true
xalix_usuariobundle_registro [apellidos]	string	true
xalix_usuariobundle_registro [email]	string	true
xalix_usuariobundle_registro [password] [first]	string	false
xalix_usuariobundle_registro [password] [second]	string	false
xalix_usuariobundle_registro [permite_email]	boolean	false
xalix_usuariobundle_registro [foto]	file	false

Responsabilidad: actualiza los datos del usuario en el sistema.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando el formulario tuvo errores.
- 401 – Retorna cuando no está autorizado.

Nombre del servicio: Credencial de usuario	**Método de envío de información:** POST

Nombre de la ruta: /api/v1/accesos.{_format}

Requerimientos			
Nombre	**Requerimiento**	**Tipo**	**Descripción**
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Parámetros de entrada		
Nombre del parámetro	**Tipo de dato**	**Requerido**
password	string	true
email	email	true

Responsabilidad: obtiene los datos de la sesión del usuario.

Parámetros de salida:

- 200 – Acción realizada con éxito. Incluye el ***token*** de la sesión del usuario.
- 400 – Retorna cuando los parámetros fueron mal enviados.

Nombre del servicio: Listar archivos	**Método de envío de información:** GET

Nombre de la ruta: /api/v1/archivos.{_format}

Requerimientos			
Nombre	**Requerimiento**	**Tipo**	**Descripción**
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Responsabilidad: lista todos los archivos almacenados en el sistema.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando la petición fue mal enviada.

- 401 – Retorna cuando no está autorizado.

Nombre del servicio: Crear archivo	**Método de envío de información:** POST

Nombre de la ruta: /api/v1/archivos.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Parámetros de entrada

Nombre del parámetro	Tipo de dato	Requerido
xalix_portafoliobundle_archivo [adjunto]	file	true

Responsabilidad: crea un nuevo archivo con los datos enviados.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando el formulario de envío tuvo errores.
- 401 – Retorna cuando no está autorizado.

Nombre del servicio: Formulario crear archivo	**Método de envío de información:** GET

Nombre de la ruta: /api/v1/archivos/new.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Responsabilidad: presenta un formulario para crear un nuevo archivo.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 401 – Retorna cuando no está autorizado.

Nombre del servicio: Archivo dado id	**Método de envío de información:** GET

Nombre de la ruta: /api/v1/archivos/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador del archivo que se desea obtener.

Responsabilidad: obtiene un archivo dado un identificador.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando la petición tuvo errores.
- 404 – Retorna cuando el archivo no fue encontrado.

Nombre del servicio: Editar archivo parcialmente	**Método de envío de información:** PATCH

Nombre de la ruta: /api/v1/archivos/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la sección que se desea obtener.

Parámetros de entrada

Nombre del parámetro	Tipo de dato	Requerido
xalix_portafoliobundle_archivo [nombre]	string	false
xalix_portafoliobundle_archivo [descripcion]	string	false

Responsabilidad: edita los datos del archivo parcialmente.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando el formulario de envío tuvo errores.
- 401 – Retorna cuando no está autorizado.
- 404 – Retorna cuando el archivo no fue encontrado.

Nombre del servicio: Editar archivo | **Método de envío de información:** PUT

Nombre de la ruta: /api/v1/archivos/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador del archivo que se desea obtener.

Parámetros de entrada

Nombre del parámetro	Tipo de dato	Requerido
xalix_portafoliobundle_archivo [nombre]	string	true
xalix_portafoliobundle_archivo [descripcion]	string	false

Responsabilidad: Edita el archivo con los datos especificados.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando el formulario de envío tuvo errores.
- 401 – Retorna cuando no está autorizado.
- 404 – Retorna cuando el archivo no fue encontrado.

Nombre del servicio: Eliminar archivo | **Método de envío de información:** DELETE

Nombre de la ruta: /api/v1/archivos/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador del archivo que se desea obtener.

Responsabilidad: elimina el archivo especificado dado su identificador.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 401 – Retorna cuando no está autorizado.
- 404 – Retorna cuando el archivo no fue encontrado.

Nombre del servicio: Listar evidencias | **Método de envío de información:** GET

Nombre de la ruta: /api/v1/evidencias.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Responsabilidad: lista todas las evidencias almacenadas en el sistema.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando la petición fue mal ejecutada.
- 401 – Retorna cuando no está autorizado.

Nombre del servicio: Crear evidencia | **Método de envío de información:** POST

Nombre de la ruta: /api/v1/evidencias.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Parámetros de entrada

Nombre del parámetro	Tipo de dato	Requerido
xalix_portafoliobundle_evidencia [nombre]	string	true
xalix_portafoliobundle_evidencia [descripcion]	string	false
xalix_portafoliobundle_evidencia [seccion]	integer	false

Responsabilidad: crea una nueva evidencia en el sistema.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando el formulario de envío tuvo errores.
- 401 – Retorna cuando no está autorizado.

Nombre del servicio: Formulario crear evidencia | **Método de envío de información:** GET

Nombre de la ruta: /api/v1/evidencias/new.{_format}

Requerimientos			
Nombre	**Requerimiento**	**Tipo**	**Descripción**
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
Responsabilidad: presenta el formulario para crear una nueva evidencia.			
Parámetros de salida: • 200 – Acción realizada con éxito. • 401 – Retorna cuando no está autorizado.			
Nombre del servicio: Evidencia dado id			**Método de envío de información:** GET
Nombre de la ruta: /api/v1/evidencias/{id}.{_format}			

Requerimientos			
Nombre	**Requerimiento**	**Tipo**	**Descripción**
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la evidencia que se desea obtener.
Responsabilidad: obtiene una evidencia dado un identificador.			
Parámetros de salida: • 200 – Acción realizada con éxito. • 400 – Retorna cuando la petición fue mal ejecutada. • 404 – Retorna cuando la evidencia no fue encontrada.			
Nombre del servicio: Editar evidencia parcial			**Método de envío de información:** PATCH
Nombre de la ruta: /api/v1/evidencias/{id}.{_format}			

Requerimientos			
Nombre	**Requerimiento**	**Tipo**	**Descripción**
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la evidencia que se desea obtener.

