Facultad Regional “Mártires de Artemisa”.

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

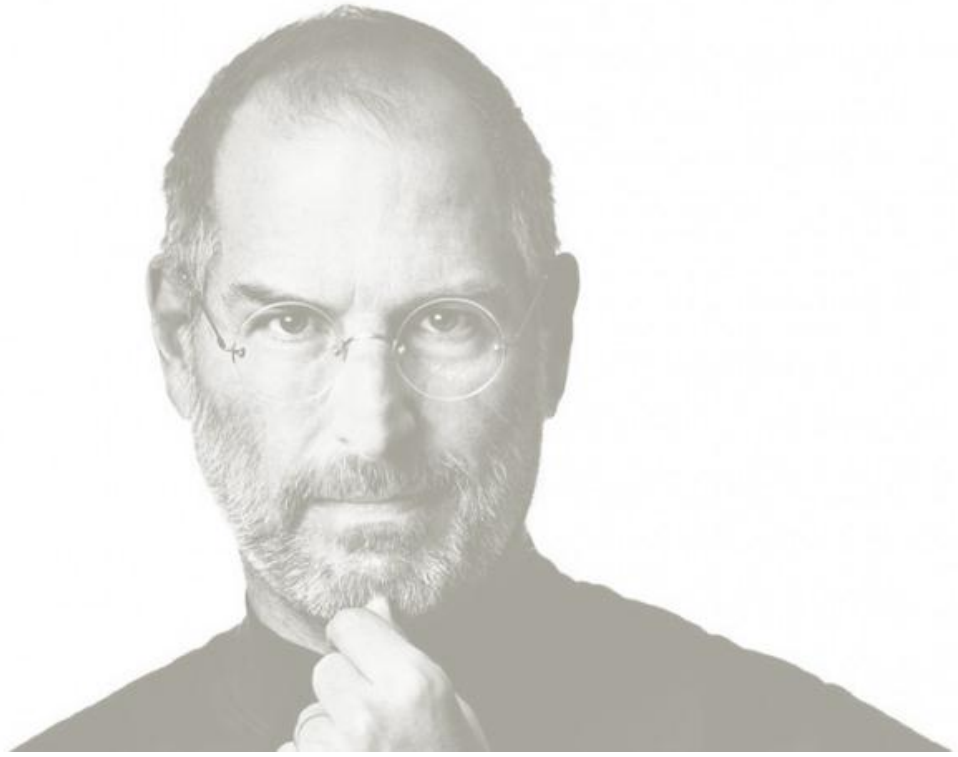
Título: Gestión remota de servidores jWebSocket.

Autor: Lester Alfonso Zaila Viejo.

Tutor: Msc. Yamila Vigil Regalado.

Cotutor: Dr. Raúl René Crespo Heredia.

Artemisa, Cuba, Junio 2012.



Tu tiempo es limitado, de modo que no lo malgastes viviendo la vida de alguien distinto. No quedes atrapado en el dogma, que es vivir como otros piensan que deberías vivir. No dejes que los ruidos de las opiniones de los demás acallen tu propia voz interior. Y, lo que es más importante, ten el coraje para hacer lo que te dicen tu corazón y tu intuición.

Steve Jobs (1955-2011)

Declaración de Autoría

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

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

_____ _____

Lester Alfonso Zaila Viejo **Yamila Vigil Regalado**

Firma del Autor Firma del Tutor

_____

Raúl René Crespo Heredia

Firma del Cotutor

Agradecimientos

A mis padres, a mi familia, a mis suegros y a mi novia. Yaku, tu eres sin dudas parte de este éxito, sin el apoyo de ustedes no hubiese sido posible estar donde estoy hoy. A mis amigos, Dayli, por toda tu paciencia, Ángel, Tilán, Rolando Santamaría, Daimí, Merly, a todo mi proyecto jWebSocket. A los profesores responsables de mi formación. Y a todos aquellos que de alguna manera u otra hicieron posible que mi sueño se hiciera realidad.

Dedicatoria

A mis padres, por ser guías, por todo su apoyo incondicional, por el amor que me han profesado, y su confianza ilimitada.

A mis suegros por acogerme como a un hijo.

A mi novia, por todo, todo, todo su amor y apoyo.

