Facultad 1

Título: Solución para la integración de los principales servicios telemáticos de la Universidad de las Ciencias Informáticas con la Red Social Universitaria.

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

Autor: Yannier Solano Farell.

Tutores: Ing. Miguel Jaeger Rodriguez Lazo.

Ing. Nayla Socarras Monzón.

La Habana, Junio del 2012

"Año 54 de la Revolución"

“El futuro de nuestra patria tiene que ser necesariamente de hombres de ciencia, tiene que ser un futuro de hombres de pensamientos, porque precisamente es lo que más estamos sembrando, lo que más estamos sembrando son oportunidades a la inteligencia (...)”

Fidel Castro Ruz.

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Centro de Informatización Universitaria de la Universidad de las Ciencias Informáticas, para que haga el uso que estime pertinente con este trabajo.

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

Autor: Yannier Solano Farell

Firma del Autor

Tutor: Ing. Miguel Jaeger Rodriguez Lazo

Firma del Tutor

Tutor: Ing. Nayla Socarras Monzón

Firma del Tutor

A mis Padres:

Por toda su confianza y haberme apoyado en todo momento a pesar de las circunstancias, daré lo mejor de mí para compensar todo lo que me han dado.

A mis Hermanos:

Por todo el apoyo prestado y brindarme su amistad, este logro es de ustedes también.

A Mili:

Por ser mi fiel compañera en las buenas y malas, por ser mi apoyo y confiar en mí en todo momento.

A mi nueva familia:

Por todo el cariño, ayuda y dedicación que me han brindado sin esperar nada a cambio, les doy las gracias a Ana, Mary y Robert.

A mis Amigos:

Por todo el apoyo y colaboración. Gracias por estar siempre conmigo.

Al Prof. Miguel Jaeger:

Por haber aceptado ser tutor de mi tesis, y por toda la enseñanza ofrecida en el transcurso de la misma y por todo el apoyo brindado en la realización de este proyecto.

A la Prof. Nayla Socarras

Por brindarme su ayuda, conocimientos y apoyo, pues fueron vitales para la realización de esta investigación.

A Todos Muchas Gracias.

A mis padres y mis hermanos.

A mi fiel compañera Mili.

A mi nueva familia Ana, Robert y Mary.

A mis amigos que me brindaron su apoyo y confianza.

A los tutores que me guiaron y apoyaron.

A la UCI por ser mi casa y escuela.

RESUMEN

El Centro de Informatización Universitaria (CENIA) lleva a cabo el desarrollo de una Red Social Universitaria (RSU), plataforma que brinda varios servicios y aplicaciones como la notificación de eventos, catálogos de software y servicio de perfil que potencian el intercambio y colaboración de la información. Debido a que el correo electrónico y la mensajería instantánea son los principales servicios informáticos brindados por la Universidad de las Ciencias informáticas que permiten a los usuarios el manejo de la información digitalizada y que se encuentran además de manera descentralizada, la presente investigación propone desarrollar una Interfaz de Programación de Aplicaciones que integre los servicios telemáticos más consumidos con la RSU y que utilice como mecanismo de integración un método que brinde rapidez y rendimiento a la hora de consumir las funcionalidades, por lo que se hace uso del estilo arquitectónico REST a través de la utilización del *framework* Restler. Este marco de trabajo define el uso del lenguaje de programación PHP, además se emplean las librerías imap, smtp, caldav y xmpphp para acceder al servicio de correo electrónico y la mensajería instantánea.

Palabras clave: Interfaz de Programación de Aplicaciones (API), Red Social Universitaria, servicios telemáticos.

ÍNDICE DE CONTENIDO

INTRODUCCIÓN 1

CAPÍTULO I. LOS SERVICIOS TELEMÁTICOS Y LAS API 5

1.1. INTRODUCCIÓN 5
1.2. GESTIÓN DE LA INFORMACIÓN EN SERVICIOS TELEMÁTICOS 5
1.3. INTERFAZ DE PROGRAMACIÓN DE APLICACIONES 11
1.4. ANÁLISIS DE LAS API 14
1.5. TECNOLOGÍAS 15
1.6. ANÁLISIS DE LAS TECNOLOGÍAS 24
1.7. CONCLUSIONES PARCIALES 25

CAPÍTULO II. DISEÑO DE LA INTERFAZ DE PROGRAMACIÓN DE APLICACIONES 26

2.1. INTRODUCCIÓN 26
2.2. DIAGNÓSTICO DEL CAMPO DE ACCIÓN 26
2.3. DESCRIPCIÓN DE LA PROPUESTA DE SOLUCIÓN 27
2.4. MODELO DE DOMINIO 28
2.5. REQUISITOS FUNCIONALES Y NO FUNCIONALES 29
2.6. PATRONES ARQUITECTÓNICOS Y DE DISEÑO 34
2.7. CONCLUSIONES PARCIALES 35

CAPÍTULO III. CONSTRUCCIÓN Y PRUEBAS DEL SOFTWARE 36

3.1. INTRODUCCIÓN 36
3.2. IMPLEMENTACIÓN 36
3.3. DIAGRAMA DE DESPLIEGUE 38
3.4. PROCESO DE PRUEBAS DE SOFTWARE 39
3.5. CONCLUSIONES PARCIALES 53

CONCLUSIONES GENERALES 54

RECOMENDACIONES 55

BIBLIOGRAFÍA REFERENCIADA 56

BIBLIOGRAFÍA CONSULTADA 59

GLOSARIO DE TÉRMINOS 62

ANEXOS 63

ANEXO I. DESCRIPCIÓN DE LOS REQUISITOS FUNCIONALES 63
ANEXO II. INTEGRACIÓN DE LA API CON SERVICIOS DE LA RSU 74

ÍNDICE DE FIGURAS

Figura 1. Elementos del proceso de software. 15
Figura 2. Estructura de clases de la API. 27
Figura 3. Modelo de dominio..... 28
Figura 4. Diagrama de despliegue. 38
Figura 5. Código de la funcionalidad Crear evento..... 42
Figura 6. Grafo de flujo asociado a la funcionalidad Crear evento. 43
Figura 7. Código de la funcionalidad Modificar evento. 45
Figura 8. Grafo de flujo asociado a la funcionalidad Modificar evento. 46
Figura 9. Código de la funcionalidad Eliminar evento..... 48
Figura 10. Grafo de flujo asociado a la funcionalidad Eliminar evento..... 49
Figura 11. Proceso de iteraciones de pruebas. 53

ÍNDICE DE TABLAS

Tabla 1 Características de REST y SOAP. 18
Tabla 2 Descripción de las principales diferencias entre REST y SOAP. 19
Tabla 3 Descripción del requisito funcional Crear evento. 30
Tabla 4 Descripción del requisito funcional Modificar evento. 31
Tabla 5 Descripción del requisito funcional Eliminar evento. 32
Tabla 6 Caminos básicos de la funcionalidad Crear evento. 43
Tabla 7 Caso de prueba 1 para el requisito funcional Crear evento. 44
Tabla 8 Caso de prueba 2 para el requisito funcional Crear evento. 44
Tabla 9 Caminos básicos de la funcionalidad Modificar evento. 46
Tabla 10 Caso de prueba 1 para el requisito funcional Modificar evento. 47
Tabla 11 Caso de prueba 2 para el requisito funcional Modificar evento. 47
Tabla 12 Caso de prueba 3 para el requisito funcional Modificar evento. 47
Tabla 13 Caso de prueba 4 para el requisito funcional Modificar evento. 48
Tabla 14 Caminos básicos de la funcionalidad Eliminar evento. 49
Tabla 15 Caso de prueba 1 para el requisito funcional Eliminar evento. 50
Tabla 16 Caso de prueba 2 para el requisito funcional Eliminar evento. 50
Tabla 17 Caso de prueba 3 para el requisito funcional Eliminar evento. 50
Tabla 18 Caso de prueba 1 para la funcionalidad Contactar usuario por correo electrónico. 51
Tabla 19 Caso de prueba 2 para la funcionalidad Contactar usuario por mensaje instantáneo. 51
Tabla 20 Caso de prueba para la funcionalidad Suscribirse a un evento. 52

ÍNDICE DE ANEXOS

Anexo 1 Descripción del requisito funcional Consultar cuota de navegación. 63
Anexo 2 Descripción del requisito funcional Enviar correo electrónico. 63
Anexo 3 Descripción del requisito funcional Mostrar correos electrónicos recibidos. 64
Anexo 4 Descripción del requisito funcional Mostrar cantidad de correos no leídos. 65
Anexo 5 Descripción del requisito funcional Crear carpeta. 65
Anexo 6 Descripción del requisito funcional Renombrar carpeta. 66
Anexo 7 Descripción del requisito funcional Eliminar carpeta. 67
Anexo 8 Descripción del requisito funcional Obtener eventos. 68
Anexo 9 Descripción del requisito funcional Crear tarea. 68
Anexo 10 Descripción del requisito funcional Modificar tarea. 69
Anexo 11 Descripción del requisito funcional Obtener tarea. 70
Anexo 12 Descripción del requisito funcional Eliminar tarea. 71
Anexo 13 Descripción del requisito funcional Enviar mensaje instantáneo. 72
Anexo 14 Descripción del requisito funcional Recibir mensaje instantáneo. 72
Anexo 15 Descripción del requisito funcional Mostrar listado de contactos. 73
Anexo 16 Integración de las cuotas de navegación con la RSU. 74

INTRODUCCIÓN

Sin duda alguna, la creación y desarrollo de los ordenadores y las redes han cambiado la forma en que vive la sociedad actual. A través de estas tecnologías son brindados un cúmulo de servicios que poseen gran calidad y brindan grandes beneficios a los usuarios. Servicios que se encuentran cimentados por un proceso evolutivo de eventos y descubrimientos que han propiciado la comunicación de la información digital.

La creación del correo electrónico constituye un elemento vital en la evolución de los servicios informáticos. Su creación se debe a Ray Tomlinson, un ingeniero que tuvo la idea de crear un programa que permitiera depositar mensajes en máquinas remotas. De esta manera, en 1971 se introduce el primer sistema de correo electrónico (DELFA, 2005).

Esta útil herramienta representa el sistema de mensajería más antiguo, además de uno de los medios más extendidos existentes en la actualidad. La gran red de computadoras Internet posee actualmente 3.146 millones de cuentas de correo electrónico, de las cuales el 75 % corresponden a usuarios privados y el 25 % restante a empresas (SARA RADICATI, 2011).

Aunque el correo electrónico juega un papel importante en el trabajo colaborativo en las redes de computadoras, se han creado tecnologías que brindan a los usuarios nuevas vías de comunicación, tal es el caso de la mensajería instantánea. Esta es creada en junio de 1996, cuando cuatro jóvenes israelíes se plantearon la necesidad de un sistema que permitiera localizar a otros usuarios en la red, dando lugar a la creación del primer sistema de mensajería instantánea (SAAVEDRA, 2009). Este servicio de gran consumo ha permitido que Internet hoy en día posea 2,6 billones de cuentas de usuarios que explotan este recurso (DÍAZ, 2012).

Notables son los avances en la transferencia y comunicación de la información que se sitúan a lo largo de la historia, aunque actualmente las redes sociales se encuentran entre los mayores medios de consumo por los usuarios. Este nuevo fenómeno es una forma de interacción social con espacios de convivencia y en conectividad, definida como un intercambio dinámico entre personas, grupos e instituciones en contextos de complejidad. Además constituyen un sistema abierto y en construcción permanente que involucra a conjuntos que se identifican en las mismas necesidades, y que se organizan para potenciar sus recursos y contribuir a la resolución de problemas, proporcionando sociabilidad, apoyo, información y un sentido de pertenencia social (CLAVIJO, 2011).

Recientemente según estudios realizados fue comprobado que redes sociales como Facebook, Twitter o MySpace poseen 640, 200 y 100 millones de usuarios registrados respectivamente, lo que implica

que alrededor de 940 millones de personas en el mundo acceden a estos sitios para obtener o intercambiar información (DESIGNERS, 2011).

Los avances creados en la rama de las tecnologías de las comunicaciones, han sido el motor impulsor hacia una nueva era donde han sido creadas nuevas herramientas y medios que permitan el trabajo colaborativo y manejo de la información. Su uso intensivo y a nivel global responde a las necesidades de comunicación de la sociedad, lo que provoca que el correo electrónico, la mensajería instantánea y las redes sociales se fusionen de forma homogénea, buscando un único objetivo en común, brindar a los usuarios un potente medio de intercambio de información.

En Cuba, la UCI como institución innovadora de excelencia científica, académica y productiva lleva a cabo en el CENIA el desarrollo de una red social adaptada a las necesidades de la universidad. Esta plataforma pretende ofrecer una gran variedad de servicios informáticos y contar con funcionalidades tanto para el desarrollo de las relaciones interpersonales, como la formación profesional de sus estudiantes y trabajadores en el ámbito académico y productivo.

La Red Social Universitaria, constituye una plataforma de gran envergadura que potencia el uso intensivo de los servicios telemáticos que brinda la UCI, servicios que representan los principales medios de comunicación en la universidad. Los usuarios acceden a servicios tales como el correo electrónico, la mensajería instantánea y el servicio de cuotas de navegación que permite conocer cuál es el porciento de la cuota de navegación que ha sido consumida.

Con la creación de la plataforma mencionada anteriormente se hace necesario el acceso frecuente a los principales servicios telemáticos brindados en la UCI, debido al constante flujo de información entre los usuarios que se encuentran navegando por la red. La universidad no brinda un mecanismo que posibilite la integración de estos servicios con otras aplicaciones, por lo que la descentralización presentada por estos, no posibilita el acceso desde un único punto a las diferentes funcionalidades que estos brindan. Esto provoca que los usuarios tengan que navegar por diferentes aplicaciones para consumirlos, incrementando el tiempo que emplean y disminuyendo la utilización de los mismos.

A partir de la problemática antes mencionada se plantea como **problema a resolver:** ¿Cómo integrar el servicio de correo electrónico, la mensajería instantánea y la consulta de cuotas de navegación de la Universidad de las Ciencias Informáticas con la Red Social Universitaria, para garantizar un espacio de convergencia de todos estos servicios, desde el cual puedan ser accedidos o consumidos por los usuarios?

El **objeto de estudio** para darle solución a este problema, está enfocado hacia la gestión de la información en servicios telemáticos, el **campo de acción** está dirigido hacia la gestión de la información en los principales servicios telemáticos brindados en la UCI.

Dadas las condiciones mencionadas anteriormente se propone como **objetivo general:** Desarrollar una Interfaz de Programación de Aplicaciones que permita integrar el servicio de correo electrónico, la mensajería instantánea y la consulta de cuotas de navegación de la Universidad de las Ciencias Informáticas con la Red Social Universitaria y sirva como interfaz única de acceso a estos.

A partir del objetivo general planteado se derivan los siguientes **objetivos específicos:**

- ✓ Caracterizar aspectos teóricos conceptuales sobre los servicios telemáticos y las Interfaces de Programación de Aplicaciones.
- ✓ Describir las herramientas y tecnologías actuales a utilizar para el desarrollo de la solución.
- ✓ Implementar el software para la integración de los servicios telemáticos más importantes de la Universidad de las Ciencias Informáticas con la Red Social Universitaria haciendo uso de las herramientas previamente seleccionadas.
- ✓ Validar el correcto funcionamiento de acuerdo a los requisitos planteados.

Para darle solución a los objetivos planteados en esta investigación fue necesaria la utilización de los **métodos científicos de la investigación** que se reflejan a continuación:

- ✓ **Histórico-lógico:** este método se utilizó para el procesamiento de la información obtenida sobre los servicios telemáticos, lo que permitió centrarse en los principales problemas que se han presentado a lo largo de la historia que han dado paso a la fusión de los mismos con otras aplicaciones.
- ✓ **Analítico-sintético:** se utilizó para el análisis y estudio de la información recopilada, separando los elementos más relevantes para facilitar el entendimiento y captura de cada una de las cualidades específicas de las API existentes en el mundo que fueron estudiadas.
- ✓ **Observación:** el uso de este método permitió la extracción de información obtenida de fuentes confiables sobre los mecanismos utilizados en la actualidad por otras aplicaciones, para llevar a cabo la integración a los servicios brindados en la universidad con la RSU.

La **justificación de esta investigación** se basa en la obtención de una API que integra los servicios telemáticos más importantes de la UCI con la RSU, proporcionando beneficios tales como:

- ✓ Proveer un medio a través del cual las aplicaciones de la RSU puedan interactuar con los servicios telemáticos de mayor consumo en la universidad.
- ✓ Dotar a los desarrolladores de un software que permita desde sus aplicaciones gestionar la información que es manipulada en el correo electrónico, la mensajería instantánea y la utilización del servicio de cuotas de navegación.
- ✓ Servir de ejemplo práctico de la utilización de nuevas tecnologías en la universidad y el país.

El presente documento de investigación está estructurado por varios capítulos, a través de los cuales son reflejados todos los conocimientos e información recopilada sobre el tema tratado, así como las características del software y la validación de sus funcionalidades, expresando el correcto funcionamiento del mismo. Los capítulos con que cuenta son:

Capítulo I: Los servicios telemáticos y las API. En este capítulo se encuentra la fundamentación teórica que sustenta este trabajo de diploma, incluyéndose un estudio de los principales servicios telemáticos brindados en la UCI, así como las distintas Interfaces de Programación de Aplicaciones existentes en el mundo relacionadas con los servicios telemáticos abordados. Se describen además las principales tecnologías y herramientas que se utilizan para el desarrollo de la aplicación.