Parámetros de entrada		
Nombre del parámetro	**Tipo de dato**	**Requerido**
xalix_portafoliobundle_evidencia [nombre]	string	true
xalix_portafoliobundle_evidencia [descripcion]	string	false
xalix_portafoliobundle_evidencia [archivos]	integer	false
xalix_portafoliobundle_evidencia [seccion]	integer	false
Responsabilidad: edita los datos de la evidencia parcialmente.		
Parámetros de salida: • 200 – Acción realizada con éxito. • 400 – Retorna cuando el formulario de envío tuvo errores.		

- 401 – Retorna cuando no está autorizado.
- 404 – Retorna cuando la evidencia no fue encontrada.

Nombre del servicio: Editar evidencia | **Método de envío de información:** PUT

Nombre de la ruta: /api/v1/evidencias/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la evidencia que se desea obtener.

Parámetros de entrada

Nombre del parámetro	Tipo de dato	Requerido
xalix_portafoliobundle_evidencia [nombre]	string	true
xalix_portafoliobundle_evidencia [descripcion]	string	false
xalix_portafoliobundle_evidencia [archivos]	integer	false
xalix_portafoliobundle_evidencia [seccion]	integer	false

Responsabilidad: edita los datos de la evidencia especificados en la petición.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando el formulario de envío tuvo errores.
- 401 – Retorna cuando no está autorizado.
- 404 – Retorna cuando la evidencia no fue encontrada.

Nombre del servicio: Eliminar evidencia | **Método de envío de información:** DELETE

Nombre de la ruta: /api/v1/evidencias/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la evidencia que se desea obtener.

Responsabilidad: elimina una evidencia especificado su identificador.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 401 – Retorna cuando no está autorizado.
- 404 – Retorna cuando la sección no fue encontrada.

Nombre del servicio: Evidencias compartidas a mi | **Método de envío de información:** GET

Nombre de la ruta: /api/v1/ami/comparticiones.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.

Responsabilidad: lista las evidencias que han sido compartidas al usuario.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 401 – Retorna cuando no está autorizado.
- 400 – Retorna cuando la petición fue mal enviada.

Nombre del servicio: Evidencias compartidas a mi dado id | **Método de envío de información:** GET

Nombre de la ruta: /api/v1/comparticiones/{id}.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la evidencia que se desea obtener.

Responsabilidad: obtiene las evidencias que le han sido compartidas al usuario dado un identificador.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 400 – Retorna cuando la petición fue mal enviada.
- 404 – Retorna cuando la sección no fue encontrada.

Nombre del servicio: Compartir evidencia a usuario | **Método de envío de información:** POST

Nombre de la ruta: /api/v1/evidencias/{id}/comparticiones.{_format}

Requerimientos

Nombre	Requerimiento	Tipo	Descripción
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
id	-	integer	Representa el identificador de la evidencia que se desea compartir.

Parámetros

Nombre del parámetro	Tipo de dato	Requerido
email	email	true

Responsabilidad: comparte una evidencia a un usuario seleccionado.

Parámetros de salida:

- 200 – Acción realizada con éxito.
- 401 – Retorna cuando no está autorizado.
- 400 – Retorna cuando el formulario de envío tuvo errores.

Nombre del servicio: Evidencias compartidas por mi | **Método de envío de información:** GET

Nombre de la ruta: /api/v1/mis/comparticiones.{_format}

Requerimientos			
Nombre	**Requerimiento**	**Tipo**	**Descripción**
_format	(xml | json | html)	-	Extensiones mediante las cuales se pueden realizar las peticiones.
Responsabilidad: lista las evidencias que han sido compartidas por un usuario.			
Parámetros de salida: • 200 – Acción realizada con éxito. • 401 – Retorna cuando no está autorizado. • 400 – Retorna cuando la petición fue mal enviada.			

Anexo 12: Diagrama de componentes.

