UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 4

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

Actualización y Soporte a la "Alternativa para el uso del Juez en Línea Caribeño en entornos con dificultades de conectividad"

Autor: Yobanni Pérez Pérez

Tutor: Ing. Yonny Mondelo Hernández

La Habana, 17 de junio del 2015

Declaración de Autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____

Firma del Autor

_____

Firma del Tutor

Agradecimientos

A mis padres que siempre han estado conmigo en las buenas y en las malas.

A mi familia por apoyarme en todo momento.

A mi tutor Yonny Mondelo Hernández.

A Michael Horta por ayudarme cuando le pedí su ayuda.

A mis compañeros y amigos de aula.

A toda persona que de una u otra forma me ayudó a pasar los años de curso de una mejor manera.

Resumen

Los jueces en línea son aplicaciones generalmente web disponibles todo el tiempo, que automatizan la evaluación de las respuestas a problemas de programación de computadoras. Creados con el objetivo de potenciar el desarrollo de habilidades y técnicas en programación, además del diseño de algoritmos y preparación para concursos de programación. En la Universidad de las Ciencias Informáticas se creó en el año 2010 el Juez en Línea Caribeño (COJ por sus siglas en inglés), con el objetivo de proveer un espacio a los usuarios de todo el mundo, donde intercambien sus experiencias y conocimientos, además de mejorar y compartir sus problemas de programación y sus soluciones, así como sus habilidades en programación. El objetivo de la presente investigación, es realizar actualización y soporte a la “Alternativa para el uso del Juez en Línea Caribeño en entornos con dificultades de conectividad”. Para su cumplimiento se realizó un estudio a los protocolos SMTP para la transferencia de mensajes e IMAP para la descarga de los mensajes, además del uso del protocolo SSL/TLS para la conexión segura en la red. Se seleccionaron las tecnologías, herramientas y metodología de desarrollo, obteniéndose como resultado un sistema actualizado capaz de interactuar con el Juez en Línea Caribeño en entornos con dificultades de conectividad. Siendo de gran ayuda, ya que sirve como una alternativa para aquellos usuarios que no pueden acceder al Juez en Línea Caribeño mediante la vía de internet, debido a problemas con la conectividad.

Abstract

The online Judges are web applications generally available all the time that automate the evaluation of responses to computer programming problems. Created with the aim of promoting the development of skills in programming techniques, algorithm design and preparation of programming contests. At the University of Information Sciences in Caribbean Online Judge (COJ for its acronym in English) it was established in 2010 with the aim of providing a space users worldwide test, enhance and share their problems and solutions and their programming skills. The objective of this research is to update and support "alternative to the use of Caribbean Online Judge in environments with connectivity difficulties." For the compliance was conducted a study the protocol SMTP to transfer messages and IMAP to download messages, and the use of SSL / TLS protocol for secure network connection. Were selected Technologies, tools and methodology development, obtaining resulting in an updated able to interact with the Caribbean Online Judge in difficult environments connectivity.

Índice

Índice de contenido

Índice

Índice de Tablas

Índice de Tablas

Índice de ilustraciones

Introducción

El ACM-ICPC es una de las competiciones de programación en la que se encuentran involucrados principalmente estudiantes y profesores de diversas universidades de todo el mundo. Desde su surgimiento ha ido obteniendo un notable auge y prestigio, debido al desarrollo de habilidades que se obtienen en el ámbito de la programación.

La Universidad de las Ciencias Informáticas (UCI) está vinculada a este prestigioso evento competitivo con la participación de sus estudiantes y profesores. Con el propósito de intercambiar experiencias y conocimientos, además de mejorar y compartir sus ejercicios y soluciones, profesores y estudiantes de la UCI crean el Juez en Línea Caribeño (COJ, por sus siglas en inglés).

El COJ cuenta ya con más de 2200 problemas de distintas fuentes y dificultades lo que va creciendo paulatinamente, permite el envío de soluciones (esta opción requiere registro) escritos en diversos lenguajes de programación tales como Pascal, Python, C/C++, Java, así como en otros lenguajes de programación. Además brinda una serie de servicios como: estadísticas, comparación entre usuarios, concursos, recomendador de problemas, entre otros (1).

El bloqueo económico impuesto a nuestro país hace más de 50 años, afecta de manera considerable todos los sectores de la sociedad cubana. Uno de estos sectores es la Educación Superior, lo que hace que la mayoría de las universidades cubanas no puedan aprovechar bien el Juez en Línea Caribeño, debido a la mala disponibilidad de la red que está sujeta a intermitencias, su velocidad es baja o porque el ancho de banda para la navegación es muy pobre, por lo que es muy complicado mantener un acceso a internet constantemente y sin que existan fallos en el momento de acceder a los servicios que brinda el COJ.

Aun cuando la red universitaria no presente problemas para acceder al COJ, otros factores pueden interferir, los mismos pueden ser organizacionales (horarios de clases, pases, o demás afectaciones del cronograma de actividades del estudiante); tecnológicos (disponibilidad de un puesto trabajo o de herramientas de programación adecuadas a la tecnología que se desea utilizar); o simplemente humanos (para prácticas en equipo, dificultad de coordinar la presencia de todos los integrantes del mismo equipo en un solo lugar, o para prácticas individuales) (2).

Introducción

Por tales motivos en 2011 se crea en la UCI una aplicación llamada CojMail, capaz de interactuar con el Juez en Línea Caribeño (COJ) estando en entornos con dificultades de conectividad a través de correo electrónico usando los protocolo SMTP (Protocolo Simple de Transferencia de Correo) para la transferencia de mensajes e IMAP en su versión 4 (Protocolo de Internet para el acceso a Mensajes) para la descarga de los mensajes.

En octubre de 2011 la arquitectura base del COJ fue sustituida, quedando obsoleta su arquitectura inicial, provocando que el CojMail dejase de ofrecer sus servicios, debido a que no era compatible con la nueva arquitectura. Más adelante se realiza una actualización al CojMail, corrigiendo las deficiencias que presentaba, además soportando un gran número se servicios que brindaba el COJ, pero esta prestó muy poco sus servicios debido a las siguientes razones:

1. Carencia de personal para dedicarle el soporte necesario a las funcionalidades existentes como:
    - Consultar perfil de usuario.
    - Consultar ver problema.
    - Consultar estadísticas de un concurso.
2. No ofrece soporte a servicios importantes como:
    - Consultar las últimas entradas.
    - Publicar una entrada.
    - Consultar estadística de un usuario en un Concurso.
    - Consultar el listado de competencias (próximas o en ejecución) de diferentes jurados.
    - Consultar premios de un concurso.

Todos estos inconvenientes no permiten aprovechar al máximo las funcionalidades que brinda el COJ.

Por lo antes planteado se identifica como **problema a resolver**: ¿Cómo mejorar la interacción de los usuarios con el Juez en Línea Caribeño, para acceder a un mayor número de servicios en entornos con dificultades de conectividad?

A partir del problema científico se puede inferir que el **objeto de estudio** es: Sistemas de evaluación automática y protocolos de comunicación vía correo electrónico.

Se tiene como **campo de acción**: Aplicaciones de correo electrónico como vía acceso a los servicios de los jueces en línea de programación.

Introducción

Objetivo: Realizar actualización y soporte a la alternativa para el uso del Juez en Línea Caribeño, que permita a los usuarios acceder a un mayor número de servicios, en entornos con dificultades de conectividad.

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

1. Establecer los referentes teóricos que sustentan la investigación.
2. Definir los aspectos funcionales y no funcionales que debe tener la propuesta de solución.
3. Realizar soporte a funcionalidades existentes.
4. Implementar y probar las nuevas funcionalidades.
5. Realizar pruebas de software a la propuesta de solución.

Se presenta como **Hipótesis**: La actualización y soporte a la alternativa para el uso del Juez en Línea Caribeño, permitirá acceder a un mayor número de servicios que brinda el Juez en Línea Caribeño en entornos con dificultades de conectividad.

Como posible **resultado** de la investigación sería: Un Sistema actualizado que interactúa con el Juez en Línea Caribeño estando en entornos con dificultades de conectividad y su documentación correspondiente.

Las **tareas de Investigación** para cumplir con los objetivos son:

1. Revisión bibliográfica para conformar el estado del arte de la presente investigación.
2. Selección de las herramientas y metodologías para la implementación del sistema.
3. Definir los aspectos funcionales y no funcionales de la actualización y soporte del CojMail.
4. Confección de los artefactos que propone la metodología de desarrollo seleccionada.
5. Implementación de la propuesta de solución.
6. Realización de las pruebas de software a la aplicación.

Los **métodos científicos** utilizados durante el transcurso de la investigación son:

Métodos teóricos y métodos empíricos.

Métodos teóricos.

Introducción

Analítico - sintético: utilizado para analizar las soluciones existentes y determinar las características que debe tener la solución a realizar, además hacer un estudio de las características de la metodología, las tecnologías y herramientas que sustentan el proceso de desarrollo.

Histórico-Lógico: utilizado para analizar la evolución y desarrollo del CojMail.

Métodos empíricos

Observación: utilizado para observar los problemas de conectividad al Juez en Línea Caribeño que presentan las universidades cubanas, los servicios que no brinda por la versión anterior del CojMail, así como sus características, entre otros elementos necesarios.

La presente investigación está organizada de la siguiente forma:

Capítulo 1: Fundamentación Teórica

En este capítulo se hará un análisis de las aplicaciones CojMail existentes. Además, se realiza un estudio de las herramientas, tecnologías y metodologías más destacadas en el desarrollo informático que se utilizarán en la presente investigación.

Capítulo 2: Descripción de la propuesta de solución, detallando cada uno de sus aspectos funcionales y no funcionales además de su planificación y diseño.

Capítulo 3: Implementación del sistema y fase de pruebas.

Capítulo 1

Capítulo1 Fundamentación Teórica

1.1 Introducción

En este capítulo se realiza un estudio a las aplicaciones existentes. Además se analizan las herramientas y tecnologías que se utilizarán en la presente investigación, así como la selección de la metodología en el proceso de desarrollo.

1.2 Conceptos relacionados con la investigación

Aplicaciones web: Una aplicación web es cualquier aplicación que es accedida vía web por una red como internet o una intranet (3).

En general, el término también se utiliza para designar aquellos programas informáticos que son ejecutados en el entorno del navegador (por ejemplo, un *applet* de Java) o codificado con algún lenguaje soportado por el navegador (como JavaScript, combinado con HTML); confiándose en el navegador web para que reproduzca (3).

Aplicaciones Desktop: Una aplicación de escritorio son aplicaciones para ser instaladas, configuradas y utilizadas en ordenadores de escritorio o portátiles sin necesidad de utilizar un navegador (4).

Aplicaciones Mobile: Un software que se ejecuta en un teléfono inteligente, tableta u otro dispositivo portátil (5).

Jueces en línea de programación: Los jueces en línea son generalmente aplicaciones web disponibles todo el tiempo, que automatizan la evaluación de las respuestas a problemas de programación de competencia (2).

1.3 Análisis de soluciones similares

Luego de realizar un estudio a los jueces en línea de programación más prestigiosos a nivel mundial (Universidad de Valladolid Online Judge (UVA), *Timus Online Judge* (TIMUS), *Sphere Online Judge* (SPOJ), *CodeForces* y *TopCoder)* sobre si contenían algún sistema capaz de interactuar con los servicios que estos brindaban, se pudo evidenciar que estos hacen uso del protocolo SMTP en el envío de notificaciones por correo electrónico, pero ninguno posee un sistema capaz de interactuar a los servicios mediante correo electrónico.