Capítulo II: Diseño de la Interfaz de Programación de Aplicaciones. Se realiza la descripción de la propuesta que aporta una solución al problema planteado. Se caracterizan y definen las funcionalidades a brindar por el software, así como la utilización de los patrones arquitectónicos y de diseño que se evidencian.

Capítulo III: Construcción y pruebas del software. En este capítulo se describen las clases más importantes que posee el software, así como los estándares de codificación y los nodos físicos o hardware necesarios para el despliegue de la solución. Se describe también el proceso de pruebas realizadas para determinar el cumplimiento de los requisitos propuestos.

CAPÍTULO I. LOS SERVICIOS TELEMÁTICOS Y LAS API

1.1. INTRODUCCIÓN

Actualmente en la UCI existen múltiples centros que llevan a cabo la construcción de servicios informáticos. Estos brindan una solución a problemas planteados por los usuarios y permiten ser integrados a otras aplicaciones a través de mecanismos estándares.

En el actual capítulo se exponen los elementos relacionados con la investigación realizada sobre los servicios telemáticos más importantes que se brindan en la UCI. Se abordan además las principales características y ventajas de las API existentes en el mundo que interactúan con los servicios telemáticos descritos en la presente investigación. Se realiza un estudio a algunos de los mecanismos de interacción entre aplicaciones, así como las tecnologías y herramientas actuales que se utilizan para el desarrollo del software.

1.2. GESTIÓN DE LA INFORMACIÓN EN SERVICIOS TELEMÁTICOS

La gestión de la información puede verse desde dos perspectivas, la organizacional y la informática.

Desde el punto de vista organizacional se compone por:

- ✓ Las Tecnologías de Información y las Comunicaciones, estas son tecnologías que se necesitan para la gestión y transformación de la información, y muy en particular el uso de ordenadores y programas que permiten crear, modificar, almacenar, proteger y recuperar esa información (VARO MARTÍNEZ, 2009).
- ✓ Los sistemas de información, ya que cuando se habla de su gestión se refiere a la gestión que se realiza dentro de estos sistemas dígase base datos, servicios informáticos, intranets, entre otros.

Por lo que se puede afirmar que la gestión de la información:

Permite captar, almacenar y distribuir información del entorno de la institución y de sus operaciones internas y externas; para apoyar la toma de decisiones, en las funciones o áreas, en la comunicación, coordinación, control y análisis de la organización. Estos sistemas se encuentran presentes en todas las empresas e instituciones, cumpliendo con la función de transformar los datos puros en información útil (FRÁNQUIZ et al., 2007).

1.2.1. Servicios telemáticos

Los servicios telemáticos cubren un campo científico y tecnológico de una considerable amplitud. Engloban el estudio, diseño, gestión y aplicación de las redes y servicios de comunicaciones para el transporte, almacenamiento y procesamiento de cualquier tipo de información digital.

Son servicios que utilizan generalmente técnicas de procesamiento de la información en forma remota, combinando el empleo de ordenadores y redes de comunicaciones. Permiten que el usuario desde su ordenador o terminal inteligente reciba o envíe información pública o privada que reside en ordenadores conectados a redes o efectúe operaciones tales como la consulta y descarga de ficheros, reservas y transacciones comerciales o bancarias. Se soportan en servicios de transmisión de datos, sobre los que añaden valor o utilidad al cliente final (REYES, 2009).

Actualmente en la UCI se brinda una gran variedad de servicios telemáticos, dentro de los que se encuentran el servicio de correo electrónico nacional e internacional, la navegación básica en Internet, el acceso remoto por teléfono a la red, audio y video conferencia, servicio de mensajería instantánea nacional e internacional y FTP. De esta gama de servicios, algunos de ellos son consumidos con mayor frecuencia que otros en dependencia de las necesidades y preferencias de los usuarios, tal es el caso del correo electrónico, la mensajería instantánea y el servicio de cuotas de navegación.

1.2.1.1. Servicio de correo electrónico

El correo electrónico, también conocido como *e-mail*, permite a los usuarios enviar y recibir mensajes de forma rápida mediante sistemas de comunicación electrónicos. Este medio permite enviar no solo textos, sino todo tipo de información digital. Este servicio está basado en protocolos de redes que posibilitan el envío y recepción de los mensajes. A continuación se reflejan algunas de las principales características que poseen estos protocolos.

- ***Simple Mail Transfer Protocol* (SMTP):** protocolo simple de transferencia de correo basado en textos utilizados para el intercambio de mensajes de correo electrónico entre computadoras u otros dispositivos. Dentro de sus principales características se encuentran que es independiente del subsistema de transmisión y sólo requiere un canal fiable ordenado por flujo de datos. Permite que un proceso pueda transferir un correo electrónico a otro proceso en la misma red o de alguna otra red, a través de un proceso de retransmisión o puerta de enlace accesible a ambas redes. Se basa en el modelo cliente-servidor (KLENSIN, 2001).

  Ventajas

  - ✓ Informa mediante códigos enumerados si el servicio se encuentra o no disponible, si fue procesado correctamente el mensaje.

- ✓ Posee servicios para la transmisión de datos que no se pueden representar con textos ASCII[1].
- ✓ Posee un esquema simple de direccionamiento.

Desventajas

- ✓ Carece de ciertos tipos de funciones.
- ✓ Su simplicidad en comparación con otros protocolos limita su utilidad.

➔ ***Post Office Protocol*** **(POP):** protocolo de la oficina de correo en clientes locales, se encuentra actualmente en la versión 3 y está diseñado no para enviar, sino obtener los mensajes de correo electrónico almacenados en un servidor remoto. Su objetivo principal es gestionar los mensajes sin tener que estar conectado.

Ventajas

- ✓ Si son administrados múltiples correos electrónicos desde un mismo cliente, pueden ser revisados los mensajes desde un mismo lugar.
- ✓ Los mensajes son almacenados localmente por lo que siempre se puede acceder a los mensajes, independientemente de si existe o no una conexión.
- ✓ La apertura de archivos adjuntos funciona generalmente más rápido ya que éstos son descargados simultáneamente con el contenido del mensaje.

Desventajas

- ✓ Si son recibidos archivos o contenidos que puedan infectar la computadora, puede afectar el equipo ya que los mensajes son descargados.
- ✓ En caso de que ocurra un problema en la computadora toda la información puede perderse.
- ✓ Si se encuentra configurado el cliente para dejar una copia en el servidor, a medida que la cuenta crezca en tamaño, tarda más en revisar los nuevos correos, ya que son revisados que correos ya fueron descargados y cuales no (VOLPL, 2009).

➔ ***Internet Message Access Protocol*** **(IMAP):** es un protocolo de red de acceso a mensajes electrónicos almacenados en un servidor. Se encuentra en la versión 4 y permite visualizar los mensajes de manera remota y no descargarlos del servidor (CRISPIN, 2003).

Ventajas

- ✓ Todos los correos están alojados en el servidor.
- ✓ Puede ser cambiado el cliente de correo que los correos son vistos exactamente igual.
- ✓ Si ocurre un problema en la computadora y es perdida toda la información, los correos se encuentran intactos ya que están alojados en el servidor.

[1] ***American Standard Code for Information Interchange*** **(ASCII):** código estándar estadounidense para el intercambio de información, es un código de caracteres basado en el alfabeto latino.

Desventajas

- ✓ Los correos están disponibles solamente en el servidor, por lo que no se puede interactuar con ellos a menos que exista una conexión de red.
- ✓ Debido a que todos los mensajes están alojados en el servidor, la velocidad de acceso depende de la conexión a la red y esto puede hacer que el manejo de correos sea un poco más lento (VOLPL, 2009).

➔ ***Multipurpose Internet Mail Extension*** **(MIME)**: extensiones multipropósito de correo de Internet, son una serie de convenciones o especificaciones dirigidas al intercambio a través de todo tipo de archivos (texto, audio, vídeo, etc.), de forma transparente para el usuario (FREED, 1996).

Una parte importante del MIME está dedicada a mejorar las posibilidades de transferencia de texto en distintos idiomas y alfabetos. En sentido general las extensiones van encaminadas a soportar:

- ✓ Texto en conjuntos de caracteres distintos de us-ascii.
- ✓ Adjuntos que no son de tipo texto.
- ✓ Cuerpos de mensajes con múltiples partes.
- ✓ Información de encabezados con conjuntos de caracteres distintos de ascii.

En la universidad actualmente para el envío de mensajes electrónicos se utiliza el cliente web de correo electrónico Zimbra, aunque puede ser configurado otro cliente usando el servidor de correo saliente o envío SMTP a través de smtp.uci.cu, este requiere la autenticación de los usuarios, tiene la opción de cifrado TSL[2] y se configura a través del puerto 25. La configuración del servidor de correo entrante IMAP se realiza mediante la dirección imap.uci.cu, requiere también la autenticación de los usuarios, ofrece cifrado de tipo TSL por el puerto 143 y SSL[3] por el puerto 993.

1.2.1.2. Servicio de mensajería instantánea

El servicio de mensajería instantánea está basado en *Extensible Messaging and Presence Protocol* (XMPP), un protocolo extensible, abierto y estándar para el intercambio en tiempo real de mensajes.

➔ **Servicios:** en este contexto un servicio es una característica o función que es utilizada por otras aplicaciones. El servicio de mensajería instantánea proporciona servicios básicos tales como:

[2] ***Transport Layer Security***: seguridad de la capa de transporte, protocolo criptográfico que proporciona comunicaciones seguras por una red.

[3] ***Secure Socket Layer****:* capa de conexión segura que proporciona sus servicios de seguridad cifrando los datos intercambiados entre un servidor y un cliente.

- ✓ **Canal cifrado:** proporciona cifrado de la conexión entre un cliente y un servidor, o entre dos servidores.
- ✓ **Autenticación:** las entidades para intentar comunicarse a través en la red deben estar autenticadas primero en el servidor, que actúa como una especie de portero para acceder a la red.
- ✓ **Presencia:** permite conocer la disponibilidad de la red de otras entidades. En el nivel más básico, un servicio de presencia expresa si la entidad está en línea y disponibles para la comunicación, o fuera de línea y no está disponible. El intercambio de información de presencia se basa en una suscripción de presencia explícita entre dos entidades con el fin de proteger la privacidad de la información del usuario.
- ✓ **Las listas de contactos:** almacena una lista de contactos en un servidor XMPP.
- ✓ **Uno a uno de mensajería:** establece el envío de mensajes a otra entidad.

➔ **Aplicaciones:** debido a que brinda un gran variedad de servicios, a partir de ellos pueden ser construidas aplicaciones como:

- ✓ **La mensajería instantánea:** los sistemas clásicos con los que los usuarios están familiarizados son la combinación de tres de los servicios básicos, la presencia, listas de contactos y mensajes de uno a uno.
- ✓ **Grupo de charla:** el servicio de multiconferencia de mensajería permite construir los sistemas de grupo de charla, utilizados para aplicaciones específicas como de sistemas de negociación en tiempo real y aulas virtuales.
- ✓ **Geolocalización:** permite la construcción de aplicaciones basadas en la localización, tales como seguimiento de vehículos o localización de personas.

En la UCI, el servicio de mensajería instantánea se encuentra configurado para permitir el intercambio de mensajes entre usuarios registrados en los servidores de la institución y otros servidores a nivel nacional e internacional. Además solo permite dos conexiones simultáneas para hacer un uso óptimo del servicio y los recursos de hardware que el mismo necesita.

1.2.1.3. Servicio de consulta de cuotas de navegación

Para la navegación por sitios de Interne a los usuarios de la UCI se les asignan cuotas que permiten acceder a estos sitios para descargar, obtener o enviar información mientras no ha sido agotada la cuota asignada. La consulta de las cuotas de navegación es un servicio web brindado en la Universidad a través del cual puede obtenerse el porciento consumido durante un período de tiempo determinado, así como la cuota total asignada y el nivel de navegación que posee un usuario. Este nivel de navegación está restringido en dependencia del rol o cargo que desempeñe en la universidad

un estudiante o trabajador, o el año en que se encuentre un estudiante. Dentro de los niveles de acceso definidos se encuentran:

- ✓ **Acceso Prohibido:** el usuario no posee una cuenta con acceso a Internet ni a los sitios de navegación nacional.
- ✓ **Acceso Pleno:** se puede acceder a cualquier sitio tanto nacional como internacional.
- ✓ **Acceso a los sitios de navegación nacional:** el usuario puede entrar a cualquier página que esté dentro del dominio del país.
- ✓ **Acceso a los sitios definidos en la navegación básica:** son una serie de sitios de Internet a los cuales se da acceso para el desarrollo de la docencia y la investigación.

Para los usuarios que no poseen cuotas de acceso pleno, se brindan sitios de navegación básica, tales como:

- ✓ Sitios de revistas, diarios, periódicos.
- ✓ Sitios de parques tecnológicos y de organizaciones.
- ✓ Sitios de programación, ingeniería de software, empresas y compañías.
- ✓ Sitios de universidades.

1.2.1.4. Los servicios telemáticos en las redes sociales

A continuación se describe como los servicios telemáticos son implementados por las redes sociales. Se selecciona un conjunto de las principales redes sociales más populares en el mundo. Esta descripción se realiza con el objetivo de determinar las estrategias que sirvan como elementos positivos para el desarrollo de la propuesta de solución.

Las redes sociales seleccionadas son:

- ✓ **Facebook:** brinda la capacidad para comunicarse con personas dentro y fuera de la red social utilizando el correo electrónico, mensajería interna, mensajería instantánea y mensajes de texto utilizando múltiples dispositivos, incluidos los móviles. El servicio de correo electrónico brindado no soporta IMAP. Posee una bandeja de entrada social, esta funcionalidad utiliza los datos de Facebook para saber qué mensajes de correo electrónico son importantes y cuáles no son tan importantes. Permite acceder al servicio de mensajería instantánea desde cualquier cliente del servicio que pueda conectarse mediante XMPP. A través de este servicio brinda el envío y recepción de mensajes de texto sin formato (mensajes no HTML).
- ✓ **Twitter:** a través del sistema *TwitterMail*, los usuarios pueden enviar mensajes electrónicos. Las respuestas son recibidas en la plataforma y en el buzón del correo electrónico. Además, brinda un servicio de mensajería instantánea que permite enviar y recibir mensajes. Permite enviar mensajes desde clientes de mensajería y enviarlos directamente a la red social sin tener

que introducir una cuenta. También los usuarios pueden recibir los *tweets*[4] directamente en un cliente de mensajería instantánea.

- ✓ **MySpace:** agrega una dirección de correo electrónico para el intercambio de mensajes entre los miembros de la aplicación. Brinda un almacenamiento ilimitado, permitiendo mantener todos los mensajes deseados. No ofrece acceso POP o IMAP y no se puede recuperar el correo desde cuentas externas. Se apoya en la popularidad de los servicios que brinda integrando el servicio de perfil y la Web de los usuarios con la mensajería instantánea. Además integra a este servicio un reproductor de música, el envío de imágenes arrastrando y soltando, avisos de nuevos mensajes en el perfil o solicitudes de nuevos amigos.
- ✓ **GooglePlus:** brinda a los usuarios la posibilidad de comunicarse por correo electrónico o chat. Esta red social brinda un *widget*[5] que se coloca en la parte derecha de la plataforma y permite enviar un mensaje instantáneo al seleccionar un usuario en una la lista de amigos mostrada. A diferencia de Facebook y Twitter, permite enviar un mensaje privado a un amigo a través del chat o directamente al buzón de correo electrónico. Los usuarios pueden elegir la opción de recibir o no un correo electrónico.

1.3. INTERFAZ DE PROGRAMACIÓN DE APLICACIONES

Según expresó Nick Stoughton en abril del 2005, las siglas API provienen del inglés *Application Programming Interface*. Estas constituyen un conjunto de funciones y procedimientos, o métodos, en la programación. Son usadas generalmente en las bibliotecas (STOUGHTON, 2005).

Características

Una interfaz de programación representa una interfaz de comunicación entre componentes de software. Se trata del conjunto de llamadas a bibliotecas que ofrecen acceso a servicios desde los procesos y representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general. De esta forma, los programadores se benefician de las ventajas de la API haciendo uso de su

[4]Proviene de la red social Twitter y significa una publicación de un mensaje o la actualización en el estado de un usuario.

[5] Pequeña aplicación o programa, usualmente presentado en archivos o ficheros pequeños. Entre sus objetivos están dar fácil acceso a funciones frecuentemente usadas y proveer información de forma visual.

funcionalidad, evitándose el trabajo de programar todo desde el principio. El software que proporciona una cierta API generalmente es llamado la implementación de esa API.

Ventajas

- ✓ Una API hace fácil el trabajo de desarrollo de un programa, ya que debe proveer todos los bloques para construirlo. El programador lo único que hace es poner todos los bloques juntos.
- ✓ Están diseñadas especialmente para los programadores, ya que garantizan que todos los programas que utilizan API posean las mismas interfaces e implementación de las funcionalidades. Esto le facilita al usuario aprender la lógica de nuevos programas.

1.3.1. Zimbra API

Interfaz de Programación de Aplicaciones desarrollada en el 2007 por Zachary Tirrell en el lenguaje PHP. Su versión más reciente es la 1.3. Permite crear aplicaciones web interactuando con el servidor web de correo electrónico Zimbra. Posee licencia *General Public License* (GPL). Esta permite conectarse a través de varios métodos de comunicación y gestionar la información contenida en el servidor de correo Zimbra desde otra aplicación. Dentro de sus funcionalidades se encuentran:

- ➔ **REST**
  - ✓ Obtener carpetas.
  - ✓ Importar mensajes.
  - ✓ Obtener contactos.
  - ✓ Importar contactos.
  - ✓ Obtener calendario.
  - ✓ Importar eventos.
  - ✓ Importar tareas.
  - ✓ Obtener archivos del maletín.
  - ✓ Exportar buzón del correo.
- ➔ **SOAP**
  - ✓ Administración de usuarios.
  - ✓ Obtener mensajes.
  - ✓ Obtener contenido de los mensajes.
  - ✓ Obtener eventos.
  - ✓ Obtener tareas.
  - ✓ Obtener suscripciones del calendario.