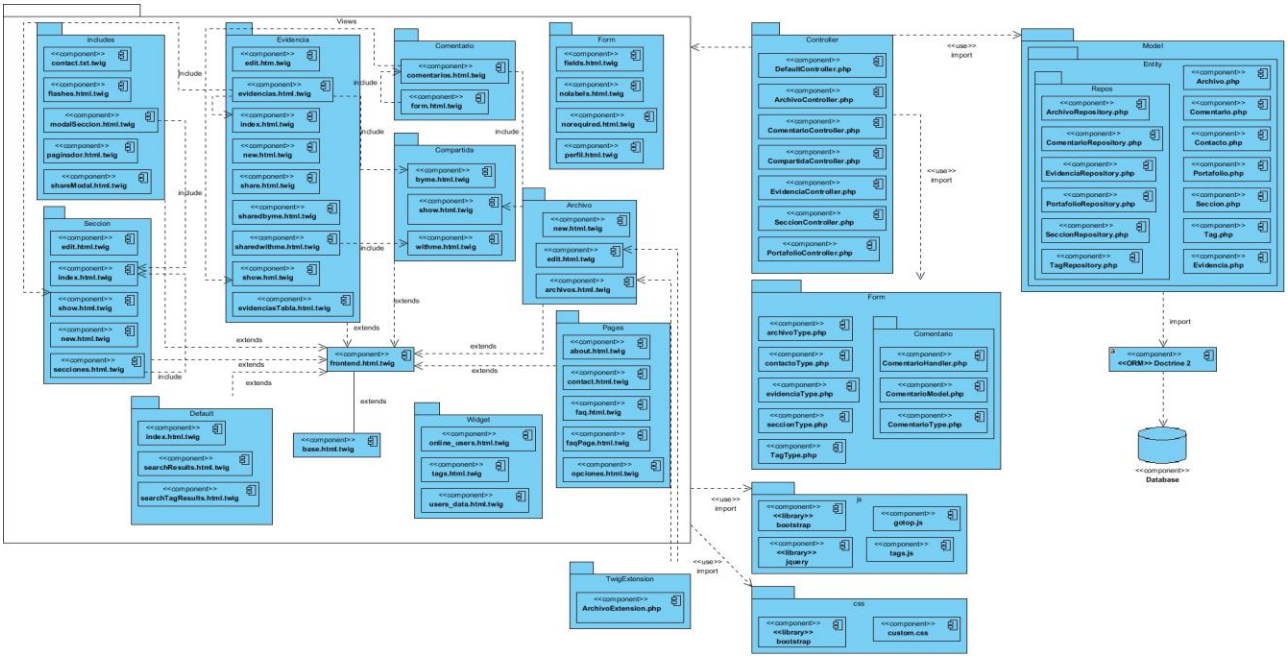


Ilustración 21. Diagrama de componentes. Paquete PortafolioBundle.

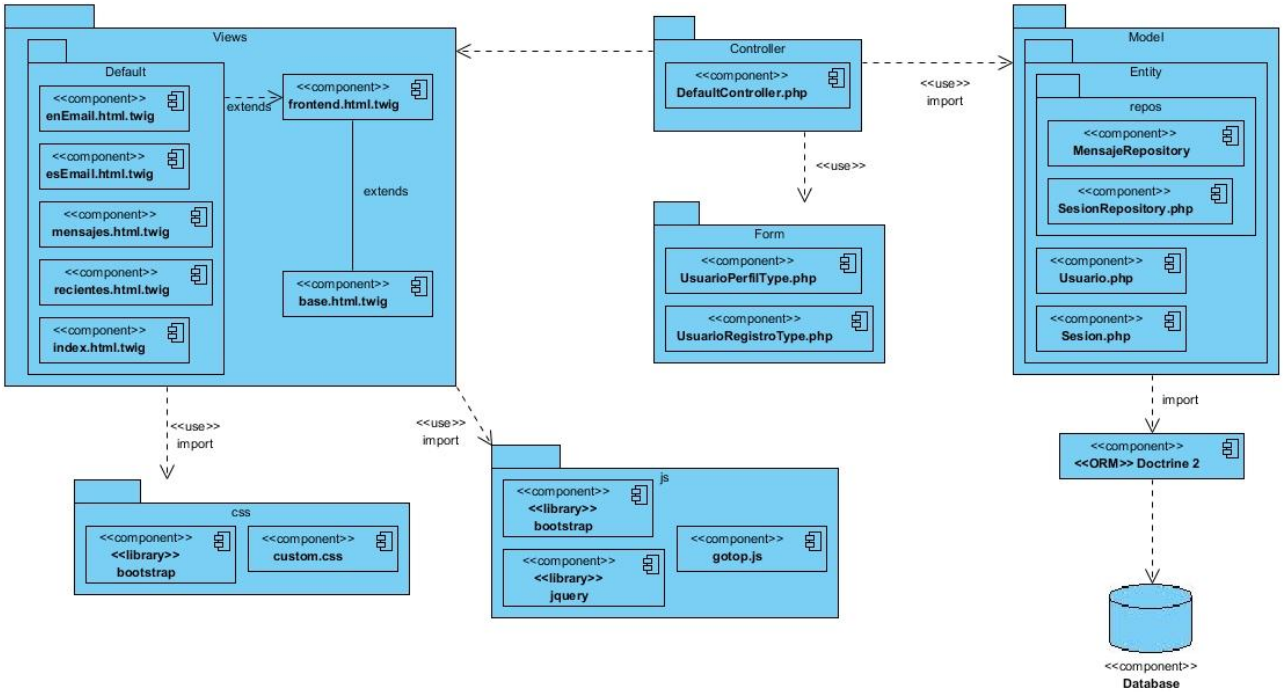


Ilustración 22. Diagrama de componentes. Paquete UsuarioBundle.