Capítulo 1

En la Universidad de las Ciencias Informáticas se desarrollaron dos aplicaciones capaces de comunicarse con el COJ y consumir los servicios que este brindaba, a continuación se hace referencia a estas aplicaciones.

Implementación y prueba de un Jurado Online sobre correo electrónico (CojMail1.0).

Esta aplicación se creó con el fin de mejorar la interacción con Juez en Línea Caribeño vía correo electrónico con problemas con la conectividad. Esta solución tenía el objetivo de interactuar con las funcionalidades del COJ mediante correo electrónico en vez de acceder directamente al COJ vía internet.

El acceso mediante vía internet al COJ dificultaba a las universidades, esto se podría ver resuelto creando una aplicación similar siendo de manera local, pero tendría el inconveniente de que solo se accedería al mismo solamente los de la propia universidad, apartando uno de los objetivos fundamentales del COJ que es medir el conocimiento de los estudiantes de diferentes universidades mediante competencias, además de compartir sus soluciones (6).

Funcionamiento general del CojMail 1.0

El CojMail es un ***daemon.*** [1] "Su funcionamiento se basa en descargar de manera constante los correos que se encuentren en la bandeja de entrada del buzón asignado. Una vez realizada la descarga, el sistema da respuesta a la petición y elimina el correo del servidor. En caso de que no haya correos para responder, la aplicación espera un tiempo definido por los administradores antes de ejecutar una nueva descarga (2).

Micro lenguaje

Para establecer comunicación con el COJ vía correo electrónico los usuarios deben enviar un correo a la dirección electrónica del correo impersonal del COJ (coj@uci.cu). En el asunto de dicho correo el usuario debe escribir unos comandos particulares conocidos como micro lenguaje. Este término representa el lenguaje común entre los usuarios y el CojMail, entendible desde luego por ambas partes. El objetivo del micro lenguaje es que los usuarios puedan especificar la información que desean gestionar u obtener, y en

[1] **Daemon**: programa o proceso que se ejecuta constantemente, por lo general en plataformas UNIX, con el fin de realizar determinadas tareas, por ejemplo la invocación de un servicio web, la recepción y envío de correos electrónicos, entre otros.

muchos casos, la forma en que quieren obtenerla. La estructura del micro lenguaje es similar a muchos de los comandos empleados en consola (existencia de pares con la forma<*criterio*><*valor*>) (2).

Principales deficiencias de la primera versión del CojMail

El principal problema de esta primera versión del CojMail (en lo adelante CojMail 1.0) es que no era compatible con la arquitectura que presentaba en aquel momento el COJ, razón por la cual no estaba en condiciones de realizar prestaciones para el juez en línea. Desde el punto de vista funcional, el COJMail1.0 no satisfacía las necesidades de los usuarios del COJ (6).

Alternativa para el uso del Juez en Línea Caribeño en entornos con dificultades de conectividad (CojMail2.0)

Es una actualización de CojMail1.0 donde quedan erradicadas las deficiencias encontradas en la versión anterior (6).

Este sistema es capaz de acceder a gran cantidad de servicios que brinda el Juez en Línea Caribeño entre los que se encuentran: Registrar un nuevo usuario, Activar una cuenta de usuario, Consultar perfil de usuario, Consultar listado de problemas, Consultar estadísticas de concursos, entre otros, pero su principal problema de esta versión es la carencia de personal para brindarle un soporte y mantenimiento a las funcionalidades existentes, además desde el punto de vista funcional el CojMail2.0 presenta el inconveniente de que:

- No brinda el soporte a los servicios consultar últimas entradas, publicar una entrada, consultar el listado de competencias existentes (próximas o en ejecución) en distintos jueces en línea, consultar los premios otorgados en un concurso determinado, consultar la estadística obtenida de un usuario en un concurso.

Esto significa que el CojMail2.0 no satisface las necesidades actuales de los usuarios de acceder a un mayor número de servicios que brinda actualmente el COJ.

1.4 Protocolos

La alternativa para el uso del Juez en Línea Caribeño en entornos con dificultades de conectividad, es un sistema que descarga los correos que se encuentran en el buzón del COJ y de inmediato envía una respuesta al destinatario del correo descargado. Para realizar estas acciones se hace uso de protocolos de

correos SMTP e IMAP y protocolos de seguridad SSL/TLS. Como se definió en el objetivo de la investigación se pretende realizar actualización y soporte a este sistema, lo que es necesario realizar un estudio a los protocolos SMTP e IMAP para el envío de correo y descarga de correos respectivamente, además del protocolo SSL/TLS para la conexión segura en la red. Estos protocolos se describen a continuación.

1.4.1 Protocolos de correos electrónicos

Un protocolo de correo electrónico es el medio a través del cual diferentes máquinas, a menudo usando sistemas operativos y clientes de correo diferentes, pueden intercambiar mensajes de correo electrónico (6).

Protocolo SMTP

SMTP es el protocolo usado para el envío de correo electrónico entre computadoras u otros dispositivos. Entre las propiedades del SMTP cabe destacar la utilización de listas de mensajería, la gestión de acuses de recibo y el reenvío de mensajes. El protocolo SMTP no especifica cómo se crean los mensajes, para este fin se necesita un programa de correo electrónico nativo o un editor local. Una vez que se ha creado el mensaje, SMTP lo acepta y hace uso del TCP para enviarlo al módulo SMTP en el computador remoto. En el receptor, el módulo SMTP utilizará su aplicación de correo electrónico local para almacenar el mensaje recibido en el buzón de correo del usuario destino (6).

Características de SMTP:

- Permite la comunicación entre servidores de diferentes plataformas.
- Utiliza el puerto 25 para la conexión y el puerto 465 para la conexión segura.
- La conexión entre emisor y receptor es unidireccional. Esto significa que el emisor puede enviar mensajes al receptor, sin embargo durante la conexión establecida el receptor no puede enviar mensajes al emisor. Para que el receptor pueda enviar mensajes debe finalizar la conexión anterior e iniciar una en sentido contrario; en este caso se invierten los roles emisor y receptor (6).

Protocolo IMAP

El protocolo IMAP, definido en el documento RFC 2060, al igual que POP es utilizado por los usuarios para acceder a los mensajes de correo electrónico contenidos en un servidor. Actualmente IMAP se

encuentra en la versión 4 (IMAP4) de su desarrollo. Con IMAP los mensajes permanecen en el servidor de correo, donde el usuario puede leerlos, eliminarlos y gestionar los buzones de correo (6).

“Además, el protocolo IMAP es totalmente compatible con importantes estándares de mensajes de Internet, como MIME, que permiten recibir ficheros adjuntos. Muchos clientes de correo electrónico que utilizan el protocolo IMAP también se pueden configurar para que se almacene temporalmente en caché una copia de los mensajes localmente, de modo que el usuario puede examinar los mensajes que ha leído anteriormente si no está conectado directamente al servidor IMAP” (6).

“IMAP lo utilizan principalmente los usuarios que pueden obtener acceso a su correo desde varias máquinas, como mensajes almacenados en una ubicación central y a los que puede acceder cualquier sistema que utilice un cliente de correo IMAP y una conexión con el servidor IMAP remoto. También los usuarios que se conectan a Internet o a una red privada a través de una conexión de ancho de banda baja utilizan a menudo el protocolo IMAP, puesto que sólo la información de cabecera del correo se obtiene inicialmente. Esto les permite posponer la descarga de mensajes que tienen ficheros adjuntos de gran tamaño hasta el momento en el que no se utilice el limitado ancho de banda. De la misma manera, el usuario puede eliminar el correo electrónico que no le interesa sin tener que ver antes el cuerpo del mensaje, lo cual evita el tener que descargar un mensaje mediante la conexión de red que utilicen” (6).

Característica de IMAP.

- A diferencia de POP, permite el acceso simultáneo desde múltiples clientes (Buzones multiusuario).
- Existencia de banderines para el control de los diferentes estados de un mensaje.
- Soporte a operaciones para crear, eliminar y cambiar el nombre de las carpetas de correos, comprobar si hay mensajes nuevos y ordenarlos en base a criterios definidos (6).

1.4.2 Seguridad en los protocolos de correos electrónicos

Inicialmente los protocolos de correo electrónico estudiados no incluían suficientes mecanismos o métodos de seguridad. En especificaciones posteriores de estos protocolos se incluyeron mecanismos para garantizar la confidencialidad e integridad de los datos. En el caso de SMTP se incluyó el comando AUTH (RFC 2554) que permite la autenticación del cliente en el servidor de correo, erradicando una de las principales vulnerabilidades del protocolo, la proliferación de correos masivos, generalmente conocidos

como (spam). Tanto SMTP, IMAP, como POP, incluyen soporte al protocolo SSL/TLS con el propósito de lograr seguridad en la información enviada y recibida a través de correo electrónico. En la siguiente tabla se muestra el identificador y puerto de las extensiones de SMTP, IMAP y POP sobre SSL/TLS (6).

Protocolo	Identificador	Puerto TCP (por defecto)
SMTP sobre SSL/TLS	Smtps	465
IMAP sobre SSL/TLS	Imaps	993
POP sobre SSL/TLS	Pops	995

Tabla 1-Protocolos

Protocolo SSL/TLS

"El protocolo SSL (Secure Sockets Layer) y su sucesor TLS (Transport Layer Security) son protocolos criptográficos que proporcionan comunicaciones seguras por una red. Se basa en un proceso de cifrado de clave pública estableciendo un canal de comunicación seguro (cifrado) entre dos equipos después de una fase de autentificación (7)."

"SSL proporciona confidencialidad mediante el uso de encriptación, integridad, ya que los datos recibidos son exactamente iguales a los datos enviados y autentificación, ya que se realiza utilizando un certificado digital por parte del servidor. (Y cliente opcionalmente) (7)."

SSL/TLS se encuentra en la capa de transporte del modelo TCP/IP, por debajo de la capa de aplicación donde se encuentran los protocolos SMTP, POP, IMAP, HTTP, FTP, entre otros. Esto quiere decir que SSL/TLS es independiente de los protocolos de aplicación, por lo que se pueden realizar conexiones a través de estos protocolos cifrando la información con SSL/TLS (7).

1.5 Metodología de desarrollo

"Una metodología es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información. Una metodología está formada por fases, cada una de las cuales se puede dividir en sub-fases, que guiarán a los desarrolladores de sistemas a elegir las técnicas más apropiadas en cada momento del proyecto y también a planificarlo, gestionarlo, controlarlo y evaluarlo" (8).

Capítulo 1

Las metodologías se clasifican en dos grandes grupos:

- **Metodologías tradicionales:** orientadas al control de los procesos, estableciendo rigurosamente las actividades a desarrollar, herramientas a utilizar y notaciones que se usarán.

- **Metodologías ágiles:** Las orientadas a la interacción con el cliente y el desarrollo incremental del software, mostrando versiones parcialmente funcionales del software al cliente en determinados intervalos de tiempo, para que pueda evaluar y sugerir cambios en el producto (6).