- ✓ Obtener suscripciones de tareas.
- ✓ Obtener carpetas.
- ✓ Obtener preferencias.
- ✓ Asignar preferencias.

1.3.2. *Javamail* API

Este API es un paquete opcional para leer, componer y enviar mensajes electrónicos. Se usa para crear programas similares al cliente de correo electrónico *Microsoft Outlook*. Su propósito principal es leer y escribir mensajes electrónicos e interactuar con los programas que se encargan del envío de estos mensajes usando el lenguaje de programación Java. Ofrece la posibilidad de desarrollar clientes y aplicaciones compatibles con el correo electrónico (ÁLAMO y HERRERO, 2003).

Permite implementar muchas funcionalidades que brindan los clientes de correo electrónico, tales como la lectura, envío y reenvío, búsqueda, borrado y adición de archivos adjuntos a mensajes. Utiliza básicamente los protocolos SMTP, POP, IMAP, MIME y NNTP[6].

Javamail está formada por un conjunto de clases abstractas que modelan las partes de un sistema de correo electrónico. Estas clases son:

- ✓ **Sesión:** define una sesión de correo básica.
- ✓ **Mensajes:** representa un mensaje de correo electrónico.
- ✓ **Direcciones:** indica la dirección a la que se va a enviar el mensaje.
- ✓ **Autenticación**: se utiliza para proteger a los recursos no autorizados mediante un nombre de usuario y una contraseña.
- ✓ **Transporte**: representa un protocolo de transporte específico.
- ✓ **Almacenamiento:** representa una base de datos de mensajes que es mantenida por un servidor de correos electrónicos y agrupada por el usuario.
- ✓ **Carpetas:** proporciona un método para recuperar los mensajes.

1.3.3. XMPPHP

Librería desarrollada en el lenguaje de programación PHP. Esta biblioteca está basada en XMPP, protocolo que permite el intercambio de mensaje utilizando el estándar XML[7].

[6] ***Networks News Transport Protocol*:** protocolo basado en el modelo cliente/servidor que define el formato, la sintaxis y el contenido de una comunicación entre dos computadoras.

Es particularmente útil para el servicio de mensajería instantánea, y consta de dos ficheros XML que inician una conexión y la finalizan en el extremo de la misma. Un fichero es el servidor que se comunica con el cliente, y el otro es el cliente de la comunicación con el servidor.

- ➔ **XMLStream:** en XMPPHP, el proceso principal se hace por XMLStream.php. Este contiene una clase que se utiliza para realizar la transmisión de un cliente o servidor de *socket*[8] XML. Se ocupa de las conexiones de la clase, cifrado TLS y la gestión de eventos entrantes.
- ➔ **StreamProcessing:** El procesamiento de un XMLStream es realizado de tres maneras:

- ✓ Procesamiento de forma indefinida.
- ✓ Procesamiento de un número determinado de segundos.
- ✓ Procesamiento con eventos. Este método es particularmente útil, ya que consiste en procesar un XML para un usuario autenticado y tener una sesión establecida. Permite recuperar la lista de contactos y enviar el mensaje de presencia.

1.4. ANÁLISIS DE LAS API

A partir del estudio realizado a algunas de las API que pueden servir en el desarrollo de la solución se decide que:

- ✓ **Zimbra API:** no cumple con las condiciones necesarias para la integración del servicio de correo electrónico con la RSU. Este software solo permite la interacción de aplicaciones con el servidor de correo electrónico Zimbra. Aunque este servidor es el que se encuentra en uso en la universidad, en caso de ser instalado un nuevo servidor sus funcionalidades dejan de ser útiles para la red social. Un aporte significativo del estudio de esta herramienta es que toma como medio de interacción con otras aplicaciones REST y SOAP, mecanismos a estudiar en esta investigación.
- ✓ ***Javamail* API:** se encuentra desarrollada para el lenguaje de programación Java, un elemento negativo en relación a lo que se desea realizar en esta investigación. Su estructura abstracta de carpetas aporta un elemento importante para el API a desarrollar, a partir de la cual se define separar las clases a implementar en dependencia del servicio telemático a brindar.

---

[7] ***eXtensible Markup Language*:** es un meta-lenguaje que permite definir lenguajes de marcado. LÓPEZ, D. C. y DÍAZ, L. B. M. *Implementación del Portafolio Digital de la Universidad de las Ciencias Informáticas.* Universidad de las Ciencias Informáticas, 2011.

[8] Designa un concepto abstracto por el cual dos programas pueden intercambiar cualquier flujo de datos de manera fiable y ordenada.

- ✓ **XMPPHP:** aporta varios beneficios dentro de los que se encuentran su desarrollo en PHP, cuenta con soporte para protocolos criptográficos que permiten comunicaciones seguras en las redes y permite la conexión a cualquier servidor XMPP.

1.5. TECNOLOGÍAS

A continuación son descritas las tecnologías estudiadas durante la investigación, de ellas se seleccionan las que cumplen con los elementos necesarios para la construcción del software.

1.5.1. Proceso de desarrollo

Según Pressman el proceso de software es un marco de trabajo de las tareas que se requieren para construir un software de alta calidad (PRESSMAN, 2002), ver Figura 1.

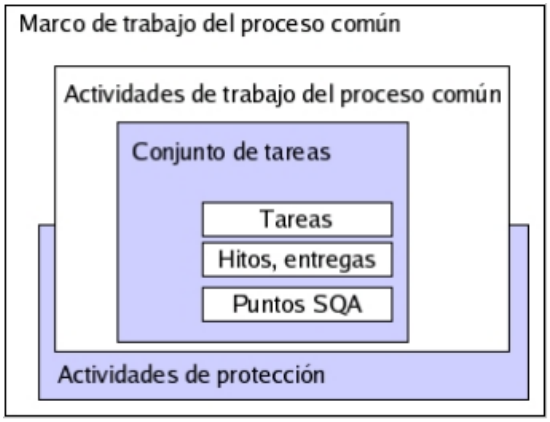


Figura 1. Elementos del proceso de software.

Los elementos involucrados se describen a continuación:

- ✓ **Un marco común del proceso**: definiendo un pequeño número de actividades del marco de trabajo que son aplicables a todos los proyectos de software, con independencia del tamaño o complejidad.
- ✓ **Un conjunto de tareas**: cada una es una colección de tareas de ingeniería del software, hitos de proyectos, entregas y productos de trabajo del software, y puntos de garantía de calidad, que permiten que las actividades del marco de trabajo se adapten a las características del proyecto de software y los requisitos del equipo del proyecto.
- ✓ **Las actividades de protección**: tales como garantía de calidad del software, gestión de configuración del software y medición, abarcan el modelo del proceso. Las actividades de

protección son independientes de cualquier actividad del marco de trabajo y aparecen durante todo el proceso.

Enfoque ágil

Este enfoque ágil de las metodologías de desarrollo de software surge como alternativa a las metodologías formales a las que consideraban excesivamente pesadas y rígidas por su carácter normativo y fuerte dependencia de planificaciones detalladas previas al desarrollo. Dicho enfoque supera los defectos de estas, ya que se basan en:

- ✓ Valorar más a los individuos y su interacción que a los procesos y las herramientas.
- ✓ Valorar más el software que funciona que la documentación exhaustiva.
- ✓ Valorar más la colaboración con el cliente que la negociación contractual.
- ✓ Valorar más la respuesta al cambio que el seguimiento de un plan.
- ✓ Valorar más el software que funciona que la documentación excesiva. Genera la documentación necesaria del desarrollo del producto (BECK *et al.*, 2001).

Integración de Modelos de Madurez de Capacidades

CMMI por sus siglas en inglés ***Capability Maturity Model Integration*** es un modelo de calidad del software que provee lineamientos para las empresas, organizaciones o áreas que deseen una mejora continua y efectiva en sus procesos de desarrollo. Para esto propone 25 áreas de procesos las cuales están agrupadas en 5 niveles (iniciado, gestionado, definido, gestionado cuantitativamente, en optimización) de madurez para clasificar a las empresas en función de qué áreas de proceso consiguen sus objetivos y se gestionan con principios de ingeniería. Los niveles de madurez son los siguientes:
El nivel 2 de CMMI es el gestionado, se enfoca en la gestión de procesos y define siete áreas:

Gestión de requisitos: mantiene los requerimientos, describe las actividades para obtener y controlar los cambios a los requerimientos, y asegura que otros planes y datos relevantes se mantengan actualizados.
Planificación del proyecto: incluye el desarrollo del plan de proyecto, la involucración de las partes interesadas de forma apropiada, la obtención de compromisos con el plan y el mantenimiento del plan.
Seguimiento y control del proyecto: incluye las actividades de monitorización y toma de acciones correctivas. Estas acciones pueden incluir la re-planificación del proyecto.

Gestión de acuerdos con proveedores: trata la necesidad del proyecto de adquirir aquellas partes del trabajo que son producidas por proveedores. Las fuentes de productos que pueden ser usadas para satisfacer los requerimientos del proyecto se identifican de forma proactiva.
Medición y análisis: da soporte a todas las áreas de proceso, proporcionando prácticas específicas que guían a los proyectos y a las organizaciones durante la alineación de las necesidades y objetivos de medición con una forma de medir que proporcionará resultados objetivos.
Aseguramiento de calidad de procesos y productos: da soporte a todas las áreas de proceso, proporcionando prácticas específicas para evaluar objetivamente los procesos, los productos de trabajo y los servicios realizados frente a las descripciones aplicables de procesos, estándares y procedimientos.
Gestión de configuración: brinda soporte a todas las áreas de proceso, estableciendo y manteniendo la integridad de los productos de trabajo usando la identificación de la configuración, el control de la configuración, los informes de estado de la configuración y las auditorías de la configuración (MÉNDEZ *et al.*, 2011).

1.5.2. Métodos de comunicación entre aplicaciones

Del estudio realizado a las API existentes en el capítulo anterior, se obtiene cuales emplean como métodos o mecanismos para la comunicación con otras aplicaciones REST y SOAP, por lo que se decide realizar un análisis de estos, para definir el mecanismo a utilizar en el desarrollo de la propuesta.

Representational State Transfer (REST)

La Transferencia de Estado Representacional define una colección de principios arquitectónicos por los cuales se diseñan servicios web haciendo foco en los recursos del sistema, incluyendo cómo se accede al estado de dichos recursos y cómo se transfieren por HTTP[9] hacia clientes escritos en diversos lenguajes (MASSÉ, 2012).
Los principios arquitectónicos especificados por REST resumen como los recursos son definidos y diseccionados. El término frecuentemente es utilizado en el sentido de describir a cualquier interfaz que transmite datos específicos sobre HTTP sin una capa adicional, como hace SOAP.
Cabe destacar que REST no es un estándar, ya que es tan solo un estilo de arquitectura, pero se encuentra basado en estándares, tales como:

[9]***Hypertext Transfer Protocol*****:** Protocolo de transferencia de hipertexto. Es un protocolo usado en cada transacción de la World Wide Web.

- ✓ HTTP.
- ✓ URL.
- ✓ Representación de los recursos: XML, HTML, GIF y JPEG.
- ✓ Tipos MIME: text/xml, text/html.

Simple Object Access Protocol (SOAP)

Es un protocolo que permite la comunicación entre aplicaciones a través de mensajes por medio de Internet. Es independiente de la plataforma y del lenguaje (LAZO y REYES, 2007). Define cómo dos objetos en diferentes procesos se comunican por medio de intercambio de datos XML. Su principal beneficio recae en ser fuertemente acoplado, lo que permite poder ser testeado y depurado antes de poner en marcha una aplicación.

Un diseño basado en SOAP es adecuado cuando:

- ✓ Se establece un contrato formal para la descripción de la interfaz que el servicio ofrece. El lenguaje de Descripción de Servicios Web (WSDL), permite describir con detalles un servicio web.
- ✓ La arquitectura debe abordar requerimientos complejos no funcionales. Muchas especificaciones de servicios web abordan tales requisitos y establecen un vocabulario común para ellos. Algunos ejemplos incluyen transacciones, seguridad y direccionamiento.

1.5.2.1. Características generales

A continuación se muestra en la Tabla 1 las principales características que poseen estos mecanismos de interacción entre aplicaciones.

Tabla 1 Características de REST y SOAP.

	REST	SOAP
Características	Las operaciones se definen en los mensajes. Una dirección única para cada instancia del proceso. Cada objeto soporta operaciones estándares definidas. Componentes débilmente acoplados.	Las operaciones son definidas como puertos WSDL. Dirección única para todas las operaciones. Múltiple instancias del proceso comparten la misma operación. Componentes fuertemente acoplados.
Ventajas	Bajo consumo de recursos.	Fácil (generalmente) de utilizar.

	Las instancias del proceso son creadas explícitamente. Generalmente fácil de construir y adoptar.	La depuración es posible. Las operaciones complejas pueden ser escondidas detrás de una fachada. Herramientas de desarrollo.
Posibles desventajas	Gran número de objetos. Manejar el espacio de nombres (URIs) puede ser engorroso. Pocas herramientas de desarrollo.	Los clientes necesitan saber las operaciones y su semántica antes del uso. Los clientes necesitan puertos dedicados para diferentes tipos de notificaciones.

1.5.2.2. Principales diferencias

En la Tabla 2, se muestran las principales diferencias que poseen los mecanismos de interacción entre aplicaciones REST y SOAP.

Tabla 2 Descripción de las principales diferencias entre REST y SOAP.

	REST	**SOAP**
Tecnología	Interacción dirigida por el usuario por medio de formularios. Pocas operaciones con muchos recursos. Mecanismo consistente de nombrado de recursos (URI).	Muchas operaciones con pocos recursos. Falta de un mecanismo de nombrado. Se centra en el diseño de aplicaciones distribuidas.
Protocolo	XML autodescriptivo. HTTP. HTTP es un protocolo de aplicación. Síncrono.	Tipado fuerte, XML *Schema*[10]. Independiente del transporte. HTTP es un protocolo de transporte. Síncrono y Asíncrono.
Descripción del servicio	Confía en documentos orientados al usuario que define las direcciones de petición y las respuestas.	WSDL. Tipado fuerte. Se pueden construir automáticamente clientes por medio del WSDL.
Gestión del estado	El servidor no tiene estado. Los recursos contienen datos y enlaces representando transiciones a estados válidos. Los clientes mantienen el estado siguiendo los	El servidor puede mantener el estado de la conversación. Los mensajes solo contienen datos. Los clientes mantienen el estado

[10] lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML.

	enlaces. Técnicas para añadir sesiones: Cookies	suponiendo el estado del servicio. Técnicas para añadir sesiones: Cabecera de sesión (no estándar)
Seguridad	HTTPS. Implementado desde hace muchos años. Comunicación punto a punto segura.	*WS-Security*[11]. Las implementaciones están comenzando a aparecer ahora. Comunicación origen a destino segura.
Metodología de diseño	Definir URL para direccionarlos. Distinguir los recursos de solo lectura (GET), de los modificables (POST, PUT, DELETE).	Listar las operaciones del servicio en el documento WSDL. Definir un modelo de datos para el contenido de los mensajes.

1.5.3. Marco de trabajo

Un marco de trabajo o framework, simplifica considerablemente el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Proporciona una estructura al código fuente, forzando al desarrollador a crear código más legible y fácil de mantener, además facilita la programación de aplicaciones pues encapsula operaciones complejas en instrucciones sencillas.

1.5.3.1. Restler 2.0

Es un marco de trabajo único y completo basado en el lenguaje de programación PHP. Permite a los usuarios enviar datos en cualquiera de los formatos habilitados o como una cadena de consulta o formulario de envío, que convierte los datos y asigna los parámetros de la función. Soporta los métodos de petición de HTTP como son GET, POST, PUT y DELETE, además deja al desarrollador la opción de solo asignar las funciones al método URL. Es gratuito y de código abierto bajo licencia GNU *Lesser General Public License*. Posee métodos públicos y protegidos que gestionan sus propias dependencias. Además permite agregar seguridad a las API desarrolladas ya que posee esquemas de autenticación conectables.

[11] Seguridad en Servicios Web es un protocolo de comunicaciones que suministra un medio para aplicar seguridad a los servicios web.

1.5.3.2. DAVE

DAVE es un acrónimo que significa eliminar, agregar, editar y ver. Estos 4 métodos constituyen la funcionalidad principal de muchas aplicaciones web transaccionales (TAHLER, 2012). Comprende 2 tipos de metodología de seguridad. Las de acciones que pueden ser llamadas por cualquier persona y las que se pueden aplicar a un usuario basado en su seguridad. Dentro de sus funcionalidades se encuentran:

- ✓ La abstracción de acciones como eliminar, añadir, editar y ver.
- ✓ Modelado de base de datos activa en la marcha o fija según sus necesidades.
- ✓ Clase basada en la abstracción de conexiones con MySQL.
- ✓ El manejo de errores simple y saneamiento de entrada.
- ✓ XML y JSON como tipos de datos de PHP.