A mi familia en general por estar pendiente de mis resultados.

A mis verdaderos amigos.

Resumen

Internet es la denominada red de redes donde millones de ordenadores están interconectados entre sí alrededor del mundo. Desde su surgimiento tiene un gran impacto, tanto para actividades de ocio como para la gestión del conocimiento a nivel mundial, pues millones de personas tienen acceso fácil e inmediato a una cantidad extensa y diversa de información.

En este siglo de grandes avances tecnológicos la gestión remota de servidores provee el acceso inmediato al servidor que se desea consultar, ganando en tiempo y recursos, ayudando a mantener la eficiencia y seguridad de las empresas informáticas.

La gestión remota de servidores es una actividad indispensable hoy para la garantía de la disponibilidad de todos los servicios y recursos que se encuentran en Internet. Se encuentra difundida hoy desde distintitas tecnologías de comunicación, por ejemplo desde los teléfonos móviles inteligentes de última generación, pues ellos brindan la ventaja de la movilidad y la administración remota en cualquier momento del día.

En esta investigación se presenta el proceso de desarrollo de jWebSocket Watchdog Client su primera versión. Esta es una aplicación que permite la gestión remota de servidores jWebSocket a través de Internet.

Índice

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Introducción

Las redes y sus servicios están creciendo diariamente a una gran velocidad, con la aparición de Internet, la *World Wide Web* (WWW) obtiene un espacio único, donde tecnologías se desarrollan a su alrededor: *Java*[1], *Javascript*[2], *XML*[3] y el protocolo websocket son algunas de las que ponen en alto su constante evolución. Internet tiene un gran impacto, tanto para actividades de ocio como para la gestión del conocimiento a nivel mundial, pues millones de personas tienen acceso fácil e inmediato a una cantidad extensa y diversa de información.

Internet es la denominada red de redes donde millones de ordenadores están interconectados entre sí alrededor del mundo. Al contrario de lo que se piensa comúnmente, Internet no es solo la Web, esta constituye sólo una porción de Internet. Internet remonta sus orígenes en la década de 1960, denominándose *ARPANet*[4] en un inicio y surgió para dar respuesta a la necesidad que tenía la empresa *ARPA*[5] de dar un mejor uso los sistemas de cómputos de ese entonces. En ese momento entre los principales problemas se encontraban en que todos necesitaban tener sus computadoras independientes, de esta manera sin una conexión en red tanto las universidades, laboratorios e investigadores tendrían el mismo contenido duplicado, malgastando esfuerzos y recursos.

El mayor peso de Internet recae en sus potentes servidores. Un servidor dedicado es un equipo físico con propiedades de hardware superiores a los equipos de cómputos de uso doméstico o empresarial, con acceso completo y sin restricciones; que formando parte de una red, brinda servicios a otras computadoras. Las

[1]Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90.

[2]JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, y dinámico.

[3]XML, siglas en inglés de *eXtensible Markup Language* ('lenguaje de marcas extensible'), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C).

[4]La red de computadoras *Advanced Research Projects Agency Network* (ARPANET) fue creada por encargo del Departamento de Defensa de los Estados Unidos ("DOD" por sus siglas en inglés) como medio de comunicación para los diferentes organismos del país.

[5]DARPA acrónimo de la expresión en inglés Defense Advanced Research Projects Agency (Agencia de Investigación de Proyectos Avanzados de Defensa) es una agencia del Departamento de Defensa de Estados Unidos responsable del desarrollo de nuevas tecnologías para uso militar.

propiedades de hardware varían según el tipo de servicio a brindar. Existen distintos tipos de servidores como por ejemplo: Servidores Proxy, Servidores de Acceso Remoto (RAS), Servidores de Base de Datos, Servidores de Aplicaciones, entre otros.