Tabla 2-Metodologías ágiles vs tradicionales

Metodologías Ágiles	**Metodologías Tradicionales**
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo.
Especialmente preparados para cambios durante el proyecto.	Cierta resistencia a los cambios.
Impuestas internamente (por el equipo de desarrollo).	Impuestas externamente.
Proceso menos controlado, con pocos principios.	Proceso mucho más controlado, con numerosas políticas/normas.
No existe contrato tradicional o al menos es bastante flexible.	Existe un contrato prefijado.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.
Grupos pequeños (menos de diez integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos.	Muchos artefactos.

Pocos roles.	Muchos roles.
Menos énfasis en la arquitectura de software.	La arquitectura de software es esencial y se expresa mediante modelos.

Selección de la metodología

De acuerdo a las características presentadas por ambas metodologías y centrándose en la presente investigación se decide optar para el desarrollo del trabajo de investigación: la metodología ágil.

Motivos de la selección

1. Puede ocurrir gran probabilidad de que el software tenga variación en los requisitos.
2. El grupo de desarrollo es pequeño.
3. No es necesario tener muchos roles.

En general las metodologías ágiles ofrecen una solución casi a la mediada para una gran cantidad de proyectos que tienen estas características. Además una de las cualidades más destacadas en una metodología ágil es su sencillez, tanto en su aprendizaje como en su aplicación, reduciéndose así los costos de implementación el equipo de desarrollo.

1.5.1 Xtreme Programming (XP)

Historia.

La programación extrema o *Xtreme Programming* (XP) es un enfoque de la ingeniería de software formulado por Kent Beck, autor del primer libro sobre la materia, *Extreme Programming Explained*: *Embrace Change* (1999). Es el más destacado de los procesos ágiles de desarrollo de software. La programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos (9).

Capítulo 1

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico (9).

A continuación se presenta las características esenciales de XP organizadas e los 3 apartados siguientes: historias de usuario, roles, procesos y prácticas.

Historias de usuario

Son la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuarios es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas (10).

Roles XP

Los roles de acuerdo con la propuesta original de Beck son (10):

- **Programador:** El programador escribe las pruebas unitarias y produce el código del sistema.
- **Cliente:** Escribe las historias de usuario y las pruebas funcionales para validar su implementación.
- **Encargado de pruebas (*Tester*):** Ayuda al cliente de escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas y soporte para pruebas.
- **Encargado de seguimiento (*Tracker*):** Proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas. Realiza el seguimiento del proceso de cada iteración.
- **Entrenador (*Coach*):** Es el responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente.
- **Consultor:** Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas.

- **Gestor (*Big Boss*):** Es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es la coordinación.

Proceso XP

El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá la calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible en cada iteración. El ciclo de vida ideal de XP consiste en 6 fases: Exploración, Planificación de la entrega (*Release*), Iteraciones, Producción, Mantenimiento y muerte del proyecto (10).

Prácticas XP

La principal suposición que se realiza en XP es la posibilidad de disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y a la aplicación disciplinada de las siguientes prácticas (10):

- **El juego de la planificación:** Hay una comunicación frecuente entre el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas y de cada iteración.
- **Entregas pequeñas:** Producir rápidamente versiones del sistema que sean operativas aunque no cuenten con toda la funcionalidad del sistema. Esta versión ya constituye un resultado de valor para el negocio. Una entrega no debería tardar más de 3 meses.

- **Diseño simple:** Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.
- **Pruebas:** La producción de código está dirigida por las pruebas unitarias. Estas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente ante cada modificación del sistema.
- **Refactorización (*Refactoring*):** Es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar posteriores cambios. Se mejora la estructura interna del código sin alterar su comportamiento externo.
- **Programación en parejas:** Toda la producción del código de realizarse con trabajo en parejas de programadores. Esto conlleva ventajas implícitas (menor tasa de errores, mejor diseño, mayor satisfacción de los programadores,…).
- **Estándares de programación:** XP enfatiza que la comunicación de los programadores es a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación para mantener el código legible.

El mayor beneficio de las prácticas se consigue con su aplicación conjunta y equilibrada puesto que se apoyan unas de otras, como se puede observar en la ilustración1 (…). La mayoría de las prácticas propuestas por XP no son novedosas sino que en alguna forma ya habían sido propuestas en ingeniería del software e incluso demostrado su valor en la práctica (…). El mérito de XP es integrarlas de una forma efectiva y complementarlas con otras ideas desde la perspectiva del negocio, los valores humanos y el trabajo en equipo (10).

Capítulo 1

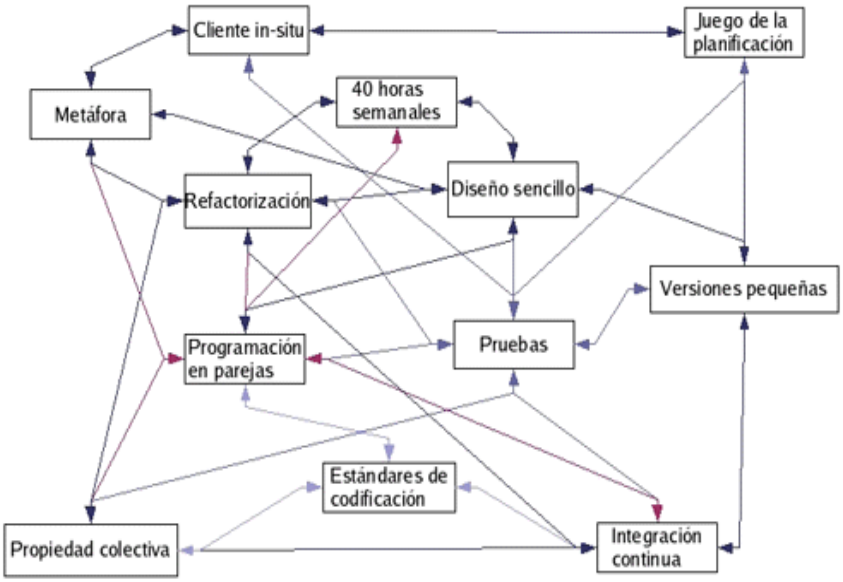

Ilustración 1-Las prácticas se refuerzan entre sí *(10)*.

Objetivos de XP (9)**.**

1. Establecer las mejores prácticas de Ingeniería de Software en los desarrollos de proyectos.
2. Mejorar la productividad de los proyectos.
3. Garantizar la Calidad del Software desarrollando, haciendo que este supere las expectativas del cliente.

Contexto XP (9)**.**

1. Cliente bien definido
2. Los requisitos pueden (y van a) cambiar
3. Grupo pequeño y muy integrado (máximo 12 personas)
4. Equipo con formación elevada y capacidad de aprender

Ventajas XP (9)**.**

1. Programación organizada.
2. Menor taza de errores.
3. Satisfacción del programador.

Desventajas XP (9).

1. Es recomendable emplearlo solo en proyectos a corto plazo.
2. Altas comisiones en caso de fallar.

1.5.2 SCRUM

Desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Define un marco para la gestión de proyectos. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas *sprints,* con una duración de 30 días. El resultado de cada *sprint* es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo del proyecto, entre ellas destaca la reunión diaria del equipo de desarrollo para coordinación e integración (10).

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos. En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales (11).

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto (11).

Requisitos importantes para utilizar Scrum (12)

Los siguientes puntos son de especial importancia para la implantación de una gestión ágil de proyectos como Scrum:

Cultura de empresa: La cultura de la empresa proveedora del proyecto debe estar alineada con la filosofía de una gestión ágil de proyectos como Scrum. Debe fomentar:

- El trabajo en equipo y la colaboración entre todas las personas implicadas en un proyecto.
- Equipos auto-gestionados a los que se ha delegado la responsabilidad y autoridad para hacer su trabajo, en contraposición a la dirección y control de personas que ejercería un jefe de proyecto tradicional (en lugar del jefe de proyecto existe la figura del Facilitador).
- La creatividad del equipo.
- La transparencia y la mejora continua, tanto del contexto de la organización y del proyecto, como de las herramientas del equipo.

Compromiso del cliente

Scrum exige del cliente una **alta implicación y una dedicación regular**:

El cliente tiene la responsabilidad de dirigir los resultados del producto o proyecto:

- El cliente debe disponer de una visión de alto nivel del producto o proyecto y tener reflejadas sus expectativas en forma de lista de requisitos priorizada donde ha indicado el valor que le aportará cada uno. A partir de los costes de desarrollo que le proporcione el equipo, priorizará los requisitos en función del Retorno de la Inversión (ROI) más rápido.
    - El cliente re-planifica el proyecto en cada iteración para maximizar este ROI de manera continua

Al tratarse de un proyecto que va entregando resultados en iteraciones regulares, el cliente debe colaborar participando en el inicio de cada iteración (reunión de planificación) y en el fin de cada iteración (demostración), y debe estar disponible durante la ejecución de cada iteración para resolver dudas.

Tamaño del equipo

El tamaño de un equipo está entre 5 y 9 personas. Por debajo de 5 personas, cualquier imprevisto o interrupción sobre un miembro del equipo compromete seriamente el compromiso que han adquirido y, por

tanto, el resultado que se va a entregar al cliente al finalizar la iteración. Por encima de 9 personas, la comunicación y colaboración entre todos los miembros se hace más difícil y se forma subgrupos.

Selección de la metodología

Siguiendo las características de ambas metodologías se puede apreciar que una como la otra tienen características similares, entre las que se encuentran: ambas están dirigidas fundamentalmente para equipos de desarrollo pequeño, cuenta con la superación continúa de los integrantes, existe simplicidad en las soluciones propuestas y el cliente se encuentra integrado en el proceso de desarrollo del software. Por lo que se puede escoger cualquiera de las dos metodologías, pero centrándose en la presente investigación, el equipo es un integrante y la metodología Scrum plantea que por debajo de 5 personas, cualquier imprevisto o interrupción sobre un miembro del equipo compromete seriamente el resultado que se va a entregar, por otro lado la metodología XP propone un trabajo en parejas lo que disminuye la tasa de errores y aumenta la satisfacción de los programadores. Por lo antes planteado se opta por la metodología *Xtreme Programming* (XP).

1.6 Lenguaje de programación

Cualquier lenguaje artificial que puede utilizarse para definir una secuencia de instrucciones para su procesamiento por un ordenador o computadora (13).

Para la actualización del CojMail se utilizará el lenguaje de programación Java por las siguientes razones.

- La versión anterior esta implementada java
- El lenguaje de programación java brinda gran cantidad de librerías relacionadas con correo electrónico

Se reutilizará el código y así no tener que migrarlo a otro lenguaje lo que ahorraría gran cantidad de tiempo.

1.6.1 Lenguaje de programación java

El lenguaje para la programación en Java, es un lenguaje orientado a objeto, de una plataforma independiente. Fue desarrollado por la compañía *Sun Microsystems*, con la idea original de usarlo para la creación de páginas web. La programación Java tiene muchas similitudes con el lenguaje C y C++, así que si se tiene conocimiento de este lenguaje, el aprendizaje de la programación Java será de fácil comprensión por un programador que haya realizado programas en estos lenguajes (14).

Capítulo 1

Con la programación en Java, se pueden realizar distintos aplicativos, como son *applets*, que son aplicaciones especiales, que se ejecutan dentro de un navegador al ser cargada una página HTML en un servidor web, Por lo general los *applets* son programas pequeños y de propósitos específicos (14).