1.5.4. Lenguaje de programación

Un lenguaje de programación es un idioma artificial diseñado para expresar funciones que pueden ser llevadas a cabo por máquinas como las computadoras. Los lenguajes de programación son usados para crear programas que controlan el comportamiento físico y lógico de una máquina o para expresar algoritmos con precisión. Cada uno está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

1.5.4.1. PHP 5.3.10

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. PHP es ampliamente utilizado para fines generales aunque es especialmente adecuado para el desarrollo web y puede ser embebido en páginas HTML (GROUP, 2011).

PHP es un Acrónimo recursivo que significa *Hypertext Pre-processor*. El lenguaje fue creado originalmente por Rasmus Lerdorf en 1994; sin embargo la implementación principal de PHP es realizada en la actualidad por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la PHP License, la *Free Software Foundation* considera esta licencia como software libre (MANES, 2011).

PHP brinda además la posibilidad de usar programación procedimental o programación orientada a objetos (POO), o una mezcla de ambos. Una de las características más fuertes y más importantes es su soporte para una amplia gama de bases de datos. Escribir una página web habilitada para una base de datos es muy simple utilizando una de las extensiones de bases de datos específicas (por ejemplo

para MySQL), o conectarse a cualquier base de datos que soporte el estándar de base de datos de conexión abierta.

1.5.5. Bibliotecas o librerías

Las librerías están orientadas a un uso pasivo y usualmente no generan código fuente en las aplicaciones, por lo que los desarrolladores le dan un gran uso a las funcionalidades que estas proveen. Permite a los desarrolladores hacer uso de un conjunto de funcionalidades de uso general. A continuación se describen algunas de las principales librerías de desarrollo para PHP.

1.5.5.1. Imap 5.3.10

Biblioteca desarrollada en PHP. Posee una gran variedad de funciones que le capacitan para operar con el protocolo IMAP y los métodos de acceso al buzón local. Estas funciones permiten conectarse a un servidor de correo, la codificación y decodificación de los mensajes del correo, obtener cabeceras y cuerpos de mensajes, entre otras. Además permite operar con las carpetas del lado servidor. Esta librería posee una gran ventaja, ya que permite visualizar los mensajes de manera remota y no descargarlos.

1.5.5.2. Phpmailer 5.1

Es una clase escrita en el lenguaje PHP que permite enviar mensajes electrónicos. Esta clase se conecta a un servidor SMTP, a través del cual envía no solo mensajes sino también datos adjuntos. Su licencia es LGPL.

1.5.5.3. Caldav

Biblioteca con soporte para el protocolo Caldav. Este protocolo es un estándar de Internet que permite a un cliente acceder a información de planificación en un servidor remoto. Esta permite integrar el calendario del correo electrónico con otras aplicaciones. Además permite crear, gestionar y compartir eventos dentro de las agendas. Dentro de las agendas más conocidas que soporta esta biblioteca están *Microsoft Outlook*, *iCal* y *Sunbird* (DELFA, 2005).

1.5.6. Servidor web apache 2.2

El servidor HTTP Apache es un potente y flexible servidor web. Es altamente configurable y extensible con módulos de terceros. Se puede personalizar mediante la escritura "módulos" a través de la API de

Apache, proporciona el código fuente completo y viene con una licencia sin restricciones (BANNISTER, 2012).

Entre sus características se destacan:

- ✓ Es multiplataforma.
- ✓ Es modular: se adapta a diferentes entornos y necesidades con los diferentes módulos de apoyo que proporciona.
- ✓ Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos.
- ✓ Se desarrolla con código abierto.
- ✓ Es actualizado de manera continua y evoluciona a gran velocidad.

1.5.7. Herramienta CASE Visual Paradigm 8.0

Es una herramienta de Ingeniería de Software Asistida por Computación (CASE[12]). La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación (VISUAL-PARADIGM.COM, 2012).

Esta herramienta de modelado esta concebida para soportar el ciclo de vida completo del proceso de desarrollo de un software a través de la representación de todo tipo de diagramas. Constituye una herramienta de software libre de probada utilidad para los analistas que diseñan una amplia gama de sistemas de software de forma fiable a través de la utilización de un enfoque orientado a objetos.

Entre sus principales características se encuentran:

- ✓ Entorno de creación de diagramas para UML.
- ✓ Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ✓ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo. Disponibilidad de múltiples versiones, para cada necesidad.
- ✓ Disponibilidad de integrarse en los principales IDEs.
- ✓ Generación de bases de datos.
- ✓ Transformación de diagramas de Entidad-Relación en tablas de una base de datos.

[12]Siglas en inglés para ***Computer Asisted Software Engeneering***.

- ✓ Importación y exportación de ficheros XML.

1.5.8. Entorno Integrado de Desarrollo (IDE Netbeans 7.0.1)

Es un Entorno de Desarrollo Integrado (IDE) para desarrolladores de software. Posee todas las herramientas necesarias para crear aplicaciones profesionales de escritorio, empresariales, web y aplicaciones móviles con la plataforma Java, así como con C + +, PHP y JavaScript (NETBEANS, 2012). Pueden ser adicionadas extensiones que permiten desarrollar módulos y depurar el código realizado. También permite incluir herramientas para la programación web como es el caso de un constructor de CSS.

1.6. ANÁLISIS DE LAS TECNOLOGÍAS

Como parte del proceso de mejora que lleva a cabo el CENIA, además de las características en las que se desarrolla el producto, el desarrollo del software está guiado por el “Proceso de desarrollo con enfoque ágil orientado al segundo nivel de CMMI”.

Dentro de los elementos que forman parte de la solución se define el uso de herramientas libres, compatibles con el desarrollo tecnológico de la universidad.

Como marco de trabajo es seleccionado Restler, ya que este *framework* brinda al igual que otros estudiados las mismas funcionalidades, posee una amplia documentación y su instalación resulta muy fácil. Este sirve de gran ayuda para la creación del API y permite la comunicación con otras aplicaciones mediante REST, mecanismo estándar estudiado que por sus características y ventajas se selecciona para la interacción de los servicios telemáticos con la RSU.

El marco de trabajo seleccionado utiliza como lenguaje de programación PHP, lo que implica la utilización de este lenguaje para la implementación de la solución, además es definida la utilización de varias bibliotecas con soporte para el lenguaje seleccionado tales como Caldav, XMPPHP, phpmailer, imap y smtp.

Como IDE que facilita el trabajo con las bibliotecas y el lenguaje de programación definido se encuentra Netbeans. También se selecciona la herramienta Visual Paradigm para la creación de diagramas y el modelado del producto, ya que la misma posee una amplia documentación y brinda un conjunto de funcionalidades para el desarrollo de un software como el diseño y la generación de código fuente.

1.7. CONCLUSIONES PARCIALES

Con la investigación realizada a los servicios telemáticos más importantes de la UCI se logra una mejor comprensión de estos y se seleccionan las posibles funcionalidades a implementar en el API. El estudio de soluciones existentes permite la extracción de rasgos característicos de estas aplicaciones, dando paso al estudio de elementos como los mecanismos de interacción con otras aplicaciones que utilizan y las herramientas empleadas en su construcción.

CAPÍTULO II. DISEÑO DE LA INTERFAZ DE PROGRAMACIÓN DE APLICACIONES

2.1. INTRODUCCIÓN

El estudio del flujo actual de los procesos influye directamente en que la solución propuesta responda a las necesidades de la integración de los servicios telemáticos con la RSU. En este capítulo se describe la solución propuesta para integrar los principales servicios telemáticos brindados en la UCI con la RSU. Se detallan las principales características de la aplicación, así como los requisitos funcionales y no funcionales. Se presenta además una descripción de las funcionalidades a implementar.

2.2. DIAGNÓSTICO DEL CAMPO DE ACCIÓN

Los trabajadores encargados de la administración de las redes en la UCI, dedican sus esfuerzos para brindar a los usuarios servicios que permiten compartir, transferir y comunicar información a través de las computadoras. Estos servicios informáticos o telemáticos, calificativo por el cual son conocidos en la universidad, están constituidos por servicios como el correo electrónico nacional e internacional y la navegación básica en Internet, este último incluye solamente una lista de sitios importantes para la docencia y la formación de los estudiantes. También los servicios especiales permiten el acceso remoto por teléfono a la red, audio y video conferencia, servicio de mensajería instantánea nacional e internacional y FTP.

Los servicios telemáticos en la actualidad están descentralizados, donde el acceso requiere la navegación por diferentes aplicaciones y existen solo pocas que integran algunos de ellos, tal es el caso de http://telematicos.uci.cu, sitio accesible por los usuarios de la universidad. Además no existe ningún mecanismo establecido que posibilite su integración con otras aplicaciones.

El Centro de Informatización Universitaria lleva a cabo el desarrollo de la RSU. Plataforma que persigue el objetivo de dotar a los usuarios de la universidad de una herramienta para la comunicación de la información. La misma necesita que el manejo de la información contenida se realice de una forma segura, precisa, garantizando los principios básicos de seguridad y confidencialidad de la información. Por tal motivo resulta vital la integración de los servicios telemáticos con esta plataforma.

2.3. DESCRIPCIÓN DE LA PROPUESTA DE SOLUCIÓN

La solución propuesta por la presente investigación define una capa de abstracción de las funcionalidades que brindan los servicios telemáticos tales como el correo electrónico, mensajería instantánea y el servicio de consulta de las cuotas de navegación. Esta capa de abstracción se basa en la implementación de una API que permite realizar operaciones de gestión a los recursos que manipulan los servicios informáticos anteriormente expuestos. Ejemplo de esto son los eventos del calendario del correo, a los que se le realizan operaciones de inserción, modificación, obtención y eliminación. Para la comunicación con el API se utiliza REST como mecanismo eficiente para la interacción de las aplicaciones la cual satisface las necesidades planteadas en la investigación.

Para el acceso a los servicios telemáticos se hace necesario contar con los datos de autenticación de los usuarios que harán uso de los mismos. El proceso de validación de estos datos se realiza a través de la pasarela de autenticación definida con este propósito en la universidad.

El acceso a las funcionalidades que brinda este software se realiza mediante HTTP, haciendo uso de los métodos GET, PUT, POST o DELETE se realizan las peticiones que permiten crear, obtener, actualizar o eliminar un recurso.

En la Figura 2**,** se ilustra un conjunto de clases que componen la capa de manipulación de la información que se obtiene de estos servicios. En esta capa son separados los servicios en clases en dependencia del nivel de complejidad que se requiere para manipular la información. En el caso del servicio de correo electrónico, se decide separar las funcionalidades que gestionan los eventos de las clases que manipulan los mensajes y carpetas, también son separadas las clases que gestionan las tareas del calendario del correo electrónico.

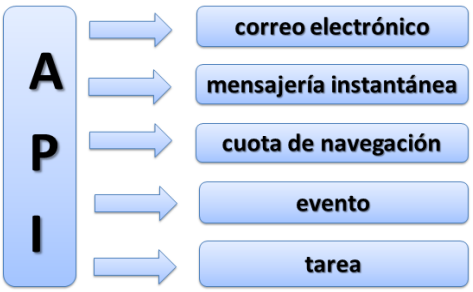


Figura 2. Estructura de clases de la API.

2.4. MODELO DE DOMINIO

El medio en que se desarrolla la solución posee varios flujos de información, lo que conlleva a la utilización del modelo de dominio como uno de los artefactos utilizados en la ingeniería de software que permite representar los procesos o la esencia del software.

El objetivo principal de este modelo es describir las clases más importantes dentro del contexto del sistema (JACOBSON, 2000). En el caso particular de esta investigación se define el modelo como se muestra en la Figura 3.

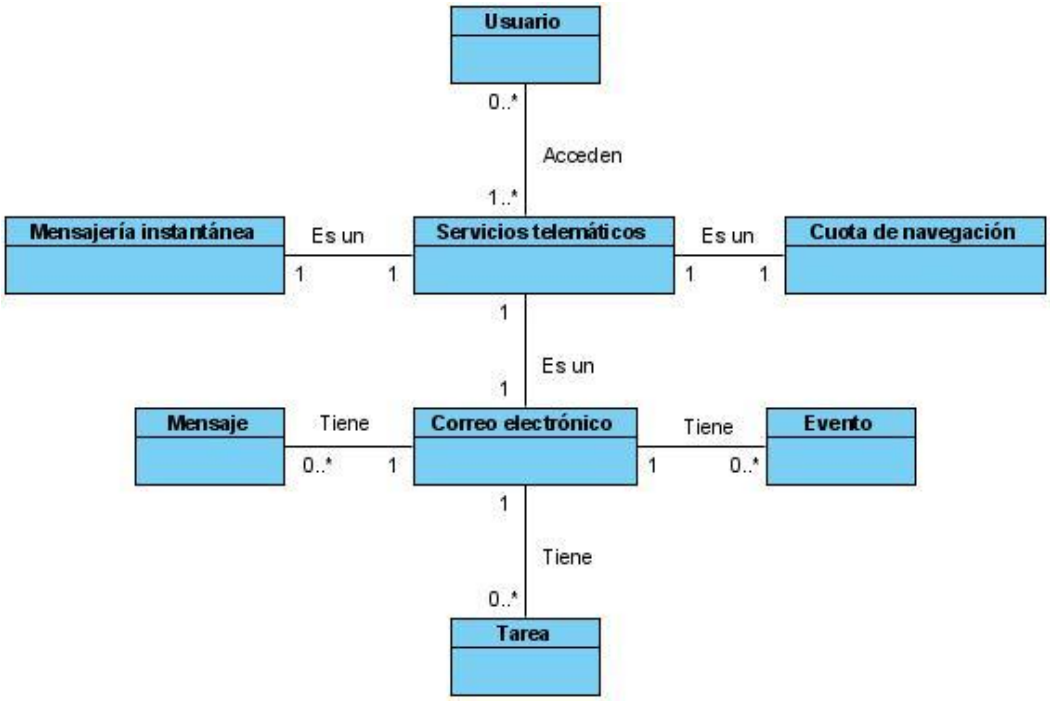


Figura 3. Modelo de dominio.

2.4.1. Descripción de las entidades del modelo de dominio

- ✓ Usuario: esta entidad representa a las personas que se encargan de consumir las funcionalidades de la interfaz de programación de aplicaciones.
- ✓ Servicios Telemáticos: constituye los servicios informáticos de mayor consumo en la UCI que se desean integrar a la RSU.
- ✓ Mensajería Instantánea: es identificado como uno de los servicios telemáticos que se implementan en el API.

- ✓ Cuota de Navegación: es identificado como uno de los servicios telemáticos que se implementan en el API.
- ✓ Correo Electrónico: es identificado como uno de los servicios telemáticos que se implementan en el API.
- ✓ Mensajes: representa una parte del servicio de correo electrónico que es desarrollado en la interfaz de programación de aplicaciones.
- ✓ Eventos: representa una parte del servicio de correo electrónico que es desarrollado en la interfaz de programación de aplicaciones. Dentro de ella se incluye la gestión de los eventos del calendario.
- ✓ Tareas: representa una parte del servicio de correo electrónico que es desarrollado en la interfaz de programación de aplicaciones. Dentro de ella se incluye la gestión de las tareas del calendario.

2.5. REQUISITOS FUNCIONALES Y NO FUNCIONALES

Los requisitos de software son las capacidades o condiciones que debe tener un sistema y están encaminados a cumplir con las exigencias que quedan plasmadas en un documento formal o contrato.

2.5.1. Requisitos funcionales

Un requisito funcional define el comportamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la prácticas (STELLMAN y GREENE, 2005).

A continuación se muestra un listado con los requisitos funcionales definidos para la implementación en la API:

- ➔ Cuotas de navegación.
  - ✓ **RF1** Consultar cuota de navegación.
- ➔ Correo electrónico.
  - ✓ **RF2** Enviar correo electrónico.
  - ✓ **RF3** Mostrar correos electrónicos recibidos.
  - ✓ **RF4** Mostrar cantidad de correos no leídos.
  - ✓ **RF5** Crear carpeta.
  - ✓ **RF6** Renombrar carpeta.

- ✓ **RF7** Eliminar carpeta.

➔ Eventos.

- ✓ **RF8** Crear evento.
- ✓ **RF9** Modificar evento.
- ✓ **RF10** Obtener evento.
- ✓ **RF11** Eliminar evento.

➔ Tareas.

- ✓ **RF12** Crear tarea.
- ✓ **RF13** Modificar tarea.
- ✓ **RF14** Obtener tarea.
- ✓ **RF15** Eliminar tarea.

➔ Mensajería instantánea.

- ✓ **RF16** Enviar mensaje instantáneo.
- ✓ **RF17** Recibir mensaje instantáneo.
- ✓ **RF18** Mostrar lista de contactos.

2.5.1.1. Descripción de algunos requisitos funcionales

A continuación son descritos algunos de los requisitos funcionales que posee la API, son detalladas características como número o código, su descripción, complejidad, campos de entrada o salida, tipos de datos, reglas o restricciones y otras observaciones.

Las descripciones de los requisitos funcionales restantes se encuentran en la sección I de los anexos, (ver Anexo I. Descripción de los requisitos funcionales).

Tabla 3 Descripción del requisito funcional Crear evento.

Nº	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para cliente**
RF8	Crear evento	Es adicionado un evento al calendario del correo electrónico.	Alta	Alta
	Prototipo			
	Campos	**Tipos de datos**	**Reglas o restricciones**	

	Usuario	Texto	El campo debe ser un usuario válido del dominio.
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.
	Asunto	Texto	Cualquier cadena de caracteres.
	Cuerpo	Texto	Cualquier cadena de caracteres.
	Lugar	Texto	Cualquier cadena de caracteres.
	Fecha de inicio	Fecha	Debe poseer un formato de fecha válido.
	Fecha final	Fecha	Debe poseer un formato de fecha válido.
	Observaciones	Los parámetros usuario, contraseña y asunto son campos obligatorios. Una vez creado el evento se muestra el etag (etiqueta) e identificador del evento.	