Muchas empresas brindan un servicio de hospedaje de aplicaciones en sus servidores. Dada la cantidad de demanda de usuarios que requieren de tales prestaciones, las empresas que lo brindaban han ido en la búsqueda de nuevas alternativas para intentar brindar un servicio óptimo a la mayor cantidad posible de clientes. Es por esta razón que se idearon las soluciones de hospedaje web y servidores virtuales. Se conoce como servidor virtual privado a cada división dentro de un servidor para habilitar varias máquinas virtuales en él, obteniendo de este modo la potencia, privacidad y seguridad de un servidor.(WAINERMAN, 2002)

En este siglo de grandes avances tecnológicos, donde el uso de las computadoras y sus servicios ha sido generalizado, cada vez un mayor número de empresas e instituciones educativas dependen de la utilización de servidores. Esta creciente expansión de las redes de comunicaciones ha hecho necesaria la adopción de técnicas y el desarrollo de herramientas de seguridad que protejan la integridad de los servicios y permitan controlar remotamente los servidores. Muchos de estos servidores se encuentran ubicados en lugares muy distantes y son administrados de manera remota sin la necesidad de estar físicamente en el mismo local para tener acceso a ellos. Actualmente existen servidores ubicados en el otro extremo del mundo y gracias a la gestión remota de servidores, se hace simple el acceso y la administración del mismo, ganando en tiempo y recursos.

La gestión remota de servidores consiste en un conjunto de técnicas y herramientas, dispuestas a mantener la eficiencia, seguridad y constante monitoreo con una planeación adecuada. La utilización y coordinación de los recursos para planificar, organizar, mantener, supervisar y evaluar la calidad de los servicios, permite administrar remotamente los servidores rápidamente y de forma segura. Además garantiza monitorear los servidores para mantener aplicaciones críticas en línea y corriendo.

La gestión remota de servidores es una actividad indispensable hoy para la garantía de la disponibilidad de todos los servicios y recursos que se encuentran en Internet. Se encuentra difundida hoy desde distintitas tecnologías de comunicación, por ejemplo desde los teléfonos móviles inteligentes de última generación, pues ellos brindan la ventaja de la movilidad y la administración remota desde cualquier momento del día.

Entre las principales dificultades de la gestión remota de servidores en Internet se encuentra los altos niveles de latencia en la comunicación entre el cliente y el servidor a administrar. Por otra parte el uso del protocolo HTTP y técnicas para aumentar la velocidad de respuesta entre cliente y servidor provocan un aumento en el uso de los recursos de la red. Es por ello que aun estas aplicaciones carecen de tiempo real durante la gestión remota de servidores y las que logran algunos niveles de tiempo real lo hacen a grandes costos. Las crecientes exigencias de soluciones y aplicaciones más rápidas, han implicado la introducción del tiempo real en la gestión remota de servidores.

El tiempo real en las aplicaciones para la gestión remota mantiene constantemente informados a los administradores de red y posibilita dar soluciones rápidas a problemas que puedan afectar la integridad de los servicios. Un sistema de tiempo real es un sistema informático que interacciona repetidamente con su entorno físico y responde a los estímulos que recibe del mismo dentro de un plazo de tiempo establecido. El tiempo real es logrado gracias al nuevo protocolo websocket que proporciona un canal de comunicación bidireccional y *full-duplex*[6] sobre un único *socket*[7] TCP (Transmission Control Protocolo, Protocolo de Control de Trasmisiones).

En la actualidad existen varios servidores que soportan este protocolo, *Kaazing Gateway* (Pasarela Kaazing), *django-websocket*, *Jetty*, *GlassFish*, jWebSocket y

---

[6]La transmisión de datos full-duplex significa que los datos pueden ser transmitidos en ambas direcciones sobre una transportadora de señales al mismo tiempo.

[7]Un socket, es un método para la comunicación entre un programa del cliente y un programa del servidor en una red. Un socket se define como el punto final en una conexión. Los sockets se crean y se utilizan con un sistema de peticiones o de *llamadas de función* a veces llamados interfaz de programación de aplicación de sockets.

otros. jWebSocket, es una tecnología orientada al desarrollo de aplicaciones basadas en websocket que proporcionan altos niveles de velocidad, escalabilidad y seguridad. (Framework Approach for WebSockets, 2011). Debido a la necesidad de tener un mayor control del estado de las aplicaciones que se ejecutan en los servidores jWebSocket, es preciso encontrar soluciones y aplicaciones más rápidas que permitan la gestión remota de servidores de este marco de trabajo.