Otra de las utilidades de la programación en Java es el desarrollo de aplicaciones, que son programas que se ejecutan en forma independiente, es decir con la programación Java, se pueden realizar aplicaciones como un procesador de palabras, una hoja que sirva para cálculos, una aplicación gráfica, etc. en resumen cualquier tipo de aplicación se puede realizar con ella. Java permite la modularidad por lo que se pueden hacer rutinas individuales que sean usadas por más de una aplicación (14).

La programación en Java, permite el desarrollo de aplicaciones bajo el esquema de Cliente Servidor, como de aplicaciones distribuidas, lo que lo hace capaz de conectar dos o más computadoras u ordenadores, ejecutando tareas simultáneamente, y de esta forma logra distribuir el trabajo a realizar (14).

Otras características de java

La independencia de plataforma es una de las razones por las que Java es interesante para Internet, ya que muchas personas deben tener acceso con ordenadores distintos. Pero no se queda ahí, Java está desarrollándose incluso para distintos tipos de dispositivos además del ordenador como móviles, agendas y en general para cualquier cosa que se le ocurra a la industria. Actualmente Java se utiliza en un amplio abanico de posibilidades y casi cualquier cosa que se puede hacer en cualquier lenguaje se puede hacer también en Java y muchas veces con grandes ventajas. Con Java podemos programar páginas web dinámicas, con accesos a bases de datos, utilizando XML, con cualquier tipo de conexión de red entre cualquier sistema. En general, cualquier aplicación que deseemos hacer con acceso a través web se puede hacer utilizando Java (15).

1.7 Gestor de Bases Datos (SGBD)

Los sistemas gestores de bases de datos, son una colección de programas que permiten a los usuarios crear y mantener una base de datos. Sistema software de propósito general que facilita los procesos de definición, construcción y manipulación de la base de datos para distintas aplicaciones (16).

1.7.1 SGBD PostgreSQL

Definiciones

Capítulo 1

PostgreSQL: Es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales. Utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Sus características técnicas la hacen una de las bases de datos más potentes y robustos del mercado. Su desarrollo comenzó hace más de 16 años, y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo (17).

PostgreSQL es un potente sistema de base de datos, de código abierto objeto-relacional. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación por su fiabilidad, integridad de datos y corrección. Se ejecuta en todos los sistemas operativos, incluyendo Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), y Windows. Es totalmente compatible con ACID, tiene soporte completo para claves foráneas, combinaciones, vistas, *triggers* y procedimientos almacenados (en varios idiomas). Incluye más 2008 tipos de datos, como: CHAR, VARCHAR, DATE, INTERVALO y TIMESTAMP. También es compatible con el almacenamiento de grandes objetos binarios, como imágenes, sonidos, o de vídeo. Cuenta con las interfaces de programación nativas para C / C ++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otros, y la documentación excepcional (18).

Características de PostgreSQL

1. Control de concurrencias para múltiples versiones (MVCC), esto permite la eficiencia para el trabajo con grandes volúmenes de datos.
2. Disponible para Linux y UNIX en todas sus variantes (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows 32/64bit.
3. Funciones/procedimientos almacenados (*stored procedures*) en numerosos lenguajes de programación, entre otros PL/pgSQL (similar al PL/SQL de Oracle), PL/Perl, PL/Python y PL/Tcl
4. Soporta consultas recursivas, Vistas, Disparadores, Herencia de tablas

Tabla 3-Límites de PostgreSQL

Límite	valor

Máximo tamaño base de dato	Ilimitado (Depende de tu sistema de almacenamiento)
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 (dependiendo del tipo)
Máximo número de índices por tabla	Ilimitado

El equipo de desarrollo del COJ (CDEVT) optó desde el inicio por PostgreSQL para el desarrollo del COJ. Este gestor ha manejado eficientemente los datos de un juez en línea que ya cuenta (hasta el momento) con más de 21500 usuarios registrados, más de 800000 soluciones enviadas, un archivo de problemas bastante extenso (más de 2300) y más de 300 concursos realizados (6). Por lo antes planteado y de acuerdo a las características que presenta el sistema gestor de base dato PostgreSQL se escoge dicho gestor.

1.8 Entorno Integrado de Desarrollo (IDE)

Un entorno de desarrollo integrado o *Integrated Development Environment* (IDE) es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un lenguaje de programación o bien, poder utilizarse para varios. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI. Los IDE pueden ser aplicaciones por si solas o pueden ser parte de aplicaciones existentes (19).

1.8.1 Entorno Integrado de Desarrollo Eclipse

El entorno de desarrollo integrado Eclipse es una herramienta de código abierto, creada inicialmente por IBM para desarrollar en java. Es una plataforma de desarrollo, diseñada para ser extendida de forma indefinida a través de *plug-ins*. Fue concebida desde sus orígenes para convertirse en una plataforma de

integración de herramientas de desarrollo. No tiene en mente un lenguaje específico, sino que es un IDE genérico, aunque goza de mucha popularidad entre la comunidad de desarrolladores del lenguaje **Java** usando el *plug-in* **JDT** que viene incluido en la distribución estándar del IDE. Proporciona herramientas para la gestión de espacios de trabajo, escribir, desplegar, ejecutar y depurar aplicaciones (20). Permite desarrollar aplicaciones desktop, web y móviles, además soporta los lenguajes Java, PHP, Ruby, C y C++.

Principales características de eclipse (20).

Perspectivas, editores y vistas: en Eclipse el concepto de trabajo está basado en las perspectivas, que no es otra cosa que una pre-configuración de ventanas y editores, relacionadas entre sí, y que permite trabajar en un determinado entorno de trabajo de forma óptima.

Gestión de proyectos: el desarrollo sobre Eclipse se basa en los proyectos, que son el conjunto de recursos relacionados entre sí, como puede ser el código fuente, documentación, ficheros configuración, árbol de directorios. El IDE proporciona asistentes y ayudas para la creación de proyectos, por ejemplo, cuando creamos uno, se abre la perspectiva adecuada al tipo de proyecto que estemos creando, con la colección de vistas, editores y ventanas pre-configurada por defecto.

Depurador de código: se incluye un potente depurador, de uso fácil e intuitivo, y que visualmente ayuda a mejorar el código. Para ello sólo debemos ejecutar el programa en modo depuración (con un simple botón.

Extensa colección de *plug-ins*: están disponibles en una gran cantidad, unos publicados por Eclipse y otros por terceros. Al haber sido un estándar de facto durante tanto tiempo (no el único estándar, pero sí uno de ellos), la colección disponible es muy grande. Los hay gratuitos, de pago, bajo distintas licencias, pero casi para cualquier cosa que imaginemos tenemos el *plug-in* adecuado (20).

1.8.2 Entorno Integrado de Desarrollo Netbeans

El entorno de desarrollo integrado Netbeans es un entorno de desarrollo gratuito y de código abierto que permite el uso de un amplio rango de tecnologías de desarrollo tanto para escritorio, como aplicaciones Web, o para dispositivos móviles, además puede instalarse en varios sistemas operativos: Windows, Linux, Mac OS (21). Netbeans soporta lenguajes como: Java, PHP, C, C++, JavaScript, Ruby entre otros.

Características principales de Netbeans (21).

- Suele dar **soporte** a casi todas las novedades en el lenguaje Java. Cualquier *preview* del lenguaje es rápidamente soportada por Netbeans.
- **Asistentes** para la creación y configuración de distintos proyectos, incluida la elección de algunos frameworks.
- Buen **editor de código**, **multilenguaje**, con el habitual coloreado y sugerencias de código, acceso a clases pinchando en el código, control de versiones, localización de ubicación de la clase actual, comprobaciones sintácticas y semánticas, plantillas de código, *coding tips*, herramientas de refactorización.
- Simplifica la **gestión de grandes proyectos** con el uso de diferentes vistas, asistentes de ayuda, y estructurando la visualización de manera ordenada, lo que ayuda en el trabajo diario. Una vez que nos metemos en una clase java, por poner un ejemplo, se mostrará distintas ventanas con el código, su localización en el proyecto, una lista de los métodos y propiedades (ordenadas alfabéticamente), también hay una vista que presenta las jerarquías que tiene la clase.
- **Optimización de código**: por su parte el *Profiler* ayuda a optimizar las aplicaciones e intentar hacer que se ejecuten más rápido y con el mínimo uso de memoria. Podemos igualmente configurarlo a nuestro gusto, aunque por defecto, ofrece opciones bastante útiles. Lo importante es que podemos ver el comportamiento de la aplicación y obtener indicadores e información de cómo y cuantos recursos consume, cuantos objetos se crean, también podemos obtener capturas del estado del sistema en diferentes momentos (*Snapshots*) y compararlos entre sí.
- **Acceso a base de datos**: desde el propio Netbeans se puede conectar a distintos sistemas gestores de bases de datos, como pueden ser Oracle, MySql y demás, y ver las tablas, realizar consultas y modificaciones, y todo ello integrado en el propio IDE.
- Se integra con diversos servidores de aplicaciones, de tal manera que podemos gestionarlos desde el propio IDE: inicio, parada, arranque en modo *debug*, despliegues. Entre otros podemos usar Apache Tomcat, GlassFish, JBoss, WebLogic, Sailfin, Sun Java System Application Server. Es fácilmente extensible a través de plug-ins (21).

Selección de IDE

Los IDE anteriormente mencionados tienen características muy similares. Ambos soportan varios lenguajes de programación entre los que se encuentra Java, el lenguaje utilizado en la propuesta solución, soportan

frameworks de desarrollo, etc. Por lo antes planteado, de acuerdo a las características de cada uno de los IDE, se puede escoger uno como el otro, pero como el equipo de desarrollo tiene mayor dominio y experiencia con Netbeans se decidió escoger este.

1.9 Servidor de aplicación Apache Tomcat

Tomcat. (También llamado Jakarta Tomcat o Apache Tomcat), es un servidor multiplataforma el cual funciona como contenedor de servlets desarrollado bajo el proyecto Jakarta en la *Apache Software Foundation* (22).

Desarrollo

Tomcat es mantenido y desarrollado por miembros de la *Apache Software Foundation* y voluntarios independientes. Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Apache *Software Licence* (22).

Entorno

Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java (22).

Características

- Implementado de *Servlet* 3.0, JSP 2.2 y EL2.2.
- Mejoras para detectar y prevenir "fugas de memoria" en las aplicaciones web.
- Limpieza interna de código.
- Soporte para la inclusión de contenidos externos directamente en una aplicación web.

1.10 Conclusiones del capítulo

El estudio de este capítulo permitió definir las herramientas y metodologías a utilizar en el proceso de desarrollo de la propuesta solución, además se hizo un estudio sobre los protocolos SMTP e IMAP para la descarga y envío de correos electrónicos además del protocolo SSL/TLS para lograr una seguridad en la información enviada. Se optó por metodología XP para llevar el proceso de desarrollo, como lenguaje de programación fue escogido el lenguaje de programación Java, como gestor de base de datos se escogió

Capítulo 1

PostgreSQL en su versión 9.3, como entorno de desarrollo integrado netbeans en su versión 8 y como servidor web el Apache Tomcat en su versión 8.0.3.