Tabla 4 Descripción del requisito funcional Modificar evento.

Nº	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para cliente**
RF9	Modificar evento	Se modifica la información de un evento que fue creado anteriormente.	Alta	Alta
	Prototipo			
	Campos	**Tipos de datos**	**Reglas o restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio.	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Identificador	Texto	Debe existir un evento creado con este identificador.	
	Nuevo asunto	Texto	Cualquier cadena de caracteres.	
	Nuevo cuerpo	Texto	Cualquier cadena de caracteres.	

	Nuevo lugar	Texto	Cualquier cadena de caracteres.
	Nueva fecha de inicio	Fecha	Debe poseer un formato de fecha válido.
	Nueva fecha final	Fecha	Debe poseer un formato de fecha válido.
	Observaciones	Los campos usuario, contraseña e identificador son obligatorios. El resto de los campos toman valores por defecto en caso de dejarse vacíos. Una vez modificado el evento, se muestra el mensaje “El evento fue modificado satisfactoriamente”.	

Tabla 5 Descripción del requisito funcional Eliminar evento.

Nº	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para cliente**
RF11	Eliminar evento	Elimina permanentemente un evento previamente creado.	Alta	Alta
	Prototipo			
	Campos	**Tipos de datos**	**Reglas o restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio.	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Identificador	Texto	Debe existir un evento creado con este identificador.	
	Observaciones	Una vez eliminado el evento, el software muestra el mensaje “Evento eliminado satisfactoriamente”.		

2.5.2. Requisitos no funcionales

Los requisitos no funcionales son las condiciones que necesita un software para que su rendimiento sea óptimo. También hacen que sea aceptado por los clientes ya que son los encargados de hacer que el software sea de interés pues definen la rapidez con que responde el mismo al usuario, la interfaz gráfica que se usa, entre otros aspectos.

➔ **Usabilidad**

- ✓ **RNF1:** El software debe permitir la integración de los servicios brindados con otras aplicaciones o sistemas.
- ✓ **RNF2**: Los usuarios que harán uso de la API deberán saber como comunicarse con esta y poseer un conocimiento básico sobre los servicios telemáticos integrados a la misma.
- ✓ **RNF3**: Requiere del correcto funcionamiento de los servicios de correo electrónico, la mensajería instantánea, y el servicio web de las cuotas de navegación.

➔ **Eficiencia**

- ✓ **RNF4:** Debe ser capaz de responder a cualquier petición de sus funcionalidades en 15 segundos o menos.
- ✓ **RNF5:** Debe soportar como máximo 1000 usuarios trabajando de forma concurrente.

➔ **Soporte**

- ✓ **RNF6:** Debe contar con un grupo de usuarios destinados a brindar soporte técnico a la API.

➔ **Restricciones de Diseño**

- ✓ **RNF7:** Utilización del *framework* Restler.
- ✓ **RNF8:** Lenguaje de programación PHP 5.3.10.
- ✓ **RNF9:** La utilización de las librerías imap, smtp, phpmailer y xmpphp.
- ✓ **RNF10:** Servidor web Apache 2.
- ✓ **RNF11:** Entorno Integrado de Desarrollo (IDE) Netbeans 7.0.1.
- ✓ **RNF12:** Herramienta de modelado Visual Paradigm 8.0.
- ✓ **RNF13:** Como metodología de desarrollo “Proceso de desarrollo con enfoque ágil orientado al segundo nivel de CMMI”.

➔ **Requisitos para la documentación de usuarios en línea y ayuda del sistema.**

- ✓ **RNF14:** Toda la documentación relacionada con el análisis, desarrollo y despliegue del software debe estar actualizada.

➔ **Componentes Comprados**

- ✓ **RNF15:** Licencia del Visual Paradigm.

➔ **Requisitos Legales, de Derecho de Autor y otros**

- ✓ **RNF16:** Todas las herramientas utilizadas están basadas en la licencia GNU/GPL.

➔ **Estándares Aplicables**

- ✓ **RNF17:** Estándares de codificación del centro CENIA (VIDAL, 2010).

2.6. PATRONES ARQUITECTÓNICOS Y DE DISEÑO

Uno de los elementos para el buen desarrollo de un software radica en la elaboración de un buen diseño, contribuyendo directamente a la calidad del producto final, gran parte de esta yace en la utilización adecuada de los patrones de diseño y arquitectónicos existentes en la actualidad.

2.6.1. Patrón de arquitectura

El marco de desarrollo a utilizar define la arquitectura en capas. Esta arquitectura describe la descomposición de servicios de forma que la mayoría de la interacción ocurre solamente entre capas vecinas. Además aporta beneficios como el aislamiento, ya que permite asilar los cambios en tecnologías a ciertas capas para reducir el impacto en el sistema total (PELAEZ, 2009).

Capa de negocio: es donde residen las funcionalidades que manejan la información obtenida por la capa de servicios, se reciben las peticiones de los usuarios y se envía una respuesta una vez realizada la conexión a los servidores de la institución. Algunas de las clases en las que residen las funcionalidades son correo.php y jabber.php, además existe un fichero de configuración denominado apiconfig.php en el que se especifican las direcciones del servidor smtp, *jabber* e imap al cual se conectan las clases pertenecientes a esta capa para obtener o enviar la información.

Capa de servicios: esta capa interactúa con la información que manipulan los servicios telemáticos implementados. Posee clases relevantes como phpmailer.php y xmpp.php, esta última encargada de conectarse con el servidor de mensajería instantánea y obtener mensajes o enviarlos, así como definir mecanismos de encriptación de la información.

2.6.2. Patrones de diseño

A continuación son descritos los patrones de diseño GRASP que se evidencian en el software desarrollado. Estos representan los patrones generales de software para asignar responsabilidades, por sus siglas en inglés *General Responsibility Assignment Software Patterns*.

Experto

Permite que se le asigne a una clase la información necesaria para que cumpla con su responsabilidad. Dentro de sus beneficios se encuentran que permite conservar el encapsulamiento,

ya que los objetos se valen de su propia información para hacer lo que se les pide. Un ejemplo del uso de este patrón se encuentra en la clase correo, ya que ella es la única encargada de todas las funcionalidades relacionadas con el envío y recepción de los mensajes electrónicos.

Alta cohesión

Expresa que la información que almacena una clase debe de ser coherente y debe estar relacionada con la clase en la medida de lo posible. La información que manipula cada clase del marco de trabajo está estrechamente relacionada con la funcionalidad que realiza la misma como componente de la plataforma.

Bajo acoplamiento

Plantea que las clases deben estar lo menos ligadas entre sí que se pueda. Se garantiza en la arquitectura del marco de trabajo la separación de las funcionalidades en clases con baja dependencia entre ellas, lo que representa una ventaja a la hora de realizar modificaciones en un servicio, posibilita que no ocurra ninguna repercusión en los otros.

GOF

Se evidencia también el uso de los patrones de diseño orientados a objetos GOF (*Gang Of Four*) facilitando la localización de los objetos que conforman la aplicación. Tal es el caso del patrón de diseño descrito a continuación.

Command

El marco de trabajo para evitar la existencia de código redundante a consecuencia de acciones comunes implementadas en varios objetos o evitar dificultades en el cambio de acciones, encapsula las operaciones en objetos, lo que permite ejecutar dicha operación sin necesidad de conocer el contenido de la misma.

2.7. CONCLUSIONES PARCIALES

La realización del presente capitulo sentó las bases para la implementación del software, tomando en consideración los patrones arquitectónicos y de diseño descritos en el mismo. Además permitió la identificación de los requerimientos necesarios para el desarrollo de las funcionalidades definidas.

CAPÍTULO III. CONSTRUCCIÓN Y PRUEBAS DEL SOFTWARE

3.1. INTRODUCCIÓN

En el actual capítulo se describen los elementos que son utilizados en la construcción de la solución. A través del diagrama de despliegue de la solución, se muestran los nodos o elementos de hardware necesarios para la implantación del producto. También se describen las pruebas realizadas al software obtenido para determinar el nivel de calidad que posee efectuando pruebas de código e integración con otros servicios, que permiten conocer cuántas de las especificaciones planteadas fueron cumplidas en su totalidad.

3.2. IMPLEMENTACIÓN

En este epígrafe son descritos elementos importantes en la construcción del software, tal es el caso de los estándares de codificación y la descripción de las clases más relevantes en la solución implementada.

3.2.1. Estándares de codificación

Los estándares de codificación representan pautas o reglas creadas para escribir el código en la programación. Usar estas reglas implica obtener un código de alta calidad que juega un papel importante en la calidad del software y en su rendimiento. Además, la utilización de forma eficiente de estos estándares de codificación permite una mejor comprensión del código por otras personas.

1. Longitud de las líneas y llaves de apertura y cierre

Las llaves” {}” utilizadas en la declaración de clases y funciones es en una nueva línea. La cantidad de caracteres en las líneas de código se encuentran entre 75 y 80, para favorecer la legibilidad del código.

2. Nomenclatura

- ✓ Las variables comienzan con letra minúscula. Cuando existen nombres compuestos por varias palabras, la primera letra de cada palabra comienza con letra mayúscula.
- ✓ Las clases siempre comienzan con mayúscula, en caso de nombres compuestos las palabras se separan con el carácter subrayado “_” y el resto en minúscula.

- ✓ Las funciones o métodos comienzan con minúscula y en caso de nombres compuestos por varias palabras, la primera letra de cada palabra comienza con mayúscula. En caso de recibir parámetros son separados por coma y luego un espacio.
- ✓ Los ficheros de clases PHP comienzan con minúsculas. Los ficheros de javascript y css siempre con minúsculas.

3. Estructuras de control

La utilización de llaves de apertura y cierre se recomienda incluso en situaciones opcionales. Esto aumenta la legibilidad y disminuye la probabilidad de errores lógicos.

4. Buenas prácticas

Los valores booleanos y nulos siempre se escriben con minúsculas, para facilitar la legibilidad del código se usa un salto de línea antes de las estructuras de control y definición de las funciones (VIDAL, 2010).

3.2.2. Descripción de clases implementadas

Como parte de la API se hace necesaria la construcción de clases para el manejo de las funcionalidades de los servicios telemáticos. Estas clases permiten la interacción con los servidores de los servicios brindados en la UCI, posibilitando a los usuarios el acceso a las funcionalidades que se detallan a continuación:

- ✓ **correo_electrónico:** esta clase brinda funcionalidades relacionadas con los mensajes del correo electrónico. Utiliza la librería de imap, smtp y phpmailer tanto para el envío como recepción de los mensajes electrónicos.
- ✓ **evento:** utiliza la librería Caldav para gestionar toda la información relacionada a los eventos del calendario.
- ✓ **tarea:** utiliza la librería Caldav para gestionar toda la información relacionada a las tareas del calendario.
- ✓ **cuota_de_navegación:** consume el servicio web de la UCI a través del cual se obtiene el estado de la cuota consumida por el usuario, visualizando los porcientos consumidos y total, además del tipo de navegación que tiene.
- ✓ **mensajería_instantánea:** esta clase permite enviar y obtener mensajes instantáneos, mostrar la lista de contactos, adicionar y eliminar contactos. Hace uso de la librería XMPPHP para conectarse al servidor de mensajería instantánea de la universidad.

3.3. DIAGRAMA DE DESPLIEGUE

Los diagramas de despliegue representan la topología del hardware sobre el que se va a ejecutar una aplicación, además se utilizan para describir la vista de despliegue estática de un sistema.

Figura 4. Diagrama de despliegue.

La Figura 4 muestra el diagrama de despliegue, representa los nodos a utilizar en el despliegue de la aplicación. La explotación y consumo de las funcionalidades de la API que se encuentra en un servidor de aplicaciones es realizada por los usuarios a través de las PC clientes mediante el protocolo HTTPS. Este servidor de aplicaciones web se conecta a un nuevo servidor web para utilizar el servicio de consultas de cuotas de navegación a través de HTTPS. Además se conecta al servidor de mensajería instantánea a través de XMPP y al servidor de correo entrante mediante IMAP y al saliente a través de SMTP para la obtención y envío de correos electrónicos o mensajes instantáneos. La utilización de las funcionalidades de la API requiere de la autenticación de los usuarios, por lo que es necesaria la

conexión al directorio activo de la institución a través del protocolo LDAP mediante un servicio web de autenticación por HTTPS.

3.4. PROCESO DE PRUEBAS DE SOFTWARE

Las pruebas de software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación (PRESSMAN, 2002). Estas constituyen un conjunto de herramientas, técnicas y métodos que evalúan el desempeño de un programa. Estas involucran las operaciones del sistema, evaluando los resultados bajo condiciones controladas, lo que hace que la realización de pruebas al software sea un factor de vital importancia.

3.4.1. Definición de las estrategias de pruebas

La estrategia de prueba describe los pasos que hay que llevar a cabo como parte de la prueba, cuando se deben planificar y realizar esos casos. Debido a que la solución desarrollada no posee ninguna interfaz con la que el usuario pueda interactuar y además se integra a la RSU, se define la realización de pruebas que se encuentren dentro del nivel de pruebas unitarias y de integración.

Pruebas de unidad

Es la escala más pequeña de la prueba, está basada en la funcionalidad de los módulos del programa, como funciones, procedimientos y módulos de clase.

Los casos de pruebas que se diseñan para este nivel, deben descubrir errores como:

- ✓ Comparaciones entre tipos de datos distintos.
- ✓ Operadores lógicos o procedencia incorrecta.
- ✓ Igualdad esperada cuando los errores de precisión la hacen poco probable.
- ✓ Las variables o comparaciones incorrectas.
- ✓ Terminaciones de bucles inapropiadas o inexistentes.
- ✓ Fallo de salida cuando se encuentra una iteración divergente.
- ✓ Bucles que manejan variables modificadas de forma inapropiada.

La realización de estas pruebas permite llegar a la fase de integración de un software con la certeza de que el código está funcionando correctamente. Además los errores están más acotados y son más fáciles de localizar, ya que existen pruebas unitarias que permiten desenmascararlos.

Método

El método de prueba seleccionado es el de caja blanca. Estas pruebas denominadas también pruebas de caja de cristal, constituyen un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba. Mediante este método, el ingeniero del software puede obtener casos de prueba que:

- ✓ Garanticen que se ejecuten por lo menos una vez todos los caminos independientes de cada módulo.
- ✓ Ejerciten todas las decisiones lógicas en sus vertientes verdadera y falsa.
- ✓ Ejecuten todos los bucles en sus límites y con sus límites operacionales.
- ✓ Ejerciten las estructuras internas de datos para asegurar su validez (PRESSMAN, 2002).

Técnica

La técnica a utilizar es la prueba del camino básico. Para determinar el número de caminos posibles que puede tomar un algoritmo y definir la cantidad mínima de pruebas a realizar calculando la complejidad ciclomática del mismo. El cálculo de este valor se hace a partir del grafo del flujo asociado al algoritmo. A partir del valor obtenido se diseñan casos de prueba que garanticen que cada camino se ejecuta al menos una vez.

El cálculo de la complejidad ciclomática puede ser realizado de tres maneras:

1. El número de regiones del grafo de flujo coincide con la complejidad ciclomática.
2. La complejidad ciclomática V(G), de un grafo de flujo G se define como V(G) = A – N + 2, donde A es el número de aristas del grafo de flujo y N es el número de nodos del mismo.
3. La complejidad ciclomática, V(G), de un grafo de flujo G también se define como V(G) = P + 1, donde P es el número de nodos predicado (son los nodos de los cuales parten dos o más aristas) que tiene contenidos el grafo de flujo G (PRESSMAN, 2002).

Luego de identificar los caminos básicos del flujo se realizan los casos de prueba para la funcionalidad, elaborando un caso de prueba por cada camino básico en el que se describa el flujo del camino, las condiciones para que su ejecución, los parámetros de entrada y el resultado esperado de la ejecución del procedimiento.

Herramientas

- ✓ PHPUnit: es un entorno para realizar pruebas unitarias en el lenguaje de programación PHP. Proporciona un marco que hace que la escritura de las pruebas de fácil, así como las funciones a ejecutar con facilidad las pruebas y analizar sus resultados (BERGMANN, 2012).
- ✓ SimpleTest: marco de trabajo de pruebas unitarias para el lenguaje de programación PHP, de código abierto y fue creado por Marcus Baker. SimpleTest soporta la utilización de objetos de imitación y se puede utilizar para automatizar las pruebas de regresión de aplicaciones web con scripts de cliente HTTP, donde se pueden analizar las páginas HTML y simular cosas como hacer clic en los enlaces y el envío de formularios (MATHONIUS, 2012).
- ✓ Atoum: al igual que SimpleTest o PHPUnit, es un marco de pruebas unitarias específicamente para el lenguaje PHP (HARDY, 2012).
- ✓ NUnit: es un framework de código abierto para pruebas unitarias de sistemas realizados con la plataforma Microsof.NET. Se encuentra desarrollado en C# y permite realizar la ejecución de clases de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera.

A través de un estudio realizado sobre las pruebas unitarias, fueron encontrados varios marcos de trabajo que permiten la realización de las mismas. Con la investigación realizada sobre algunos de ellos, se selecciona SimpleTest por su fácil instalación y utilización, además de ser uno de los marcos de trabajo más usados por los usuarios para la realización de este tipo de pruebas.