Atendiendo a los elementos anteriores, se describe la siguiente **situación problemática:**

La gestión remota de servidores permite tener un mayor control de las aplicaciones, contribuyendo a la eficiencia y productividad de las empresas informáticas, así como a la alta disponibilidad de sus servicios. El marco de trabajo jWebSocket permite crear aplicaciones es además un servidor web y no permite que su servidor sea monitorizado y gestionado remotamente a través de Internet, sino por una aplicación de terceros que hace el proceso muy complejo, trayendo como consecuencia deficiencias en el control del funcionamiento de los servidores y baja disponibilidad de las aplicaciones desarrolladas bajo este marco de trabajo.

Mantener un correcto funcionamiento de los servidores es la máxima prioridad de los administradores de red. Un mal funcionamiento trae como consecuencias inestabilidad en los servicios tanto para los clientes como para las empresas falta de disponibilidad y pérdidas económicas, transmitiendo falta de credibilidad a las empresas que utilicen el marco de trabajo jWebSocket. En el caso de poseer numerosos servidores el administrador de redes requiere de mucho esfuerzo y tiempo para hacer las revisiones manualmente de las aplicaciones, ya que no reciben un aviso del estado actual. El trabajo del administrador de red se vuelve engorroso y costoso ante esta situación, ya que no conoce el estado actual de las aplicaciones en funcionamiento.

Para darle solución a la situación descrita, se plantea como **problema de investigación**, ¿Cómo garantizar mediante la gestión remota de servidores jWebSocket a través de Internet mayor nivel de disponibilidad en la aplicaciones desarrolladas con este marco de trabajo?

Del problema anterior se puede definir que el **objeto de estudio** de este trabajo de diploma es la gestión remota de servidores a través de Internet. Delimitando su **campo de acción** a la gestión remota de servidores jWebSocket a través de Internet.

Se plantea como **objetivo general** de la investigación, desarrollar una aplicación de escritorio que posibilite la gestión remota de servidores jWebSocket a través de Internet que logre mayor nivel de disponibilidad en las aplicaciones desarrolladas con este marco de trabajo.

Para complementar este objetivo general se definen las siguientes **preguntas científicas**:

- ¿Cuáles son los fundamentos teórico-metodológicos de la gestión remota de servidores jWebSocket a través de Internet que orientan la investigación?
- ¿Cuál es la situación actual de la gestión remota de servidores a través de Internet?
- ¿Cómo desarrollar una aplicación de escritorio que posibilite la gestión remota de servidores jWebSocket a través de Internet que logre mayor nivel de disponibilidad en las aplicaciones desarrolladas con este marco de trabajo?
- ¿Cuáles son los resultados obtenidos al utilizar la aplicación de escritorio desarrollada para garantizar mayor nivel de disponibilidad en las aplicaciones jWebSocket mediante la gestión remota de servidores?

Para dar respuesta las preguntas anteriores se plantean las siguientes **tareas de la investigación**:

- ✓ Fundamentación teórico-metodológica de la gestión remota de servidores jWebSocket a través de Internet.
- ✓ Análisis de la situación actual de la gestión remota de servidores a través de Internet.
- ✓ Desarrollo de una aplicación de escritorio que posibilite la gestión remota de servidores jWebSocket a través de Internet que logre mayor nivel de disponibilidad en las aplicaciones desarrolladas con este marco de trabajo.

- ✓ Validación del funcionamiento de la aplicación de escritorio que garantice mayor nivel de disponibilidad en las aplicaciones jWebSocket mediante la gestión remota de servidores.

Para llevar a cabo esta investigación se utilizan los siguientes **métodos de la investigación**:

Métodos Teóricos:

Histórico-Lógico: Permite analizar la trayectoria completa acerca de las aplicaciones de escritorio que posibilite gestionar remotamente los servidores en tiempo real, así como el estudio histórico de las mismas que permite ver deficiencias y proponer soluciones acordes a las necesidades.

Analítico-Sintético: Mediante este método se va a analizar toda la teoría recopilada a través de los diferentes medios bibliográficos que pueda servir para desarrollar mejor el diseño del sistema, y poder aplicar así estos conocimientos en la práctica de manera que se adquiera una mayor preparación sobre el tema en cuestión.