Tabla 4-Herramientas y tecnologías

Clasificación	Herramienta
Lenguaje de programación	Java
Gestor de Base de Datos	PostgreSQL
Entorno de Desarrollo Integrado	NetBeans
Servidor de aplicación	Apache Tomcat
Metodologías de desarrollo	Extreme Programming

Capítulo 2

Capítulo2 Descripción de la propuesta de solución

2.1 Introducción

Luego de haber realizado un estudio a las soluciones existentes (CojMail1.0 y CojMail2.0), haber seleccionado las herramientas, metodología y lenguaje de programación que se utilizará en la presente investigación, en este capítulo se realizará una descripción de la presente solución, aspectos funcionales y no funcionales, así como sus principales características. Se realizará el diseño y planificación de la presente investigación, teniendo en cuenta el enfoque de la metodología XP donde marca los pasos iníciales hacia la implementación de la solución. Los aspectos a enfatizar según la metodología planteada serán las Historias de Usuarios, Iteraciones, Plan de las Entregas y las Tarjetas CRC. Estas se explicarán en su debido apartado.

2.2 Descripción de la solución

La propuesta de solución tiene como objetivo realizar actualización y soporte al CojMail2.0 donde se le incorporará nuevas funcionalidades, y así ponerlo en funcionamiento para que los usuarios de universidades cubanas con problemas de conexión puedan acceder a un mayor número de servicios que brinda el COJ.

Esta aplicación es un ***daemon*** que está constantemente realizando descargas de los correos del buzón del COJ, luego se conecta a la base de datos del COJ y envía una respuesta de acuerdo a la petición realizada en dicho correo. Más adelante se describirá detalladamente dicho funcionamiento.

Características del CojMail

1. Soporta los protocolos SMTP para enviar los correos y el IMAP para descargar los correos.
2. Da una respuesta en formato HTML.

Principio de funcionamiento del CojMail

1. Esta aplicación realiza constantemente descargas de los correos que se encuentran en el buzón del COJ.
2. En caso de que existan correos el CojMail da respuesta a la petición
3. En caso de que no existan correos para responder el mismo espera un tiempo definido y vuelve a realizar el mismo procedimiento.

Capítulo 2

2.3 Funcionalidades

Tabla 5-Nuevas funcionalidades

2 Nuevas funcionalidades soportadas en la presente solución 3	Funcionalidades para actualizar
Consultar últimas entradas.	Consultar perfil de usuario
Publicar una entrada.	Consultar estadísticas de un concurso
Consultar premios de un concurso.	Ver problema.
Consultar estadística de un usuario en un concurso.	
Listar competencias (próximas o en ejecución).	

2.4 Aspectos no funcionales

Tabla 6-Aspectos no funcionales

Aspecto	Descripción
Software	➢ El Gestor de Base Datos debe ser versión 9.1 o superior. ➢ El usuario debe tener instalado la máquina virtual de java
Portabilidad	➢ La aplicación debe ejecutarse en Windows como Linux.

Diseño e Implementación	➢ Utilizar el lenguaje de programación java para su desarrollo. ➢ Utilización de la programación orientada a objeto. ➢ Utilización de la metodología ágil XP para el desarrollo del software.

2.5 Descripción del micro-lenguaje de las nuevas funcionalidades

Dado que se añadieron nuevas funcionalidades al CojMail es necesario definir un micro lenguaje para cada una de estas operaciones. La notación se define con la siguiente estructura **cm: acción<variables>** donde.

- **cm:** abreviatura de CojMail.
- **acción:** funcionalidad que se desea consultar**.**
- **variables:** término conformado por uno a varios pares criterio valor

Los pares criterio valor pueden estar encerrado en corchetes, esto quiere decir que pueden ser especificado a conveniencia por el usuario (opcional) en caso contrario es de carácter obligatorio (6).

Comando para consultar últimas entradas

cm: entradas

Comando para publicar una entrada

cm: publicar –w <mensaje>

Comando para consultar listado de competencias (próximas o en ejecución)

cm: competencias

Comando para consultar premios de un concurso

cm: premios_concurso<id concurso>

Comando para consultar la estadística de un usuario en un concurso

cm: estadistica_usuario_concurso<id concurso><user>

2.6 Funcionamiento general

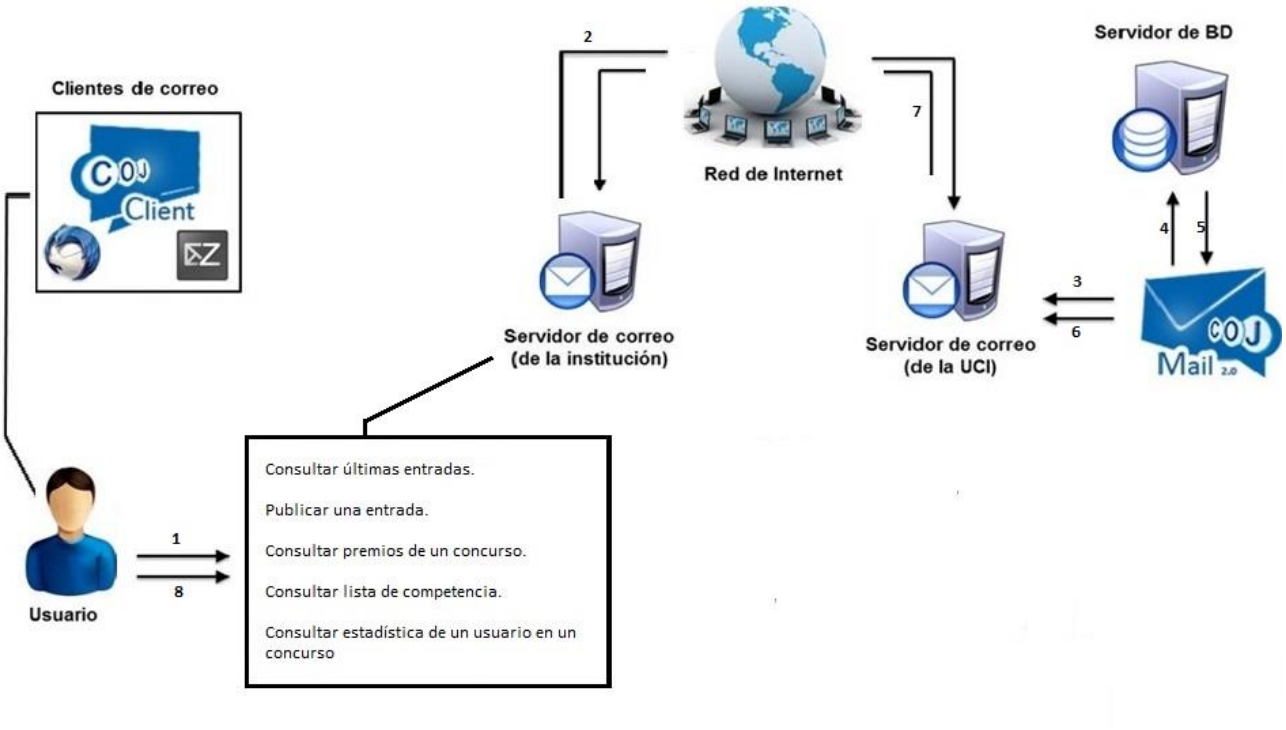

Ilustración 2-Funcionamiento general del proceso.

Descripción del proceso

1. El usuario realiza una petición, esta es enviada a través del correo de la institución donde radica el usuario
2. Usando un cliente de correo convencional, donde el usuario selecciona redactar un nuevo correo y escribe en el campo **Para** la dirección **coj@uci.cu** y en el campo **Asunto** el micro lenguaje correspondiente a la acción a que desea solicitar.
3. El correo donde radica el usuario se comunica con el servidor de correo de la UCI y envía la solicitud a la dirección electrónica del buzón del COJ, haciendo uso del protocolo SMTP.
4. El CojMail que está constantemente revisando el buzón del COJ, obtiene los correos de las

peticiones.

5. El CojMail consulta la base de datos.
6. El CojMail construye una vista a partir de la información obtenida.
7. El CojMail envía un correo de respuesta a través del servidor de correo de la UCI, desde el buzón de correo del COJ.
8. El servidor de correo de la UCI se comunica con el servidor de correo de la institución donde radica el usuario y envía la respuesta hacia su buzón de correo, haciendo uso del protocolo SMTP.
9. El usuario consulta la respuesta, haciendo uso del protocolo IMAP o POP3, en dependencia de cuál esté funcionando en su institución.

2.7 Planificación

La metodología XP plantea la planificación como un dialogo continuo entre las partes involucradas en el proyecto, incluyendo al cliente, a los programadores y a los coordinadores o gerentes. El proyecto comienza recopilando “Historias de usuarios”, las que sustituyen a los tradicionales “casos de uso”. Una vez obtenidas las “historias de usuarios”, los programadores evalúan rápidamente el tiempo de desarrollo de cada una (...). Una vez realizadas estas estimaciones, se organiza una reunión de planificación, con los diversos actores del proyecto (cliente, desarrolladores, gerentes), a los efectos de establecer un plan o cronograma de entregas (“Ralease Plan”) en los que todos estén de acuerdo. Una vez acordado este cronograma, comienza una fase de iteraciones, y en cada una de ellas, se desarrollan, prueban e instalan unas pocas “historias de usuarios” (23).

2.7.1 Historias de usuario

Las historias de usuario (HU) son utilizadas en la metodología XP, en la que son escritas por el cliente en un lenguaje proprio como descripciones cortas de lo que el sistema debe realizar. Las historias de usuarios deben tener el detalle mínimo para que los programadores puedan realizar una estimación del riesgo del tiempo que llevará la implementación del sistema. La estructura de la HU se muestra a continuación.

Plantilla de la historia de usuario (HU)

Capítulo 2

Tabla 7-Historias de usuario

Historia de Usuario	
Número: Número de la HU.	**Nombre:** Nombre de la HU.
Prioridad: Importancia para el cliente: Alta, Media, Baja.	**Complejidad:** Dificultad para el desarrollador: Alta, Media, Baja.
Iteración: Iteración a la que corresponde.	**Estimación:** Tiempo estimado para realizar la actividad.
Descripción: Descripción correspondiente a la HU, describiendo las acciones del usuario.	
Notas: Alguna observación que resulte de interés.	

Historias de usuario de las nuevas funcionalidades

A continuación se muestra una representación de las historias de usuario, las restantes se encuentran en los anexos de la investigación.

Tabla 8-HU últimas entradas

Historia de Usuario	
Número: 1	**Nombre:** Últimas entradas.
Prioridad: Alta	**Complejidad:** Media.
Iteración: 2	**Estimación:** 7 días
Descripción: El usuario tiene la posibilidad de acceder a las últimas entradas que han sido publicadas por los usuarios. Este puede realizarlo de la siguiente forma. 1. Mediante el cliente de correo convencional de su institución, escribiendo un correo al buzón del COJ. Este en el asunto debe contener el micro lenguaje correspondiente a la petición deseada.	

