Universidad de las Ciencias Informáticas, Centro FORTES, Facultad 4

Desarrollo de una capa de servicios para las herramientas que componen el Observatorio Tecnológico del Grupo de Gestión de la Información y el Conocimiento

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

Autor: Randy La Rosa Alvarez

Tutor(es): Ing. Maikel Aparicio Reytor
Lic. Yenieris Morales Norchales

La Habana, Junio de 2012

Declaración de autoría

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

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

Autor

_____

Randy La Rosa Alvarez

Tutor

_____

Ing. Maikel Aparicio Reytor

Cotutor

_____

Lic. Yenieris Moyares Norchales

Resumen

Desde el surgimiento de las aplicaciones informáticas, las organizaciones las han utilizado como herramienta para la realización de sus procesos internos. Gran parte de estas desarrolladas de manera independiente y/o sobre tecnologías incompatibles. Con el constante cambio del negocio en las organizaciones, muchas aplicaciones heredadas no son capaces de realizar los procesos más complejos de las organizaciones por sí solas, por lo que se hace necesaria su integración. En la Universidad de las Ciencias Informáticas, específicamente en el Centro FORTES, el Grupo de Gestión de la Información y el Conocimiento (GIC) cuenta con un Observatorio Tecnológico. Dentro de las principales actividades del Grupo está la gestión y socialización de las fuentes de información. Para llevar a cabo este proceso se apoyan en un conjunto de aplicaciones independientes y sin comunicación entre ellas. Esto trae consigo que exista una descentralización y ralentización en el proceso, así como un cierto nivel de complejidad en el trabajo con las herramientas. El objetivo de esta investigación es desarrollar una capa de servicios que permita integrar estas herramientas al Portal del Observatorio. Se emplearon los métodos de investigación científica analítico – sintético e histórico – lógico. Además se utilizó la metodología de desarrollo WSIM-XP, destinada para Servicios Web, y el lenguaje de programación PHP. Como resultado se obtuvo la capa de Servicios Web que permitió integrar las herramientas que componen el observatorio con el portal, contribuyendo de esta forma al proceso de gestión y socialización de las fuentes de información gestionadas por los miembros del Grupo GIC.

Palabras claves: aplicaciones informáticas, integración de aplicaciones, servicios web.

Índice

Introducción

Desde los inicios del surgimiento de las computadoras, las aplicaciones informáticas se han convertido en una necesidad, debido a su gran utilización. Con el transcurso de los años, el desarrollo del software fue progresando, las aplicaciones se fueron mejorando, ganando en complejidad, independencia y en gran parte de los casos en calidad. Disímiles organizaciones y empresas acogieron las aplicaciones informáticas con el objetivo de mejorar su proceso productivo, logrando grandes resultados en cuanto a la agilización de la producción y la toma de decisiones.

En la actualidad la mayoría de las empresas poseen un conjunto muy variado de aplicaciones informáticas, herramientas y sistemas de información y control, utilizados para apoyar sus diferentes procesos de negocios. Gran parte de estos sistemas han sido desarrollados en momentos diferentes y usando tecnologías de software heterogéneas y en muchos casos incompatibles. Es común que estos sistemas operen en equipos de computación de diferentes tecnologías y se ejecuten en sistemas operativos diferentes.

Según Montilva en su informe del año 2005 titulado *Automatización de Sistemas e Integración de Software*: “La falta de comunicación entre estos sistemas, la ausencia de reutilización de funcionalidades y la fragmentación de las áreas de proceso debido a la independencia, heterogeneidad e incompatibilidad de los sistemas trajo consigo la necesidad de integrarlos. Para la solución de este problema han surgido en la última década, propuestas que van desde la integración a través de una interfaz gráfica común, hasta llegar a la integración de los procesos de negocios inter-empresa o intra-empresas basados en las nuevas Tecnologías de Información y Comunicaciones (TICs)”. (1)

Para resolver el problema de la integración de software han surgidos diferentes enfoques, entre los cuales se destacan la Integración de Aplicaciones Empresariales (conocidas como EAI: Enterprise Application Integration). La EAI es el proceso de integrar múltiples aplicaciones desarrolladas independientemente, que utilizan tecnología incompatible y/o son gestionadas de forma independiente, permitiendo que se comuniquen e intercambien datos entre sí, además de permitir la reutilización de sus funcionalidades. Uno

de los principales objetivos de la EAI es proporcionar acceso transparente a la amplia gama de aplicaciones que existen en una organización. Las características más importantes de esta tecnología es que se utiliza para la integración de aplicaciones a otras aplicaciones y proporciona un enfoque de integración orientado a procesos.

La integración de aplicaciones dentro de las empresas es un tema en desarrollo en los últimos años. Pero no solo es necesaria la integración de las aplicaciones internas en una organización; sino que debido a la forma en la que se han desarrollado las TICs y al uso del protocolo HTTP[1] por las organizaciones, han pasado a ser parte de las tendencias tecnológicas.

Tendencias de las que nuestro país no está ajeno. En Cuba varias son las experiencias en cuanto a la integración de aplicaciones; una de estas constituyó el proceso de informatización llevado a cabo en la Contraloría General de la República de Cuba; donde se identificó que existía versatilidad de tecnologías y ausencia de comunicación entre los sistemas existentes. Por lo que se realizó un proyecto informático que requirió la integración entre los sistemas que conformaban la solución.

También en proyectos para la automatización de procesos internos de la Universidad de las Ciencias Informáticas (UCI) de gestión académica (proyecto Akademos), y gestión de la residencia, se han identificado procesos donde era necesaria la reutilización de funcionalidades, por lo que se acudió a la integración de aplicaciones para satisfacer las necesidades identificadas.

El Centro de Tecnologías para la Formación (FORTES) perteneciente a la UCI dispone del Grupo de Gestión de la Información y el Conocimiento (GIC), el cual cuenta con un Observatorio Tecnológico que está compuesto por una serie de aplicaciones web tales como: Open Journal System (OJS[2]), Open Harvester System (OHS[2]), Open Conference System (OCS[2]) y Alfresco. Además se espera contar en un futuro con un portal web desarrollado en el CMS[3] Drupal, el cual brindará los servicios del observatorio. Este portal será uno de los principales sistemas que utilizarán los miembros del Grupo para la socialización y gestión de las fuentes de información. Actualmente se encuentra en fase de construcción. Las herramientas anteriormente mencionadas les permiten a los miembros del Grupo el almacenamiento,

[1] Hypertext Transfer Protocol.
[2] http://pkp.sfu.ca/
[3] Content Manager System.

gestión, consulta y recuperación (de manera independiente), de las diferentes fuentes de información relacionadas con las áreas temáticas que desarrolla e investiga el centro.

Las herramientas OJS, OHS y OCS son aplicaciones web desarrolladas por el Public Knowledge Project[2]. Cada una tiene funcionalidades diferentes que son utilizadas por los miembros del Grupo GIC para realizar el proceso de gestión de las fuentes de información en la entidad. Dicho proceso de gestión implica la ejecución de forma independiente de las herramientas, de modo que, al estar separadas, el trabajo con las mismas se hace de diferentes maneras, lo que trae consigo un cierto grado de dificultad y ralentización en la búsqueda, recuperación y gestión de la información. Debido a la independencia y falta de comunicación entre las aplicaciones se dificulta el control íntegro de las fuentes gestionadas, provocando descentralización del proceso, lo que trae como consecuencia posible pérdida de las fuentes de información gestionadas. Además actualmente el Grupo no cuenta con una aplicación integral que permita realizar todo el proceso de gestión de las fuentes desde un solo sistema, o al menos facilite el proceso.

A partir de los planteamientos anteriores se evidencia como **problema a resolver**: ¿Cómo integrar las herramientas que componen el Observatorio Tecnológico con el portal?

Por lo que se plantea como **objetivo general**: Desarrollar una capa de Servicios Web que permita integrar las herramientas que componen el Observatorio Tecnológico del Grupo de Gestión de la Información y el Conocimiento, del Centro FORTES, con el portal web del observatorio.

Por ende el **objeto de estudio** es el proceso de desarrollo de Servicios Web para la integración de aplicaciones. El **campo de acción** está enmarcado en el proceso de desarrollo de Servicios Web a las aplicaciones que componen el Observatorio Tecnológico del Grupo de Gestión de la Información y el Conocimiento del Centro FORTES.

Para dar cumplimiento al objetivo general se plantean los siguientes **objetivos específicos**:

- Caracterizar los enfoques teórico-conceptuales de la integración de aplicaciones y los Servicios Web.

- Diseñar los Servicios Web necesarios para la integración de las herramientas utilizadas por el Grupo GIC con el portal.
- Implementar los Servicios Web.
- Validar los Servicios Web implementados.

Para dar cumplimiento a estos objetivos fueron trazadas las siguientes **tareas**:

- Sistematización de los referentes teóricos que sustentan la integración de aplicaciones y los Servicios Web.
- Caracterización de las aplicaciones que componen el Observatorio Tecnológico.
- Selección de la metodología y las herramientas de desarrollo a utilizar.
- Identificación de las funcionalidades a implementar como servicios.
- Diseño de los Servicios Web para las aplicaciones que componen el observatorio.
- Implementación de los Servicios Web previamente diseñados.
- Validación de los Servicios Web implementados.

Se tiene como **idea a defender**: El desarrollo de una capa de Servicios Web permitirá la integración de las herramientas que componen el Observatorio Tecnológico del Grupo de Gestión de la Información y el Conocimiento, del Centro FORTES, con el portal.

Los resultados obtenidos son:

- Una capa de servicios que permite exponer las principales funcionalidades de las herramientas que componen el observatorio como Servicios Web.
- La integración de las herramientas que componen el observatorio con el portal.
- Una gestión centralizada de las fuentes de información analizadas por el Grupo.

El presente trabajo consta de tres capítulos formados por epígrafes tal como se describe a continuación:

En el primer capítulo, "**Fundamentación teórica**", se analizan los principales conceptos de la integración de aplicaciones y Servicios Web abordados por diferentes autores. Se plasman las ventajas y desventajas del empleo de los Servicios Web como técnica de integración de aplicaciones. Además se realiza una

breve descripción de las herramientas que componen el observatorio. Por último se hace una caracterización de las metodologías, tecnologías y herramientas que posibilitan el desarrollo de la propuesta.

En el segundo capítulo, "**Planificación y diseño de la solución"**, se definen y describen las características de la solución propuesta. Se identifican los requerimientos que se deben tener en cuenta para la construcción de los Servicios Web y se muestran los artefactos relevantes generados en el diseño de la solución.

El tercer capítulo, "**Implementación y prueba**", ilustra el diseño de los Servicios Web enfocándose en la implementación de los mismos y tomando como base para ello, los resultados obtenidos en los capítulos anteriores. Además se valida la investigación a partir de casos de pruebas realizados.

Los métodos científicos que se utilizaron para el desarrollo de la investigación fueron:

Métodos Teóricos:

Histórico-Lógico: Se utilizó durante el estudio de las tendencias de integración de aplicaciones, así como en el análisis de las experiencias en Cuba sobre la integración mediante Servicios Web.

Analítico-Sintético: Se utilizó en el estudio de los aspectos teóricos referentes a la integración de aplicaciones y a los Servicios Web; en la selección de la metodología, lenguajes de programación y las herramientas de desarrollo a utilizar. También se hizo uso de este método durante el análisis de la arquitectura de las herramientas que componen el observatorio.

Capítulo 1: Fundamentación teórica

Introducción

En la actualidad las aplicaciones de software son de suma importancia para las organizaciones debido a que la mayoría de sus procesos y actividades dependen de sus sistemas de información. El constante cambio del negocio en las organizaciones aumenta la complejidad de sus procesos, imposibilitando su gestión por las aplicaciones existentes de forma individual, lo cual potencia la creación de tecnologías que satisfagan las nuevas necesidades. Una de estas son los Servicios Web que permiten la comunicación entre diversas aplicaciones y la reutilización de funcionalidades.

1.1 Aplicaciones informáticas

Una aplicación, en informática, es un tipo de programa informático diseñado para facilitar al usuario la realización de un determinado tipo de trabajo.

La definición dada por Salazar Pacios en su tesis para optar por el título de Ingeniero en Ciencias Informáticas del año 2009 referente a aplicación informática es: “componente lógico de un computador digital que permite a los usuarios realizar una o varias tareas específicas”. (2)

En dependencia de su entorno de ejecución las aplicaciones informáticas se pueden dividir en dos grandes grupos, aplicaciones de escritorio y aplicaciones web.

1.1.1 Aplicaciones de escritorio

Las aplicaciones de escritorio son: “aquellas que se instalan en el sistema operativo del usuario y almacenan sus datos en ficheros locales. Dependen en gran medida de la plataforma (Sistema Operativo) del usuario y es relativamente complejo portarlas a plataformas diferentes. Su capacidad de respuesta es muy rápida y posibilitan un alto nivel de interactividad”. (2)

1.1.2 Aplicaciones web

Existen varias definiciones de aplicaciones web, entre las cuales está la definida en el Glosario del sitio Hooping.net como: “Aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través

de Internet o de una intranet mediante un navegador. Aplicación que se codifica en un lenguaje soportado por los navegadores web en la que se confía la ejecución al navegador". (3)

Por su parte el Ingeniero en Informática Luján Mora expone en su libro titulado "Programación de aplicaciones web Historia, principios básicos y clientes web" que: "*una aplicación web se puede definir como una aplicación en la cual un usuario por medio de un navegador realiza peticiones a una aplicación remota accesible a través de Internet (o a través de una intranet) y que recibe una respuesta que se muestra en el propio navegador*". (4)

El autor de la presente investigación coincide con la definición brindada anteriormente por Salazar Pacios referente a aplicación de escritorio. En cuanto a la definición de aplicación web dada por Luján Mora se está totalmente de acuerdo, aunque se considera que se debe adicionar que son aplicaciones que están publicadas en servidores web. Generalmente la información que gestionan las aplicaciones web es almacenada en servidores de bases de datos y pueden ser usadas por cualquier cliente independientemente del Sistema Operativo del mismo.
Tanto las aplicaciones de escritorio como las web tienen como objetivo hacerle más fácil el trabajo a los usuarios que las utilicen. En ocasiones utilizarlas de manera individual no satisface las necesidades de las organizaciones, por lo que se utilizan en conjunto, de forma integrada entre ellas o con otros sistemas.

1.2 Integración de aplicaciones

La integración de aplicaciones consiste en reemplazar la variedad de aplicaciones independientes por una centralizada y ordenada, utilizando un sistema central. Permite a diferentes aplicaciones de una organización trabajar de manera conjunta, compartir datos y procesos de negocio.

La integración de aplicaciones proporciona un conjunto de ventajas como:

- Acceso a la información en tiempo real.
- Acceso a la información desde una misma interface.
- Obtención y distribución de los datos en distintas aplicaciones desde un mismo sistema.
- Facilidad de uso por medio de interfaces web.

- Desarrollo de nuevas funcionalidades, independientes del sistema operativo, de lenguajes de programación y de soporte de datos.

1.2.1 Tendencias sobre la integración de aplicaciones

Gran parte de las organizaciones vigentes poseen multitud de aplicaciones informáticas con funcionalidades muy específicas, asociadas frecuentemente a las áreas de negocio y usualmente fabricadas bajo distintas plataformas y lenguajes de programación. Estas aplicaciones han constituido el resultado de una forma de desarrollo de software dirigida por la información. “Cada vez que se necesitaba procesar ciertos datos, se creaba una aplicación aislada que los calculaba” (5). Los sistemas Enterprise Resource Planning (ERP) surgen para resolver este problema planteando sistemas de información integrales que envolvían todas las áreas de negocio. No obstante, estos sistemas son caros y en muchas ocasiones presentan dificultades a la hora de adaptarlos a una organización concreta. En la actualidad, son numerosas las organizaciones que disponen de una gran gama de sistemas heredados diseñados para ejecutarse de forma aislada, que deberían ser integrados para llevar a cabo eficientemente sus procesos de negocio.