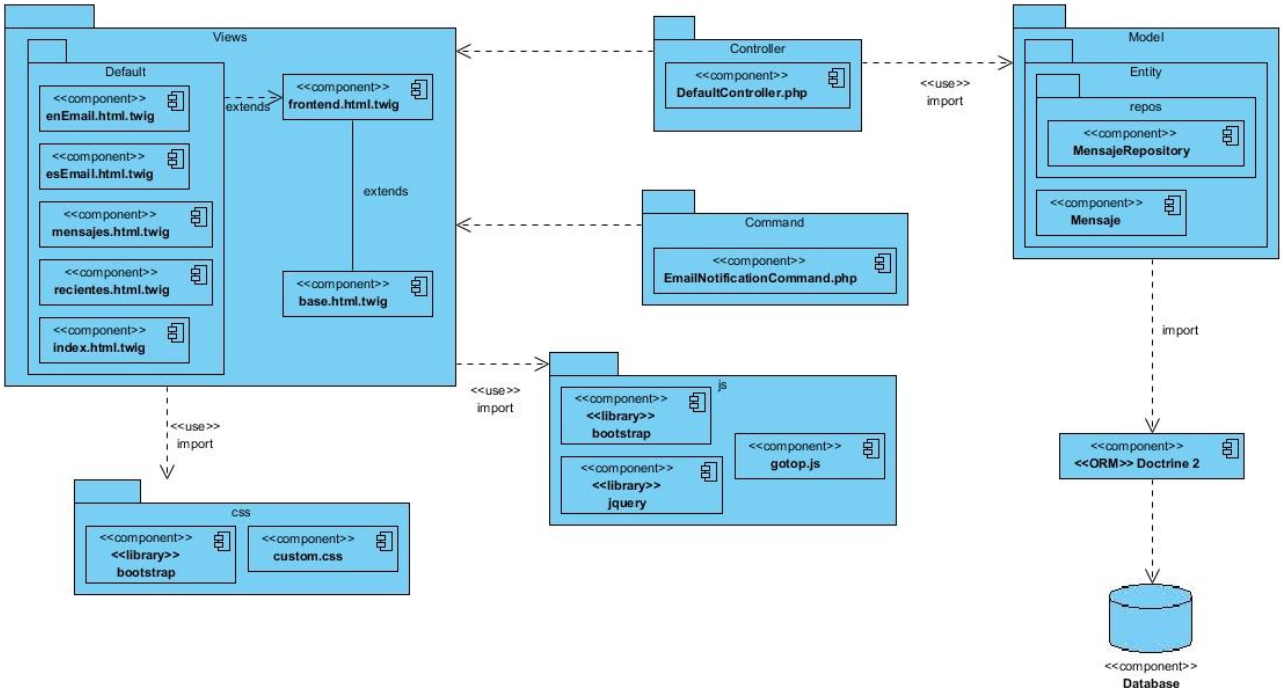


Ilustración 23. Diagrama de componentes. Paquete MensajeBundle.

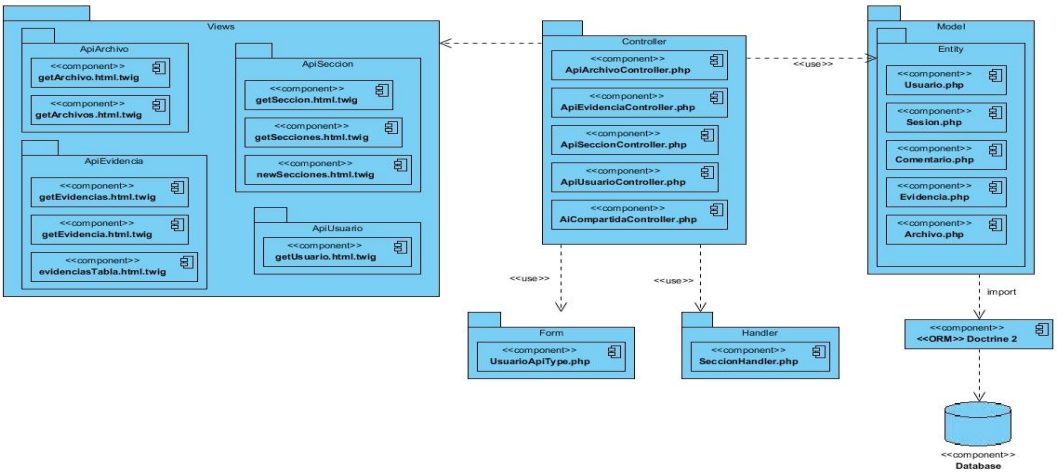


Ilustración 24. Diagrama de componentes. Paquete ServiciosBundle.

Anexo 13: Diagrama de Clases del Sistema

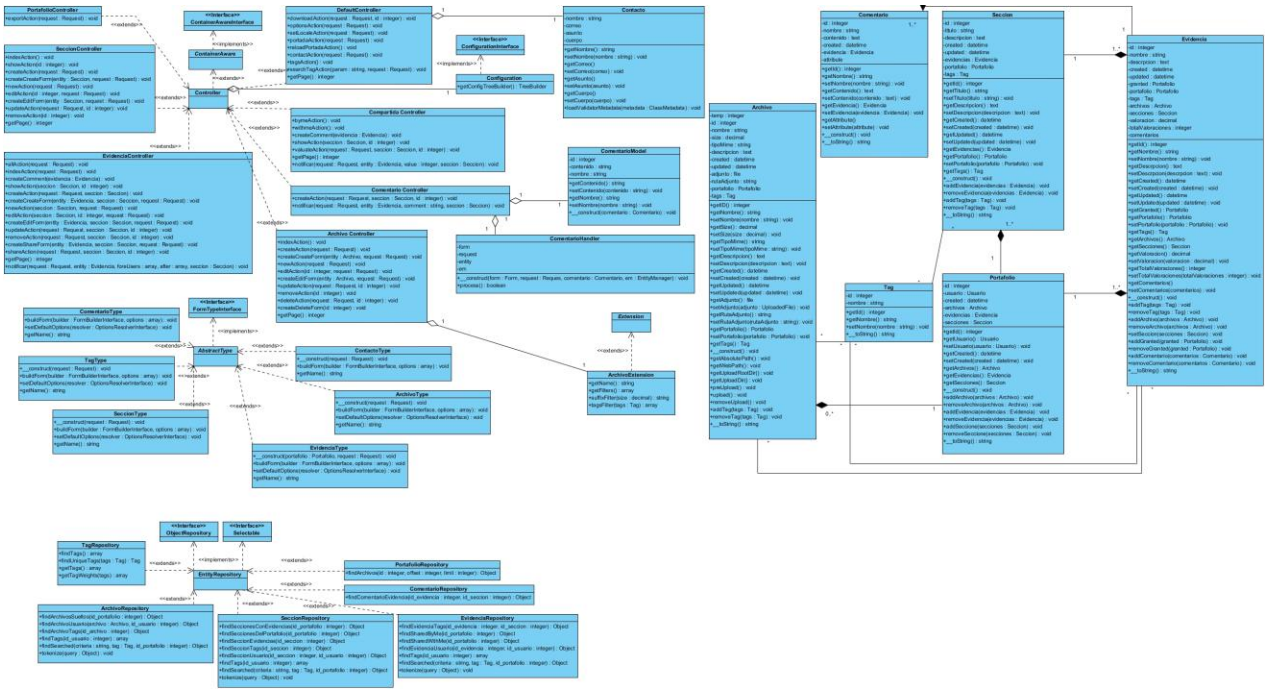


Ilustración 25. Diagrama de Clases del Sistema. Paquete PortafolioBundle.