El sistema mostrará las últimas entradas publicadas por los usuarios.
Notas:

Tabla 9-HU publicar entrada

Historia de Usuario	
Número: 2	**Nombre:** Publicar una entrada.
Prioridad: Alta	**Complejidad:** Alta.
Iteración: 2	**Estimación:** 7 días
Descripción: El usuario tiene la posibilidad de publicar una entrada. Este puede realizarlo de la siguiente forma. 1. Mediante el cliente de correo convencional de su institución, escribiendo un correo al buzón del COJ. Este en el asunto debe contener el micro lenguaje correspondiente a la petición deseada. El sistema enviará una notificación si se publicó correctamente la entrada escrita por el usuario.	
Notas:	

Tabla 10-HU consultar premios de un concurso

Historia de Usuario	
Número: 3	**Nombre:** Consultar permios de un concurso.
Prioridad: Alta	**Complejidad:** Media.
Iteración: 2	**Estimación:** 7 días
Descripción: El usuario tiene la posibilidad de consultar los premios otorgados en un concurso determinado. Este puede realizarlo de la siguiente forma. 1. Mediante el cliente de correo convencional de su institución, escribiendo un correo al buzón del	

COJ. Este en el asunto debe contener el micro lenguaje correspondiente a la petición deseada. El sistema mostrará los premios correspondientes al concurso seleccionado.
Notas:

2.7.2 Iteraciones

Las iteraciones fueron realizadas de acurdo a como se realizará la presente solución. La primera iteración está enmarcada hacia la actualización de algunas funcionalidades existentes con su debido mantenimiento luego en la segunda iteración está dirigida a la implementación de 3 de las nuevas funcionalidades que se añadirán y por último la tercera iteración las últimas nuevas funcionalidades.

Primera iteración

- Actualizar “Consultar perfil de usuario”.
- Actualizar “Consultar estadística de un concurso”.
- Actualizar “Ver problema”.

Segunda iteración

- Consultar últimas entradas.
- Publicar una entrada.
- Consultar premios de un concurso

Tercera iteración

- Consultar estadística de un usuario en un concurso.
- Consultar listado de competencias (próximas o en ejecución).

Tabla 11-Iteraciones

Iteraciones	Orden en que se implementaran las Historias de Usuario.	Tiempo estimado

Iteración 1	HU6. Actualizar consultar perfil de usuario. HU7. Actualizar consultar estadística de un concurso. HU8.Actualizar ver problema.	3 semanas.
Iteración 2	HU1. Consultar últimas entradas. HU2. Publicar una entrada. HU3. Consultar premios de un concurso.	3 semanas.
Iteración 3	HU4. Consultar estadísticas de un usuario en un concurso. HU5. Consultar listado de competencias (próximas o en ejecución).	2 semanas.

2.7.3 Plan de entrega

El plan de entrega debe ser lo más real posible ya que es el documento oficial por el cual los desarrolladores exigirán a los programadores como irían realizando las entregas de la aplicación.

Tabla 12-Plan de entrega

Nombre de la aplicación	Primera entrega	Segunda entrega	Tercera entrega
Actualización y Soporte del CojMail.	Tercera semana de abril	Tercera semana mayo	Tercera semana junio

Tabla 13-Plan de entrega con las HU

Historia de Usuario	Primera entrega	Segunda entrega	Tercera entrega
HU6. Actualizar perfil de usuario.	**x**	**x**	**x**
HU7. Actualizar Consultar estadísticas de un concurso.	**x**	**x**	**x**
HU8. Actualizar ver problema.	**x**	**x**	**x**
HU1. Consultar últimas entradas.		**x**	**x**
HU2. Publicar una entrada.		**x**	**x**
HU3. Consultar premios de un concurso.		**x**	**x**
HU4. Consultar estadística de un usuario en un concurso.			**x**
HU5. Consultar listado de competencias (próximas o en ejecución).			**x**

2.8 Tarjetas CRC (Clase-Responsabilidad-Colaboración)

En estas tarjetas se define un conjunto de clases que son necesarias para la implementación del sistema. Como en la presente solución se trata del soporte y actualización del CojMail, se hace referencia a clases principales que se describieron en la versión anterior, además de algunas que se incluirán en nueva versión.

Más detalle acerca del resto de las tarjetas CRC, consultar el documento de investigación de la solución anterior “Alternativa para el uso del Juez en Línea Caribeño en entornos con dificultades de conectividad” de los autores Nellis Margarita Cabrera Mallea y Michael Horta Fleitas.

Plantilla general de la tarjeta CRC

Tabla 14-Tarjeta CRC

Tarjeta CRC

Nombre de la Clase: Nombre de la clase que se está modelando	
Responsabilidades: • Descripción de las responsabilidades de la clase.	**Colaboradores:** • Relación existente con otras clases para cumplir su responsabilidad.

Tabla 15-Tarjeta CRC Concurso

Tarjeta CRC	
Nombre de la Clase: Concurso	
Responsabilidades: • Definir una entidad que represente un concurso con los atributos: Identificador, Nombre, Fecha de inicio, Fecha fin, Tipo de acceso y Creador.	**Colaboradores:** • Ninguno.

Tabla 16-Tarjeta CRC ComandoListadoConcurso

Tarjeta CRC	
Nombre de la Clase: ComandoListadoConcursos	
Responsabilidades: • Validar los parámetros especificados para obtener un listado de concursos. • Ejecutar la acción de obtener un listado de concursos.	**Colaboradores:** • ComandoNoAutenticado. • ParametosComando. • ParametrosVista. • Concurso.

	• DBFacade.

Tabla 17-Tarjeta CRC Entrada

Tarjeta CRC	
Nombre de la Clase: Entrada	
Responsabilidades: • Definir una entidad que represente una entrada con los siguientes atributos: Identificador, fecha, Texto, usuario.	**Colaboradores:** • Ninguno.

Tabla 18-Tarjeta CRC ComandoUltimasEntradas

Tarjeta CRC	
Nombre de la Clase: ComandoUltimasEntradas	
Responsabilidades: • Validar los parámetros especificados para obtener un listado de las últimas entradas. • Ejecutar la acción de obtener un listado	**Colaboradores:** • ComandoNoAutenticado. • ParametosComando. • ParametrosVista. • Entrada.

de las últimas entradas.	• DBFacade.

2.9 Conclusiones del capítulo

Durante el desarrollo del capítulo se dejó plasmado los artefactos generados por las fases de la metodología XP, se recopiló las Historias de Usuario, se priorizó y se colocó las Iteraciones para la implementación, se generó un Plan de Entregas con la fecha de cada una de las versiones entregadas de la solución y se creó las tarjetas CRC de las clases principales para la implementación del sistema.

Capítulo 3

Capítulo3 Implementación y pruebas

3.1 Introducción

En el presente capítulo se pasará a la implementación de las historias de usuarios definidas en el capítulo anterior, donde se dará cumplimiento a los requisitos funcionales del sistema.

3.2 Tareas de ingeniería

La metodología XP propone en la fase de implementación las tareas de ingeniería donde en estas, se exponen las tareas realizadas para implementar cada historia de usuario. A continuación se muestra una representación de dichas tareas. En los anexos se encuentran el resto de las tareas definidas.

Tabla 19-Tarea de Ingeniería

Tarea de ingeniería	
Número de tarea: Número de la tarea.	**Número de la historia de usuario:** número de la historia de usuario a la que está dirigida la tarea.
Nombre de la tarea: nombre de la tarea a desarrollar.	
Tipo de tarea: Tipo de tarea (Configuración, Desarrollo, otros).	**Estimación:** Tiempo en que se estima para el desarrollo de la tarea.
Fecha de inicio: fecha en que se inicia el desarrollo de la tarea.	**Fecha fin:** fecha en que se da cumplimiento la tarea.
Programador responsable: Autor de la tarea.	
Descripción: Breve descripción de la tarea. **Clases:** Nombre de las clases implementadas.	

Tarea de ingeniería para cada historia de usuario

Primera iteración

Tabla 20-Tarea de ingeniería Actualizar perfil de usuario

Tarea de ingeniería

Número de tarea: 1.	**Número de la historia de usuario:** 6.
Nombre de la tarea: Actualizar consultar perfil de usuario.	
Tipo de tarea: Desarrollo.	**Estimación:** 7dias.
Fecha de inicio: 23 de marzo 2015.	**Fecha fin:** 30 de marzo 2015.
Programador responsable: Yobanni Pérez.	
Descripción: Re-implementación de las clases encargadas para actualizar consultar perfil de usuario. **Clases:** VistaPerfilUsuario, PerfilUsuario, Usuario.	

Tabla 21-Tarea de ingeniería Actualizar consultar estadística de un concurso

Tarea de ingeniería	
Número de tarea: 2.	**Número de la historia de usuario:** 7.
Nombre de la tarea: Actualizar consultar estadística de un concurso.	
Tipo de tarea: Desarrollo	**Estimación:** 7dias.
Fecha de inicio: 1ero de abril 2015.	**Fecha fin:** 8 de abril 2015.
Programador responsable: Yobanni Pérez.	
Descripción: Re-implementación de las clases encargadas para actualizar consultar estadística de un concurso. En dichas clases se obtienen la cantidad de envíos realizados correctamente en el concurso con resultados de ac, ce, sle, ivf, mle, ole, pe, rte, tle, wa y total por cada lenguaje de programación. Con dichos datos se construye una tabla con cada uno de los resultados y otra tabla con los totales de cada envío. **Clases:** VistaEstadísticasConcurso.	

Segunda iteración

Tabla 22-Tarea de ingeniería consultar últimas entradas

Tarea de ingeniería	
Número de tarea: 4.	**Número de la historia de usuario:** 1.

Nombre de la tarea: Consultar últimas entradas	
Tipo de tarea. Desarrollo.	**Estimación:** 7dias.
Fecha de inicio: 17 de abril 2015.	**Fecha fin:** 24 de abril 2015.
Programador responsable: Yobanni Pérez.	
Descripción: Desarrollo de las clases encargadas consultar últimas entradas publicadas en el COJ en forma de lista. En dicha lista se mostrará las entradas publicadas, con los datos siguientes: Usuario, Texto publicado y fecha en que se publicó dicha entradas. **Clases:** ComandoUltimasEntrada, Entrada, VistaUltimasEntradas.	

Tercera iteración

Tabla 23-Tarea de ingeniería Consultar listado de próximas o futuras competencias

Tarea de ingeniería	
Número de tarea: 8.	**Número de la historia de usuario:** 5.
Nombre de la tarea: Consultar listado de competencias.	
Tipo de tarea: Desarrollo.	**Estimación:** 7 días.
Fecha de inicio 20 de mayo del 2015	**Fecha fin: 27 de mayo del 2015.**
Programador responsable: Yobanni Pérez.	
Descripción: Desarrollo de las clases encargadas consultar las competencias que se realizarán en diferentes jueces en línea ya sean próximas o en ejecución, en forma de lista. En dicha lista se mostrará las competencias con los siguientes datos: nombre que tendrá la competencia, URL, fecha de inicio, fecha fin. **Clases:** ComandoListadoCompetencia, WboardContest, Concurso, VistaListadoCompetencia.	

3.3 Pruebas

La metodología XP propone realizarle pruebas al sistema implementado. Estas son de gran importancia debido a que ayudan a detectar y eliminar posteriormente los errores cometidos en la implementación del sistema. XP propone pruebas unitarias y de aceptación.

Pruebas unitarias

Capítulo 3

Al desarrollar un nuevo software o sistema de información, la primera etapa de pruebas a considerar es la etapa de pruebas unitarias o también llamada pruebas de caja blanca (White Box). El objetivo fundamental de las pruebas unitarias es asegurar el correcto funcionamiento de las interfaces, o flujo de datos entre componentes. Por lo general son realizadas por los mismos desarrolladores ya tienen un dominio total del código y se le hace más fácil la tarea de generar datos de pruebas (24).