La integración de las aplicaciones no solo es necesaria en el interior de una organización, sino que, debido a la forma en la que han evolucionado los mercados, gracias al establecimiento intensivo de Internet y al uso del protocolo HTTP en las organizaciones, se ha convertido en un tema importante la integración de las aplicaciones de una organización con las de otras instituciones. Este proceso de integración de aplicaciones suele ser muy costoso, con un nivel de complejidad alto, una gran demanda de tiempo y suele requerir soluciones informáticas distribuidas.

Adolfo de Soto en su artículo “*Nuevas Tendencias en Sistemas de Información: Procesos y Servicios*” del año 2006 plantea que: “Para lograr la construcción rápida de sistemas distribuidos se necesita una nueva capa software que realice la abstracción de la comunicación entre sistemas heterogéneos, esta capa se coloca entre las aplicaciones distribuidas y los servicios de red, los sistemas operativos y el hardware de comunicaciones” (5). Esta nueva capa se denomina middleware y pretende resolver el problema de la comunicación entre procesos de forma independiente del lenguaje y de la plataforma hardware o software subyacente. La definición formal de middleware dada en el sitio oficial de Middleware.org es "software de

conectividad que consiste en un conjunto de servicios, que permiten interactuar a múltiples procesos que se ejecutan en distintas máquinas a través de la red". (6)

Existen multitud de soluciones de middleware para integrar aplicaciones informáticas dentro de una organización, como sistemas RPC (Remote Procedure Call), monitores TP (Transaction Processing), intermediarios de objetos (Object Brokers) o Middleware orientados a mensajes. En los últimos años se ha producido un uso masivo de la web en las organizaciones, lo que constituyó un paso importante en la integración de aplicaciones. Esto trajo protocolos de interacción estándar, como el HTTP, y formatos de datos, como el XML, que fueron adoptados rápidamente por las compañías, creando por tanto un estándar donde establecer una infraestructura de middleware común que reduce la heterogeneidad. En este contexto toman fuerza los Servicios Web, los cuales según Vicente Pelechano, "*proporcionan la plataforma tecnológica ideal para conseguir la completa integración de los procesos de negocio de una organización con diferentes organizaciones*". "*Los Servicios Web prometen ser el mecanismo adecuado para la implementación de las Arquitecturas Orientadas a Servicios para sistemas de información integrados y distribuidos*". (7)

1.2.2 Experiencias nacionales sobre la integración de aplicaciones mediante Servicios Web

Numerosas son las experiencias de integración de aplicaciones web mediante Servicios Web a nivel mundial. Nuestro país no está ajeno a estas experiencias. En muchos de los grandes y medianos proyectos como el ERP Cubano, SCADA[4] Guardián del Alba, así como en proyectos para la automatización de los procesos de gestión académica y de la residencia de la UCI, se han provisto funcionalidades como Servicios Web para facilitar los mecanismos de integración con otras aplicaciones y acceso remoto.

Una de estas experiencias constituyó el proceso de informatización llevado a cabo en la Contraloría General de la República de Cuba. Se identificó que existía versatilidad de tecnologías, por lo cual se realizó un proyecto informático donde además de automatizar los procesos, se integraron los sistemas que conformaban la solución.

[4] Speaker builders Computer Aided Design and Analysis tool.

El Sistema Informático de Gestión de Auditoría y Control (SIGAC), desarrollado por la UCI, para gestionar los procesos de la planificación y control de las auditorías; y procesos de otro sistema, llamado Alfresco, para la gestión documental de todos los archivos que dentro de este órgano del estado se manejan.

Durante el desarrollo de este proyecto se analizaron diferentes enfoques de Integración:

- Réplica de Datos.
- Extracción, Transformación y Carga de Datos o ETL.
- Integración de Información Empresarial o EII.
- Integración de Aplicaciones Empresariales o EAI.

"Se utilizó la última técnica que es la más se ajustaba a la situación del ministerio. Para llevar a cabo la integración, se optó por usar los Servicios Web. Alfresco en su arquitectura, ofrece una capa dedicada a exponer Servicios Web permitiendo así interactuar con otras aplicaciones". (8)

"*Esta solución permitió integrar ambas aplicaciones y evitar la fragmentación del negocio automatizado y evitar dentro de la solución brindada la 'isla de información'*" (8). La Contraloría General de la República de Cuba puede tener una mejor visión de sí misma al terminar el proyecto de informatización. El costo en el desarrollo de esta tecnología es bastante bajo por la posibilidad de haber usado tecnologías gratuitas y la técnica EAI de integración de aplicaciones informáticas basadas en Servicios Web.

Otro ejemplo es el sistema SCADA Guardián del ALBA que permite supervisión, control y adquisición de procesos industriales para el control de las instalaciones de la corporación estatal de la República Bolivariana de Venezuela a cargo de la exploración, producción, transporte y comercialización de los hidrocarburos Petróleos de Venezuela Sociedad Anónima (PDVSA). Este sistema expone como Servicios Web funcionalidades relacionadas con la gestión de la información de las alarmas y eventos, para la integración del sistema con terceros.

1.2.2 Características de la integración de aplicaciones

Algunas de las características que debe cumplir la integración de aplicaciones para que su utilización en una organización sea factible son (9):

- Las aplicaciones integradas no deben alterar su funcionamiento después de realizada la integración.
- Las aplicaciones, después de realizado el proceso de integración, deben mantenerse con un nivel de desacoplamiento igual al que tenían antes de la integración.
- La solución de integración no debe generar dependencias entre las aplicaciones integradas, aparte de las ya existentes antes del proceso de integración.
- La solución de integración debe realizarse de acuerdo a las necesidades de la entidad.

La realización de un proceso de integración de aplicaciones no es sencillo, los equipos de desarrollo implicados a menudo se enfrentan a diversas problemáticas, tales como:

- Sistemas aislados, heterogéneos y solapados en funcionalidad.
- Necesidad de realizar procesos que involucran a varios sistemas.
- Información inconsistente en distintos sistemas.
- Costo de implementación de los mecanismos de integración.

De igual manera luego de lograda la integración pueden surgir otros problemas:

- Alta dependencia entre sistemas.
- Alteración en el funcionamiento de los sistemas.

A pesar de las inconvenientes y problemas a los que se pueden enfrentar tanto el equipo involucrado en la integración de aplicaciones como la misma institución, los beneficios obtenidos propiciarán grandes ventajas, como la socialización de la información dispersa entre ellas y la integración de sus funcionalidades. Se puede definir un único entorno de acceso que facilite la realización de los procesos del negocio, permitiendo una toma de decisiones más efectiva para su organización.

Según Gail Corbitt en su libro *Enterprise Integration* del año 2001*:* “la integración de aplicaciones resulta, cada vez más, un factor estratégico a tener en cuenta por las empresas. La serie de ventajas que trae consigo su aplicación son puntos bien recibidos por las mismas”. (10)

Entre las principales ventajas que proporciona la integración de aplicaciones podemos mencionar las siguientes (11):

- Acceso a la información dispersa por toda la organización: En una empresa dotada de una integración adecuada de sus aplicaciones se puede acceder a cualquiera de sus datos. Se reduce el tiempo de búsqueda, acceso y uso de la información global de la empresa.
- Capacidad de administrar la información: Administrar la información global de una empresa es una tarea casi imposible si las aplicaciones no están integradas de forma coherente.
- Sincronización de la información replicada por las distintas aplicaciones: Uno de los problemas más habituales en las empresas con sistemas de información es que esta suele estar muchas veces replicada. Al establecer una integración correcta entre las distintas aplicaciones puede conseguir que la información se propague de forma natural. De esta forma todos los sistemas se encontrarán actualizados con la información correcta y de su origen natural. Es lo que se conoce como SCM[5].
- Integración de procesos y no solo de la información: Cuando la integración llega al nivel de procesos se consigue que series de procesos a lo largo de varias aplicaciones se conviertan en una secuencia de provisión automática. En cierta manera se logra que se convierta en un solo proceso más general. Todos ellos alimentados con información en el momento preciso, y sincronizados de manera que el rendimiento sea el máximo posible.
- Mayor rapidez y efectividad de costos en la adaptación de nuevos requerimientos: La existencia de una infraestructura de integración adecuadamente definida asegura no solo el aprovechamiento de los sistemas existentes, sino que permite nivelar el camino de aquellos elementos que seguro tendrán que integrar. Que el tiempo de adaptación a la oportunidad sea lo más reducido posible facilita el disponer de la ventaja competitiva antes que el resto de empresas, y si además esta adaptación es de un coste bajo se consigue un retorno de la inversión más rápido.

Permite también:

- Introducir nuevas aplicaciones y tecnologías más eficientes.

[5] Supply Chain Management.

- Modificar y automatizar sus procesos de negocio de manera más fácil para cumplir con sus nuevos requisitos.
- Proveer más canales de distribución para su organización.
- Compartir datos entre aplicaciones web por XML o Servicios Web.

El autor de la presente investigación concuerda en que la integración no es un proceso sencillo y puede muchas veces no ser factible. Es necesario realizar previamente un estudio de la organización, de los procesos y de las tecnologías de las que dispone, para analizar la factibilidad de la integración. Un correcto levantamiento de información sobre las tecnologías, las problemáticas existentes y el uso de la técnica de integración que más se adecue a las características de la entidad, les garantiza a la empresa gran parte de los beneficios anteriormente mencionados.

1.2.3 Técnicas de integración

La integración de aplicaciones puede ser llevada a cabo por varias técnicas, entre las más utilizadas están las siguientes:

1. Protocolos de comunicación entre aplicaciones.
2. Plataformas de integración.
3. Integración mediante Servicios Web.

A continuación se realiza una breve descripción de las técnicas anteriormente enunciadas.

Protocolos de comunicación entre aplicaciones

Esta técnica consiste en el empleo de estándares o protocolos como base para lograr una comunicación rápida y segura entre las diversas aplicaciones. Algunos de los más utilizados son (12):

- Arquitectura Común de Intermediarios en Peticiones a Objetos (CORBA): Es un modelo de soporte para la programación distribuida orientada a objetos. Hace posible que los objetos interactúen a través de lenguajes de programación, protocolos de comunicación y plataformas heterogéneas.
- Modelo de Objetos Componentes Distribuidos (DCOM): Es una plataforma de Microsoft para comunicar diferentes aplicaciones. Es utilizada por los desarrolladores para crear componentes de software reutilizables. Los objetos pueden ser creados en distintos lenguajes.

- Protocolo de Acceso Simple a Objetos (SOAP): Provee una forma de comunicación entre aplicaciones que utilizan tecnologías y lenguajes de programación distintos. Un mensaje SOAP es un documento XML[6] que puede transportarse utilizando cualquier protocolo capaz de transmitir texto. Permite la interoperabilidad entre múltiplos entornos, pues, su desarrollo, se basa en estándares existentes.

Plataformas de integración

Tradicionalmente la integración de sistemas se realizaba punto a punto; sin embargo, este enfoque requiere de grandes esfuerzos que se ven incrementados por el número de aplicaciones que se desean integrar.

Como solución a esta problemática surgen las tecnologías llamadas Middleware, que se encargan de gestionar la integración de las aplicaciones a nivel de servicios[7]. El Dr. James Mckeen define Middleware, como "*la capa de software que se coloca entre el usuario y el entorno distribuido, abstrayendo al usuario de la complejidad y la heterogeneidad de las arquitecturas, protocolos, sistemas operativos, lenguajes de programación; es decir, otorga la posibilidad de intercomunicar aplicaciones desarrolladas en distintos lenguajes de programación, sistemas operativos y plataformas*". (13)

Integración mediante Servicios Web

Las Arquitecturas Orientadas a Servicios, también conocidas como SOA por sus siglas en inglés, son una de las tecnologías más utilizadas para la integración de aplicaciones web. "Establecen un marco de diseño para la integración de aplicaciones independientes de manera que desde la red pueda accederse a sus funcionalidades, las cuales se ofrecen como servicios. La forma más habitual de implementarla es mediante Servicios Web, una tecnología basada en estándares e independiente de la plataforma, con la que se puede descomponer aplicaciones monolíticas en un conjunto de servicios e implementar esta funcionalidad en forma modular" (14). Juan A. Brena en su artículo publicado en el sitio desarrolloweb.com afirma que: "Los Servicios Web son la revolución informática de la nueva generación de aplicaciones que trabajan colaborativamente en las cuales el software está distribuido en diferentes servidores". (15)

[6] Extensible Markup Language.

[7] En el desarrollo basado en componentes se pueden identificar tres niveles de integración: El nivel de servicios es cuando las aplicaciones no son capaces de cubrir por sí solas todos los procesos que necesitan las entidades y deben integrarse con otras aplicaciones que las soporten.

Por otra parte la especialista en Dirección estratégica en Tecnologías de la Información y las Comunicaciones de la Universidad Politécnica de Madrid, Liliana M. Arboleda en su publicación *Servicios Web: Distribución e integración*, del año 2004 expone que: "Entre los principales beneficios que se exponen al hablar de los Servicios Web, generalmente se encuentran aquellos que tienen que ver con su granularidad e interoperabilidad, es decir, con la posibilidad de desarrollar componentes de software totalmente independientes que tienen funcionalidad propia, pero que son capaces de exponer tal funcionalidad y compartirla con otros servicios y aplicaciones para lograr crear sistemas más complejos e integrados" . (16)

Los Servicios Web proporcionan una plataforma tecnológica para conseguir la completa integración de los procesos de negocio tanto de una organización con diferentes organizaciones, como de forma interna en la entidad. Como dice Vicente Pelechano "*prometen ser el mecanismo adecuado para la implementación de las Arquitecturas Orientadas a Servicios para sistemas de información integrados y distribuidos*". (7)

Después de un análisis sobre las diferentes técnicas de integración de aplicaciones se decidió utilizar la de Servicios Web, que utilizan el protocolo HTTP en las invocaciones para el intercambio de información, lo que facilita el uso de la infraestructura web existente en la empresa. Asimismo, este intercambio es posible sin agregar más protocolos a los Firewalls[8], lo cual sería necesario si se utiliza algún otro protocolo. Además, las otras tecnologías mencionadas como CORBA o DCOM normalmente son muy complejas (difíciles de implementar) y requieren de una gran cantidad de recursos para funcionar correctamente. Esto implica requerimientos importantes de hardware tanto para el servidor como para el cliente y también sobre la infraestructura de la red de soporte.

En el siguiente epígrafe se describe de forma detallada las características más importantes sobre los Servicios Web, así como las ventajas que proporcionan la utilización de los mismos.

1.3 Servicios Web

Los Servicios Web permiten a las organizaciones el intercambio de datos entre diferentes aplicaciones de distintas fuentes. Estos proporcionan mecanismos de comunicación que permiten presentar información

[8] Software que filtra y bloquea gran parte del tráfico de Internet.

dinámica al usuario. Varios son los autores que han dado su definición referente a Servicios Web, a continuación se mencionan algunas.

Valcárcel y Munilla, en su libro *E-business Colaborativo*, definen a los Servicios Web como: "aplicaciones modulares auto-descriptivas que se pueden publicar, ubicar e invocar desde cualquier punto de red o desde el interior de una red local, basados en estándares abiertos de Internet. No es necesario que el proveedor y el usuario de un Servicio Web tengan el mismo sistema operativo y utilicen el mismo lenguaje de programación, dado que se basan en estándares aceptados plenamente por la industria como, XML, HTML[9] y Simple Mail Transfer Protocol (SMTP, Protocolo para la transferencia de correo)". (17)

Por su parte Dewit, en su libro *ASP.Net*, describe a los Servicios Web como: "una biblioteca de clases accesible vía HTTP. Su interfaz está descrita en XML, de forma estándar e independiente del lenguaje de implementación subyacente". (18)

El término Servicio Web presentado por Saffirio en su blog describe una forma estandarizada de integrar aplicaciones web mediante el uso de XML, SOAP, WSDL y UDDI sobre los protocolos de la Internet. "Uno de los usos principales es permitir la comunicación entre las empresas y entre las empresas y sus clientes. Los Servicios Web permiten a las organizaciones intercambiar datos sin necesidad de conocer los detalles de sus respectivos sistemas de información". (19)

La W3C[10] lo define como: "un sistema de software diseñado para apoyar la interacción interoperable máquina a máquina sobre una red. Tiene una interfaz descrita en un formato procesable por máquina (específicamente Web Services Description Language). Otros sistemas interactúan con el Servicio Web de una manera prescrita por su descripción usando mensajes SOAP, típicamente transmitido a través de HTTP con una serialización XML en conjunción con otras normas relacionadas con la Web". (20)

Para la investigación, se asume el concepto de Saffirio, que se ajusta a la explicación que se desea brindar sobre este tipo de servicios.