3.4.2. Ejecución de las pruebas

➔ **Requisito funcional crear evento**

En la Figura 5 se observa el fragmento de código correspondiente a la funcionalidad de creación de un evento, destacando con números los pasos lógicos para la creación del grafo de flujo.

```php
private function crear($request_data=NULL){ ← 1
    $user = $_SERVER['PHP_AUTH_USER'];
    $pass = $_SERVER['PHP_AUTH_PW'];
    $asunto = $this->extract_data_value('asunto', $request_data);
    //Campos vacíos
    if(empty($user) || empty($pass) || empty($asunto)) ← 2
        throw new RestException(417, "Datos vacíos"); ← 3
    //Datos del evento
    $mail = $this->getUserMail($user);
    $cuerpo = $this->extract_data_value('cuerpo', $request_data, '');
    $lugar  = $this->extract_data_value('lugar', $request_data, '');
    $now = time();
    $fecha_inicio = $this->extract_data_value('fecha_inicio', $request_data, $now);
    $onehour = 60*60;
    $fecha_fin = $this->extract_data_value('fecha_fin', $request_data, ($fecha_inicio + $onehour));

    $caldav_url = $this->getCalDavURL($mail);
    $cal = new CalDAVClient(
        $caldav_url,
        $user,
        $pass,
        "calendar"
    );
    $v = new vcalendar( array( 'unique_id' => 'uci.cu' ));
    $e = & $v->newComponent( 'vevent' );
    $e->setProperty( 'dtstart' ,  date('Y', $fecha_inicio), date('n', $fecha_inicio), date('j', $fecha_inicio),
    date('G', $fecha_inicio), date('i', $fecha_inicio), 00 );
    $e->setProperty( 'dtend'   ,  date('Y', $fecha_fin), date('n', $fecha_fin), date('j', $fecha_fin),
    date('G', $fecha_fin), date('i', $fecha_fin), 00 );
    $e->setProperty( 'summary' , $asunto );
    $e->setProperty( 'description' , $cuerpo );
    $e->setProperty( 'location' , $lugar );

    $icalendar = $v->createCalendar() ;
    $uid = $e->getProperty('uid');
    $result = $cal->DoPUTRequest($uid.'.ics', $icalendar, '*'); ← 4
    $status_code = $cal->GetStatusHTTPResponse();
    return array('code' => $status_code, 'result'=> array('etag' =>$result, 'uid' => $uid)); ← 5
}
```

Figura 5. Código de la funcionalidad Crear evento.

A partir del flujo de la subrutina mostrada anteriormente queda construido el grafo de la Figura 6.

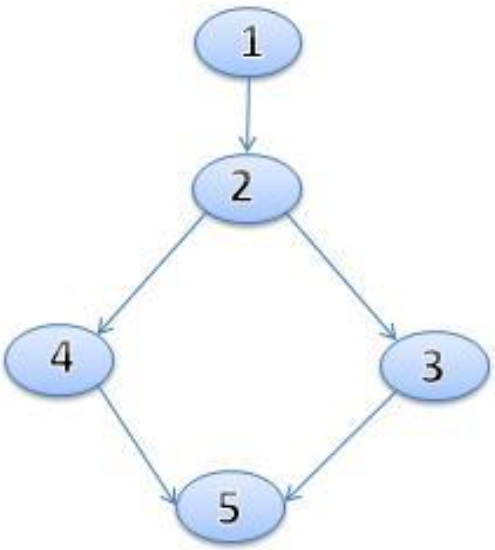


Figura 6. Grafo de flujo asociado a la funcionalidad Crear evento.

El cálculo de la complejidad ciclomática queda de la siguiente forma:

1. El grafo de flujo tiene 2 regiones.
2. V(G) = 5 aristas – 5 nodos + 2 = 2
3. V(G) = 1 nodos predicado + 1 =2

Una vez realizado el cálculo de la complejidad ciclomática, se determina que el algoritmo implementado posee una complejidad ciclomática de 2, lo que deriva que presenta a lo sumo 2 caminos lógicos por donde ejecutarse, ver Tabla 6.

Tabla 6 Caminos básicos de la funcionalidad Crear evento.

No.	Camino básico
1	1, 2, 3, 5
2	1, 2, 4, 5

- ✓ **Caso de prueba para el camino 1**

Tabla 7 Caso de prueba 1 para el requisito funcional Crear evento.

Camino	1, 2, 3, 5
Descripción	Los datos requeridos no poseen valor.
Condición de ejecución	Se introducen datos vacíos.
Entrada	$usuario= “vacío”, $contrasena = “vacío”, $uid= “vacío”, $asunto = “vacío”.
Respuesta esperada	Se muestra el mensaje “Campo (nombre del campo) no especificado”.

- ✓ **Caso de prueba para el camino 2**

Tabla 8 Caso de prueba 2 para el requisito funcional Crear evento.

Camino	1, 2, 4, 5
Descripción	Es creado un evento correctamente.
Condición de ejecución	Los datos introducidos son válidos.
Entrada	$usuario = “válido”, $contrasena = “válido”, $uid= “válido”, $asunto = “válido”.
Respuesta esperada	Se muestra la etiqueta y el identificador del evento creado.

- ➔ **Requisito funcional modificar evento**

En la Figura 7 se muestra el código perteneciente a la funcionalidad de modificar un evento. Este requisito funcional permite cambiar la información correspondiente a un evento creado con anterioridad, en caso que no haya sido creado, muestra un mensaje de error al usuario indicando que esta operación no ha podido ser realizada.

```php
private function actualizar($request_data=NULL){ ← 1
    //Campos requeridos
    $user = $_SERVER['PHP_AUTH_USER'];
    $pass = $_SERVER['PHP_AUTH_PW'];
    $uid = $this->extract_data_value('uid', $request_data);
    $asunto = $this->extract_data_value('asunto', $request_data);
    //Campos vacíos
    if(empty($user) || empty($pass) || empty($uid) ||empty($asunto)) ← 2
        throw new RestException(417, "Datos vacios"); ← 3
    //Obtener el correo del usuario
    $mail = $this->getUserMail($user); ← 4
    //Busca si el evento existe
    $test = $this->obtener(array('uid' => $uid)); ← 5
    if($test['code'] != 200)
        return array('code' => $test['code'], 'result'=> "El evento no existe"); ← 6
    //El resto de los datos toman valores por defecto si estan vacios
    $cuerpo = $this->extract_data_value('cuerpo', $request_data, ''); ← 7
    $lugar  = $this->extract_data_value('lugar', $request_data, '');
    $now = time();
    $fecha_inicio = $this->extract_data_value('fecha_inicio', $request_data, $now);
    $onehour = 60*60;
    $fecha_fin = $this->extract_data_value('fecha_fin', $request_data, ($fecha_inicio + $onehour));
    //Obtener la ruta del calendario
    $caldav_url = $this->getCalDavURL($mail);
    //Inicializar el objeto caldav_client
    $cal = new CalDAVClient(
        $caldav_url,
        $user,
        $pass,
        "calendar"
    );
    //Generando el objeto calendario
    // inicializar el uid
    $v = new vcalendar( array( 'unique_id' => 'uci.cu' ));
    $e = & $v->newComponent( 'vevent' );
    $e->setProperty( 'uid' , $uid );
    // fecha de inicio
    $e->setProperty( 'dtstart' ,
        date('Y', $fecha_inicio), date('n', $fecha_inicio), date('j', $fecha_inicio),
        date('G', $fecha_inicio), date('i', $fecha_inicio), 00
    );
    $e->setProperty( 'dtend'   ,
        date('Y', $fecha_fin), date('n', $fecha_fin), date('j', $fecha_fin),
        date('G', $fecha_fin), date('i', $fecha_fin), 00
    );
    $e->setProperty( 'summary' , $asunto );
    $e->setProperty( 'description' , $cuerpo );
    $e->setProperty( 'location' , $lugar );
    // generar el calendario
    $icalendar = $v->createCalendar() ;
    //Realizar la petición de actualización
    $result = $cal->DoPUTRequest($uid.'.ics', $icalendar, '*');
    //Obtener la respuesta del servidor
    $status_code = $cal->GetStatusHTTPResponse(); ← 8
    //¿ Todo satisfactorio ?
    if($status_code == 201)
        $response = array('code' => 200, 'result'=> array('etag' =>$result, 'uid' => $uid)); ← 10
    else
        $response = array('code' => $status_code, 'result'=> "No se pudo actualizar el evento"); ← 9
    return $response; ← 11
}
```

Figura 7. Código de la funcionalidad Modificar evento.

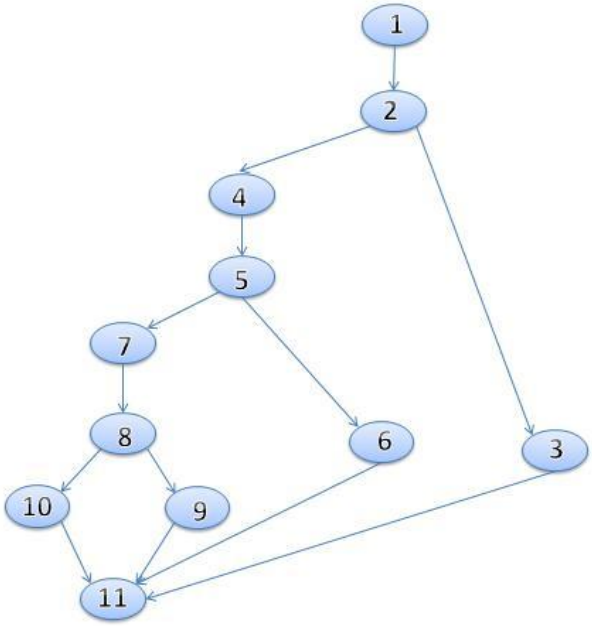


Figura 8**.** Grafo de flujo asociado a la funcionalidad Modificar evento.

Para el caso del algoritmo de la funcionalidad presentada, refiriéndonos al grafo de flujo presentado en la Figura 8, se puede calcular la complejidad ciclomática de las siguientes formas:

4. El grafo de flujo tiene 4 regiones.
5. V(G) = 13 aristas – 11 nodos + 2 = 4
6. V(G) = 3 nodos predicado + 1 = 4

Una vez realizado el cálculo se determina que el algoritmo implementado tiene una complejidad ciclomática de 4, lo que deriva que presenta a lo sumo 4 caminos lógicos por donde ejecutarse. En la Tabla 9 se observan los caminos básicos por los que puede transitar la funcionalidad.

Tabla 9 Caminos básicos de la funcionalidad Modificar evento.

No.	Camino básico
1	1, 2, 3, 11
2	1, 2, 4, 5, 6, 11

3	1, 2, 4, 5, 7, 8, 9, 11
4	1, 2, 4, 5, 7, 8, 10, 11

- ✓ **Caso de prueba para el camino 1**

Tabla 10 Caso de prueba 1 para el requisito funcional Modificar evento.

Camino	1, 2, 3, 11
Descripción	Los datos requeridos por el método no poseen valor.
Condición de ejecución	Los valores de los parámetros obligatorios se encuentran vacíos.
Entrada	$usuario = “vacío”, $contrasena = “vacío”, $uid= “vacío”, $asunto = “vacío”.
Respuesta esperada	Se muestra el mensaje “Campo (nombre del campo) no especificado”.

- ✓ **Caso de prueba para el camino 2**

Tabla 11 Caso de prueba 2 para el requisito funcional Modificar evento.

Camino	1, 2, 4, 5, 6
Descripción	El evento que se desea ser actualizado no existe.
Condición de ejecución	El identificador del evento no existe porque no ha sido creado un evento con el mismo.
Entrada	$usuario = “dato correcto”, $contrasena = “dato correcto”, $uid= “no existe”, $asunto = “dato correcto”.
Respuesta esperada	Se muestra el mensaje “El evento no existe”.

- ✓ **Caso de prueba para el camino 3**

Tabla 12 Caso de prueba 3 para el requisito funcional Modificar evento.

Camino	1, 2, 4, 5, 7, 8, 9, 11
Descripción	La petición de actualización realizada al servidor muestra error.
Condición de ejecución	No se pudo conectar por problemas de la red o el servidor del correo. La petición de actualización no obtuvo el código 201 Creado.
Entrada	$usuario = “correcto”, $contrasena = “correcto”, $uid= “existe”, $asunto = “correcto”.
Respuesta esperada	Se muestra el mensaje “No se pudo actualizar el evento”.

✓ **Caso de prueba para el camino 4**

Tabla 13 Caso de prueba 4 para el requisito funcional Modificar evento.

Camino	1, 2, 4, 5, 7, 8, 10, 11
Descripción	Es actualizado un evento se forma satisfactoria.
Condición de ejecución	Los datos introducidos son correctos, el identificador del evento existe y la petición de actualización obtuvo una respuesta satisfactoria.
Entrada	$usuario = “correcto”, $contrasena = “correcto”, $uid= “existe”, $asunto = “correcto”.
Respuesta esperada	Se muestra el mensaje “El evento ha sido modificado satisfactoriamente”.

➔ **Requisito funcional eliminar evento**

```
private function eliminar($request_data=NULL){ ← 1
    $user = $_SERVER['PHP_AUTH_USER'];
    $pass = $_SERVER['PHP_AUTH_PW'];
    $uid = $this->extract_data_value('uid', $request_data);

    //Datos vacíos
    if(empty($user) || empty($pass) || empty($uid)) ← 2
    throw new Exception("Datos vacíos"); ← 3

    //Buscar si el evento existe
    $test= $this->obtener(array('uid'=>$uid)); ← 4
    if($test['code'] !=200)
        return array('code'=> $test['code'], 'result'=> "El evento no existe");
    $mail = $this->getUserMail($user);
    $caldav_url = $this->getCalDavURL($mail);

    $cal = new CalDAVClient(
        $caldav_url,
        $user,
        $pass,
        "calendar"
    );

    $result = $cal->DoDELETERequest($uid.'.ics', '*'); ← 6
    $status_code = $cal->GetStatusHTTPResponse();
    return array('code' => $status_code, 'result'=> $result); ← 7
}
```

5

Figura 9. Código de la funcionalidad Eliminar evento.

En la Figura 9 se puede observar el fragmento de código correspondiente a la funcionalidad de eliminación de un evento, destacando con números los pasos lógicos para la creación del grafo de

flujo. En este funcionalidad se reflejan condiciones necesarias que deben cumplirse para la ejecución del método como la inserción de datos vacíos o el intento de eliminar un evento que no ha sido creado.

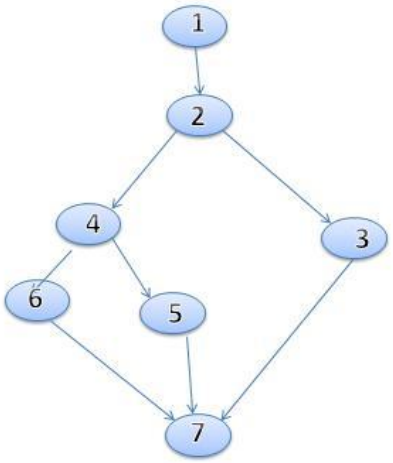


Figura 10. Grafo de flujo asociado a la funcionalidad Eliminar evento.

El cálculo de la complejidad ciclomática para el caso del grafo mostrado en la Figura 10 quedaría de las siguientes maneras:

1. El grafo de flujo tiene 3 regiones.
2. V(G) = 8 aristas – 7 nodos + 2 = 3
3. V(G) = 2 nodos predicado + 1 = 3

Una vez realizado el cálculo se determina que el algoritmo implementado tiene una complejidad ciclomática de 3, lo que deriva que presenta a lo sumo 3 caminos lógicos por donde ejecutarse. La siguiente tabla ilustra los caminos mínimos definidos en el algoritmo, (ver Tabla 14).

Tabla 14 Caminos básicos de la funcionalidad Eliminar evento.

No.	Camino básico
1	1, 2, 3, 7
2	1, 2, 4, 5, 7
3	1, 2, 4, 6, 7

- ✓ **Caso de prueba para el camino 1**

Tabla 15 Caso de prueba 1 para el requisito funcional Eliminar evento.

Camino	1, 2, 3, 7
Descripción	Son introducidos datos vacíos.
Condición de ejecución	Los campos de entrada no poseen valor.
Entrada	$usuario = “vacío”, $contrasena = “vacío”, $uid= “vacío”
Respuesta esperada	Se muestra el mensaje “Campo (nombre del campo) no especificado”.

- ✓ **Caso de prueba para el camino 2**

Tabla 16 Caso de prueba 2 para el requisito funcional Eliminar evento.

Camino	1, 2, 4, 5, 7
Descripción	El evento no ha sido creado.
Condición de ejecución	No existe el evento.
Entrada	$usuario = “válido”, $contrasena = “válido”, $uid= “no existe”
Respuesta esperada	Se muestra el mensaje “El evento no existe”.

- ✓ **Caso de prueba para el camino 3**

Tabla 17 Caso de prueba 3 para el requisito funcional Eliminar evento.

Camino	1, 2, 4, 6, 7
Descripción	El evento es eliminado correctamente.
Condición de ejecución	Los parámetros son correctos y el evento fue previamente creado.
Entrada	$usuario = “válido”, $contrasena = “válido”, $uid= “existe”
Respuesta esperada	Se muestra el mensaje “Evento eliminado satisfactoriamente”.

3.4.3. Pruebas de integración de la API con servicios de la RSU

Como parte del proceso de pruebas realizadas a la API, se desarrollan estrategias para comprobar que la misma se integra correctamente a la RSU. Estas estrategias consisten en la descripción del proceso que ocurre en una muestra de los servicios que brinda esta plataforma que hacen uso del API y un

caso de prueba en el que se detallan las condiciones de ejecución, datos de entrada y resultados del sistema al ejecutarse la funcionalidad.