Por qué realizar pruebas unitarias (25)

- **Asegura la calidad del entregado.** Es la mejor forma detectar errores temporalmente en el desarrollo del software. No obstante, esto no asegura detectar todos los errores por lo que las pruebas de integración y aceptación siguen siendo necesarias.
- **Ayuda a definir los requerimientos y responsabilidades de cada método en cada clase probada.**
- **Permite encontrar errores o bugs temporalmente en el desarrollo.**

Resultado de las pruebas Unitarias

Para la realización de las pruebas unitarias se utilizó el framework para pruebas o testing que trabaja con java TestNG. El mismo ya viene integrado en el IDE Netbeans 8.0. Este es un framework basado en JUnit (para java) pero introduce nuevas funcionalidades que lo hacen más poderoso y fácil de usar tales como:

- Anotaciones JDK 5 (*Annotations*).
- Configuración flexible de pruebas.
- Soporte de pasaje de parámetros.

En la realización de las pruebas utilizando TestNG se generaron clases y métodos para probar cada tarea de implementación. Como resultado de realizar esta prueba se encontró la mayor cantidad de errores en las tareas *Publicar una entrada y Consultar estadística de un usuario en un concurso,* debido a que estas tienen presentan el mayor grado de complejidad.

Pruebas de aceptación

Capítulo 3

El uso de cualquier producto de software tiene que estar justificado por las ventajas que ofrece. Sin embargo, antes de empezar a usarlo es muy difícil determinar si sus ventajas realmente justifican su uso. El mejor instrumento para esta determinación es la llamada prueba de aceptación. En esta prueba se evalúa el grado de calidad del software con relación a todos los aspectos relevantes para que el uso del producto se justifique.

Estas pruebas las realiza el cliente. Son básicamente pruebas funcionales, sobre el sistema completo, estas no se realizan durante el desarrollo, pues sería impresentable al cliente; sino que se realizan sobre el producto terminado e integrado o pudiera ser una versión del producto o una iteración funcional pactada previamente con el cliente. Los casos de pruebas de aceptación restantes se especifican en los anexos.
A continuación se muestra la plantilla para la realización de las pruebas de aceptación (26).

Tabla 24-Pruebas de aceptación

Caso de prueba de aceptación	
Número: número de la historia de usuario.	**Nombre:** nombre de la historia de usuario.
Prioridad: importancia para el cliente (alta, media, baja).	**Complejidad:** nivel de complejidad para el desarrollador (alta, media, baja).
Iteración: iteración a la que corresponde.	**Estimación:** Tiempo que se estima para realizar la actividad.
Descripción: Breve reseña de la historia de usuario, describiendo las acciones del usuario.	
Notas: observaciones de interés.	
No conformidades: Errores encontrados en las pruebas realizadas.	
Resultado esperado: resultado obtenido al realizar la prueba.	

Tabla 25-Prueba de aceptación Actualizar perfil de usuario

Caso de prueba de aceptación	
Número: 6.	**Nombre:** Actualizar perfil de usuario.
Prioridad: alta.	**Complejidad:** media.
Iteración: 1.	**Estimación:** 7 días.

Descripción: El usuario tiene la posibilidad de consultar el perfil de usuario. Puede realizarlo mediante la siguiente vía: • Escribir directamente desde su cliente de correo, un correo al buzón del COJ. En el asunto del correo debe contener el micro lenguaje definido para esta operación.
Notas:
No conformidades: ninguna.
Resultado esperado: El usuario recibe un correo con el perfil del usuario especificado con los datos actualizados.

Tabla 26-Prueba de aceptación Actualizar estadística de un concurso

Caso de prueba de aceptación	
Número: 7.	**Nombre:** Actualizar estadística de un concurso.
Prioridad: Alta.	**Complejidad:** media.
Iteración: 1.	**Estimación:** 7 días.
Descripción: El usuario tiene la posibilidad consultar la estadística de un concurso determinado. Puede realizarlo mediante las siguientes vías: • El usuario selecciona la opción estadística de un concurso especificando el identificador del concurso. • Escribir directamente desde su cliente de correo, un correo al buzón del COJ. En el asunto del correo debe contener el micro lenguaje definido para esta operación.	
Notas:	
No conformidades: ninguna.	
Resultado: El usuario recibe un correo con las estadísticas referente al concurso.	

Tabla 27-Pruebas de aceptación últimas entradas

Caso de prueba de aceptación	
Número: 1.	**Nombre:** Consultar últimas entradas.

Prioridad: alta.	**Complejidad:** media.
Iteración: 2.	**Estimación:** 7 días.
Descripción: El usuario tiene la posibilidad de consultar las últimas entradas publicadas en el COJ. Puede realizarlo mediante la siguientes vía: • Escribir directamente desde su cliente de correo, un correo al buzón del COJ. En el asunto del correo debe contener el micro lenguaje definido para esta operación.	
Notas:	
No conformidades: ninguna.	
Resultado esperado: El usuario recibe un correo con el listado de las últimas 10 entradas publicadas.	

Ilustración 3-Resultado de la prueba de aceptación Consultar últimas entradas.

Resultado de las pruebas de aceptación

Capítulo 3

Las pruebas se realizaron en tres iteraciones correspondiéndose junto con la planificación de las entregas. En la primera iteraciones se encontraron un grupo de no conformidades que fueron erradicadas. Luego en la segunda iteración se encontraron otras no conformidades que igualmente se resolvieron y por último en la tercera iteración no se encontraron no conformidades.

3.4 Conclusiones del capítulo

En este capítulo se continuó con el proceso de desarrollo de software centrándose en la fase de implementación y prueba teniéndose como resultado:

- La implementación de las nuevas funcionalidades las cuales quedaron plasmadas en las tareas de ingeniería.
- Realización de pruebas unitarias y de aceptación para verificar el funcionamiento y calidad del software.

Conclusiones Generales

El presente trabajo se enmarcó en la actualización y soporte a la ¨*Alternativa para el uso del juez en línea caribeño en entornos con dificultades de conectividad*¨, con el objetivo de satisfacer las necesidades que tienen los usuarios en diferentes universidades de acceder a la mayor cantidad de servicios que brinda el COJ.

En el transcurso de la investigación se dio cumplimiento a los objetivos inicialmente propuestos, lo que permitió arribar a las siguientes conclusiones:

- Durante la realización de un estudio al COJ y al CojMail2.0, se pudo evidenciar que el CojMail2.0 no satisfacía las necesidades de los usuarios ya que no soportaba un gran número de servicios que brinda el COJ de gran importancia para los usuarios, además algunas de las funcionalidades que soportaba el CojMail2.0 debían ser actualizadas.
- Al estudiar las metodologías de desarrollo de software se pudo evidenciar que para realizar entregas rápidas en un entorno en que los requisitos pueden cambiar contantemente la más adecuada es la metodología de desarrollo ágil XP.
- Al obtener la aplicación actualizada se le pudo realizar varias pruebas, para comprobar su funcionamiento y aceptación del cliente, siendo estas pruebas unitarias y pruebas de aceptación.

Recomendaciones

- Se recomienda internacionalizar el CojMail donde brinde servicios en múltiples lenguajes, donde a la hora de interpretar el micro-lenguaje las palabras claves pueden en español, inglés portugués entre otros.
- Se recomienda actualizar el cliente de correo CojClient, y así evitarle al usuario tener que memorizar el micro-lenguaje de las nuevas funcionalidades.
- Se recomienda ofrecer el soporte a funcionalidades de administración. Donde un usuario que posee permisos administrativos y se encuentre con problemas de conectividad, lo hagan mediante la vía de correo electrónico
- Se recomienda seguir extendiendo el número de funcionalidades al CojMail.

Bibliografía

Referencias Bibliográficas

1. Caribean Online Judge. [En línea] 2010-2015. http://coj.uci.cu.

2. Labrada, Nersa Doraines Acosta. *Implementación y prueba de un Jurado Online sobre correo electrónico.* La Habana: s.n., 2011.

3. Alegsa.com.ar. [En línea] 1998-2015. http://www.alegsa.com.ar/Dic/aplicacion%20web.php.

4. PCMag. [En línea] 1996-2015. http://www.pcmag.com/encyclopedia/term/41158/desktop-application.

5. Pcmag. [En línea] 1996-2015. http://www.pcmag.com/encyclopedia/term/60015/mobile-app.

6. Michael Horta Fleitas, Nellis Margarita Cabrera Mallea. Alternativa para el uso del Juez en Línea Caribeño en entornos con dificultades de conectividad. *Alternativa para el uso del Juez en Línea Caribeño en entornos con dificultades de conectividad.* La Habana: s.n., 2013.

7. Alonso-Rodríguez, Antonio Jesús Caro. MAN IN THE MIDDLE ATTACKS ON SSL/TLS. 2013.

8. Riola, Jose Carlos Carvajal. Metodologías Ágiles: Herramientas y modelo de desarrollo para aplicaciones java EE como metodología empresarial. Barcelona: s.n., 2008.

9. Ingeniería de Software. [En línea] http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html.

10. *Metodologías Ágiles en el Desarrollo de Software.* José H. Canós, Patricio Letelier, Ma Carmen Penades. DSIC -Universidad Politécnica de Valencia: s.n., 2003.

11. Proyectos Ágiles. [En línea] http://www.proyectosagiles.org/que-es-scrum.

12. Proyectos Ágiles. [En línea] http://www.proyectosagiles.org/requisitos-de-scrum.

13. EcuRed. [En línea] http://www.ecured.cu.

14. Lenguajes de Programación. [En línea] 2009. http://www.lenguajes-de-programacion.com/programacion-java.shtml.

15. Álvarez, Miguel Ángel. Desarrollo Web. [En línea] http://www.desarrolloweb.com/articulos/497.php.

16. PÉREZ, Mª TERESA GARZÓN. SISTEMAS GESTORES DE BASES DE DATOS. 2010.

Bibliografía

17. Martínez, Rafael. PostgreSql-es. [En línea] http://www.postgresql.org.es/sobre_postgresql.

18. PostgreSql. [En línea] http://www.postgresql.org/about/.

19. Laura. Programación Java. [En línea] http://programacion-laura.blogspot.com/2011/08/entorno-de-desarrollo-integrado-ide.html.

20. GENBETA: Desarrollo y Software. [En línea] http://www.genbetadev.com/herramientas/eclipse-ide.

21. GENBETA:dev . [En línea] http://www.genbetadev.com/herramientas/netbeans-1.

22. Apache Tomcat. [En línea] http://tomcat.apache.org.

23. Joskowicz, I. J. Reglas y Prácticas en eXtreme Programming. Universidad de Vigo, España: s.n., 2008.

24. CalidadySoftware. [En línea] http://www.calidadysoftware.com/testing/pruebas_unitarias1.php.

25. Rodríguez, Jorge. Pruebas Unitarias. 2006.

26. PruebasdeSoftware. [En línea] http://pruebasdesoftware.com/pruebadeaceptacion.htm.

Glosario de términos

Glosario de término

Buzón de correo: Espacio virtual en el servidor de correo donde se almacenan los mensajes de correo electrónico de un usuario determinado.

ACM-ICPC: Prestigioso concurso de programación realizado anualmente a diferentes niveles (Local, Nacional, Regional y Mundial) donde solo participan equipos universitarios, buscando obtener buenas posiciones que le clasifiquen hacia el nivel superior. Consiste en intentar resolver un conjunto de problemas en un tiempo determinado (por lo general 4 horas), donde cada equipo obtiene una puntuación según el tiempo en que tardó en solucionar determinado problema; por cada envío incorrecto se penaliza el tiempo total.