[9] Hypertext Markup Language.
[10] World Wide Web Consortion. Comunidad internacional para el desarrollo de estándares de la web.

1.3.1 Estándares y protocolos utilizados en los Servicios Web

Los Servicios Web se acoplan en una capa adicional en las aplicaciones, de tal forma que pueden interaccionar con las demás capas, usando para comunicarse estándares y protocolos. Ejemplo de estos son:

Lenguaje Extensible de Etiquetas, XML

Se entiende por XML como lo define la W3C: "un lenguaje con una importante función en el proceso de intercambio, estructuración y envío de datos en la web. Describe los datos de tal manera que es posible estructurarlos utilizando para ello etiquetas, como lo hace HTML pero que no están predefinidas, delimitando de esta manera los datos, a la vez que favoreciendo la interoperabilidad de los mismos." (21)

Basado en la definición anterior se puede definir que XML es un lenguaje de etiquetas o tags (similar en sintaxis a HTML) desarrollado por la W3C, para permitir la descripción de información contenida en la WWW[11] a través de estándares y formatos comunes. No impone un conjunto de etiquetas, sino sólo unas pocas normas de uso. Se puede decir que XML es la base para toda la tecnología entorno al desarrollo y despliegue de Servicios Web.

Web Services Description Language, WSDL

Benjamín González lo define como: "un dialecto basado en XML sobre el esquema que describe un Servicio Web. Un documento WSDL proporciona la información necesaria al cliente para interaccionar con el Servicio Web. El WSDL es extensible y se pude utilizar para describir, prácticamente, cualquier servicio de red, incluyendo SOAP sobre HTTP e incluso protocolos que no se basan en XML como DCOM (Distribute Component Object Model) sobre UDP[12]". (22)

Según la W3C: "permite definir lo que hace un Servicio Web según la funcionalidad que ofrece. Mediante este lenguaje se representa la interfaz de uso del servicio, lo que tendrán que tener en cuenta otros servicios a la hora de acceder a su funcionalidad". (21)

[11] World Wide Web.
[12] User Datagram Protocol.

Por otro lado, permite que un servicio y un cliente establezcan un acuerdo en lo que se refiere a los detalles de transporte de mensajes y su contenido, a través de un documento procesable por dispositivos. El WSDL representa una especie de contrato entre el proveedor y el que solicita. El Dr. Ignacio García recomienda en su artículo *Servicios Web* utilizar la versión 2.0. (23)

Protocolo Simple de Acceso a Objetos, SOAP

Se trata de un protocolo que permite la interacción entre varios dispositivos y que tiene la capacidad de transmitir información compleja. Los datos pueden ser transmitidos a través de HTTP y SMTP.

Jim Webber, Savas Parastatidis e Ian Robinson en su artículo *Rest in practice* refiriéndose a SOAP expresan: "La especificación SOAP menciona que las aplicaciones deben ser independientes del lenguaje de desarrollo, por lo que, las aplicaciones cliente y servidor pueden estar escritas con HTML, DHTML[13], Java[14], Visual Basic u otras herramientas y lenguajes disponibles. Las peticiones con el uso del protocolo HTTP emplean el comando POST para transmitir información entre el cliente y el servidor". (24)

Es un protocolo de comunicación, basado en el lenguaje XML, simple y extensible. Contiene un formato para el envío de mensajes y la comunicación vía Internet a través de HTTP. Además define cómo dos objetos en procesos diferentes pueden comunicarse por medio de intercambio de datos XML.

Existen varios protocolos específicamente creados para facilitar la comunicación entre aplicaciones como son:

- RPC[15] de Sun
- DCE[16] de Microsoft
- RMI[17] de Java
- ORPC[18] de CORBA

[13] Dynamic Hypertext Markup Language.
[14] Lenguaje de programación.
[15] Remote Procure Call.
[16] Distributed Computing Environment.
[17] Remote Method Invocation.
[18] Object Remote Procedure Call.

¿Pero por qué se ha prestado tanta atención en los últimos años a SOAP?

Según Benjamín González “una de las razones principales es que SOAP ha recibido un increíble apoyo por parte de la industria. Además es el primer protocolo de su tipo que ha sido aceptado prácticamente por todas las grandes compañías de software del mundo. Compañías que en raras ocasiones cooperan entre sí están ofreciendo su apoyo a este protocolo. Algunas de las mayores compañías que soportan SOAP son: Microsoft, IBM, SUN Microsystems, SAP[19] y Ariba[20]”. (25)

Para la implementación del mismo, en ocasiones, se utilizan librerías que ayudan y aportan funcionalidades a este proceso, algunos ejemplos son: Microsoft.Net, NuSoap, Soap de PHP, WSO2, entre otras.

Universal Description, Discovery and Integration of Web Services, UDDI

UDDI es una especificación para un registro distribuido de información acerca de los Servicios Web. Define la forma en la cual se publica y descubre información acerca de estos. “UDDI se concibió como un registro de negocio, un servicio de directorio y nombrado sofisticado. Especifica un marco para describir y descubrir Servicios Web”. (7)

Algunas personas se preguntan el por qué realizar Servicios Web, si la creación de los mismos es un proceso un poco complejo. Además ya existen aplicaciones de escritorios que utilizan tecnología cliente-servidor y pueden comunicarse con otros sistemas perfectamente. A continuación se darán algunos argumentos de por qué utilizar Servicios Web.

1.3.2 Razones para crear un Servicio Web

La principal razón para usar Servicios Web es que estos se basan en HTTP sobre TCP (Transmission Control Protocol) en el puerto 80. Dado que las organizaciones protegen sus redes mediante Firewalls, cierran casi todos los puertos TCP, exceptuando el 80 que es justamente el que usan los navegadores. Los Servicios Web se enrutan por este puerto, por la simple razón de que no resultan bloqueados.

[19] www.sap.com/
[20] www.ariba.com/

Otra razón es la que propone Góngora Marcel R. en su trabajo de diploma del año 2009 *Extensión de la capa de servicios web del Gestor de Contenido*, “antes de que existiera SOAP, no había buenas interfaces para acceder a las funcionalidades de otros ordenadores en red. Las que había eran poco conocidas, tales como EDI (Electronic Data Interchange), RPC, u otras APIs (Application Programming Interface)”. (26)

Una tercera razón por la que los Servicios Web son muy prácticos, también dada por Góngora es: “pueden aportar gran independencia entre la aplicación que usa el Servicio Web y el propio servicio. De esta manera, los cambios a lo largo del tiempo en uno no deben afectar al otro. También permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados”. (26)

La utilización de los Servicios Web trae consigo una serie de ventajas y desventajas, las cuales se detallan en el siguiente epígrafe.

1.3.3 Ventajas y desventajas de los Servicios Web

Una vez descrito en qué consiste un Servicio Web y los elementos que lo constituyen, se hace un resumen de las principales ventajas y las desventajas que presentan.

En cuanto a las ventajas Santiago Márquez expone (27):

- Aporta interoperabilidad entre aplicaciones de software, independientemente de sus propiedades o de las plataformas sobre las que se instalen.
- Los Servicios Web fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- Al apoyarse en HTTP, los Servicios Web pueden aprovecharse de los sistemas de seguridad sin necesidad de cambiar las reglas de filtrado.
- Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan intercambiar información.
- “El utilizar el protocolo HTTP elimina la conocida restricción de procedimientos remotos debido a que SOAP opera bajo HTTP, casi siempre es permitido este tipo de tráfico por el firewall”. (28)

A pesar de estas ventajas que resultan incuestionables, los Servicios Web también presentan algunos inconvenientes que no se debe dejar pasar por alto, los más significativos son (27):

- Para realizar transacciones no pueden compararse en su grado de desarrollo con los estándares abiertos de computación distribuida como CORBA.
- Su rendimiento es bajo si se compara con otros modelos de computación distribuida, tales como: RMI, CORBA, o DCOM. Es uno de los inconvenientes derivados de adoptar un formato basado en texto, ya que como lo que usa es XML, el mismo no tiene en cuenta la concisión ni la eficacia de procesamiento.

Entre las ventajas mencionadas anteriormente está la interoperabilidad entre plataformas, que permiten el intercambio de información. Dicho intercambio puede convertirse también en una amenaza a la seguridad de la información que se gestiona si no se tiene en cuenta algunos aspectos de seguridad.

1.3.4 Seguridad en los Servicios Web

Publicar un conjunto de Servicios Web accesibles a todo el mundo puede suponer una ventaja, sin embargo la seguridad en determinados servicios puede ser de vital importancia, por lo que es necesario evitar acceso de terceros a la información que viaja en los mensajes SOAP o la alteración de los mismos.

Por lo que será necesario tener en cuenta los siguientes factores para asegurar la seguridad del servicio (29):

- Confidencialidad: los mensajes no puedan ser leídos por terceros.
- Integridad: para asegurar a los receptores que el mensaje no ha sido modificado.
- Autenticación: asegurar la procedencia del mensaje, realmente es quién dice ser.
- Autorización: para determinar si se tienen privilegios para realizar una determinada acción.
- No repudio: para poder demostrar que un determinado cliente realizó una determinada acción.

El doctor en Informática Vadim Paz y otros exponen en su informe del año 2007 titulado “Servicios Web” que: “*debido a la tecnología que es usada por los Servicios Web, y en concreto al uso de SOAP, las técnicas de seguridad convencionales que se han venido usando en Internet, ya no son suficientes. Con SOAP, cada mensaje simple que se intercambia realiza múltiples saltos y es enrutado a través de*

numerosos puntos antes de que alcance su destino final. Es por ello, que los Servicios Web necesitan tecnologías que protejan los mensajes desde el principio hasta el final. Existen un conjunto de técnicas que se pueden usar para garantizar la seguridad a nivel de mensaje". (30)

Estas técnicas son:

- Encriptación: Evita que los datos se vean expuestos a lo largo de su recorrido. Esta técnica garantiza la confidencialidad del mensaje.
- Firma Digital XML: Asocia los datos del mensaje al usuario que emite la firma, de modo que este usuario es el único que puede modificar dichos datos.
- Validación de datos: Permite que los Servicios Web reciban datos dentro de los rangos esperados.
- Security Assertion Mark-up Language (SAML) y Autorización: Posibilita que los Servicios Web intercambien información de autentificación y autorización entre ellos, de modo que un Servicio Web confíe en un usuario autentificado por otro Servicio Web.

Vadim Paz continúa explicando que: "En ocasiones, la seguridad se ha conseguido mediante el uso de métodos externos a los Servicios Web, mediante el uso de SSL (Security Socket Layers) o HTTPS (Hypertext Transfer Protocol Security), esta solución no abarca la seguridad completa de los servicios, debido a que los Servicios Web no se limitan únicamente al protocolo HTTP, y deben poseer un sistema pensado específicamente para su mecánica y procedimientos. Para solucionar este problema, se crea la especificación WS-Security que define una serie de extensiones para el protocolo SOAP, que permite el intercambio entre el cliente y el servidor de tokens de seguridad. El intercambio de tokens junto con mecanismos de encriptación y firmado, permiten conseguir los factores anteriormente nombrados". (30)

WS-Security es un estándar que ofrece funciones para proporcionar seguridad en los Servicios Web que se basa en estándares establecidos respecto a la criptografía y firmado de XML. Se recomienda su utilización cuando es necesaria la autenticación, integridad y confidencialidad. Incorpora características de seguridad en el encabezado de un mensaje SOAP, trabajando en la capa aplicación, así permite la seguridad de extremo a extremo. Proporciona un requisito especializado de seguridad llamado: UsernameToken, que es un dispositivo simple y robusto para la comunicación de la autenticación básica de un usuario a un servicio. (31)

Después de analizadas algunas de las técnicas más utilizadas para garantizar la seguridad en los Servicios Web, se exponen las que se utilizan en la presente investigación para garantizar los factores: confidencialidad, autenticación y autorización.

Confidencialidad:

El cuerpo de todos los mensajes SOAP contenidos en las peticiones realizadas por el cliente y en las respuestas dadas por el servidor se encuentra encriptado mediante una función que utiliza como algoritmo de encriptación el 3DES y una palabra clave (Keyword). Dicha palabra clave es definida y enviada (con codificación base64[21]) por el cliente mediante un token en el encabezado del mensaje SOAP. El servidor captura el token que contiene la Keyword, la cual es utilizada para descencriptar los mensajes. Posteriormente procesa el mensaje y encripta la respuesta que le devolverá al cliente.

Autenticación:

El sistema cliente debe enviar el usuario y la contraseña en un UsernameToken, el cual es capturado en el servidor y verificado para asegurar que la procedencia del mensaje es de quién dice ser. El UsernameToken es brindado por el estándar WS-Security.

Autorización:

Una vez que el servidor verifica que las credenciales del UsernameToken son válidas atiende la petición, y le da autorización al cliente a realizar determinada función.

1.4 Descripción de las herramientas que componen el observatorio tecnológico del Grupo GIC

En el Centro FORTES el Grupo GIC lleva cabo la vigilancia tecnológica permitiendo conocer todos los hechos relevantes relacionados con la Tecnología Educativa que surgen actualmente. Por necesidad de los miembros del Grupo, para llevar a cabo una mejor vigilancia, se está desarrollando un Observatorio Tecnológico que permitirá el desarrollo y potenciación de programas y líneas estratégicas de investigación en el ámbito de la Tecnología Educativa. Este integrará algunas de las herramientas necesarias para desarrollar funciones de vigilancia: observación, análisis e interpretación y distribución de la información

[21] Es un grupo de esquemas de codificación similares que representan los datos binarios en un formato de cadena de caracteres ASCII.

científico-tecnológica referente a Tecnología Educativa que permita cubrir las necesidades de información del Centro FORTES.

A continuación se hace una descripción de las principales herramientas que componen el Observatorio Tecnológico.

1.4.1 Open Journal System

Open Journal Systems es un sistema de publicación y gestión de revistas, desarrollado por el Public Knowledge Project. Es un software de código abierto libremente a disposición de revistas en todo el mundo con el propósito de expandir y mejorar el acceso y la calidad de la investigación referida, tanto la académica como la pública. (32)

OJS permite controlar todo el proceso de publicación (32):

- El envío de los manuscritos, por parte de los autores.
- Selección de los revisores, por parte de los editores.
- La revisión, por parte de los revisores primero y luego por los autores mismos.
- La corrección de estilos y de sintaxis.
- La diagramación/maquetación.
- La publicación misma del artículo
- Lectura (con herramientas de lectura).
- Indexación en bases de datos y buscadores.

Algunas de las características de OJS (32):

- Maneja notificaciones por correo electrónico del avance que hay en el proceso editorial y permite a los usuarios registrados comentar los artículos publicados.
- Sistema de indexación de documentos a través del texto completo y el uso de metadatos.
- Se instala y administra localmente.
- Los artículos y todo el proceso editorial es gestionado en línea.

1.4.2 Open Harvester System

El Open Harvester System es un software de código abierto, desarrollado también por el Public Knowledge Project. Es un sistema de indexación de metadatos para ampliar y mejorar el acceso a la investigación. Está diseñado para ser una herramienta flexible para indexar el almacenamiento y la búsqueda de datos de una variedad de diferentes tipos de fuentes de información. (32)

OHS le permite crear un índice de búsqueda de los metadatos, compatible con archivos, tales como los sitios que utilizan Open Journal System u Open Conference System.