Modelación: Este método permite realizar una representación de la situación que se analiza. Permite obtener mediante diagramas y objetos una mayor compresión del problema y desarrollar un modelo para la aplicación a desarrollar a partir de la situación problemática.

Análisis Documental: Mediante este método se realiza un estudio de la documentación referente a la gestión remota de servidores o cualquier otra temática que proporcione conocimientos necesarios que pudieran ser incorporados a la investigación.

Método Particular:

Certificación de calidad de software: Este método se utiliza en la liberación que realiza el equipo de calidad de software del Centro de Desarrollo de la Facultad Regional “Mártires de Artemisa”. Mediante este método se certifica la capacidad funcional de la aplicación de escritorio desarrollada y su usabilidad.

Se definen las siguientes **variables de la investigación**:

- **Variable independiente**: Aplicación de escritorio que posibilite la gestión remota de servidores jWebSocket a través de Internet.
- **Variable dependiente**: Disponibilidad en las aplicaciones jWebSocket.

Al concluir el trabajo se contarán con los **posibles resultados** siguientes:

- ✓ Aplicación de escritorio que posibilite la gestión remota de servidores jWebSocket a través de Internet que logre mayor nivel de disponibilidad en las aplicaciones desarrolladas con este marco de trabajo

Para una mejor comprensión de la investigación, el contenido ha sido desglosado en tres capítulos, además de las conclusiones generales, recomendaciones, referencias bibliográficas y bibliografía utilizada, glosario de términos en el cual se detallan los términos técnicos y poco claros utilizados en la elaboración del documento, y los anexos que complementan el trabajo realizado.

Los capítulos han sido estructurados de la siguiente manera:

- ✓ **Capítulo 1. Fundamentación Teórica:** Se realiza la fundamentación teórica de la investigación. Se expone un estudio del estado del arte de las aplicaciones de escritorio que posibiliten gestionar remotamente los servidores en tiempo real.
- ✓ **Capítulo 2. Características, Análisis y Diseño del Sistema:** Brinda una fundamentación de la solución propuesta, a partir de la cual se describen las actividades de análisis de la solución, seguidas por la descripción de los procesos del sistema y de la etapa de diseño.
- ✓ **Capítulo 3. Implementación y Validación del Sistema:** Se describe la etapa de implementación que conlleva a la obtención de la aplicación, y se elaboran y documentan las pruebas realizadas a la solución propuesta para demostrar el cumplimiento de los requerimientos de la misma. Se realiza un análisis de los resultados de la aplicación en un entorno real, comparando indicadores antes y luego de la solución.

CAPÍTULO1. FUNDAMENTACIÓN TEÓRICA

Introducción

El objetivo principal que se persigue con la elaboración de este capítulo es la comprensión de las bases teóricas de la gestión remota de servidores a través de Internet, tratar sus principales conceptos y aspectos más significativos. Además se hace un estudio del estado del arte que muestra la situación actual de la gestión remota de servidores en Internet. Se aborda acerca de la metodología de desarrollo, lenguajes de programación, tecnologías y herramientas utilizadas para el desarrollo de la solución, siempre teniendo en cuenta la necesidad del uso de tecnologías de código abierto.

1.1. Fundamentos teórico-metodológicos.

En la actualidad los administradores de red utilizan la gestión remota de servidores a través de Internet con el objetivo de mantener su correcto funcionamiento. El objeto de estudio y campo de acción están centrados en este tema. Es por esto que la gestión remota, el proceso de realización de la misma, los principales servicios que ofrecen los servidores en Internet y las herramientas denominadas *Watchdog* (perro guardián) constituyen los conceptos que guiaron la presente investigación.

El término gestión remota es usado para describir cualquier proceso en el cual un dispositivo no esté presente físicamente en la unidad actual donde será usado. Hay cuatro aspectos de la gestión remota: el método de comunicación, el nivel de control, capacitación de los operadores y los problemas de rendimiento.

Las aplicaciones de administración remota se pueden encontrar en la construcción, la minería, el transporte y las industrias de tecnología de la información. El número de dispositivos que se pueden administrar de forma remota se ha incrementado debido a la mejora de la calidad de la programación de computadoras y diseño de sistemas. (Conjecture Corporation, 2012).