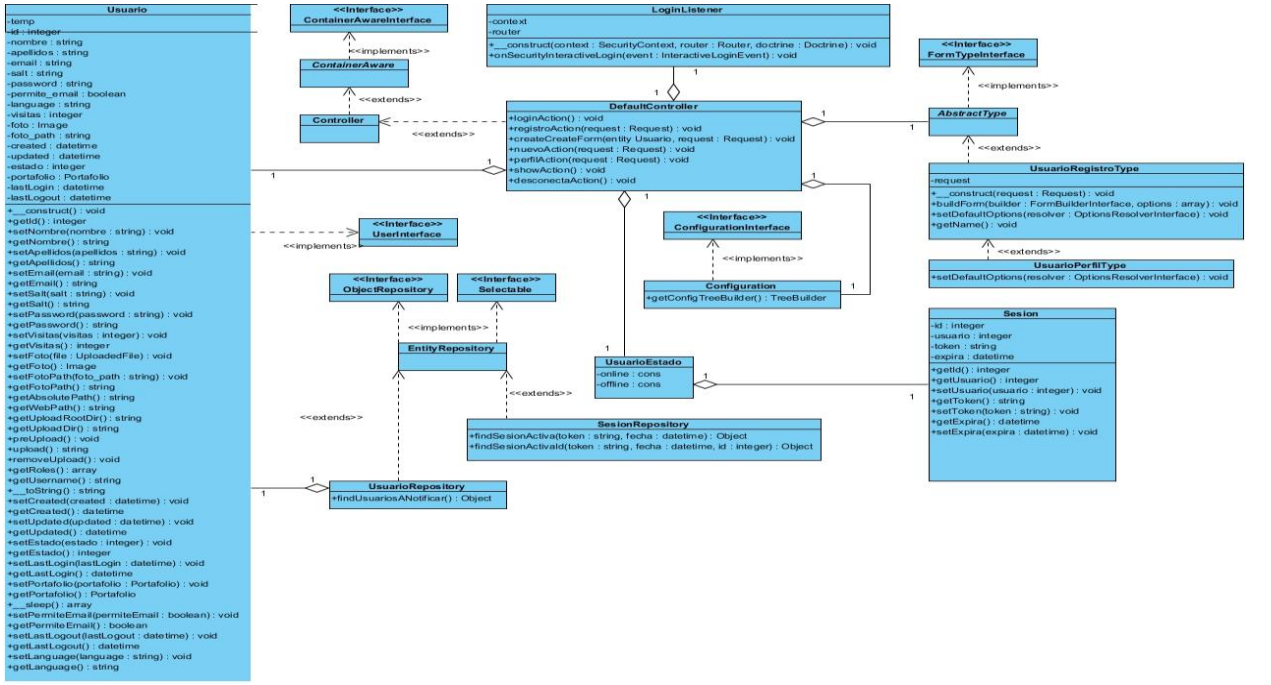


Ilustración 26. Diagrama de Clases del Sistema. Paquete UsuarioBundle.