OHS, en su versión 2.x incluye las siguientes características (32):

- Capacidad de la recolección de metadatos tipo “Iniciativa de Archivos Abiertos” (OAI, por sus siglas en inglés) en una variedad de esquemas. Posee esquemas adicionales que son compatibles a través de plugins.
- Interfaz de búsqueda flexible que permite la búsqueda simple y avanzada utilizando campos crosswalks (“mapeo de los elementos, sintaxis y semántica desde un esquema de metadatos a otro y que permite transferir un esquema a otro” (33)) de todos los archivos indexados. Búsqueda avanzada de archivos que comparten el mismo esquema siendo posible utilizar los campos como se define en el esquema. Al crear pasos de crosswalks para la búsqueda, los administradores pueden definir los elementos de texto, fecha o HTML.
- Capacidad de realizar la indexación granular con setSpec (variable de especificación de un recolector de metadatos de un OAI) y marcas de tiempo.
- Capacidad para llevar a cabo después de la recolección y antes de la indexación el filtrado/normalización de los metadatos.
- Interfaz de usuario con CSS y HTML basado en plantillas para facilitar la personalización.
- La búsqueda es muy escalable (crea un índice inverso para la búsqueda).

1.4.3 Open Conference System

El Open Conference System es una aplicación gratuita y de código abierto para la gestión y publicación de congresos académicos en la Web. Creada al igual que las dos aplicaciones anteriormente por el Public Knowledge Project. (32)

El sistema OCS permite:

- Crear un sitio web de la conferencia.
- Redactar y enviar una convocatoria de ponencias.
- Aceptar electrónicamente el envío de resúmenes.
- Permitir a los remitentes editar su trabajo.
- Inscribir a los participantes.
- Integración con conferencias post-debates en línea.

1.4.4 Arquitectura Tecnológica del OJS, OHS y OCS

El PKP ha desarrollado un conjunto de software gratuito de código abierto para la gestión, publicación, y la indexación de las revistas y conferencias. Las herramientas OHS y OCS han sido desarrolladas en el lenguaje de programación PHP, basadas en la plataforma OJS, por lo que sus arquitecturas son similares. La arquitectura sobre la que están construidas estas aplicaciones es una arquitectura 3 capas, específicamente un Modelo-Vista-Controlador. Las capas Vista-Controlador están compuestas por las Page Classes, las Action Classes, los Forms y los Templates.

Las Pages Classes reciben las peticiones de los navegadores de los usuarios, delegan cualquier procesamiento requerido a otras clases, y acceden a la plantilla de Smarty[22] adecuada para generar una respuesta (si es necesario). (34)

Las Action Classes son utilizadas por las Pages Classes para realizar el procesamiento no trivial de peticiones del usuario. Por ejemplo, la clase SectionEditorAction es invocada por la clase SectionEditorHandler o sus subclases para realizar la mayor parte del trabajo. (34)

Las clases Forms (classes/form/Form.inc.php) y sus diversas subclases son utilizadas por el journal manager (administrador de revista) para modificar un artículo, centralizar la ejecución de tareas comunes relacionadas con la forma de procesamiento, tales como la validación y el control de errores. (34)

Las plantillas de Smarty son accedidas y gestionadas a través de la clase TemplateManager (classes/template/TemplateManager.inc.php), que lleva a cabo numerosas tareas comunes tales como el

[22] www.smarty.net/

registro adicional de Smarty, funciones tales como {traducir...}, que se utiliza para la localización, y el establecimiento comúnmente utilizado por las variables de la plantilla, tales como direcciones URL y formatos de fecha. (34)

Las Model Classes son clases PHP responsable sólo para representar las entidades de la base de datos en memoria. Por ejemplo, la tabla de artículos almacena información del artículo en la base de datos, hay una Model Class correspondiente llamada Artículo y la clase DAO llamada ArticleDAO. (34)

Las clases Data Access Objects se utilizan para recuperar datos de la base de datos en forma de Model Classes, para actualizar la base de datos dado una clase del modelo modificado, o eliminar las filas de la base de datos. Toda la comunicación entre PHP y el servidor de base de datos se lleva a cabo en las clases de DAO. (34)

1.5 Metodologías de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos y técnicas que apoyan el proceso de desarrollo de productos de software. Surgen por la necesidad de elevar la calidad del software y la eficacia y eficiencia de su proceso de desarrollo. No existe una metodología universal, sino una amplia gama de estas que nos permite seleccionar la que más se ajuste a los requerimientos del software a desarrollar.

1.5.1 Rational Unified Process, RUP

Pressman la define como "*la metodología de desarrollo más popular y una de las más utilizadas a nivel mundial en grandes proyectos por su robustez. Muchas metodologías de surgimiento posterior son derivadas o adaptaciones de esta metodología tradicional. RUP es el resultado de la evolución e integración de diferentes metodologías de desarrollo de software. Permite sacar el máximo provecho de los conceptos asociados a la orientación a objetos y al modelado visual. Cuenta con las mejoras prácticas del modelo de desarrollo de un software en particular*". (35)

Mantiene una estructura bien definida del ciclo de vida de un proyecto y de forma general, está caracterizado por los siguientes aspectos:

- **Guiado por los casos de uso**: Los cuales sirven para describir el comportamiento del sistema y elaborar los casos de prueba con los que se comprueba que el sistema desarrollado hace lo que el cliente quiere.
- **Centrado en la arquitectura**: La arquitectura del sistema es la columna vertebral de todo el desarrollo del mismo. Cada iteración gira en torno a la misma, fortaleciendo y corrigiendo sus características.
- **Iterativo e incremental**: En cada ciclo de iteración se produce una nueva versión del software.

Según la metodología RUP el ciclo de vida de un proyecto se divide en las siguientes fases:

- Fase de concepción.
- Fase de elaboración.
- Fase de construcción.
- Fase de transición.

Utiliza Lenguaje Unificado de Modelado (UML) como lenguaje de modelado y cuenta con varias fases de trabajo en las cuales se desarrolla una serie de flujos fundamentales del desarrollo del proyecto.

Entre los beneficios que podemos obtener de RUP es que unifica todo el equipo de desarrollo de software, lo que favorece la comunicación, asegurando la asignación de recursos en forma eficiente, la entrega de los artefactos correctos y el cumplimiento de los tiempos límite.

Sin embargo también presenta deficiencias, entre las cuales se encuentran: constituye una metodología pesada en la cual se genera documentación en exceso. Se aconseja solo a proyectos de larga duración y gran cantidad de recursos, de lo contrario retrasaría el proceso de desarrollo del software. Además es poco flexible y menos adaptable a los cambios de requisitos durante el proceso de desarrollo.

1.5.2 Extreme Programming, XP

Shore y Warden la consideran como: "*La más popular entre las llamadas metodologías ágiles*" (36). Debe su popularidad entre los desarrolladores a la escasa documentación que genera.

A continuación se enumeran sus características más distintivas:

- **Retroalimentación con el cliente**: Conceptualmente, al menos uno de los miembros del equipo de trabajo del proyecto, es un cliente. Esto propicia una constante interacción del mismo con el producto en desarrollo.
- **Cortas iteraciones**: En cada iteración, se obtiene un producto listo para entregar y que tiene valor para el cliente. La entrega continua de resultados compromete a ambas partes en la evolución del proyecto e indirectamente influye de forma positiva en la calidad del producto final.
- **Muy flexible a cambios**: Una de las características más reconocidas de XP. La constante retroalimentación con los clientes permite prever futuros cambios y evita llegar a momentos que paralizan el desarrollo del producto.

Cuenta con una serie de prácticas destinadas a aumentar la productividad del equipo de trabajo como la programación en pares, reuniones diarias y planes de entrega a corto plazo, por mencionar algunos. "*XP es una metodología aplicable a proyectos de corta duración, en el cual efectuar un cambio es más importante que respetar la planificación realizada*". (36)

1.5.3 Web Service Implementation Methodology (WSIM) – XP

Es el resultado de la aplicación práctica del "OASIS Framework for Web Services Implementation (FWSI)" a la metodología XP. Según el consorcio OASIS "*El FWSI tiene como objetivo facilitar la implementación de Servicios Web robustos a partir de la definición de una metodología práctica y extensible basada en un proceso de implementación y elementos funcionales comunes que los desarrolladores puedan adoptar para crear sistemas de Servicios Web de alta calidad sin re-inventarlos para cada implementación*". (37)

Esta metodología se centra en las actividades de Servicios Web, artefactos, roles y responsabilidades que se pueden incorporar a una metodología de desarrollo de software ágil.

"En el desarrollo de los mismos se necesita que el cliente asigne prioridades, defina el dominio del problema y tome decisiones claves, porque es el cliente quien comprende completamente el proceso que el software está intentando emular". (38)

"*FWSI define las funcionalidades necesarias para ayudar a los desarrolladores a crear sólidas aplicaciones para Servicios Web y arquitecturas orientadas a servicios, acelerará la adopción de Servicios Web debido a la gran reducción de riesgos, la complejidad y el esfuerzo de implementación involucrados*"

(37) expresó Roberto B. Pascual de Infocomm, Copresidente propuesto del Comité Técnico de OASIS FWSI.

La metodología XP define las siguientes fases:

- Planificación
- Diseño
- Codificación
- Pruebas
- Entrega

Estas fases son adaptadas y se aplican al proceso de implementación y desarrollo de Servicios Web.

Fases definidas por WSIM – XP:

- Análisis y requerimientos
- Diseño
- Codificación
- Pruebas
- Entrega

1.5.4 Consideraciones de la metodología de desarrollo de software

Luego del estudio realizado sobre las metodologías de desarrollo más difundidas a nivel internacional se decidió aplicar la metodología “OASIS Web Service Implementation Methodology” por ser una adaptación adecuada de la metodología ágil XP al marco de la solución propuesta teniendo en cuenta los siguientes factores:

- Se requiere un corto plazo de tiempo para la culminación del producto, como máximo 6 meses.
- El equipo de trabajo es pequeño. En este caso de un solo desarrollador y el cliente.
- El cliente conoce ampliamente el funcionamiento del proceso del negocio.
- El cliente forma parte activa del equipo de desarrollo.
- Permite la creación de Servicios Web robustos y de alta calidad.

- Reduce los riesgos, la complejidad y el esfuerzo de implementación involucrados.

1.6 Tecnologías y herramientas a utilizar

1.6.1 Lenguajes de programación

Actualmente existen diferentes lenguajes de programación para desarrollar en la web, estos han ido surgiendo debido a las tendencias y necesidades de las plataformas. Se han investigado dos de los lenguajes de programación más utilizados para la programación de Servicios Web: Java, y PHP.

Java

Java es un lenguaje de programación desarrollado por la compañía Sun Microsystems[23] enfocado a cubrir las necesidades tecnológicas más punteras. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general. (39)

Una de las ventajas que presenta Java es que es un lenguaje independiente de la plataforma. Esto es posible porque se ha creado una máquina virtual de Java para cada sistema que hace de puente entre el sistema operativo y el programa de Java, permitiendo que se entienda perfectamente.

Otra ventaja es que Java es de distribución libre, no es necesario pagar una licencia para poder comenzar a desarrollar en él. Asimismo es un lenguaje muy completo y poderoso, pues posee librerías y utilidades muy completas que facilitan la programación.

PHP5

Preprocessed Hypertext Pages (PHP) es un lenguaje script[24], utilizado para el desarrollo web y puede ser embebido en páginas HTML. Con una sintaxis similar a C, Java y Perl, este lenguaje hace posible el desarrollo de páginas web que se generen dinámicamente y de forma rápida, y puede ejecutarse en distintos tipos de servidores web. (40)

[23] www.java.com/es/
[24] Script: son lenguajes que no necesitan ser compilados.

El lenguaje PHP es capaz de realizar determinadas acciones de una forma fácil y eficaz sin tener que generar programas desarrollados en un lenguaje distinto al HTML. Esto se debe a que PHP ofrece un extenso conjunto de funciones para la explotación de bases de datos sin complicaciones.

Algunas de las ventajas que presenta PHP de manera general:

- Se caracteriza por ser un lenguaje muy rápido, y fácil de aprender.
- Soporta la orientación a objeto. Clases y herencia.
- Es un lenguaje multiplataforma: Linux, Windows, entre otros.
- Capacidad de conexión con la mayoría de los manejadores de base de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, entre otras.
- No requiere definición de tipos de variables ni manejo detallado del bajo nivel.
- Integración con varias bibliotecas externas, que permite desde generar documentos en PDF (Portable Document Format) hasta analizar código XML.
- Ofrece una solución simple y universal para las paginaciones dinámicas del web de fácil programación.

Consideraciones de los lenguajes de programación

Después de un estudio realizado sobre los lenguajes de programación anteriormente mencionados y analizar las características y ventajas de cada uno, se decidió utilizar PHP, en su versión 5.3.3. Principalmente por ser el lenguaje en el que están desarrolladas las herramientas a integrar. Además de poseer un conjunto de funciones que permiten la construcción de Servicios Web.

1.6.2 Bibliotecas y frameworks

Los frameworks son definidos como marco de aplicación o conjunto de bibliotecas orientadas a la reutilización a muy gran escala de componentes software para el desarrollo rápido de aplicaciones.

WSO2 WSF-PHP

Según la WSO2[25] "WSO2 Web Services Framework para PHP es una extensión de PHP para proporcionar y consumir Servicios Web en PHP. Bajo licencia Apache versión 2.0, es una alternativa mucho más robusta. Consiste en un conjunto de extensiones, utilidades y clases con una amplia documentación y ejemplos de gran calidad. Además, soporta las últimas especificaciones de Servicios Web (WSDL 2.0/1.1, SOAP 1.2/1.1), y otras adicionales". (41)

Extensión de PHP5 para SOAP

La extensión SOAP recientemente fue agregada dentro de PHP5. Su soporte nativo implica una mejora en la eficiencia y la rapidez. La extensión SOAP puede ser utilizada para escribir servidores y clientes SOAP. Es compatible con los subconjuntos de SOAP 1.1, SOAP 1.2 y WSDL 1.1. Posee ventajas en cuanto a su fácil uso y no requiere de la instalación de módulos adicionales para su utilización.

Consideraciones de las bibliotecas y frameworks

Si se analiza las características de las alternativas, la mejor opción sería claramente WSF/PHP. Sin embargo, nos encontramos con un contratiempo, para la instalación de este framework necesitamos añadir extensiones, nuevas bibliotecas, modificar el php.ini y algunas variables de entorno del propio servidor. Por el contrario, la extensión SOAP no necesita ningún tipo de complemento adicional, solo utilizar la versión 5 de PHP. Por lo que se propone utilizar esta extensión como herramienta para exponer y consumir Servicios Web.

1.6.3 IDEs

Un Entorno de Desarrollo Integrado o IDE (acrónimo en inglés de Integrated Development Environment), es un programa informático compuesto por un conjunto de herramientas de programación. Es un entorno de programación que ha sido empaquetado como un programa de aplicación; consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica.

[25] Compañía que desarrolla aplicaciones de software abierto enfocadas en proveer arquitecturas SOA.

Zend Studio

Creado por la compañía Zend, es una herramienta propietaria que permite el desarrollo de aplicaciones Web mediante PHP. Permite generar el WSDL correspondiente a una clase o función escrita en PHP, basándose en elementos de configuración previamente definidos por el desarrollador, como la localización del WSDL, convención de nombres y opciones de enlaces, entre otros.

NetBeans 7.0

NetBeans es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso. La misma es capaz de generar el código necesario para la creación tanto de Servicios Web como de Clientes de Servicios Web para Java y PHP.

Consideraciones de los IDEs

Luego de analizados los entornos de desarrollo integrados se concluye que el más conveniente para implementar la propuesta es el NetBeans, pues además de estar entre los más potentes para el desarrollo de aplicaciones web, adquirirlo reportaría beneficios y un costo mínimo porque es de código abierto y gratuito. A pesar de los grandes beneficios que presenta Zend Studio se descarta la posibilidad de usarlo por ser de código propietario.