El proceso de gestión remota consiste en conectarse a otro equipo de cómputo, el cual puede estar dentro de tu propia red local (*LAN*[8]), o en el otro extremo del

[8]Una red de área local, red local o LAN (del inglés *local area network*) es la interconexión de una o varias computadoras y periféricos.

mundo utilizando la red de redes (*WAN*[9]). En muchos casos, el tiempo que lleva a un administrador caminar desde su ordenador hasta el servidor problemático en cuestión, puede ser más que suficiente para resolver el problema. Gestionar los servidores remotamente es lo más apropiado, sobre todo en el caso de servidores alojados en una red WAN. El software de administración remota debería por consiguiente proporcionar a su administrador con los medios para ejecutar las acciones que harían frente al servidor desde la comodidad de su ordenador.(NTRglobal, 2011)

La necesidad de gestionar remotamente los servidores está dada por la cantidad de servicios que se desea mantener bajo estricto control. Antes de dar a conocer cuáles son los principales servicios de Internet primero reparemos en qué consisten los servicios.

Varios expertos han definido a los servicios como:

"… actividades identificables e intangibles que son el objeto principal de una transacción ideada para brindar a los clientes satisfacción de deseos o necesidades".(Stanton, 2004)

"…son actividades, beneficios o satisfacciones que se ofrecen en renta o a la venta, y que son esencialmente intangibles y no dan como resultado la propiedad de algo". (Richard, 2002)

"… es el resultado de la aplicación de esfuerzos humanos o mecánicos a personas u objetos. Los servicios se refieren a un hecho, un desempeño o un esfuerzo que no es posible poseer físicamente".(Lamb, y otros).

Atendiendo a los conceptos de servicios definidos anteriormente para la presente investigación se asume que los servicios son actividades identificables e intangibles, que se ofrecen en renta o venta, formando parte del resultado de la aplicación de esfuerzos humanos o mecánicos a personas u objetos y que no es posible poseer físicamente.

---

[9]Una red de área amplia, con frecuencia denominada WAN, acrónimo de la expresión en idioma inglés *wide area network*.

Los principales servicios brindados por Internet en mayor o menor medida, tienen que ver con las funciones de información, comunicación e interacción. Algunos de los servicios disponibles en Internet aparte de la Web, son el acceso remoto a otros ordenadores (a través de *telnet*[10] o siguiendo el modelo cliente/servidor), la transferencia de ficheros, el correo electrónico, los boletines electrónicos y grupos de noticias, las listas de distribución, los foros de debate y las conversaciones en línea. Los servicios que hoy ofrece Internet no sólo se han multiplicado, sino que han evolucionado hacia nuevas y mejoradas funciones y han ganado en facilidad de uso y rendimiento. A este cambio han contribuido no sólo la velocidad de transferencia de los bits que permiten los *modems*[11] y *routers*[12] actuales y la mayor eficiencia y capacidad de las líneas de telecomunicaciones con un gran ancho de banda, sino también, mejoras en el *software*[13] y las aplicaciones y en el hardware[14] mejorando la capacidad de almacenamiento y memoria, incremento exponencial de la velocidad de los procesadores, capacidad de tratar todo tipo de datos no sólo los textuales, sino también los datos multimedia.

Para lograr mantener un buen control de los servicios, se ha hecho necesario utilizar herramientas que verifiquen el estado actual de los servidores, para mantener la eficiencia de los servicios que se están brindando.

Un tipo de herramienta que es muy utilizada para la gestión remota de servidores son las denominadas *Watchdog*. Este término en español significa perro guardián, representando las funciones principales que realizan estas herramientas para el control y seguimiento de servidores en Internet.

---

[10]Es el nombre de un protocolo de red que sirve para acceder mediante una red a otra máquina para manejarla remotamente como si estuviéramos sentados delante de ella.

[11]Un módem (Modulador Demodulador) es un dispositivo que sirve para enviar una señal llamada moduladora mediante otra señal llamada portadora.

[12]Un router (enrutador) es un dispositivo de interconexión de redes de computadoras Este dispositivo interconecta segmentos de red o redes enteras. Hace pasar paquetes de datos entre redes tomando como base la información de la capa de red.