- ➔ **Servicio de perfil:** plataforma que permite gestionar toda la información asociada a los usuarios, posibilita la comunicación entre ellos y la interacción con el resto de los servicios brindados en la RSU. La integración con la solución propuesta permite a los usuarios contactar a otros mediante un mensaje electrónico o instantáneo. En la sección II de los anexos se muestran imágenes con varias funcionalidades del API integradas a la RSU, (ver Anexo II Integración de la API con servicios de la RSU).
  - ✓ **Caso de prueba 1**

Tabla 18 Caso de prueba 1 para la funcionalidad Contactar usuario por correo electrónico.

Funcionalidad integrada	Enviar mensaje electrónico.
Condición de ejecución	El usuario realiza la acción de contactar a un amigo por correo electrónico.
Descripción de la prueba	Comprobar que al amigo seleccionado le llegue un mensaje electrónico con los datos introducidos desde la plataforma.
Entrada/Pasos de ejecución	Se requiere la autenticación del usuario en la plataforma y que introduzca todos los datos requeridos en la misma.
Respuesta del sistema	La plataforma muestra el mensaje “Correo enviado”.

  - ✓ **Caso de prueba 2**

Tabla 19 Caso de prueba 2 para la funcionalidad Contactar usuario por mensaje instantáneo.

Funcionalidad integrada	Enviar mensaje instantáneo.
Condición de ejecución	El usuario introduce los datos correctos para enviar un mensaje instantáneamente.
Descripción de la prueba	Comprobar que al amigo seleccionado le llegue un mensaje instantáneo con los datos introducidos desde la plataforma.
Entrada/Pasos de ejecución	Se requiere la autenticación del usuario en la plataforma y que introduzca todos los datos requeridos en la misma.
Respuesta del sistema	La plataforma muestra el mensaje “Jabber enviado”.

- ➔ **Servicio de eventos y notificaciones:** este sistema permite la notificación de la información referente a la gran variedad de eventos que se llevan a cabo en la UCI. A través de la RSU los

usuarios tienen acceso a este sistema que le permite conocer información sobre los eventos que se realizan en la universidad tales como: Jornada Científica Estudiantil, Seminario Juvenil Martiano, Copas de Programación y otros. Con la utilización de la API una vez que un usuario se inscribe en un evento, es adicionado un evento al calendario del correo electrónico con la información correspondiente.

- ✓ **Caso de prueba 3**

Tabla 20 Caso de prueba para la funcionalidad Suscribirse a un evento.

Funcionalidad integrada	Crear evento.
Condición de ejecución	Se ejecuta cuando un usuario crea un nuevo evento en la aplicación.
Descripción de la prueba	Comprobar que al usuario crear un evento en la aplicación este sea creado en el calendario del correo electrónico con los datos correspondientes al mismo.
Entrada/Pasos de ejecución	Se requiere la autenticación del usuario en la aplicación y que la misma envíe al API la información necesaria para que el evento sea creado en el calendario del correo.
Respuesta del sistema	La plataforma muestra el mensaje “Evento (nombre del evento) se ha creado”.

3.4.4. Resultados de las pruebas

El proceso de pruebas del software se realiza mediante tres iteraciones. Durante la realización de las pruebas en la primera iteración a las clases relacionadas con el servicio de correo electrónico, es decir, las clases correo, eventos y tareas fueron encontradas cuatro no conformidades dentro de las que se encontró una no conformidad crítica, a la hora de actualizar un evento, ya que si no existía el evento era creado uno nuevo, error que fue corregido. En la segunda iteración durante las pruebas realizadas a las funcionalidades de la mensajería instantánea y cuotas de navegación se encuentran no conformidades de nivel bajo como los mensajes de confirmación durante la ejecución de una funcionalidad. En la tercera iteración se verifican que fueron resueltas todas las no conformidades encontradas durante las iteraciones realizadas anteriormente y resulta que no se encontró ninguna. Para observar el proceso de iteraciones realizadas al software, ver Figura 11. Además se demuestra a través de las pruebas de integración realizadas a una muestra de los servicios de la RSU en desarrollo el correcto desempeño de algunas de las funcionalidades implementadas.

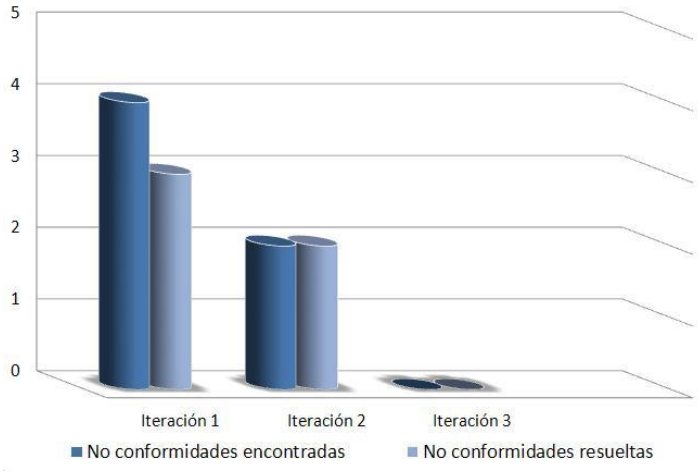


Figura 11. Proceso de iteraciones de pruebas.

3.5. CONCLUSIONES PARCIALES

La realización del modelo de despliegue permitió representar el hardware necesario para el funcionamiento de la solución brindada. La construcción de la solución hizo posible la creación de una capa de abstracción de las funcionalidades que brindan el servicio de correo electrónico, la mensajería instantánea y las consultas de navegación, lo que permitió la creación de importantes clases en el API y el acceso a estas desde la RSU. Además la ejecución de las pruebas plasmadas en el actual capítulo demuestra la correcta ejecución del software.

CONCLUSIONES GENERALES

Una vez concluido el presente trabajo de diploma se logra el cumplimiento de los objetivos trazados en la investigación. A partir de ello se arriba a las siguientes conclusiones:

- ✓ El análisis de los servicios telemáticos y las API, facilitó la selección de elementos para el desarrollo de la solución como el uso del mecanismo de interacción entre aplicaciones REST.
- ✓ El estudio de los mecanismos de interacción entre aplicaciones y las tecnologías usadas en la actualidad para la creación de API, permitió la selección de las tecnologías utilizadas en la construcción de la solución.
- ✓ La implementación de la propuesta de solución permitió que las funcionalidades definidas sean consumidas por los usuarios desde la RSU.
- ✓ La realización de las pruebas unitarias y de integración demostró que la información manipulada en los servicios telemáticos de mayor consumo en la UCI puede ser gestionada a través de la RSU.

RECOMENDACIONES

Enfocados en una mejor aceptación y mayor utilización por parte de los usuarios de la UCI se recomienda:

- ✓ Llevar a cabo la implementación de nuevas funcionalidades como la gestión de la información almacenada en el maletín del correo electrónico y la gestión de los usuarios en el servicio de mensajería instantánea, con el objetivo de aumentar las prestaciones presentadas.
- ✓ Incentivar el uso de la API por todas aquellas aplicaciones que requieran la utilización de los principales servicios telemáticos de la UCI para lograr centralizar estos servicios de manera eficiente.

BIBLIOGRAFÍA REFERENCIADA

ÁLAMO, A. R. y HERRERO, R. R. *El API JavaMail* Departamento de Informática y Automática Universidad de Salamanca 2003.

BANNISTER, T. *What is Apache?* [Consultado el: 17 de enero de 2012]. Disponible en: http://wiki.apache.org/httpd/FAQ#What_is_Apache.3F.

BECK, K.; BEEDLE, M.*, et al. Manifiesto ágil* [Consultado el: 18 de mayo de 2012]. Disponible en: http://es.wikipedia.org/wiki/ManifiestoAgil.

BERGMANN, S. *PHPUnit* github, Última actualización: 21 de abril de 2012. [Consultado el: 5 de mayo de 2012]. Disponible en: https://github.com/sebastianbergmann/phpunit/.

CLAVIJO, K. R. Concepción del sistema Red Social Académica de la UCI UciSociaLista. 2011, nº

CRISPIN, M. *Internet Message Access Protocol* The Internet Society, [Consultado el: 25 de marzo de 2012]. Disponible en: http://www.ietf.org/rfc/rfc3501.txt.

DELFA, C. V. *El correo electrónico: el nacimiento de un nuevo género.* Departamento de lengua española y teoría de la literatura y literatura comparada. Universidad complutense de Madrid, 2005.

DESIGNERS, S. W. *Social Networking - Facts and Usage Statistics [Infographic]* [Consultado el: 21 de noviembre de 2011]. Disponible en: http://www.go-gulf.com/blog/social-networking.

DÍAZ, M. H. *Internet 2011 en números* [Consultado el: 22 de febrero de 2012]. Disponible en: http://www.opcionweb.com/index.php/2012/01/24/internet-2011-en-numeros/.

FRÁNQUIZ, M. M.; PÉREZ, E. A.*, et al. Diseño de un Sistema de Gestión de la Información Económica.* La Habana: 2007, 9 p. Disponible en: www.bibliociencias.cu/gsdl/collect/eventos/archives/HASHc3c9.dir/doc.pdf.

BIBLIOGRAFÍA REFERENCIADA

FREED, N. *Multipurpose Internet Mail Extensions(MIME) Part One:Format of Internet Message Bodies* The Internet Society, [Consultado el: 25 de marzo de 2012]. Disponible en: http://tools.ietf.org/html/rfc2045.

GROUP, T. P. *PHP* [Consultado el: 13 de enero de 2011]. Disponible en: http://www.php.net/.

HARDY, F. *Atoum: A simple, modern and intuitive unit testing framework for PHP!* github, Última actualización: 26 de abril. [Consultado el: 2 de mayo de 2012]. Disponible en: https://github.com/mageekguy/atoum.

JACOBSON. *El Proceso Unificado de Desarrollo de software.* EE.UU: 2000.

KLENSIN, J. *Simple Mail Transfer Protocol* The Internet Society, [Consultado el: 28 de marzo de 2012]. Disponible en: http://www.ietf.org/rfc/rfc2821.txt.

LAZO, M. J. R. y REYES, M. S. *Módulo Control de Acceso del proyecto Intranet del Centro Rector de Universidad para Todos* Universidad de las Ciencias Informáticas, 2007.

LÓPEZ, D. C. y DÍAZ, L. B. M. *Implementación del Portafolio Digital de la Universidad de las Ciencias Informáticas.* Universidad de las Ciencias Informáticas, 2011.

MANES, J. P. P. *PHP* EcuRed, [Consultado el: 10 de diciembre de 2011]. Disponible en: http://www.ecured.cu/index.php/Php.

MASSÉ, M. *REST API Design Rulebook.* United States of America.: O'Reilly Media, Inc. , 2012. ISBN 978-1-449-31050-9.

MATHONIUS. *SimpleTest* Wikipedia.org, Última actualización: 13 de febrero de 2012. [Consultado el: 2 de mayo de 2012]. Disponible en: http://en.wikipedia.org/wiki/SimpleTest.

MÉNDEZ, A. S.; VÁZQUEZ, Y. R., *et al.* Procedimiento para el desarrollo de software con un enfoque ágil y CMMI nivel 2. 2011, nº Disponible en: http://uciencia.uci.cu/es/node/1310.

NETBEANS. *NetBeans IDE Features* Oracle Corporation and/or its affiliates, [Consultado el: 25 de febrero de 2012]. Disponible en: http://netbeans.org/features/index.html.

PELAEZ, J. C. *Arquitectura basada en capas* [Consultado el: 15 de mayo de 2012]. Disponible en: http://geeks.ms/blogs/jkpelaez/archive/2009/05/29/arquitectura-basada-en-capas.aspx.

BIBLIOGRAFÍA REFERENCIADA

PRESSMAN, R. S. *Ingeniería de software. Un enfoque ágil.* Quinta ed. McGraw-Hill, 2002. 600 p. ISBN 8448132149.

REYES, J. L. G. *Plataforma de Gestión de los Servicios Telemáticos en GNU\Linux. Módulo DNS.* Universidad de las Ciencias Informáticas, 2009.

SAAVEDRA, C. A. S. *La mensajería instantánea* [Consultado el: 12 de enero de 2012]. Disponible en: http://www.icesi.edu.co/blogs_estudiantes/emicasanchez/2009/08/09/la-mensajeria-instantanea/.

SARA RADICATI, Q. H. Email Statistics Report, 2011-2015. 2011, nº Disponible en: http://www.radicati.com.

STELLMAN, A. y GREENE, J. *Applied Software Project Management.* O'Reilly Media, 2005. ISBN 0-596-00948-8.

STOUGHTON, N. Update on Standards. abril 2005, nº Disponible en: https://db.usenix.org/publications/login/2005-04/openpdfs/standards2004.pdf.

TAHLER, E. *PHP-DAVE-API* Última actualización: 24 de mayo del 2012. [Consultado el: 4 de junio de 2012]. Disponible en: https://github.com/evantahler/PHP-DAVE-API.

VARO MARTÍNEZ, E. P. T. P., TERESA *Inclusión de las tecnologías de la información y la comunicación en las diferentes materias.* Granada: 2009, 14 p. Disponible en: http://www.csi-csif.es/andalucia/modules/mod_ense/revista/pdf/Numero_18/VARIAS_TIC%20EN%20EDUCACION_02.pdf. ISBN 1988-6047.

VIDAL, Y. G. *Estándar de código.* Universidad de las Ciencias informáticas: Dirección de Calidad de la Infraestructura Productiva, 2010,

VISUAL-PARADIGM.COM. *Visual Paradigm for UML* [Consultado el: 15 de marzo de 2012]. Disponible en: http://www.visual-paradigm.com/product/vpuml/. .

VOLPL, J. M. Z. *Métodos de acceso al email: POP3 vs IMAP* [Consultado el: 15 de febrero de 2012]. Disponible en: http://www.duplika.com/blog/metodos-de-acceso-al-email-pop3-vs-imap.

BIBLIOGRAFÍA CONSULTADA

Zimbra REST API. Zimbra, [Consultado el: 13 de Noviembre de 2011]. Disponible en: http://wiki.zimbra.com/index.php?title=ZCS_6.0:Zimbra_REST_API_Reference.

ÁLAMO, A. R. y HERRERO, R. R. *El API* JavaMail Departamento de Informática y Automática Universidad de Salamanca 2003.

ALARCÓN, R. y WILDE, E. *RESTLER: Crawling RESTful Services*. Departamento de Ciencia de la Computación Pontificia Universidad Católica de Chile 2010.

COSTELLO, R. L. y KEHOE, T. D*. Representational State Transfer* [Consultado el: 15 de marzo de 2012]. Disponible en: http://happytreeflash.com/file_9f5873cbf6b4f91327a03d45ea96cfaa.html.

CRISPIN, M. *Internet Message Access Protocol* The Internet Society, [Consultado el: 25 de marzo de 2012]. Disponible en: http://www.ietf.org/rfc/rfc3501.txt.

DABOO, C.; DESRUISSEAUX, B., et al. *Calendaring Extensions to WebDAV (CalDAV)* The IETF Trust, [Consultado el: 5 de junio de 2012]. Disponible en: http://tools.ietf.org/html/rfc4791.

DISATTENTION. *XMPP with PHP* http://disattention.com/, [Consultado el: 15 de enero de 2012]. Disponible en: http://disattention.com/notes/xmpp-with-php/.

FIELDING, T. R. *Architectural Styles and the Design of Network-based Software Architectures*. University of California, 2000.

FOUNDATION, T. A. S. *Apache* [Consultado el: 16 de Enero de 2012]. Disponible en: http://www.apache.org/.

FREED, N. *Multipurpose Internet Mail Extensions* (MIME) Part One: Format of Internet Message Bodies The Internet Society, [Consultado el: 25 de marzo de 2012]. Disponible en: http://tools.ietf.org/html/rfc2045.

BIBLIOGRAFÍA CONSULTADA

FRITZ, N. C.; WENTZ, S., et al. *XMPPHP: The PHP XMPP* Library Google, [Consultado el: 15 de enero de 2012]. Disponible en: http://code.google.com/p/xmpphp/.

GARCÍA, M. *Tutorial Sobre Como Crear Tu API* http://manelgarcia.com/ , [Consultado el: 16 de febrero de 2012]. Disponible en: http://manelgarcia.com/recursos-web/como-crear-tu-api.

GARCÍA, M. *Tutorial Sobre Como Crear Tu API* manelgarcia.com, [Consultado el: 8 de mayo de 2012]. Disponible en: http://manelgarcia.com/recursos-web/php/tutorial-sobre-como-crear-tu-api/.

GOUDARZI, N. *How to REST*. Julio del 2007 2007, nº

GUERRERO, B. *Crear un simple API con PHP* [Consultado el: 16 de mayo de 2012]. Disponible en: http://experiencias-web.blogspot.com/2012/01/crear-un-simple-api-con-php.html.

JACOBSON, D. *APIs: A Strategy Guide*. O'Reilly Media, 2011. ISBN 978-1449308926.

KLENSIN, J. *Simple Mail Transfer Protocol* The Internet Society, [Consultado el: 28 de marzo de 2012]. Disponible en: http://www.ietf.org/rfc/rfc2821.txt.

LANE, K. *Short List of RESTful API Frameworks for PHP* Última actualización: 23 de septiembre de 2011. [Consultado el: 4 de junio de 2012]. Disponible en: http://blog.programmableweb.com/2011/09/23/short-list-of-restful-api-frameworks-for-php/.

LANE, K. y WATTERS, A. *The Business of APIs*. 2011. ISBN 978-1461113881.