1.7 Conclusiones parciales del capítulo

Después de analizar las técnicas de integración de aplicaciones, se concluye que la de Servicios Web es la más viable para la solución. Las herramientas utilizadas por el Observatorio poseen arquitecturas similares, todas sobre un modelo 3 capas, específicamente un MVC, lo que facilitó la comprensión de su funcionamiento. Asimismo como metodología de desarrollo se seleccionó Web Service Implementation Methodology (WSIM) por ser una adaptación adecuada de la metodología ágil XP al marco de la solución propuesta. Para implementar la aplicación se decidió utilizar PHP por ser el lenguaje sobre el cual se encuentran desarrolladas las herramientas a integrar, además de las funciones que presenta para la creación de los Servicios Web. También se eligió NetBeans como IDE por ser de código libre y ofrecer las herramientas necesarias para crear Servicios Web.

Capítulo 2: Planificación y diseño de la solución

Introducción

Las aplicaciones que componen el Observatorio Tecnológico del Grupo GIC necesitan integrarse al Portal del Observatorio para reutilizar sus funcionalidades en función de agilizar y centralizar los procesos de gestión y socialización de las fuentes de información. Para solucionar esta problemática, en el presente capítulo, se realiza una propuesta de solución por medio de la metodología WSIM. Se describen de forma general los requisitos de Servicios Web especificados mediante historias de usuarios. Asimismo se explica la dinámica del proyecto a través de las diferentes etapas y procesos que sigue esta metodología.

2.1 Propuesta del sistema

Como solución al problema planteado se propone desarrollar una capa de Servicios Web para el Open Journal System, Open Conference System y Open Harvester System. Esta capa permite al Portal del Observatorio, y a otros sistemas informáticos (con previo acuerdo con los miembros del Grupo), reutilizar las principales funcionalidades de dichas herramientas y obtener información actualizada e imprescindible para el proceso de gestión de las fuentes de información. Sin importar la tecnología o plataforma de implementación empleada.

La capa de servicios debe servir como interfaz de comunicación entre el portal y las otras aplicaciones, brindando la posibilidad de acceso a una serie de operaciones. Estas operaciones serán de lectura y escritura de información, dependiendo siempre de los permisos que posean los usuarios registrados en los sistemas.

2.2 Descripción del sistema propuesto.

El sistema, para su correcto funcionamiento y aceptación por parte del cliente, debe cumplir una serie de especificaciones, las cuales han sido establecidas por el cliente en su papel como especialista en el funcionamiento del negocio dentro del equipo de trabajo.

2.2.1 Descripción del personal relacionado con el sistema.

Dada la naturaleza de la aplicación se toma como personal relacionado con el sistema al Portal del Observatorio del Centro FORTES. A diferencia de las soluciones de software tradicionales, los Servicios Web están pensados para ser consumidos por otras aplicaciones y no por usuarios que interactúen directamente con los mismos.

Personal relacionado	Justificación
Portal del Observatorio del Centro FORTES.	En todos los casos, las operaciones serán iniciadas por el Portal del Observatorio, que necesita obtener y/o modificar alguna información como parte del desarrollo del proceso de gestión de las fuentes de información.

Tabla 2.1. Descripción del personal relacionado con el sistema.

2.3 Planificación: Análisis y requerimientos de los Servicios Web

El objetivo de esta fase es analizar los requerimientos de negocio y traducirlos a los requerimientos de Servicios Web en función de sus características. Asimismo realizar un análisis de la arquitectura, para definir una estructura a alto nivel e identificar la interfaz de los Servicios Web.

2.3.1 Listado de requerimientos de Servicios Web

Uno de los elementos más importantes en un proceso de desarrollo de software lo constituyen los requerimientos, pues estos permiten una comunicación efectiva entre los usuarios finales y el equipo de desarrollo, con el objetivo de llegar a un entendimiento de lo que hay que realizar, siendo así la clave del éxito en la producción de un software.

La IEEE Standard Glossary of Software Engineering Terminology define requerimiento como: "*condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo. Así como la condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente*". (42)

A continuación se presenta el listado de los requisitos de Servicios Web identificados.

Requisitos para los Servicios Web de la herramienta Open Journal System:

R.1 Autenticar/desautenticar usuario: La capa de Servicios Web debe brindar la posibilidad de que los usuarios puedan autenticarse y poder tener acceso a los demás servicios que brinda la aplicación, según los privilegios que tengan estos. También debe permitir desautenticar el usuario.

R.2 Listar revistas habilitadas: La capa de Servicios Web deberá mostrar un listado con todas las revistas que se encuentran habilitadas.

R.3 Listar revistas deshabilitadas: La capa de Servicios Web deberá mostrar un listado con todas las revistas que se encuentran deshabilitadas.

R.4 Gestionar revistas: La capa de Servicios Web debe permitir crear, editar, mostrar y eliminar revistas.

R.5 Habilitar/deshabilitar revista: La capa de Servicios Web debe permitir habilitar o deshabilitar una revista.

R.6 Listar revistas por usuario: La capa de Servicios Web debe mostrar un listado con todas las revistas a las cuales el usuario autenticado está registrado.

R.7 Gestionar formularios de revisión: La capa de Servicios Web debe permitir crear, editar, mostrar y eliminar formularios de revisión.

R.8 Listar formularios de revisión: La capa de Servicios Web de permitir mostrar un listado con todos los formularios de revisión perteneciente a una revista.

R.9 Activar/desactivar formulario de revisión: La capa de Servicios Web debe permitir activar o desactivar un formulario de revisión.

R.10 Gestionar usuarios: La capa de Servicios Web debe permitir crear, editar, mostrar y eliminar usuarios.

R.11 Listar usuarios: La capa de Servicios Web debe permitir mostrar un listado con todos los usuarios.

Requisitos para los Servicios Web de la herramienta Open Conference System:

R.12 Autenticar/desautenticar usuario: La capa de Servicios Web debe brindar la posibilidad de que los usuarios puedan autenticarse y poder tener acceso a los demás servicios que brinda la aplicación, según los privilegios que tengan estos. También debe permitir desautenticar el usuario.

R.13 Listar conferencias habilitadas: La capa de Servicios Web deberá mostrar un listado con todas las conferencias que se encuentran habilitadas.

R.14 Listar conferencias deshabilitadas: La capa de Servicios Web deberá mostrar un listado con todas las conferencias que se encuentran deshabilitadas.

R.15 Gestionar conferencias: La capa de Servicios Web debe permitir crear, editar, mostrar y eliminar conferencia.

R.16 Habilitar/deshabilitar conferencia: La capa de Servicios Web debe permitir habilitar o deshabilitar una conferencia.

R.17 Gestionar usuarios: La capa de Servicios Web debe permitir crear, editar, mostrar y eliminar usuarios.

R.18 Listar usuarios: La capa de Servicios Web debe permitir mostrar un listado con todos los usuarios.

Requisitos para los Servicios Web de la herramienta Open Harvester System:

R.19 Autenticar/desautenticar usuario: La capa de Servicios Web debe brindar la posibilidad de que los usuarios puedan autenticarse y poder tener acceso a los demás servicios que brinda la aplicación, según los privilegios que tengan estos. También debe permitir desautenticar el usuario.

R.20 Listar archivos: La capa de Servicios Web deberá mostrar un listado con todos los archivos.

R.21 Gestionar archivos: La capa de Servicios Web debe permitir crear, editar, mostrar y eliminar archivos.

R.22 Gestionar usuarios: La capa de Servicios Web debe permitir crear, editar, mostrar y eliminar usuarios.

R.23 Listar usuarios: La capa de Servicios Web debe permitir mostrar un listado con todos los usuarios.

R.24 La capa de Servicios Web debe estar disponible las 24 horas.

R.25 Los servicios deben contar con una documentación apropiada que describa todas las funcionalidades del sistema desarrollado así como una guía para su uso.

R.26 La información manejada por el sistema debe estar protegida ante el acceso no autorizado.

R.27 La comunicación con el sistema debe ser a través del protocolo SOAP.

2.3.2 Historias de usuario

La metodología seleccionada para el proceso de desarrollo de la solución utiliza las Historias de Usuario (HU) para describir y organizar las preferencias del cliente. Estas historias están escritas en un lenguaje no técnico con ayuda del cliente.

La estructura de las mismas es la siguiente:

- **Nombre**: Nombre descriptivo de la HU.
- **Prioridad**: Grado de prioridad que le asigna el equipo de desarrollo a la HU en dependencia de las necesidades del cliente. Los valores que puede tomar son (Alta, Media o Baja)
- **Complejidad**: Grado de complejidad que le asigna el equipo de desarrollo a la HU luego de analizarla. (Alta, Media o Baja)
- **Estimación**: Unidades de tiempo estimadas por el equipo de desarrollo para darle cumplimiento a la HU.
- **Iteración**: Número de la iteración en la cual será implementada la HU.
- **Descripción**: Descripción simple que brinda el analista con ayuda del cliente sobre lo que debe hacer la funcionalidad en cuestión.

- **Información Adicional:** Una breve información que ayude a comprender algún dato no especificado antes.
- **Número:** Adicionalmente a cada HU se le asigna un número para facilitar su identificación.

Sobre las Historias de usuario

Para la confección del modelo de las H se aplicaron una serie de conformidades con el objetivo de adaptarlas a la situación actual, las cuales son descritas a continuación:

- **Omitir el campo de "Usuario" en las historias:** Este campo se utiliza cuando con el sistema interactúan más de un tipo de usuario. La metodología seleccionada, permite que sea omitido puesto que, al tratarse de Servicios Web, estos siempre serán iniciados por un sistema externo, en este caso el Portal del Observatorio, estando todas sus funcionalidades disponibles por igual.
- **Unidades de estimación**: Dadas las características de las funcionalidades a implementar en el desarrollo de Servicios Web, se ha tomado como unidad de estimación una semana.

A continuación se muestran las HU identificadas.

Historias de usuarios correspondientes a los requisitos identificados de la herramienta Open Journal System:

Historia de Usuario						**Número:**	**1**
Nombre:	Autenticar/desautenticar usuario en el sistema OJS.						
Prioridad:	Alta	**Complejidad:**	Alta	**Estimación:**	1	**Iteración:**	1
Descripción:							
Autenticar: Dado el username y la contraseña, autenticar el usuario en el sistema Open Journal System. **Desautenticar:** Desautenticar el usuario.							
Información Adicional:							
Da cumplimiento al requisito R.1.							

Tabla 2.2: Descripción de la historia de usuario: Autenticar/desautenticar usuario en el sistema OJS.

Historia de Usuario		**Número:**	**2**
Nombre:	Listar revistas.		

Prioridad	Alta	**Complejidad**:	Media	**Estimación**:	1	**Iteración**:	1
Descripción:							
Listar revistas habilitadas: Obtener un listado con los datos de las revistas habilitadas alojadas en el sistema. **Listar revistas deshabilitadas:** Obtener un listado con los datos de las revistas deshabilitadas en el sistema. **Listar revistas por usuario:** Obtener un listado con los datos de las revistas en las que se encuentra registrado el usuario autenticado.							
Información Adicional:							
Da cumplimiento a los requisitos R.2, R.3 y R.6. Para los requisitos R.3 y R.6: Se debe tener en cuenta el nivel de acceso del usuario que solicita la información.							

Tabla 2.3: Descripción de la historia de usuario: Listar revistas.

Historia de Usuario						**Número:**	**3**
Nombre:	Gestionar revistas.						
Prioridad:	Alta	**Complejidad:**	Alta	**Estimación:**	1	**Iteración:**	1
Descripción:							
Gestionar revistas: Crear una revista nueva a partir de los parámetros requeridos; mostrar los datos de una revista seleccionada; modificar los valores deseados de una revista mostrada; eliminar una revista. **Habilitar/deshabilitar revista:** Habilitar o deshabilitar una revista seleccionada.							
Información Adicional:							
Da cumplimiento a los requisitos R.4 y R.5. Se debe tener en cuenta el nivel de acceso del usuario que solicita la información.							

Tabla 2.4: Descripción de la historia de usuario: Gestionar revistas.

Historia de Usuario						**Número:**	**4**
Nombre:	Gestionar formularios de revisión.						
Prioridad:	Alta	**Complejidad:**	Alta	**Estimación:**	2	**Iteración:**	2
Descripción:							
Gestionar formulario de revisión: Crear un formulario de revisión nuevo a partir de los parámetros requeridos; mostrar los datos de un formulario de revisión seleccionado, modificar los valores deseados de un formulario de revisión mostrado; eliminar un formulario de revisión. **Listar formularios de revisión:** Obtener un listado con los datos de los formularios de revisión. **Activar/desactivar formulario de revisión:** Activar o desactivar un formulario de revisión seleccionado.							
Información Adicional:							

Da cumplimiento a los requisitos R.7, R.8 y R.9. Se debe tener en cuenta el nivel de acceso del usuario que solicita la información.

Tabla 2.5: Descripción de la historia de usuario: Gestionar formularios de revisión.

Historia de Usuario						**Número:**	**5**
Nombre:	Gestionar usuarios del sistema OJS.						
Prioridad:	Alta	**Complejidad:**	Alta	**Estimación:**	1	**Iteración:**	2
Descripción:							
Gestionar usuarios: Registrar un nuevo usuario en el sistema, editar su perfil, mostrar los datos de su perfil y eliminar un usuario seleccionado en el sistema Open Journal System. **Listar usuarios:** Obtener un listado con los datos del perfil de los usuarios registrados en el sistema Open Journal System.							
Información Adicional:							
Da cumplimiento a los requisitos R.10 y R.11. Se debe tener en cuenta el nivel de acceso del usuario que solicita la información.							

Tabla 2.6: Descripción de la historia de usuario: Gestionar usuarios del sistema OJS.

Historias de usuarios correspondientes a los requisitos identificados de la herramienta Open Conference System:

Historia de Usuario						**Número:**	**6**
Nombre:	Autenticar/desautenticar usuario en el sistema OCS.						
Prioridad:	Media	**Complejidad:**	Alta	**Estimación:**	1	**Iteración:**	3
Descripción:							
Autenticar: Dado el username y la contraseña autenticar el usuario en el sistema Open Conference System. **Desautenticar:** Desautenticar el usuario.							
Información Adicional:							
Da cumplimiento al requisito R.12.							

Tabla 2.7: Descripción de la historia de usuario: Autenticar/desautenticar usuario en el sistema OCS.

Historia de Usuario						**Número:**	**7**
Nombre:	Gestionar conferencias.						
Prioridad:	Media	**Complejidad:**	Alta	**Estimación:**	1	**Iteración:**	3

Descripción:
Gestionar conferencias: Crear una conferencia nueva a partir de los parámetros requeridos; mostrar los datos de una conferencia seleccionada, modificar los valores deseados de una conferencia mostrada; eliminar una conferencia. **Listar conferencias habilitadas:** Obtener un listado con los datos de las conferencias habilitadas alojadas en el sistema. **Listar conferencias deshabilitadas:** Obtener un listado con los datos de las conferencias deshabilitadas alojadas en el sistema. **Habilitar/deshabilitar conferencia:** Habilitar o deshabilitar una conferencia seleccionada.
Información Adicional:
Da cumplimiento a los requisitos R.13, R.14, R.15 y R.16. Para los requisitos R.14, R.15 y R.16: Se debe tener en cuenta el nivel de acceso del usuario que solicita la información.

Tabla 2.8: Descripción de la historia de usuario: Gestionar conferencias.

Historia de Usuario						**Número:**	**8**
Nombre:	Gestionar usuarios del sistema OCS.						
Prioridad:	Media	**Complejidad:**	Alta	**Estimación:**	1	**Iteración:**	3
Descripción:							
Gestionar usuarios: Registrar un nuevo usuario en el sistema, editar su perfil, mostrar los datos de su perfil y eliminar un usuario seleccionado en el sistema Open Conference System. **Listar usuarios:** Obtener un listado con los datos del perfil de los usuarios registrados en el sistema Open Conference System.							
Información Adicional:							
Da cumplimiento al requisito R.17 y R.18. Se debe tener en cuenta el nivel de acceso del usuario que solicita la información.							

Tabla 2.9: Descripción de la historia de usuario: Gestionar usuarios del sistema OCS.