Correo: El correo electrónico o email es un mensaje enviado por el usuario al jurado o viceversa, que contiene la información referida a cada una de las peticiones o funcionalidades del sistema.

Daemon: Programa de computadora que se ejecuta en el trasfondo y no bajo el control de un usuario. Normalmente se usan para tareas de mantenimiento constante que no requieren de la presencia de un usuario.

Usuario: Individuo que utiliza el sistema mediante el envío y recepción de correos electrónicos. El público objetivo del mismo se espera esté compuesto principalmente por estudiantes universitarios de Informática.

Entrada: Comentario publicado por un usuario. En este comentario el usuario puede estar pidiendo ayuda referente algún tema o dando respuesta a cualquier interrogante que se publicada por otro usuario, puede ser además alguna sugerencia sobre sobre el sitio.

RFC: Proveniente de *Request for Coments*. Son una serie de notas sobre protocolos de Internet (TCP, IP, SMTP, POP, IMAP, NNTP), donde se describe detalladamente sus características.

Anexos

Historias de usuario

Tabla 28-Anexo#1: HU7 Actualizar perfil de usuario.

Historia de Usuario	
Número: 6	**Nombre:** Actualizar perfil de usuario
Prioridad: Alta	**Complejidad:** Media
Iteración: 1	**Estimación:** 7 días
Descripción: El usuario tiene la posibilidad Consultar el su perfil de usuario. Este puede realizarlo de la siguiente forma. 1. Mediante el cliente de correo convencional de su institución, escribiendo un correo al buzón del COJ. Este en el asunto debe contener el micro lenguaje correspondiente a la petición deseada. El sistema mostrará el perfil del usuario especificado.	
Notas: Para el perfil el usuario debe estar previamente autenticado. Si desea consultar un perfil de un usuario, debe especificar el usuario a quien desee consultar su perfil.	

Tabla 29-Anexo#2: HU8 Actualizar consultar estadísticas de un concurso.

Historia de Usuario	
Número: 7	**Nombre:** Actualizar consultar estadísticas de un concurso
Prioridad: Alta.	**Complejidad:** Media.

Iteración: 1	**Estimación:** 7 días
Descripción: El usuario tiene la posibilidad consultar las estadísticas de un concurso. Este puede realizarlo de las siguientes formas. **1.** Mediante la aplicación CojClient Consultar estadística de un concurso. El usuario debe especificar el identificador del concurso. **2.** Mediante el cliente de correo convencional de su institución, escribiendo un correo al buzón del COJ. Este en el asunto debe contener el micro lenguaje correspondiente a la petición deseada. El sistema mostrará la estadística del concurso especificado.	
Notas:	

Tabla 30-Anexo#3: HU6 Actualizar ver problema.

Historia de Usuario	
Número: 8	**Nombre:** Actualizar ver problema
Prioridad: Alta	**Complejidad:** Media
Iteración: 2	**Estimación:** 7 días
Descripción: El usuario tiene la posibilidad ver un problema. Este puede realizarlo por la siguiente vía. 1. Mediante el CojClient seleccionar la opción ver problema, especificando el id del problema. 2. Mediante el cliente de correo convencional de su institución, escribiendo un correo al buzón del COJ. Este en el asunto debe contener el micro lenguaje correspondiente a la petición deseada.	

El sistema mostrará el problema especificado con todos sus datos.
Notas:

Tarjetas CRC

Tabla 31-Anexo#4: Tarjeta CRC MailBombingController.

Tarjeta CRC	
Nombre de la Clase: MailBombingController	
Responsabilidades: • Identificar si determinada cuenta de correo está realizando ataque mail bombing al sistema. • Bloquear y desbloquear la cuenta de correo de un usuario que previamente fue identificado como atacante.	**Colaboradores:** • Ninguno.

Tabla 32-Anexo#5: Tarjeta CRC ComandoPerfilUsuario.

Tarjeta CRC	
Nombre de la Clase: ComandoPerfilUsuario.	
Responsabilidades: • Validar los parámetros especificados para obtener el perfil de un usuario.	**Colaboradores:** • ComandoNoAutenticado. • ParametosComando. • ParametrosVista.

	• PerfilUsuario. • DBFacade.

Tabla 33-Anexo#6: Tarjeta CRC ComandoListadoCompetencia.

Tarjeta CRC	
Nombre de la Clase: ComandoListadoCompetencia.	
Responsabilidades: • Validar los parámetros especificados para obtener un listado de Competencia. • Ejecutar la petición en la base de datos.	**Colaboradores:** • ParametosComando. • ParametrosVista. • Competencia. • DBFacade.

Tabla 34-Anexo#7: Tarjeta CRC ComandoPremiosConcurso.

Tarjeta CRC	
Nombre de la Clase: ComandoPremiosConcurso.	
Responsabilidades: • Validar los parámetros especificados para obtener un los premios de un concurso.	**Colaboradores:** • ComandoNoAutenticado. • ParametosComando.

• Ejecutar la acción de obtener los premios de un concurso.	• ParametrosVista. • Concurso. • Premio. • DBFacade.

Tabla 35-Anexo#8: Tarjeta CRC ComandoDescripcionProblema.

Tarjeta CRC	
Nombre de la Clase: **ComandoDescripcionProblema**.	
Responsabilidades: • Validar los parámetros especificados para obtener la descripción de un problema. • Ejecutar la acción obtener descripción de un problema.	**Colaboradores:** • ComandoNoAutenticado. • ParametosComando. • ParametrosVista. • Problema. • DBFacade.

Tareas de ingeniería

Tabla 36-Anexo#9: Tarea de ingeniería Consultar premios de un concurso.

Tarea de ingeniería	
Número de tarea: 6	**Número de la historia de usuario:** 3
Nombre de la tarea: Consultar premios de un concurso.	

Tipo de tarea: Desarrollo	**Estimación:** 7dias.
Fecha de inicio: 3 de mayo del 2015	**Fecha fin:10 de mayo del 2015**
Programador responsable: Yobanni Pérez.	
Descripción: implementación de las clases encargadas para actualizar consultar los premios de un concurso. **Clases:** ComandoPremiosConcurso, Premio, Concurso, VistaPremiosConcurso.	

Tabla 37-Anexo#10: Tarea de ingeniería Consultar estadísticas de un usuario en un concurso.

Tarea de ingeniería	
Número de tarea: 7	**Número de la historia de usuario:** 4
Nombre de la tarea: Consultar estadísticas de un usuario en un concurso.	
Tipo de tarea: Desarrollo	**Estimación:** 7dias.
Fecha de inicio: 11 de mayo del 2015	**Fecha fin: 18 de mayo del 2015**
Programador responsable: Yobanni Pérez.	
Descripción: Implementación de las clases encargadas para consultar la estadística de un usuario en un concurso. **Clases:** ComandoEstadisticaUsuarioConcurso, EstadisticasConcursos, Usuario, Concurso, VistaEstadísticasUsuarioConcurso.	

Tabla 38-Anexo#11: Tarea de Ingeniería Publicar una entrada.

Tarea de ingeniería	
Número de tarea: 5	**Número de la historia de usuario:** 2
Nombre de la tarea: Publicar una entrada.	
Tipo de tarea: Desarrollo	**Estimación:** 7dias.
Fecha de inicio: 25 de abril	**Fecha fin:** 2 de mayo
Programador responsable: Yobanni Pérez.	
Descripción: implementación de las clases encargadas para publicar una entrada.	

Clases:

Tabla 39-Anexo#12: Tarea de ingeniería Actualizar ver problema.

Tarea de ingeniería	
Número de tarea: 3.	**Número de la historia de usuario:** 8.
Nombre de la tarea: Actualizar ver problema.	
Tipo de tarea: Desarrollo	**Estimación:** 7dias
Fecha de inicio: 9 de abril del 2015	**Fecha fin:** 16 de abril del 2015
Programador responsable: Yobanni Pérez.	
Descripción: Implementación de las clases encargadas para Ver un problema. **Clases:**	

Pruebas de aceptación

Tabla 40-Anexo#13: Prueba de aceptación Ver problema.

Caso de prueba de aceptación	
Número: 8	**Nombre:** Ver problema
Prioridad: alta	**Complejidad:** media
Iteración: 3	**Estimación:** 7 días
Descripción: El usuario puede consultar la descripción de un problema determinado. Puede realizarlo mediante la siguientes vía: • Escribir directamente desde su cliente de correo, un correo al buzón del COJ. En el asunto del correo debe contener el micro lenguaje definido para esta operación.	
Notas:	
No conformidades: ninguna	
Resultado esperado: El usuario recibe un correo con descripción del problema especificado.	

Anexos

Tabla 41-Anexo#14 Prueba de aceptación Listado de Competencia.

Caso de prueba de aceptación	
Número: 5	**Nombre:** Listado de competencia.
Prioridad: alta	**Complejidad:** media
Iteración: 3	**Estimación:** 7 días
Descripción: El usuario puede consultar el listado de competencia publicado en el COJ. Puede realizarlo mediante la siguientes vía: • Escribir directamente desde su cliente de correo, un correo al buzón del COJ. En el asunto del correo debe contener el micro lenguaje definido para esta operación.	
Notas:	
No conformidades: ninguna	
Resultado esperado: El usuario recibe un correo con el listado de las competencias publicadas en el COJ.	

Tabla 42-Anexo#15 Prueba de aceptación Publicar una entrada

Caso de prueba de aceptación	
Número: 2	**Nombre:** Publicar una entrada.
Prioridad: alta.	**Complejidad:** media
Iteración: 2	**Estimación:** 7 días
Descripción: El usuario puede publicar una entrada en el COJ. Puede realizarlo mediante la siguientes vía: • Escribir directamente desde su cliente de correo, un correo al buzón del COJ. En el asunto del correo debe contener el micro lenguaje definido para esta operación.	
Notas:	
No conformidades: ninguna	
Resultado esperado: El usuario recibe un correo con una confirmación sobre la entrada publicada.	

Tabla 43-Anexo#16 Prueba de aceptación Consultar premios de un concurso.

Caso de prueba de aceptación	
Número: 3	**Nombre:** Consultar premios de un concurso
Prioridad: alta	**Complejidad:** media
Iteración: 2	**Estimación:** 7 días
Descripción: El usuario puede consultar los premios de un concurso. Puede realizarlo mediante la siguientes vía: • Escribir directamente desde su cliente de correo, un correo al buzón del COJ. En el asunto del correo debe contener el micro lenguaje definido para esta operación.	
Notas:	
No conformidades: ninguna	
Resultado esperado: El usuario recibe un correo con los premios del concurso especificado.	

Tabla 44-Anexo#17 Prueba de aceptación Consultar estadística de un usuario en un concurso.

Caso de prueba de aceptación	
Número: 4	**Nombre:** Consultar estadística de un usuario en un concurso.
Prioridad: alta	**Complejidad:** media
Iteración: 3	**Estimación:** 7 días
Descripción: El usuario puede consultar la estadística de un usuario en un concurso. Puede realizarlo mediante la siguientes vía: • Escribir directamente desde su cliente de correo, un correo al buzón del COJ. En el asunto del correo debe contener el micro lenguaje definido para esta operación.	
Notas:	
No conformidades: ninguna	
Resultado esperado: El usuario recibe un correo con la estadística del usuario en el concurso especificado.	

Anexos