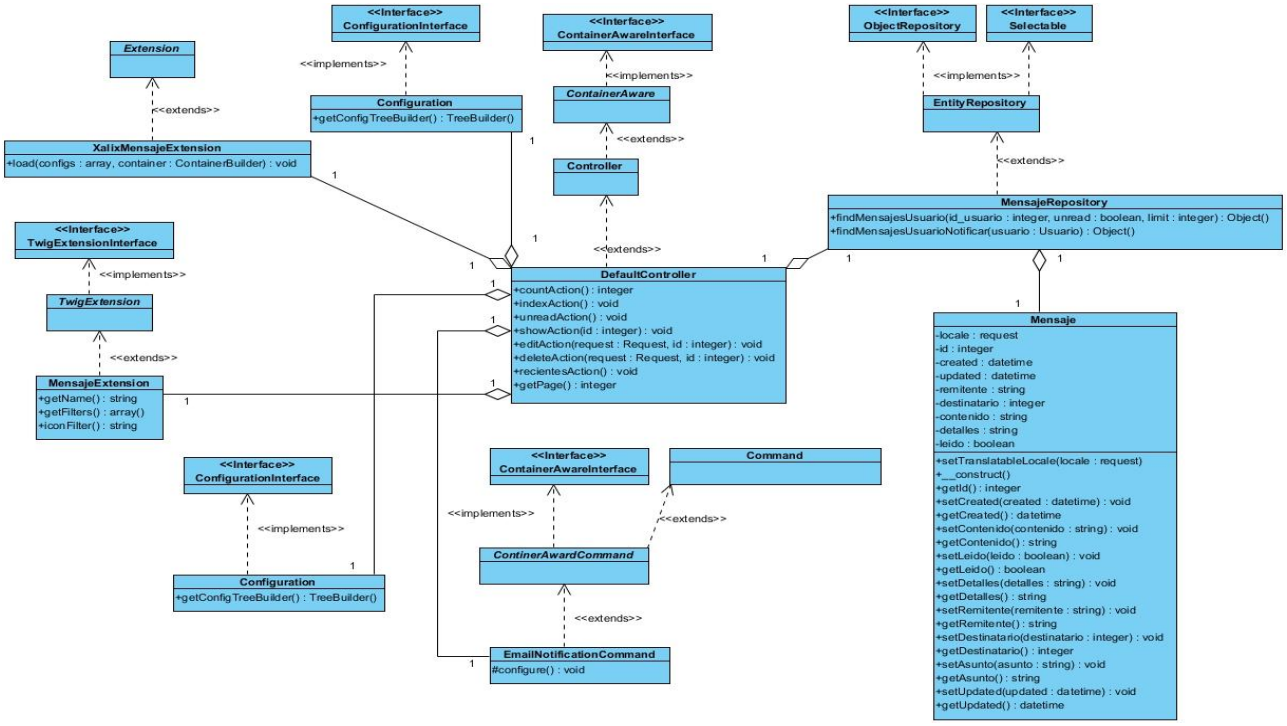


Ilustración 27. Diagrama de Clases del Sistema. Paquete MensajeBundle.

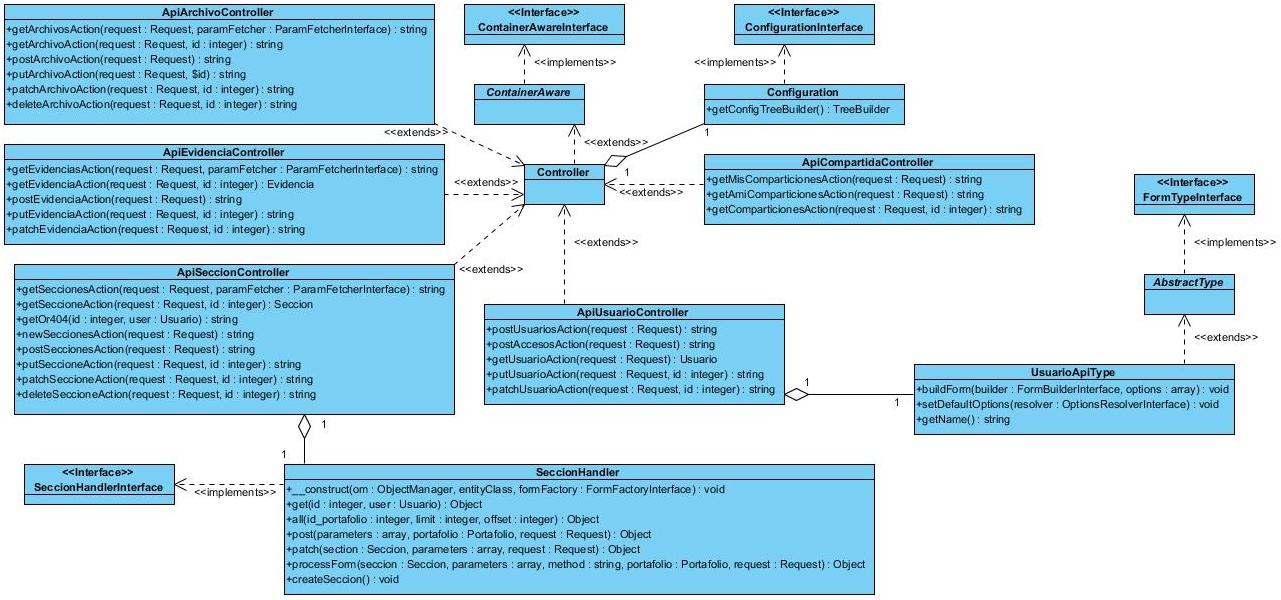


Ilustración 28. Diagrama de Clases del Sistema. Paquete ServiciosBundle.

Anexo 14: Especificación de HU

HU2 Autenticar usuario en el sistema

RF relacionados:

- **RF 2.** Autenticar usuario en el sistema.

Historia de usuario	
No.: 2 **Nombre:** Autenticar usuario en el sistema	
Usuario: Administrador, Personal	
Prioridad en el negocio: Media	**Nivel de complejidad:** Medio
Estimación: 2d	**Iteración asignada:** 1
Descripción: el usuario puede acceder al sistema introduciendo su correo electrónico y una contraseña.	
Información adicional (observaciones): da cumplimiento al *RF 2. Autenticar usuario en el sistema.*	

HU3 Gestionar archivos personales

RF relacionados:

- **RF 3.1.** Visualizar archivos personales.
- **RF 3.2.** Crear un nuevo archivo personal.
- **RF 3.3.** Editar archivo personal.
- **RF 3.4.** Eliminar archivo personal.
- **RF 3.5.** Ver archivo personal.

Historia de usuario
No.: 3 **Nombre:** Gestionar archivos personales

Usuario: Administrador, Personal	
Prioridad en el negocio: Media	**Nivel de complejidad:** Medio
Estimación: 3d	**Iteración asignada:** 1
Descripción: el usuario puede almacenar en el sistema sus archivos personales, para incluirlos posteriormente en las evidencias. El sistema brinda la posibilidad de visualizar todos los archivos creados por el usuario, además de poder editar el que estime conveniente. Puede ver el archivo que se almacenó, así como eliminar aquellos que desee.	
Información adicional (observaciones): da cumplimiento al *RF 3. Gestionar archivos personales.*	

HU4 Gestionar evidencias personales

RF relacionados:

- **RF 4.1.** Visualizar evidencias personales.
- **RF 4.2.** Incluir nueva evidencia personal.
- **RF 4.3.** Editar parámetros de la evidencia.
- **RF 4.4.** Eliminar evidencia personal.
- **RF 4.5.** Visualizar detalles de la evidencia.
- **RF 4.6.** Ver archivo(s) personal(es) asociado(s) a la evidencia.