Historias de usuarios correspondientes a los requisitos identificados de la herramienta Open Harvester System:

Historia de Usuario						**Número:**	**9**
Nombre:	Autenticar/desautenticar usuario en el sistema OHS.						
Prioridad:	Baja	**Complejidad:**	Alta	**Estimación:**	1	**Iteración:**	4
Descripción:							

Autenticar: Dado el username y la contraseña autenticar el usuario en el sistema Open Harvester System. **Desautenticar:** Desautenticar el usuario.
Información Adicional:
Da cumplimiento al requisito R.19.

Tabla 2.10: Descripción de la historia de usuario: Autenticar/desautenticar usuario en el sistema OHS.

Historia de Usuario						**Número:**	**10**
Nombre:	Gestionar archivos.						
Prioridad:	Baja	**Complejidad:**	Alta	**Estimación:**	1	**Iteración:**	4
Descripción:							
Gestionar archivos: Crear un archivo nuevo a partir de los parámetros requeridos; mostrar los datos de un archivo seleccionado; modificar los valores deseados de un archivo mostrado. Eliminar un archivo. **Listar usuarios:** Obtener un listado con los datos de los archivos alojados en el sistema.							
Información Adicional:							
Da cumplimiento al requisito R.20 y R.21. Se debe tener en cuenta el nivel de acceso del usuario que solicita la información.							

Tabla 2.11: Descripción de la historia de usuario: Gestionar archivos.

Historia de Usuario						**Número:**	**11**
Nombre:	Gestionar usuarios del sistema OHS.						
Prioridad:	Baja	**Complejidad:**	Alta	**Estimación:**	1	**Iteración:**	4
Descripción:							
Gestionar usuarios: Registrar un nuevo usuario en el sistema, editar su perfil, mostrar los datos de su perfil y eliminar un usuario seleccionado en el sistema Open Harvester System. **Listar usuarios:** Obtener un listado con los datos del perfil de los usuarios registrados en el sistema Open Harvester System.							
Información Adicional:							
Da cumplimiento al requisito R.22 y R.23. Se debe tener en cuenta el nivel de acceso del usuario que solicita la información.							

Tabla 2.12: Descripción de la historia de usuario: Gestionar usuarios del sistema OHS.

2.3.3 Propuesta de especificación de la arquitectura para los Servicios Web

Una vez identificados los objetivos del cliente se procede a identificar las opciones en cuanto a tipos de contenidos a ser ofrecidos como Servicios Web. Se toman todas las opciones en las que el cliente se muestra interesado y el próximo paso es evolucionar estas opciones a posibles Servicios Web. Definiendo así una arquitectura a alto nivel, donde se identifican las funcionalidades que serán expuestas como Servicios Web.

2.3.4 Definición de la granularidad de los Servicios Web

Se han identificado a partir de las HU varios grupos importantes de funcionalidades que según su tipo pueden ser agrupadas y contenidas dentro de Servicios Web.

Para la aplicación OJS se identificaron 4 grupos de funcionalidades, los cuales están definidos de la siguiente forma:

- Autenticación en el sistema: Contiene las funcionalidades de autenticar y desautenticar un usuario en el sistema. HU contenida: 1.
- Gestión de Revistas: Está compuesto por las funcionalidades de listar revistas habilitadas, listar revistas deshabilitadas, listar revistas en las que el usuario autenticado está registrado, gestionar las revistas (crearlas, editarlas, mostrarlas o eliminarlas) y habilitar o deshabilitar una revista. Algunas de estas funciones dependen del nivel de acceso del usuario autenticado. HU contenidas: 2 y 3.
- Gestión de Formularios de revisión: Contiene un conjunto de funcionalidades que permiten listar los formularios de revisión que pertenecen a una revista, gestionar los formularios de revisión (crearlos, editarlos, mostrarlos o eliminarlos) y activar o desactivar algún formulario. Todas estas funciones dependen del nivel de acceso del usuario autenticado. HU contenida: 4.
- Gestión de Usuarios: Está compuesto por las funcionalidades de listar usuarios registrados en una revista, gestionar los usuarios (crearlos, editarlos, mostrarlos o eliminarlos). Todas estas funciones dependen del nivel de acceso del usuario autenticado. HU contenida: 5.

Para la aplicación OCS se identificaron 3 grupos de funcionalidades, los cuales son:

- Autenticación en el sistema: Contiene las funcionalidades de autenticar y desautenticar un usuario en el sistema. HU contenida: 6.
- Gestión de Conferencias: Está compuesto por las funcionalidades de listar conferencias habilitadas o deshabilitadas, gestionar las conferencias (crearlas, editarlas, mostrarlas o eliminarlas) y habilitar o deshabilitar una conferencia. Algunas de estas funciones dependen del nivel de acceso del usuario autenticado. HU contenida: 7.
- Gestión de usuarios: Está compuesto por las funcionalidades de listar usuarios registrados, gestionar los usuarios (crearlos, editarlos, mostrarlos o eliminarlos). Todas estas funciones dependen del nivel de acceso del usuario autenticado. HU contenida: 8.

Para la aplicación OHS se identificaron 3 grupos de funcionalidades:

- Autenticación en el sistema: Contiene las funcionalidades de autenticar y desautenticar un usuario en el sistema. HU contenida: 9.
- Gestión de Archivos: Está compuesto por las funcionalidades de listar archivos, gestionar los archivos (crearlos, editarlos, mostrarlos o eliminarlos). Todas estas funciones dependen del nivel de acceso del usuario autenticado. HU contenida: 10.
- Gestión de usuarios: Está compuesto por las funcionalidades de listar usuarios registrados en una revista, gestionar los usuarios (crearlos, editarlos, mostrarlos o eliminarlos). Todas estas funciones dependen del nivel de acceso del usuario autenticado. HU contenida: 11.

2.3.5 Servicios Web reutilizables

La metodología WSIM plantea que, como parte del proceso de definición de las especificaciones de arquitectura de software, se realice un estudio de los posibles Servicios Web existentes que brinden algunas de las funcionalidades solicitadas por el cliente, con el objetivo de reutilizarlos como parte de la solución propuesta. En este caso no existe ningún servicio aprovechable por lo tanto queda descartada esta posibilidad.

2.3.6 Plan de iteraciones

A continuación se procede a distribuir las HU entre varias iteraciones ordenándolas por prioridad y agrupándolas por iteración. De esta manera se garantiza que al menos una vez cada 3 semanas el cliente tendrá un prototipo del sistema con un porciento de funcionalidades listas para ser probadas y tomar decisiones en cuanto a cambios que puedan requerirse para mantener el camino correcto rumbo a los objetivos propuestos.

Iteración	Historias de Usuarios	Tiempo
1	1. Autenticar/desautenticar usuario en el sistema OJS. 2. Listar revistas. 3. Gestionar revistas.	3 semanas
2	4. Gestionar formularios de revisión. 5. Gestionar usuarios del sistema OJS	3 semanas
3	6. Autenticar usuario en el sistema OCS 7. Gestionar conferencias 8. Gestionar usuarios del sistema OCS	3 semanas
4	9. Autenticar usuario en el sistema OHS 10. Gestionar archivos 11. Gestionar usuarios del sistema OHS	3 semanas

Tabla 2.13: Distribución de historias de usuario por iteración.

Como resultado de la planificación se estima que la codificación de las 11 HU sea completado en aproximadamente 12 semanas. Se tiene como referencia que se comienzan a realizar las iteraciones en la fecha 12/21/2011 culminando el 12/4/2012.

2.4 Diseño

El diseño detallado de los Servicios Web se lleva a cabo en esta fase. En esta se definen las interfaces de los servicios, identificando los elementos y los tipos de datos, así como las clases que formarán la arquitectura de los servicios. Para llevar a cabo el diseño, la metodología WSIM hereda de XP la recomendación de utilizar sencillos esquemas y técnicas como las tarjetas CRC (Clase, Responsabilidad, Colaborador) en lugar de diagramas de clases UML para describir el diseño del software.

2.4.1 Interfaces de servicios

Las interfaces de servicios son clases que detallan cuáles son las operaciones que ofrece el servicio. En la implementación de la solución propuesta se han definido diez Servicios Web con varias funcionalidades. Para cada uno de estos servicios se define una interfaz, en la cual se describen los nombres de las funcionalidades que brinda el mismo así como las políticas en cuanto a datos requeridos o parámetros y datos de respuesta.

A continuación se muestran algunas interfaces, específicamente para los servicios del OJS, las restantes se muestran en el **Anexo 1**.

WebServiceAuth		
Nombre: Autenticar usuario en el sistema.		**Historia de Usuario**: 1
Función: loginIn		**Retorno**: SessionWS
Tipo string	**Nombre** username	**Descripción**: Nombre del usuario que se va a autenticar.
Tipo string	**Nombre** password	**Descripción**: Contraseña del usuario que se va a autenticar.
Nombre: Desautenticar usuario del sistema.		**Historia de Usuario**: 1
Función: loginOut		**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.

Tabla 2.14: Definición de la interface del Servicio Web: WebServiceAuth.

WebServiceJournal		
Nombre: Obtener listado de todas las revistas habilitadas en el sistema.		**Historia de Usuario**: 2
Función: listEnabledJournals		**Retorno**: JournalWS [ ]
Tipo -	**Nombre** -	**Descripción**: -
Nombre: Obtener listado de todas las revistas deshabilitadas en el sistema.		**Historia de Usuario**: 2
Función: listDisabledJournals		**Retorno**: JournalWS [ ]
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.
Nombre: Crear una revista nueva en el sistema.		**Historia de Usuario**: 3

Función: createJournal			**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.	
Tipo string	**Nombre** title	**Descripción**: Título de la revista a crear.	
Tipo string	**Nombre** path	**Descripción**: Ruta donde la revista a crear se va a alojar.	
Tipo bool	**Nombre** enable	**Descripción**: Valor que define si la revista está habilitada o deshabilitada.	
Nombre: Editar una revista alojada en el sistema.			**Historia de Usuario**: 3
Función: editJournal			**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.	
Tipo integer	**Nombre** journalId	**Descripción**: Id de la revista que se desea editar.	
Tipo string	**Nombre** title	**Descripción**: Título de la revista a editar.	
Tipo string	**Nombre** path	**Descripción**: Ruta donde la revista estará alojada.	
Tipo bool	**Nombre** enable	**Descripción**: Valor que define si la revista está habilitada o deshabilitada.	
Nombre: Mostrar datos de una revista habilitada.			**Historia de Usuario**: 3
Función: displayDataEnableJournal			**Retorno**: JournalWS
Tipo integer	**Nombre** journalId	**Descripción**: Id de la revista que se desea mostrar.	
Nombre: Eliminar revista seleccionada.			**Historia de Usuario**: 3
Función: deleteJournal			**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.	
Tipo integer	**Nombre** journalId	**Descripción**: Id de la revista que se desea eliminar.	
Nombre: Habilitar o deshabilitar una revista.			**Historia de Usuario**: 3
Función: journalEnableDisable			**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.	
Tipo integer	**Nombre** journalId	**Descripción**: Id de la revista que se desea habilitar o deshabilitar.	
Nombre: Obtener un listado de las revistas en las que el			**Historia de Usuario**: 2

usuario autenticado está registrado.			
Función: journalsByUser			**Retorno**: JournalWS [ ]
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.	

Tabla 2.15: Definición de la interface del Servicio Web: WebServiceJournal.

WebServiceReviewForm			
Nombre: Obtener listado de todos los formularios de revisión.			**Historia de Usuario**: 4
Función: listReviewForms			**Retorno**: ReviewFormWS [ ]
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.	
Nombre: Crear un nuevo formulario de revisión en el sistema.			**Historia de Usuario**: 4
Función: createReviewForm			Retorno: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.	
Tipo integer	**Nombre** journalId	**Descripción**: Id de la revista en la que se desea crear el formulario de revisión.	
Tipo string	**Nombre** title	**Descripción**: Título del formulario de revisión a crear.	
Tipo string	**Nombre** desc	**Descripción**: Descripción del formulario de revisión.	
Nombre: Editar un formulario de revisión.			**Historia de Usuario**: 4
Función: editReviewForm			**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.	
Tipo integer	**Nombre** reviewFormId	**Descripción**: Id del formulario de revisión que se desea editar.	
Tipo integer	**Nombre** journalId	**Descripción**: Id de la revista de la que se desea editar el formulario de revisión.	
Tipo string	**Nombre** title	**Descripción**: Título del formulario de revisión a editar.	
Tipo string	**Nombre** desc	**Descripción**: Descripción del formulario de revisión a editar.	
Nombre: Mostrar datos de un formulario de revisión.			**Historia de Usuario**: 4
Función: displayDataReviewForm			**Retorno**: ReviewFormWS

Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.
Tipo integer	**Nombre** reviewFormId	**Descripción**: Id del formulario de revisión que se desea mostrar.
Tipo integer	**Nombre** journalId	**Descripción**: Id de la revista de la que se desea mostrar el formulario de revisión.
Nombre: Eliminar formulario de revisión seleccionado.		**Historia de Usuario**: 4
Función: deleteReviewForm		**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.
Tipo integer	**Nombre** reviewFormId	**Descripción**: Id del formulario de revisión que se desea eliminar.
Tipo integer	**Nombre** journalId	**Descripción**: Id de la revista de la que se desea eliminar el formulario de revisión.
Nombre: Activar o desactivar un formulario de revisión seleccionado.		**Historia de Usuario**: 4
Función: reviewFormActivateDisactivate		**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.
Tipo integer	**Nombre** journalId	**Descripción**: Id de la revista de la que se desea activar o desactivar el formulario de revisión.
Tipo integer	**Nombre** reviewFormId	**Descripción**: Id del formulario de revisión que se desea activar o desactivar.

Tabla 2.16: Definición de la interface del Servicio Web: WebServiceReviewForm.

WebServiceUser		
Nombre: Obtener listado de todos los usuarios registrados en el sistema.		**Historia de Usuario**: 5
Función: listUsers		Retorno: UserWS [ ]
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.
Nombre: Registrar un nuevo usuario en el sistema.		**Historia de Usuario**: 5
Función: createUser		**Retorno**: bool

Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.
Tipo array	**Nombre** parameters	**Descripción**: Un arreglo con todos los parámetros necesarios para registrar un usuario. (journalId, password, salutation, userName, firstName, middleName, lastName, initials, gender, affiliation, signature, email, url, phone, fax, interest, mailingAddress, country, biography, roleName)
Nombre: Editar un perfil de usuario.		**Historia de Usuario**: 5
Función: editUser		**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.
Tipo array	**Nombre** parameters	**Descripción**: Un arreglo con todos los parámetros necesarios para registrar un usuario. (journalId, userId, password, salutation, userName, firstName, middleName, lastName, initials, gender, affiliation, signature, email, url, phone, fax, interest, mailingAddress, country, biography, roleName)
Nombre: Mostrar datos del perfil de un usuario.		**Historia de Usuario**: 5
Función: displayDataUser		**Retorno**: UserWS
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.
Tipo integer	**Nombre** userId	**Descripción**: Id del usuario del que se desea mostrar su perfil.
Nombre: Eliminar usuario del sistema.		**Historia de Usuario**: 5
Función: deleteUser		**Retorno**: bool
Tipo SessionWS	**Nombre** session	**Descripción**: Objeto sesión que contiene el id de la sesión creada, id y nombre del usuario autenticado.
Tipo integer	**Nombre** userId	**Descripción**: Id del usuario que se desea eliminar.

Tabla 2.17: Definición de la interface del Servicio Web: WebServiceUser.

Consideraciones sobre los WSDL de los servicios

El consumo de Servicios Web utilizando la extensión SOAP de PHP5 se hace a través de la clase SoapClient, a la cual se le debe pasar en el constructor la dirección del WSDL que constituye el contrato y describe el servicio; y un parámetro *options* que constituye un arreglo.

Algo importante a tener en cuenta es que desde varias partes de la aplicación se podría necesitar consumir un Servicio Web determinado, para lo cual habría que especificar la dirección del WSDL cada vez que se establezca la conexión con el servicio. Esto puede provocar que si ocurre un cambio en la dirección, habría que actualizarla en cada punto donde se consumió el Servicio Web.