[13]Se conoce como software al equipamiento lógico o soporte lógico de un sistema informático, comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas

[14]Corresponde a todas las partes tangibles de un sistema informático; sus componentes son: eléctricos, electrónicos, electromecánicos y mecánicos.

Un Watchdog es quien se encarga de monitorizar un aspecto del sistema para llevar a cabo una acción cuando algo falla. En aplicaciones de tiempo real, de manera determinista responde a las fallas o eventos del sistema si es necesario. Si un componente crítico de un sistema falla este evita que ocurran pérdidas de información o daños físicos de hardware. Cuando una conexión de red entre un integrado de registro de datos de aplicaciones y un servidor falla, es posible continuar ejecutando la aplicación en tiempo real y acceder a un disco hasta la conexión de red se restablece.(National Instrument)

Un Watchdog es una identidad de sincronización basado en tecnología Java que inicia, reinicia y detiene cada proceso de Java. El *Watchdog* no supervisa subcomponentes, colas de mensajes, o la sincronización de identidades para la consola de Windows. El *Watchdog* puede iniciar como un demonio en el software Solaris™, *Red Hat Linux*[15], o como un servicio de *Windows*.(Oracle, 2010)

El Watchdog detecta un bloqueo del sistema, una caída o un fallo de aplicación, en caso de que se produzcan. Es un mecanismo de control que vela continuamente por una aplicación de usuario, siempre y cuando el sistema operativo y la aplicación del usuario se están ejecutando. Una alerta puede ser generada por:

- Fallos de las solicitudes del sistema.
- Fallo del sistema operativo.
- Fallo de la aplicación.

(Oracle, 2010)

El *Watchdog* debe garantizar las características en tiempo real del sistema. Determinar la causa de la violación de los requisitos de tiempo real y una detección temprana de fallos de temporización no es fácil debido a las complicadas situaciones que conducen a los errores de sincronización. Una violación de una restricción de tiempo se puede colocar en una de las dos categorías siguientes:

[15]**Red Hat Linux,** una distribución del sistema operativo GNU/Linux que lleva el mismo nombre de la empresa.

1. Un objeto falla como resultado de un recurso solicitado y está bloqueado, ya sea por el propio objeto o algún otro objeto, el tiempo suficiente para violar la restricción de tiempo.
2. Un objeto es muy demandado para su ejecución.

De acuerdo a la categorización, el Watchdog identifica las dos situaciones mediante el control de la tasa de vitalidad.(Application of Software Watchdog as a Dependability Software Service for Automotive Safety Relevant Systems, 2007)

Un Watchdog es un mecanismo de control utilizado en muchos sistemas. Se representa como un temporizador que se activa, y cuando llega a una hora predefinida se dispara y requiere inmediata atención. Las medidas adoptadas por el mecanismo de control dependen de los requisitos del sistema.(Sherif, 2006)

Un Watchdog es un mecanismo de control que toma ciertas acciones a ejecutar. La acción podría ser tan simple como reiniciar el sistema o tan compleja como la de realizar una prueba de diagnóstico del sistema. Su tarea es la de poder configurarse para ajustar cada cuanto tiempo se ejecuta.(Pohronská, y otros, 2010)

Esta investigación asume la definición de *Watchdog* dada por Mária Pohronská y Tibor Krajcovic.

1.2 Análisis de Soluciones Existentes

En el mundo las herramientas de gestión remota de servidores han tomado una importancia significativa por las ventajas que proveen al usuario, numerosas empresas de software continúan desarrollando este tipo de aplicaciones. Estas herramientas han sido utilizadas desde el surgimiento Internet, permitiendo la administración de los primeros dispositivos que se conectaron a la red de redes, aunque en ese entonces las rústicas consolas de administración era lo más avanzado que existía. Con el avance inminente de la tecnología y el desarrollo de Internet se ha hecho necesario mejorar las herramientas de gestión de remota, dotando al usuario de una amena y sencilla interfaz gráfica, con alto rendimiento y con baja latencia en la red que les posibilite administrar los servidores.