Historia de usuario	
No.: 4 **Nombre:** Gestionar evidencias personales	
Usuario: Administrador, Personal	
Prioridad en el negocio: Alta	**Nivel de complejidad:** Alta
Estimación: 5d	**Iteración asignada:** 2
Descripción: el sistema brinda la posibilidad de incluir, eliminar y consultar las evidencias personales que forman parte de la trayectoria de cada usuario. Además permite que el usuario modifique los parámetros de la evidencia que desee, mostrando el progreso reflexivo sobre su desarrollo personal. Puede ver un listado de las evidencias que ha creado y además ver los detalles de cada una.	
Información adicional (observaciones): da cumplimiento al *RF 4. Gestionar evidencias personales.*	

HU5 Compartir evidencia personal

RF relacionados:

- **RF 5.1.** Mostrar un listado de los usuarios registrados en el sistema a los que se pueda compartir una evidencia.
- **RF 5.2.** Compartir evidencia personal.

Historia de usuario	
No.: 5 **Nombre:** Compartir evidencia personal	
Usuario: Administrador, Personal	
Prioridad en el negocio: Alta	**Nivel de complejidad:** Alta
Estimación: 3d	**Iteración asignada:** 2
Descripción: el usuario tiene la posibilidad de mostrar su desarrollo personal compartiendo sus evidencias personales con los demás usuarios registrados en el sistema. Puede seleccionar uno o varios usuarios a los que compartir la evidencia.	
Información adicional (observaciones): da cumplimiento al *RF 5. Compartir evidencia personal.*	

HU6 Consultar evidencias compartidas

RF relacionados:

- **RF 6.1.** Mostrar las evidencias compartidas por el usuario.
- **RF 6.2.** Mostrar las evidencias que le han sido compartidas al usuario.
- **RF 6.3.** Valorar evidencia compartida.

Historia de usuario	
No.: 6 **Nombre:** Consultar evidencias compartidas	
Usuario: Administrador, Personal	
Prioridad en el negocio: Alta	**Nivel de complejidad:** Alta
Estimación: 4d	**Iteración asignada:** 2
Descripción: el usuario puede visualizar las evidencias que ha compartido y las que le han compartido a él. Además puede emitir una valoración a aquellas evidencias que le han sido compartidas.	
Información adicional (observaciones): da cumplimiento al *RF 6. Consultar evidencias compartidas.*	

HU7 Gestionar secciones

RF relacionados:

- **RF 7.1.** Visualizar secciones creadas.
- **RF 7.2.** Incluir nueva sección.
- **RF 7.3.** Editar datos de la sección.
- **RF 7.4.** Visualizar datos de la sección.
- **RF 7.5.** Eliminar sección.

Historia de usuario	
No.: 7 **Nombre:** Gestionar secciones	
Usuario: Administrador, Personal	
Prioridad en el negocio: Alta	**Nivel de complejidad:** Alta
Estimación: 5d	**Iteración asignada:** 1
Descripción: el usuario tiene la posibilidad de crear sus secciones donde pueda incluir evidencias personales y la información que desee relacionada con su trayectoria. Además el sistema brinda la posibilidad de modificar y visualizar los datos de cada sección. Las secciones pueden ser eliminadas y con esto toda la información que se había almacenado dentro de ella.	
Información adicional (observaciones): da cumplimiento al *RF 7. Gestionar secciones.*	

HU8 Consultar notificaciones personales

RF relacionados:

- **RF 8.1.** Mostrar un listado de notificaciones.
- **RF 8.2.** Eliminar notificaciones.
- **RF 8.3.** Ver detalles de la notificación.

Historia de usuario
No.: 8 **Nombre:** Consultar mis evidencias personales

Usuario: Administrador, Personal	
Prioridad en el negocio: Media	**Nivel de complejidad:** Media
Estimación: 2d	**Iteración asignada:** 4
Descripción: el usuario tiene la posibilidad de visualizar todas las notificaciones que le han sido emitidas a través de acciones realizadas en el portafolio.	
Información adicional (observaciones): da cumplimiento al *RF 8. Consultar notificaciones personales.*	

HU9 Exportar datos del portafolio

RF relacionados:

- **RF 9.** Exportar datos del portafolio.

Historia de usuario	
No.: 9 **Nombre:** Exportar datos del portafolio	
Usuario: Administrador, Personal	
Prioridad en el negocio: Alta	**Nivel de complejidad:** Alta
Estimación: 6d	**Iteración asignada:** 4
Descripción: el usuario puede exportar todo el contenido almacenado dentro de su portafolio personal, en un formato especificado bajo el estándar Leap2A.	
Información adicional (observaciones): da cumplimiento al *RF 9. Exportar datos del portafolio.*	

HU10 Gestionar comentarios

RF relacionados:

- **RF 10.1.** Incluir comentario en evidencia.
- **RF 10.2.** Ver cometarios de la evidencia.

Historia de usuario	
No.: 10 **Nombre:** Gestionar comentarios	
Usuario: Administrador, Personal	
Prioridad en el negocio: Baja	**Nivel de complejidad:** Baja
Estimación: 2d	**Iteración asignada:** 4
Descripción: el usuario puede incluir comentarios a las evidencias compartidas y a sus propias evidencias creadas. Además visualizar todos los comentarios realizados a una evidencia compartida.	
Información adicional (observaciones): da cumplimiento al *RF 10. Gestionar comentarios.*	

HU11 Brindar servicios web

RF relacionados:

- **RF 11.** Brindar servicios web.