Para evitar esta inconveniente existen varias formas: una de ellas es crear un fichero con todas las direcciones de los servicios a utilizar, y pasar referencias a este fichero; la otra es utilizar el SoapClient en el modo non-WSDL. En este modo no se le pasa la dirección del WSDL sino el arreglo *options* definiendo la opción *location* y *uri*, donde *location* es el URL del servidor SOAP a enviar la petición y *uri* es el espacio de nombres destino del servicio SOAP.

En la solución propuesta se decidió no publicar el WSDL, ya que los Servicios Web a desarrollar son privados, por lo que el cliente conoce todas las funcionalidades que son expuestas y no necesita la descripción del servicio. Esto disminuye cualquier intento de ataque a la seguridad, debido a que el posible atacante, al no disponer de las descripciones de los servicios, desconoce los parámetros a pasar y las funcionalidades expuestas.

2.4.2 Tarjetas CRC

Las tarjetas CRC son una herramienta de reflexión en el diseño de software orientado a objetos, son caracterizadas por ser sencillas, fáciles de entender y permiten al equipo de trabajo en su totalidad participar en el diseño de los Servicios Web.

Con el objetivo de hacer entendible las clases que exponen los Servicios Web se define una tarjeta CRC por cada una, con la finalidad de obtener un diseño simple y no incurrir en la implementación de características no necesarias.

Las partes que componen una tarjeta CRC son las siguientes:

- **Clase:** Nombre de la clase con que se está modelando.
- **Súper Clase:** Nombre de la clase padre en la herencia.
- **Sub Clase(s):** Nombre de las clases hijas en la herencia.
- **Responsabilidades:** Descripción de alto nivel del propósito de la clase.
- **Colaboraciones:** Indica con cuáles otras clases se requiere relación para cumplir con la responsabilidad.

A continuación, algunas de las tarjetas CRC, específicamente de las clases correspondientes a la capa de Servicios Web de la aplicación Open Journal System, las restantes se muestran en el **Anexo 2.**

Tarjeta CRC	
Clase: WebServiceAuthHandler **Súper Clase:** Handler **Sub Clase(s):**	
Responsabilidades	**Colaboraciones**
Define y expone el servidor SOAP del servicio Autenticación. Se encarga de chequear que el encabezado de la solicitud SOAP contenga un token de seguridad con las credenciales válidas del sistema cliente. Además atiende todas las peticiones hechas por el cliente a este servicio.	- ChekCredentials - WebServiceAuth

Tabla 2.18: Tarjeta CRC de la clase WebServiceAuthHandler.

Tarjeta CRC	
Clase: WebServiceJournalHandler **Súper Clase:** Handler **Sub Clase(s):**	
Responsabilidades	**Colaboraciones**
Define y expone el servidor SOAP del servicio	- ChekCredentials

Gestión de revistas. Se encarga de chequear que el encabezado de la solicitud SOAP contenga un token de seguridad con las credenciales válidas del sistema cliente. Además atiende todas las peticiones hechas por el cliente a este servicio.	- WebServiceJournal

Tabla 2.19: Tarjeta CRC de la clase WebServiceJournalHandler.

Tarjeta CRC	
Clase: WebServiceFormReviewHandler **Súper Clase:** Handler **Sub Clase(s):**	
Responsabilidades	**Colaboraciones**
Define y expone el servidor SOAP del servicio Gestión de formularios revisión. Se encarga de chequear que el encabezado de la solicitud SOAP contenga un token de seguridad con las credenciales válidas del sistema cliente. Además atiende todas las peticiones hechas por el cliente a este servicio.	- ChekCredentials - WebServiceFormReview

Tabla 2.20: Tarjeta CRC de la clase WebServiceFormReviewHandler.

Tarjeta CRC	
Clase: WebServiceUserHandler **Súper Clase:** Handler **Sub Clase(s):**	
Responsabilidades	**Colaboraciones**
Define y expone el servidor SOAP del servicio Gestión de usuarios. Se encarga de chequear que el encabezado de la solicitud SOAP contenga un token de seguridad	- ChekCredentials - WebServiceUser

con las credenciales válidas del sistema cliente. Además atiende todas las peticiones hechas por el cliente a este servicio.	

Tabla 2.21: Tarjeta CRC de la clase WebServiceUserHandler.

Tarjeta CRC	
Clase: FormReviewWS **Súper Clase:** **Sub Clase(s):**	
Responsabilidades	**Colaboraciones**
Brinda los datos necesarios de la clase entidad FormReviewWS. - Devolver el id del formulario de revisión. - Devolver el id de la revista a la que pertenece el formulario. - Devolver el título del formulario. - Devolver la descripción del formulario.	-

Tabla 2.22: Tarjeta CRC de la clase FormReviewWS.

Tarjeta CRC	
Clase: JournalWS **Súper Clase:** **Sub Clase(s):**	
Responsabilidades	**Colaboraciones**
Brinda los datos necesarios de la clase entidad JournalWS. - Devolver el id de la revista. - Devolver la ruta de la revista. - Devolver el título de la revista. - Devolver la descripción de la revista. - Devolver la url de la revista.	-

Tabla 2.23: Tarjeta CRC de la clase JournalWS.

Tarjeta CRC	
Clase: SessionWS **Súper Clase:** **Sub Clase(s):**	
Responsabilidades	**Colaboraciones**
Brinda los datos necesarios de la clase entidad SessionWS. - Devolver el id de la sesión. - Devolver el username del usuario autenticado. - Devolver el id del usuario autenticado.	-

Tabla 2.24: Tarjeta CRC de la clase SessionWS.

Tarjeta CRC	
Clase: UserWS **Súper Clase:** **Sub Clase(s):**	
Responsabilidades	**Colaboraciones**
Brinda los datos necesarios de la clase entidad UserWS. - Devuelve los datos de un usuario.	-

Tabla 2.25: Tarjeta CRC de la clase UserWS.

2.5 Conclusiones parciales del capítulo

A partir de las especificaciones de los requisitos mediante las historias de usuarios, se pudo identificar varios grupos de funcionalidades que fueron agrupados en varios Servicios Web según el contenido a devolver. La realización del diseño de la propuesta de solución generó artefactos que permitió definir su arquitectura y sirven como antesala a su implementación.

Capítulo 3: Implementación y prueba

Introducción

Partiendo de los artefactos resultantes del diseño de la capa de servicio, en el presente capítulo, se describirán los aspectos relacionados con su implementación, la cual estará dividida en varias iteraciones. Obteniéndose al finalizar cada iteración versiones funcionales del producto. Para la construcción se utilizarán pautas de codificación con vistas a generar un código de alta calidad de gran importancia para la calidad de los servicios. Una vez implementada la solución será sometida a una serie de pruebas para verificar que satisfaga las necesidades del cliente.

3.1 Implementación de los Servicios Web

En esta fase se descomponen las HU en tareas de desarrollo, donde se le asignan estas tareas al programador, con la responsabilidad de su implementación. Durante el proceso de desarrollo de los Servicios Web se llevaron a cabo 4 iteraciones, permitiendo obtener finalizada cada iteración una versión producto; con las características solicitadas por el cliente y listo para realizarles las pruebas correspondientes.

3.1.1 1ra Iteración

El objetivo de esta iteración es desarrollar las HU 1, 2 y 3. Al final se deben tener todas las funcionalidades propuestas en cada HU, teniéndose así una versión del producto para mostrar el resultado al cliente.

Las tareas por historias de usuarios definidas en esta iteración se muestran en el **Anexo 3**.

3.1.2 2da Iteración

En esta iteración se implementarán las HU 4 y 5, con el fin de obtener una segunda versión, de la capa de Servicios Web, con nuevas funcionalidades. Las tareas por HU definidas en esta iteración se muestran en el **Anexo 3**.

3.1.3 3ra Iteración

En esta iteración se implementarán las HU 6, 7 y 8, con el fin de obtener una tercera versión de la capa de Servicios Web con nuevas funcionalidades. Las tareas por HU definidas en esta iteración se muestran en el **Anexo 3**.

3.1.4 4ta Iteración

En esta iteración se implementarán las HU 9, 10 y 11, con el objetivo de obtener la última versión del producto con todas las funcionalidades definidas por el cliente. Las tareas por HU definidas en esta iteración se muestran en el **Anexo 3**.

3.1.5 Pautas de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad de los servicios y para obtener un buen rendimiento.

Además, si se aplica de forma continuada un estándar de codificación bien definido, se utilizan técnicas de programación apropiadas y posteriormente, se efectúan revisiones del código de rutinas, caben muchas posibilidades de que un proyecto de software se convierta en un sistema de software fácil de comprender y de mantener.

Consideraciones generales

Con el objetivo de ajustarse a los estándares internacionales se establece utilizar el idioma inglés para todos los elementos que intervienen en este proceso de codificación, dígase nombre de clases, variables, comentarios, ficheros, entre otros.

Comentarios

Los comentarios no deben ser encerrados en grandes cajas dibujadas con asteriscos u otros caracteres, y no deben nunca incluir caracteres especiales.

Formato de los comentarios en la implementación

Estilos de comentarios utilizados: bloque y de fin de línea. Ver **Figura 3.1** y **Figura 3.2.**

Código A.1: Bloque de comentario

```
/*
 *    Este es un bloque de comentarios.
 */
```

Figura 3.1: Bloque de comentario.

Los bloques de comentarios son utilizados para proveer descripciones de ficheros, métodos, estructuras de datos y algoritmos. Estos fueron utilizados al inicio de cada fichero y al inicio de cada método.

Código A.4: Comentarios de fin de línea

```
if (foo > 1)
{
  // Hacer algo.
  ...
}
else {
  return false;       // Explicar algo.
}
```

Figura 3.2: Comentarios de fin de línea.

El delimitador de comentario // puede convertir en comentario una línea completa o una parte de una línea. No debe ser usado para hacer comentarios de varias líneas consecutivas; sin embargo, puede usarse en líneas consecutivas para comentar secciones de código.

Identación, llaves de apertura y de cierre, y tamaño de las líneas

La identación se realizó con tabulaciones, con un equivalente a cuatro espacios. El uso de las llaves de apertura y de cierre se realizó en una nueva línea. Se utilizó no más de 80 caracteres por cada línea de código.

Conversión de nomenclatura

Se siguió el estándar de codificación utilizado por los desarrolladores de las herramientas OJS, OCS, y OHS. Las variables siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de la segunda palabra comienza con mayúscula. Las clases siempre comienzan con mayúscula, en caso de nombre compuesto, las palabras que componen el nombre de la clase comienzan con mayúsculas. En las funciones se utilizó el estilo lowerCamelCase, donde siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula, exceptuando la primera palabra. Los parámetros son separados por espacio luego de la coma que los separa.

Estructuras de control

Se incluyen las palabras reservadas tales como: if y foreach, entre las estructuras de control y entre los paréntesis no debe de existir espacio, en caso de estar vacíos. Se recomienda utilizar siempre llaves de apertura y cierre, incluso en situaciones en las que técnicamente son opcionales, esto aumenta la legibilidad y disminuye la probabilidad de errores lógicos. Si las condiciones son muy largas que sobrepasan el tamaño de la línea, estas se dividen en varias líneas. En el mejor de los casos cuando la condición es muy extensa, se puede dividir esta en variables y compararlas dentro de la estructura de control. Ver **Figura 3.3** y **Figura 3.4.**

```
public function listEnabledConferences()
{
    //type your code here;
}
```

Figura 3.3: Ejemplo de codificación (function ListEnabledConferences ()).

```
foreach ($arrayConferences as $Conference)
{
    $id = $Conference->getId();
    $title = $Conference->getTitle('es_ES');
    $path = $Conference->getPath();
    $url = str_replace('localhost', gethostbyname(gethostname()),$Conference->getUrl());
    $desc = $Conference->getDescription('es_ES');
    $arrayReturn[$i]= object_to_array(new ConferenceWS($id, $title, $path, $desc, $url));
    $i++;
}
```

Figura 3.4: Ejemplo de codificación (foreach).

3.2 Pruebas del sistema

Luego de implementados los Servicios Web se hace necesario verificar su funcionamiento y el cumplimiento de los requisitos que satisfacen las necesidades del cliente. Esta verificación se realiza a través de pruebas que permiten detectar, documentar y rectificar errores que comprometan el funcionamiento de los servicios.

La metodología WSIM propone las pruebas de aceptación con el objetivo de evaluar al final de cada iteración si se alcanzaron las metas propuestas por el cliente para cada funcionalidad. Estas fueron diseñadas, por el equipo de desarrollo conjuntamente con el cliente, basándose en las historias de usuarios.

A continuación se muestran los casos de prueba de aceptación realizados, con el cliente, en la iteración 1. Los restantes casos de prueba se muestran en el **Anexo 4**.

Caso de prueba de aceptación	**Código: HU01-P01**
Nombre:	Autenticar usuario en el sistema OJS.
Descripción:	Prueba para la funcionalidad de autenticar usuario en el sistema a partir del usuario y contraseña.
Condiciones:	Ninguna.
Entrada:	Se envía un username y un password. Se utilizarán credenciales válidas.
Resultado esperado:	Se autentica el usuario en el sistema. Se retorna un objeto de tipo SessionWS con el id de la sesión creada, el id y el username del usuario autenticado.

Evaluación de la prueba:	**1ra Iteración de la prueba:** No se autentica el usuario en el sistema. No se retorna el objeto SessionWS. **2da Iteración de la prueba:** No se retorna el objeto SessionWS. **3ra Iteración de la prueba:** Prueba satisfactoria.

Tabla 3.1: Caso de prueba de aceptación: HU01-P01.

Caso de prueba de aceptación	**Código: HU01-P02**
Nombre:	Desautenticar usuario en el sistema OJS.
Descripción:	Prueba para la funcionalidad de desautenticar usuario del sistema a partir de una sesión creada previamente por el usuario autenticado.
Condiciones:	El usuario debe estar autenticado.
Entrada:	Se envía el objeto SessionWS.
Resultado esperado:	Se desautentica el usuario del sistema. Se retorna un mensaje confirmando que el usuario ha sido desautenticado correctamente.
Evaluación de la prueba:	**1ra Iteración de la prueba:** El mensaje mostrado presenta errores ortográficos. **2da Iteración de la prueba:** Prueba satisfactoria.

Tabla 3.2: Caso de prueba de aceptación: HU01-P02.

Caso de prueba de aceptación	**Código: HU02-P01**
Nombre:	Listar revistas habilitadas.
Descripción:	Prueba para la funcionalidad de obtener un listado con el id, título, ruta, descripción y url de las revistas habilitadas.
Condiciones:	Ninguna.
Entrada:	Ninguna.
Resultado esperado:	Se retorna un listado con el id, título, ruta, descripción y url de las revistas habilitadas.
Evaluación de la prueba:	**1ra Iteración de la prueba:** Prueba satisfactoria.

Tabla 3.3: Caso de prueba de aceptación: HU02-P01.

Caso de prueba de aceptación	Código: HU02-P02
Nombre:	Listar revistas deshabilitadas.
Descripción:	Prueba para la funcionalidad de obtener un listado con el id, título, ruta, descripción y url de las revistas deshabilitadas.
Condiciones:	El usuario debe estar autenticado y tener privilegios de administración.
Entrada:	Ninguna.
Resultado esperado:	Se retorna un listado con el id, título, ruta, descripción y url de las revistas deshabilitadas.
Evaluación de la prueba:	**1ra Iteración de la prueba:** Prueba satisfactoria.

Tabla 3.4: Caso de prueba de aceptación: HU02-P02.

Caso de prueba de aceptación	Código: HU03-P01
Nombre:	Crear nueva revista.
Descripción:	Prueba para la funcionalidad de crear una nueva revista.
Condiciones:	El usuario debe estar autenticado y tener privilegios de administración.
Entrada:	Se envía el título, ruta, descripción y url de la revista que se va a crear. Se utiliza una ruta que no está siendo utilizada por otra revista.
Resultado esperado:	Se crea una nueva revista en el sistema. Se retorna un mensaje confirmando que se ha creado una revista correctamente.
Evaluación de la prueba:	**1ra Iteración de la prueba:** El mensaje mostrado presenta errores ortográficos. **2da Iteración de la prueba:** Prueba satisfactoria.