LITTLE, C.; HU, J., et al. *PHPicalendar* http://phpicalendar.net/ , Última actualización: 14 de abril de 2010. [Consultado el: 4 de junio de 2012]. Disponible en: http://phpicalendar.net/documentation/index.php/Main_Page.

LURACAST. *Restler 2.0* http://luracast.com , [Consultado el: 20 de mayo de 2012]. Disponible en: http://luracast.com/products/restler/.

MALKA, L. *How to Design APIs for Cryptographic Protocols*. University of Maryland, 2010.

MARSET, R. N. *Modelado, Diseño e Implementación de Servicios Web*. 2006-2007, nº

MASSÉ, M. *REST API Design Rulebook*. United States of America.: O'Reilly Media, Inc. , 2012. ISBN 978-1-449-31050-9.

BIBLIOGRAFÍA CONSULTADA

MOFFITT, J. *XMPP Programming with Javascript and jQuery*. Indianapolis: Wiley Publishing, Inc. , ISBN 978-0-470-54071-8.

PETER SAINT-ANDRE, K. S., REMKO TRONÇON. *XMPP: The Definitive Guide. Building Real-Time Applications with Jabber Technologies*. Mary E. Treseler ed. United States of America: O'Reilly Media, 2009.

PHPGROUP. *Imap* http://www.php.net/ , [Consultado el: 12 de noviembre de 2011]. Disponible en: http://www.php.net/manual/es/book.imap.php.

PREVOST, A.; BOINTON, M., et al. *PHP email class* http://sourceforge.net/, [Consultado el: 12 de octubre de 2011]. Disponible en: http://sourceforge.net/projects/phpmailer/

PUIG, A. *Desarrollando un API con REST* [Consultado el: 10 de mayo de 2012]. Disponible en: http://www.slideshare.net/apuig75/desarrollando-un-api-con-rest.

TAHLER, E. *PHP-DAVE-API* Última actualización: 24 de mayo del 2012. [Consultado el: 4 de junio de 2012]. Disponible en: https://github.com/evantahler/PHP-DAVE-API.

TIRRELL, Z. *zimbra-api-php* Google Project Hosting, [Consultado el: 22 de noviembre de 2011]. Disponible en: http://code.google.com/p/zimbra-api-php/.

WERDMULLER, B. *Write real-time web applications with XMPP, PHP, and JavaScript.* 2010, nº

XMPP-STANDARDS-FOUNDATION. *XMPP con php* xmpp.org, [Consultado el: 12 de abril de 2012]. Disponible en: http://xmpp.org/xmpp-software/libraries/.

ZULIAN, E. R. *Implementación de un framework para el desarrollo de aplicaciones web utilizando patrones de diseño y arquitectura MVC/REST*. Departamento de Investigaciones. Universidad de Belgrano, 2010.

GLOSARIO DE TÉRMINOS

***Cascading Style Sheets* (CSS):** las Hojas de Estilo en Cascada son un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML.

***File Transfer Protocol* (FTP):** el Protocolo de Transferencia de Archivos es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red.

***Lightweight Directory Access Protocol* (LDAP):** el Protocolo Ligero de Acceso a Directorios es un protocolo a nivel de aplicación el cual permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red.

MySQL: es un sistema de gestión de bases de datos relacional, multihilo y multiusuario.

Servicio web: es una colección de protocolos y estándares que sirve para intercambiar datos entre aplicaciones. Se comunica codificando los mensajes en XML y enviando estos mensajes a través de protocolos estándares de Internet.

***Uniform Resource Identifier* (URI):** un Identificador Uniforme de Recurso es una cadena de caracteres corta que identifica inequívocamente un recurso (servicio, página, documento, dirección de correo electrónico, enciclopedia, etc.).

***Unified Modelling Language* (UML):** el Lenguaje Unificado de Modelado es el lenguaje de modelado de sistemas de software más conocido en la actualidad.

***Uniform Resource Locator* (URL):** un Localizador de Recursos Uniforme es una secuencia de caracteres de acuerdo a un formato estándar que se usa para nombrar recursos en Internet para su localización o identificación.

ANEXOS

Anexo I. Descripción de los requisitos funcionales

Anexo 1 Descripción del requisito funcional Consultar cuota de navegación.

No	Nombre	Descripción	Complejidad	Prioridad para el cliente
RF1	Consultar cuota de navegación	Se muestra al usuario la cuota de navegación asignada, cuota consumida y el nivel de navegación.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio.	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Observaciones	En caso de no estar funcionando este servicio web de la UCI, la funcionalidad no podrá ser brindada por la API. Se muestra el usuario, cuota de navegación asignada, cuota de navegación usada y el nivel de navegación.		

Anexo 2 Descripción del requisito funcional Enviar correo electrónico.

No	Nombre	Descripción	Complejidad	Prioridad para el cliente
RF2	Enviar correo electrónico	Se envía un mensaje a una dirección de correo electrónico.	Alta	Alta

	Prototipo		
	Campos	**Tipos de Datos**	**Reglas o Restricciones**
	Usuario	Texto	El campo debe ser un usuario válido del dominio.
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.
	Para	Texto	El valor del campo debe ser una dirección de correo válida.
	Asunto	Texto	Cualquier cadena de caracteres.
	Cuerpo	Texto	Cualquier cadena de caracteres.
	Nombre del que lo envía	Texto	Cualquier cadena de caracteres.
	Observaciones	En caso de ser enviado un mensaje a una dirección de correo no válida, el mensaje rebota para el buzón del correo del usuario que lo envió. Si no son introducidos todos los datos requeridos se muestra el mensaje “campo (nombre del campo) no especificado”. Si se envía correctamente se muestra el mensaje “Correo enviado”.	

Anexo 3 Descripción del requisito funcional Mostrar correos electrónicos recibidos.

No	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para el cliente**
RF3	Mostrar correos electrónicos recibidos	Muestra un listado con las cabeceras de los mensajes de la bandeja de entrada	Alta	Alta
	Prototipo			

	Campos	Tipos de Datos	Reglas o Restricciones
	Usuario	Texto	El campo debe ser un usuario válido del dominio
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.
	Observaciones	Se muestran solos los correos de la bandeja de entrada. Muestra la fecha en que fue enviado, quien lo envió y el asunto.	

Anexo 4 Descripción del requisito funcional Mostrar cantidad de correos no leídos.

No	Nombre	Descripción	Complejidad	Prioridad para el cliente
RF4	Mostrar cantidad de correos no leídos	Muestra la cantidad de correos que no han sido marcados como leídos en el buzón.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Observaciones	Muestra la cantidad de correos no leídos que posee el usuario en la bandeja de entrada.		

Anexo 5 Descripción del requisito funcional Crear carpeta.

No	Nombre	Descripción	Complejidad	Prioridad para el cliente

RF5	Crear carpeta	Es añadida una nueva carpeta al correo.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Nombre de la carpeta	Texto	Cualquier cadena de caracteres	
	Observaciones	En caso de existir una carpeta con igual nombre se muestra el mensaje "Error en la creación de la carpeta", sino el mensaje “Carpeta creada”.		

Anexo 6 Descripción del requisito funcional Renombrar carpeta.

No	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para el cliente**
RF6	Renombrar carpeta	Se modifica el nombre de una carpeta del correo.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio	
	Contraseña	Texto	Debe ser una contraseña válida para un	

			usuario del dominio.
	Nombre de la carpeta que existe	Texto	Cualquier cadena de caracteres
	Nuevo nombre para la carpeta	Texto	Cualquier cadena de caracteres
	Observaciones	En caso de no existir el valor de la carpeta que se desea modificar, se muestra un mensaje de error.	

Anexo 7 Descripción del requisito funcional Eliminar carpeta.

No	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para el cliente**
RF7	Eliminar carpeta	Se eliminar completamente una carpeta del correo	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Nombre de una carpeta que existe	Texto		
	Observaciones	En caso de no existir el valor de la carpeta que se desea eliminar, se muestra un mensaje de error.		

Anexo 8 Descripción del requisito funcional Obtener eventos.

No	Nombre	Descripción	Complejidad	Prioridad para el cliente
RF10	Obtener eventos	Muestra un listado con el identificador, etag, fecha inicial, fecha final, cuerpo y otros atributos de los eventos.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Observaciones	Esta funcionalidad muestra todos los eventos que posee un usuario en su correo electrónico, en caso de introducir además de los datos requeridos el identificador de un evento, se muestra la información correspondiente a ese evento.		

Anexo 9 Descripción del requisito funcional Crear tarea.

No	Nombre	Descripción	Complejidad	Prioridad para el cliente
RF12	Crear tarea	Es adicionada una tarea al calendario del correo electrónico.	Alta	Alta
	Prototipo			

	Campos	**Tipos de Datos**	**Reglas o Restricciones**
	Usuario	Texto	El campo debe ser un usuario válido del dominio
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.
	Asunto	Texto	Cualquier cadena de caracteres.
	Cuerpo	Texto	Cualquier cadena de caracteres.
	Lugar	Texto	Cualquier cadena de caracteres.
	Fecha inicio	Fecha	Debe poseer un formato de fecha válido.
	Fecha fin	Fecha	Debe poseer un formato de fecha válido.
	Observaciones	Los parámetros usuario, contraseña y asunto son campos obligatorios. Una vez creada la tarea se muestra el etag e identificador de la misma. En caso de no introducir campos obligatorios se muestra el mensaje “Campo (nombre del campo) no especificado”.	

Anexo 10 Descripción del requisito funcional Modificar tarea.

No	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para el cliente**
RF13	Modificar tarea	Es cambiada la información de una tarea que fue creada anteriormente.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	

	Usuario	Texto	El campo debe ser un usuario válido del dominio
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.
	Identificador	Texto	Debe existir una tarea creada con este identificador.
	Nuevo asunto	Texto	Cualquier cadena de caracteres.
	Nuevo cuerpo	Texto	Cualquier cadena de caracteres.
	Nuevo lugar	Texto	Cualquier cadena de caracteres.
	Nueva fecha inicio	Fecha	Debe poseer un formato de fecha válido.
	Nueva fecha fin	Fecha	Debe poseer un formato de fecha válido.
	Observaciones	Los campos usuario, contraseña e identificador son obligatorios. El resto de los campos toman valores por defecto. En caso de no existir la tarea con el identificador introducido se muestra el mensaje “La tarea no existe”. Una vez modificada la tarea, se muestra el etag e identificador de la tarea.	

Anexo 11 Descripción del requisito funcional Obtener tarea.

No	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para el cliente**
RF14	Obtener tarea	Muestra un listado con el identificador, etag, fecha inicial, fecha final, cuerpo y otros atributos de las tareas.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	

	Usuario	Texto	El campo debe ser un usuario válido del dominio
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.
	Observaciones	Esta funcionalidad muestra todas las tareas que posee un usuario en su correo electrónico, en caso de introducir además de los datos requeridos el identificador de una tarea, se muestra la información correspondiente a esa tarea.	

Anexo 12 Descripción del requisito funcional Eliminar tarea.

No	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para el cliente**
RF15	Eliminar tarea	Elimina permanentemente una tarea previamente creada.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Identificador	Texto	Debe existir un evento creado con este identificador.	
	Observaciones	En caso de que existe una tarea con el valor del campo identificador, se muestra el mensaje de confirmación “Tarea eliminada satisfactoriamente”.		

Anexo 13 Descripción del requisito funcional Enviar mensaje instantáneo.

No	Nombre	Descripción	Complejidad	Prioridad para el cliente
RF16	Enviar mensaje instantáneo	Es enviado un mensaje de forma instantánea a una dirección válida para un usuario del dominio.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Receptor	Texto	Dirección de un contacto válida para un usuario del dominio.	
	Mensaje	Texto	Cualquier cadena de caracteres.	
	Observaciones	En caso de no ser introducidos los campos obligatorios se muestra el mensaje “Campo (nombre del campo) no especificado”, sino se muestra el mensaje “Mensaje enviado”.		

Anexo 14 Descripción del requisito funcional Recibir mensaje instantáneo.

No	Nombre	Descripción	Complejidad	Prioridad para el cliente
RF17	Recibir mensaje instantáneo		Alta	Alta

	Prototipo		
	Campos	**Tipos de Datos**	**Reglas o Restricciones**
	Usuario	Texto	El campo debe ser un usuario válido del dominio
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.
	Observaciones	Se muestra la dirección del contacto y los mensajes que han sido enviados a el por ese contacto.	

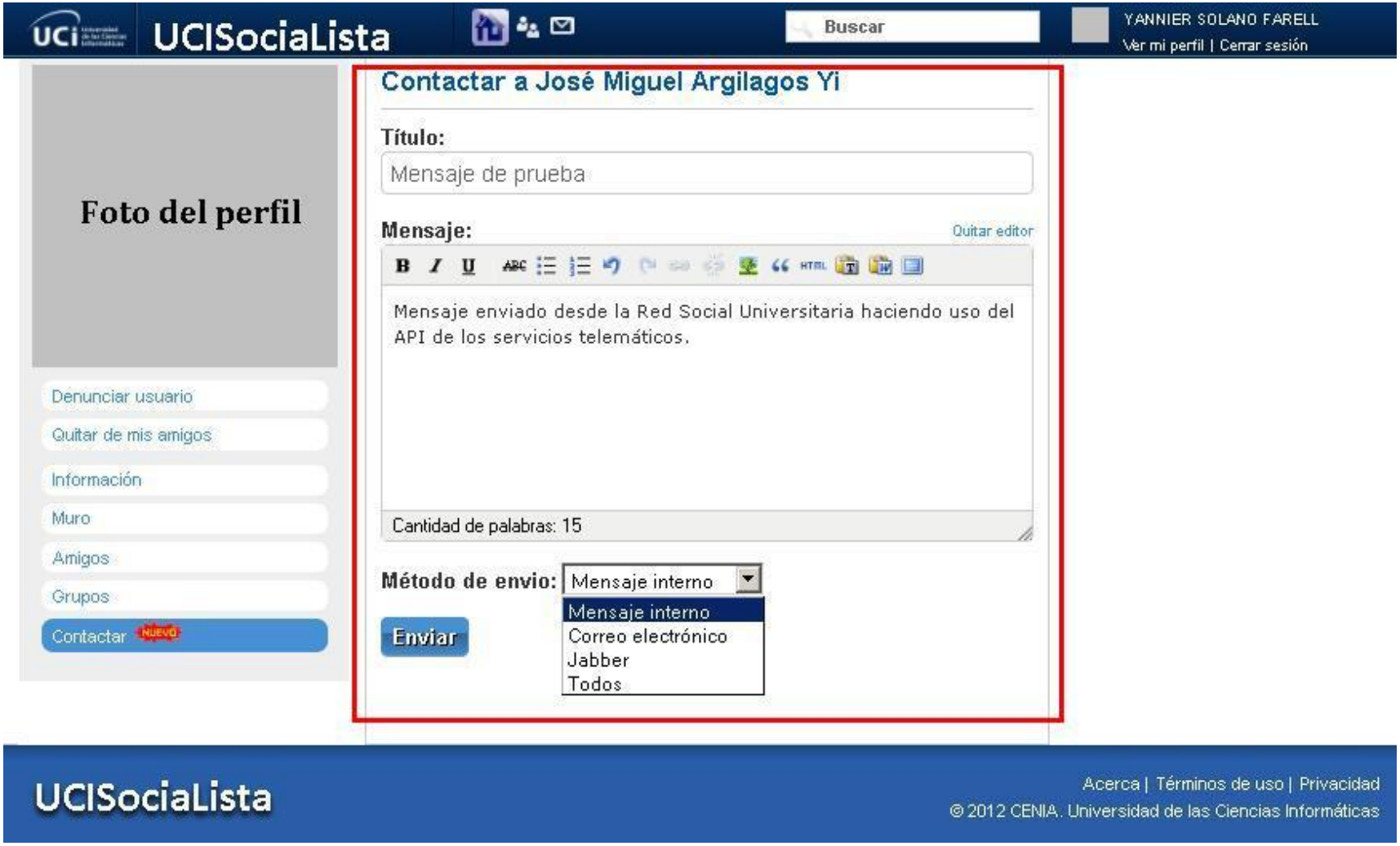
Anexo 15 Descripción del requisito funcional Mostrar listado de contactos.

No	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para el cliente**
RF18	Mostrar listado de contactos	Se muestra una lista con todos los contactos que posee un usuario en su roster.	Alta	Alta
	Prototipo			
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Usuario	Texto	El campo debe ser un usuario válido del dominio	
	Contraseña	Texto	Debe ser una contraseña válida para un usuario del dominio.	
	Observaciones	Se muestran las direcciones de los contactos, su estado, mensaje de estado y la prioridad del cliente en el que se encuentra conectado.		

Anexo II. Integración de la API con servicios de la RSU

Anexo 16 Integración de las cuotas de navegación con la RSU.

ANEXOS


UCISociaLista
Buscar
YANNIER SOLANO FARELL
Ver mi perfil | Cerrar sesión
Foto del perfil
Denunciar usuario
Quitar de mis amigos
Información
Muro
Amigos
Grupos
Contactar
NUEVO
Contactar a José Miguel Argilagos Yi
Título:
Mensaje de prueba
Mensaje:
Quitar editor
Mensaje enviado desde la Red Social Universitaria haciendo uso del API de los servicios telemáticos.
Cantidad de palabras: 15
Método de envio:
Mensaje interno
Mensaje interno
Correo electrónico
Jabber
Todos
Enviar
UCISociaLista
Acerca | Términos de uso | Privacidad
© 2012 CENIA. Universidad de las Ciencias Informáticas