Historia de usuario
No.: 11 **Nombre:** Brindar servicios web
Usuario: Herramienta portafolio

Prioridad en el negocio: Alta	**Nivel de complejidad:** Alta
Estimación: 13d	**Iteración asignada:** 3
Descripción: la herramienta debe ser capaz de brindar una serie de servicios, de acuerdo a las funcionalidades que se especifiquen, dando la posibilidad a otros sistemas informáticos hacer uso de estos servicios.	
Información adicional (observaciones): da cumplimiento al *RF 11. Brindar servicios web.*	

HU12 Mensajes del sistema

RF relacionados:

- **RF 12.1.** Enviar mensajes de contacto.
- **RF 12.2.** Enviar mensaje de notificación del portafolio.

Historia de usuario	
No.: 12 **Nombre:** Mensajes del sistema	
Usuario: Todos	
Prioridad en el negocio: Media	**Nivel de complejidad:** Media
Estimación: 4d	**Iteración asignada:** 4
Descripción: el sistema debe permitirle al usuario la posibilidad de enviar mensajes de contacto a los administradores de la aplicación y esta a su vez debe permitir enviar mensajes de notificaciones a los usuarios.	
Información adicional (observaciones): da cumplimiento al *RF 12. Mensajes del sistema.*	

HU13 Sistema de búsqueda del portafolio

RF relacionados:

- **RF 13.1.** Filtrar búsqueda de archivos.
- **RF 13.2.** Filtrar búsqueda de usuarios.
- **RF 13.3.** Búsqueda global.
- **RF 13.4.** Búsqueda por etiquetas.

Historia de usuario	
No.: 13 **Nombre:** Sistema de búsqueda del portafolio	
Usuario: Administrador, Personal	
Prioridad en el negocio: Media	**Nivel de complejidad:** Media
Estimación: 3d	**Iteración asignada:** 2
Descripción: el sistema debe permitirle al usuario la posibilidad de realizar búsquedas de elementos en cualquier espacio de la aplicación.	
Información adicional (observaciones): da cumplimiento al *RF 13. Sistema de búsqueda del portafolio.*	

HU14 Internacionalización del idioma

RF relacionados:

- **RF 14.1.** Internacionalización del idioma del portafolio.

Historia de usuario	
No.: 14 **Nombre:** Internacionalización del idioma	
Usuario: Todos	
Prioridad en el negocio: Baja	**Nivel de complejidad:** Baja
Estimación: 1d	**Iteración asignada:** 4
Descripción: el sistema debe permitirle al usuario la posibilidad de cambiar el idioma en cualquier espacio de trabajo.	
Información adicional (observaciones): da cumplimiento al *RF 14. Internacionalización del idioma del portafolio.*	

HU15 Administración de usuarios del sistema

RF relacionados:

- **RF 15.1.** Visualizar usuarios registrados en el sistema.
- **RF 15.2.** Eliminar usuarios del sistema.

Historia de usuario	
No.: 15 **Nombre:** Administración de usuarios del sistema	
Usuario: Administrador	
Prioridad en el negocio: Alta	**Nivel de complejidad:** Media
Estimación: 2d	**Iteración asignada:** 3
Descripción: el sistema debe permitirle al administrador visualizar y eliminar la información personal de cada usuario del sistema.	
Información adicional (observaciones): da cumplimiento al *RF 15. Administración de usuarios del sistema.*	

Anexo 15: Pruebas unitarias

```
public function testValidarNombre()
{
    $user = new Usuario();

    //nombre correcto
    $user->setNombre('John');
    $this->assertRegExp("/^([a-zA-ZáéíóúÁÉÍÓÚñÑ ])*$/", $user->getNombre());

    //nombre con formato incorrecto
    $user->setNombre('J0#n');

    $listaErrores = $this->validator->validate($user);
    $this->assertGreaterThan(0, $listaErrores->count(), 'Formato incorrecto.'
    );
    $error = $listaErrores[0];
    $this->assertRegExp("/This value is not valid./", $error->getMessage());

    //nombre en blanco
    $user->setNombre('');
    $listaErrores = $this->validator->validate($user);
    $this->assertGreaterThan(0, $listaErrores->count(), 'Formato incorrecto.'
    );

    $error = $listaErrores[1];
    $this->assertRegExp("/This value should not be blank./", $error->getMessage());
}
```

Ilustración 29. Prueba unitaria - Validar nombre. Entidad Usuario.

```php
public function testGet()
{
    $id = 1;
    $seccion = $this->getSeccion();
    $this->repository->expects($this->once())->method('find')
            ->with($this->equalTo($id))
            ->will($this->returnValue($seccion));

    $this->seccionHandler = $this->createSeccionHandler($this->om, static::SECCION_CLASS, $this->formFactory);

    $this->seccionHandler->getSeccion($id);
}
```

Ilustración 30. Prueba unitaria - Obtener Sección por ID. Servicios Web.

```php
public function testAll()
{
    $offset = 1;
    $limit = 2;

    $secciones = $this->getSecciones(2);
    $this->repository->expects($this->once())->method('findBy')
            ->with(array(), null, $limit, $offset)
            ->will($this->returnValue($secciones));

    $this->pageHandler = $this->createSeccionHandler($this->om, static::SECCION_CLASS, $this->formFactory);

    $all = $this->pageHandler->getAllSecciones($limit, $offset);

    $this->assertEquals($secciones, $all);
}
```

Ilustración 31. Prueba unitaria - Obtener Todas las Secciones. Servicios Web.

```
ned@ned-probook:/var/www/testing/Better$ phpunit -c app
PHPUnit 3.6.10 by Sebastian Bergmann.

Configuration read from /var/www/testing/Better/app/phpunit.xml.dist

......................

Time: 1 second, Memory: 9.50Mb

OK (22 tests, 112 assertions)
```

Ilustración 32. Resultado de las pruebas unitarias utilizando ***phpunit***.