Tabla 3.5: Caso de prueba de aceptación: HU03-P01.

Caso de prueba de aceptación	Código: HU03-P02
Nombre:	Editar revista.
Descripción:	Prueba para la funcionalidad de editar una revista.
Condiciones:	El usuario debe estar autenticado y tener privilegios de administración.
Entrada:	Se envía el id de la revista, título, ruta, descripción y url. Se utiliza una ruta que no esté siendo utilizada por ninguna otra revista.
Resultado esperado:	Se edita la revista seleccionada. Se retorna un mensaje confirmando que se ha editado la revista correctamente.
Evaluación de	**1ra Iteración de la prueba:**

la prueba:	Prueba satisfactoria.

Tabla 3.6: Caso de prueba de aceptación: HU03-P02.

Caso de prueba de aceptación	**Código: HU03-P03**
Nombre:	Mostrar datos de revista habilitada.
Descripción:	Prueba para la funcionalidad de mostrar datos de una revista habilitada.
Condiciones:	El usuario debe estar autenticado.
Entrada:	Se envía el id de la revista. Se utiliza un id de una revista habilitada.
Resultado esperado:	Se retorna el id, título, ruta, descripción y url de la revista seleccionada.
Evaluación de la prueba:	**1ra Iteración de la prueba:** Prueba satisfactoria.

Tabla 3.7: Caso de prueba de aceptación: HU03-P03.

Caso de prueba de aceptación	**Código: HU03-P04**
Nombre:	Eliminar revista.
Descripción:	Prueba para la funcionalidad de eliminar una revista.
Condiciones:	El usuario debe estar autenticado y tener privilegios de administración.
Entrada:	Se envía el id de la revista. Se utiliza un id de una revista existente.
Resultado esperado:	Se elimina la revista seleccionada. Se retorna un mensaje confirmando que la revista ha sido eliminada correctamente.
Evaluación de la prueba:	**1ra Iteración de la prueba:** Prueba satisfactoria.

Tabla 3.8: Caso de prueba de aceptación: HU03-P04.

Caso de prueba de aceptación	**Código: HU03-P05**
Nombre:	Habilitar/deshabilitar revista.
Descripción:	Prueba para la funcionalidad de habilitar/deshabilitar revista.
Condiciones:	El usuario debe estar autenticado.
Entrada:	Se envía el id de la revista. Se utiliza un id de una revista habilitada.
Resultado esperado:	Se deshabilita la revista seleccionada. Se retorna un mensaje confirmando que la revista ha sido deshabilitada correctamente.
Evaluación de la prueba:	**1ra Iteración de la prueba:** Prueba satisfactoria.

Tabla 3.9: Caso de prueba de aceptación: HU03-P05.

Caso de prueba de aceptación	Código: HU02-P03
Nombre:	Listar revistas por usuario.
Descripción:	Prueba para la funcionalidad de obtener listado de revistas en la que el usuario autenticado está registrado.
Condiciones:	El usuario debe estar autenticado.
Entrada:	Se envía el id del usuario autenticado.
Resultado esperado:	Se retorna un listado con el id, título, ruta, descripción y url de las revistas en las que el usuario autenticado se encuentra registrado.
Evaluación de la prueba:	**1ra Iteración de la prueba:** Prueba satisfactoria.

Tabla 3.10: Caso de prueba de aceptación: HU02-P03.

3.3 Resultados de las pruebas

Durante las pruebas de aceptación realizadas, en las 4 iteraciones en las cuales fue dividido el proceso de desarrollo de los Servicios Web, se detectaron 10 no conformidades, clasificadas según su importancia en Significativas, No Significativas y Recomendaciones (Ver **Figura 3.5**). Entre las Significativas se encontraban errores en las funcionalidades autenticar usuario en el OJS, desautenticar usuario en el OCS.

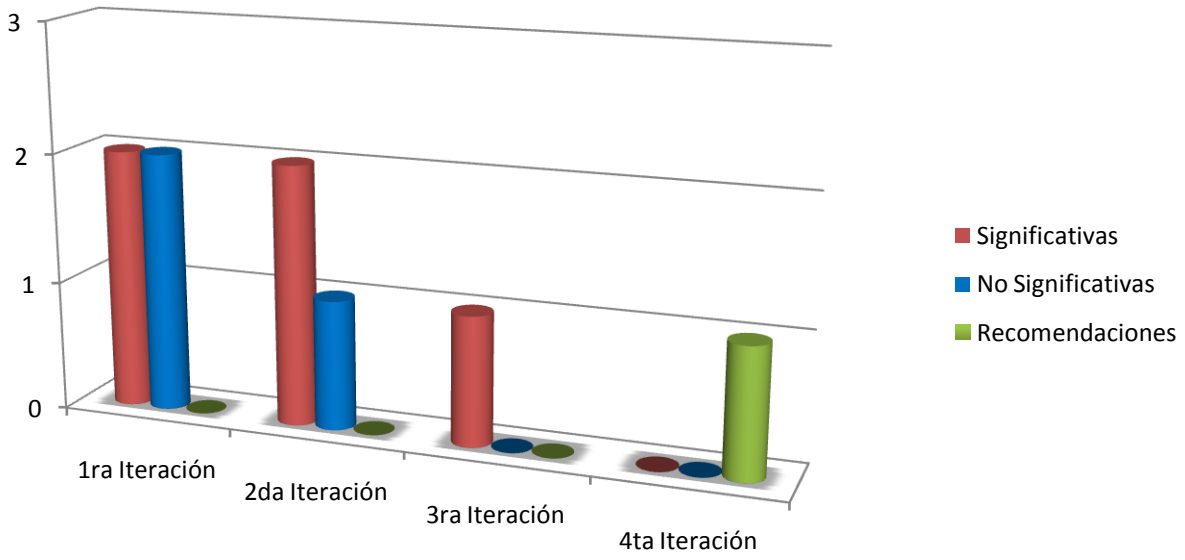

Figura 3.5: Gráfico de No conformidades.

Asimismo otras no conformidades Significativas detectadas fueron en las funcionalidades crear nuevo formulario de revisión y eliminar usuario del sistema OJS, que no mostraban ningún mensaje de confirmación.

Otros de los fallos, No Significativos, se referían a la existencia de errores ortográficos encontrados en los mensajes de confirmación de las funcionalidades autenticar usuario en el OJS, crear revista y eliminar formulario de revisión.

La recomendación sugería cambio en la funcionalidad crear archivo, en la cual se introducía entre los parámetros el valor del publicId de cada archivo. Se recomendó generar un publicId único y de forma automática para cada archivo.

3.3 Conclusiones parciales del capítulo

La construcción de la capa de servicios por iteraciones permitió la obtener versiones del producto al culminar cada iteración. Mediante las pruebas de aceptación aplicadas se pudo documentar las no conformidades detectadas en el sistema durante su desarrollo, las cuales fueron posteriormente corregidas. Al finalizar la última iteración se obtuvo una capa de servicios completamente funcional que cumple con las especificaciones establecidas por el cliente.

Conclusiones generales

Con el desarrollo de la presente investigación se arribó a las siguientes conclusiones:

- Se caracterizaron los diversos enfoques teórico-conceptuales referentes a la integración de aplicaciones, lo que permitió identificar y seleccionar a los Servicios Web como la técnica de integración más adecuada para el desarrollo de la solución.
- Se identificaron varias funcionalidades que fueron agrupadas como Servicios Web, los cuales se diseñaron según propone la metodología WSIM.
- Se obtuvo como resultado fundamental una capa de servicios permitiendo la integración de las herramientas que componen el observatorio con el portal.
- Se realizaron pruebas de aceptación a la capa de servicios, lo que permitió comprobar el correcto cumplimiento de las solicitudes del cliente.

Recomendaciones

A partir de los resultados que proporciona este trabajo de diploma, se proponen las siguientes recomendaciones:

- Extender la capa de Servicios Web con otras funcionalidades de las herramientas OJC, OCS y OHS como: navegación por ficheros, gestionar secciones de las revistas, estadísticas e informes y búsqueda de archivos.
- En aras de mejorar aspectos en la seguridad e interoperabilidad de los servicios, utilizar otros estándares WS-*.
- Realizarles pruebas de interoperabilidad, para comprobar el consumo de los servicios desde otras plataformas.

Referencias bibliográficas

1. **Montilva, Jonás A.** *Automatización de Sistemas e Integración de Software.* Mérida : s.n., 2005.

2. **Pacios, Humberto Salazar.** *Servicio Web del Sistema de Control de Tecnologías Informática y Comunicaciones.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2009. pág. 71, Tesis de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

3. **Hooping.net.** hooping.net. [En línea] 2008. [Citado el: 12 de Enero de 2012.] http://www.hooping.net/glossary/aplicaciones-web-146.aspx.

4. **Luján Mora, Sergio.** *Programación de aplicaciones web: Historia, principios básicos y clientes web.* San Vicente, España : Editorial Club Universitario, 2002. pág. 349. ISBN:9788484542063.

5. **de Soto, Adolfo R y Cuervo Fernández, Eva.** *Nuevas Tendencias en Sistemas de Información: Procesos y Servicios.* s.l. : Pecvnia, 2006.

6. **Middleware.** [En línea] 2005. [Citado el: 22 de Enero de 2012.] http://www.middleware.org/.

7. **Pelechano, Vicente.** *Servicios Web. Estándares, Extensiones y Perspectivas.* España : Universidad Politécnica de Valencia, 2005.

8. **Azán Basallo, Yasser, Díaz Estrada, Anay y González Gómez, Salvador.** *Una experiencia en integración de aplicaciones empresariales.* La Habana : Universidad de las Ciencias Informáticas, 2009.

9. **Fernández Lino, Carlos Roberto.** *Desarrollo de aplicaciones basadas en servicios aplicado a incorporar nuevas tecnologías.* Universidad Mayor San Andrés. La Paz, Bolivia : Universidad Mayor San Andrés, 2009. Tesis de Grado.

10. **Kent Sandoe, Gail Corbitt, Raymond Boykin, Aditya Saharia.** *Enterprise Integration.* s.l. : Wiley, 2001. pág. 272. ISBN: 047135993.

11. **Technology, Labrys.** Labrys Technology. [En línea] 2008. [Citado el: 24 de Enero de 2012.] http://www.labrys-tech.com/servicios/ventajas-integracion-aplicaciones.jsp#datos.

12. **Pérez, María, Mendoza, Luis E y Carvajal, Yorka.** *Orientaciones para la selección de tecnologías de integración de sistemas de software.* Caracas, Venezuela : Laboratorio de Investigación en Sistemas de Información (LISI), 2009.

13. **Mckeen, James y Smith, Heather A.** *New Developments in Practice IV: Managing the Technology Portfolio.* s.l. : Communications of the Association for Information Systems, 2002. Vol. 9.

14. **CORPORATION, MICROSOFT.** *Whitepaper: La Arquitectura Orientada a Servicios (SOA) de Microsoft aplicada al mundo real.* Diciembre, 2006.

15. **Moral, Juan Antonio Brena.** Los servicios web son la revolución informática de la nueva generación. *Desarrollo Web.* [En línea] 12 de Noviembre de 2002. [Citado el: 25 de Enero de 2012.] http://www.desarrolloweb.com/articulos/957.php.

16. **Arboleda, Liliana M.** *Servicios WEB: Distribución e integración.* Madrid, España : Universidad Icesi, 2004.

17. **Valcárcel, Ignacio García y Munilla Calvo, Eduardo.** *E-business Colaborativo.* España : Fundación Confemental, 2003. ISBN:84-95428-98-9.

18. **Dewit, Oliver.** *Asp.net Programación Web con Visual Studio y Web Matrix.* s.l. : Ediciones ENI, 2003.

19. **Saffirio, Mario.** Tecnologías de Información y Gestión de Procesos de Negocios. [En línea] 5 de Febrero de 2006. [Citado el: 12 de Enero de 2012.] http://msaffirio.wordpress.com/2006/02/05/%C2%BFque-son-los-web-services/.

20. **W3C.** W3C.org. [En línea] 2001. [Citado el: 16 de Enero de 2012.] http://www.w3c.org/TR/ws-gloss/.

21. **W3C.es.** W3C.es. [En línea] 16 de Diciembre de 2008. [Citado el: 16 de Enero de 2012.] http://www.w3c.es/divulgacion/a-z/..

22. **González, Benjamín C.** WSDL para la documentación de Servicios Web. [En línea] 2004. [Citado el: 8 de Enero de 2012.]

23. **García Rodríguez de Guzmán, Ignacio.** *Servicios Web.* Toledo, España : Universidad de Castilla, 2005.

24. **Webber, Jim, Savas, Parastatidis y Robinson, Ian.** *Rest in practice.* United States of America: Farnham : O'Reilly, 2010. ISBN: 978-0-596-80582-1.

25. **González, Benjamín.** Desarrolloweb. *Desarrolloweb.* [En línea] 7 de Julio de 2004. [Citado el: 10 de Enero de 2012.] http://www.desarrolloweb.com/articulos/1557.php.

26. **Góngora, Marcel R. Sánchez.** *Extensión de la capa de servicios web del Gestor de Contenido.* La Habana : s.n., 2009. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

27. **Márquez, Santiago.** La Güeb de Joaquín. *Introducción a la creación de servicios Web con .NET.* [En línea] 23 de Octubre de 2007. [Citado el: 12 de Enero de 2012.] http://jms32.eresmas.net/2007/textos/CreacionDeServiciosWeb.html.

28. **Gonzalez, Marco A.** authorStream. [En línea] 20 de Agosto de 2009. [Citado el: 15 de Enero de 2012.] http://www.authorstream.com/Presentation/mgonzalezr-228864-Introducci-n-los-servicios-web-services-Science-Technology-ppt-powerpoint/.

29. **Seely, S.** Seguridad HTTP y servicios Web de ASP.NET. *Microsoft Corporation.* [En línea] 2002. [Citado el: 15 de Febrero de 2012.] http://www.microsoft.com/spanish/msdn/articulos/archivo/111002/voices/httpsecurity.asp.

30. **Paz Madrid Gorelov, Vadim y De Paz Santana, Juan Francisco.** *Servicios Web.* España : Universidad de Salamanca, 2007.

31. **OASIS.** OASIS. [En línea] 2006. [Citado el: 16 de Febrero de 2012.] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.

32. **Project, Publick Knowlegde.** pkp.sfu.ca. [En línea] Simon Fraser University. [Citado el: 18 de Febrero de 2012.] http://www.pkp.sfu.ca.

33. **Lamarca Lapuente, María Jesús.** *Hipertexto, el nuevo concepto de documento en la cultura de la imagen.* Universidad Complutense de Madrid. Madrid : s.n., 2011. Tesis doctoral.

34. **Smecher, A.** *OJS Technical Reference.* Canada : Simon Fraser University, 2008. pág. 52.

35. **Pressman, Roger.** *Ingeniería del Software: Un enfoque Práctico.* s.l. : McGraw-Hill, 2005. ISBN:9701054766.

36. **Shore y Warden.** *The Art of Agile Development.* s.l. : O'Reilly, 2007.

37. **OASIS.** OASIS. *Web Service Implementation Methodology - Rational Unified Process (Example).* [En línea] 2005. [Citado el: 21 de Enero de 2012.] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=fwsi.

38. —. *FWSI IM Case Example using XP. Working Draft 03.* 2006.

39. **Alvarez, Miguel Angel.** DesarrolloWeb.com. [En línea] 18 de Julio de 2009. [Citado el: 19 de Enero de 2012.] http://www.desarrolloweb.com/articulos/497.php.

40. **PHP.** PHP: Hypertext Processor . [En línea] [Citado el: 14 de Enero de 2012.] www.php.net.

41. WSO2.org. [En línea] [Citado el: 17 de Enero de 2012.] http://wso2.com/products/web-services-framework/php/.

42. **IEEE.** *Compilation of IEEE Standard Computer Glossaries.* 1991. ISBN:1559370793.