Varias aplicaciones *Watchdog* han surgido, permitiendo al usuario la gestión de sus servidores de forma rápida y efectiva, resultando de gran utilidad a la hora de

brindar servicios y evitar recorrer largas distancias hasta el sitio físico donde se encuentran alojados. Existen varias herramientas dedicadas a la gestión remota de servidores. Una de las más eficaces es JMX (Java Management Extensions, Administración de Extensiones de Java) para la gestión y el control de los recursos, tales como aplicaciones, dispositivos, servicios, y la máquina virtual de Java. Los principales objetivos de JMX es permitir que una amplia gama de aplicaciones y sistemas de gestión sean capaces de acceder y controlar las aplicaciones administradas, pero no hace función de *Watchdog*.

Otra de las soluciones existentes para aplicaciones web, es *Drupal Watchdog*, que está dirigida a funciones específicas como supervisar el sistema y registrar eventos del sistema en un registro que puede examinar. Esto le ayuda a obtener una visión rápida de la actividad en su sitio. Dado que los registros eventos en secuencia, sino que también puede ser útil para la depuración de errores del sitio. Los registros de vigilancia de registro de errores, las advertencias, los datos de uso, datos de rendimiento y funcionamiento. Permite filtrar las entradas ("Filtrar por tipo de mensaje") para ver sólo un tipo particular de mensaje. Usted debe revisar el informe de vigilancia regular para asegurarse de que su sitio está funcionando correctamente. Una desafortunada limitación de *Drupal Watchdog* es que si quieres usarlo en aplicaciones personalizadas que no utilizan todos los módulos de Drupal este falla. No soporta el uso del protocolo websocket.

De las aplicaciones de escritorio existen varias soluciones, una de ellas es *ENCO Watchdog* que utiliza un pulso entre la aplicación de destino y él mismo. Si la aplicación de destino no puede "responder" a los pulsos, *ENCO Watchdog* intentará reiniciar el programa. Un ejemplo sería si el programa bajo supervisión del *Watchdog* retornara "No responde".

Pero *ENCO Watchdog* no genera notificaciones a los administradores del sistema, no es multiplataforma, solo corre en *Windows*, no es capaz de supervisar una aplicación en otra estación de trabajo, y no tiene soporte para el protocolo websocket.

Otra de las soluciones existentes de aplicaciones de escritorio, pero esta vez implementada en el lenguaje *Java* es *Super Watchdog* es una herramienta

programadora de tareas con la interfaz gráfica. *Super Watchdog* es una mejor alternativa para los programadores, como para la automatización de la carga de trabajo. Trabaja para la red homogénea o heterogénea, en la medida que Java está disponible. *Super Watchdog* puede ser usado en cualquier máquina en la red en cualquier momento. Sus tareas pueden ser controladas programáticamente, y a pesar de ser una excelente solución multiplataforma, no soporta el uso del protocolo websocket.

Los *Watchdog* se han hecho muy populares, no solo se encuentran en la web y en las aplicaciones de escritorio, sino que también se pueden encontrar en los teléfonos inteligentes. Aplicaciones como *Watchdog Task Manager*, *3G Watchdog, Permission Watchdog* son algunas con las que cuentan hoy en día los teléfonos inteligentes. Todas con funcionalidades prometedoras pero ninguna utiliza el protocolo *websocket*.

Actualmente son varias las aplicaciones que monitorean servidores, pero jWebSocket no cuenta con una herramienta destinada con ese fin, y las existentes no son compatibles por razones tales como: no soportan el protocolo websocket para monitorear aplicaciones en tiempo real, no se ejecutan desde un equipo de cómputo remoto sino del mismo equipo y en el caso de que el equipo deje de funcionar no tendría sentido utilizarlo, por lo tanto se hace necesaria una investigación que logre una aplicación de escritorio para gestionar servidores jWebSocket a través de Internet.

1.3. Metodología usada para el desarrollo de la solución

En una aplicación informática, el proceso de desarrollo debe estar regido y orientado por una metodología de desarrollo de software, que guíe los procesos y permita tener un registro detallado del avance de la investigación. Las metodologías pueden ser robustas o ágiles. Las metodologías robustas o pesadas están concebidas para guiar el proceso de desarrollo de los software de gran envergadura, cuando un proyecto requiere de gran cantidad de documentación, vaya a ser realizado en un tiempo considerablemente largo y existe la posibilidad de que pase por las manos de varios equipos de trabajo. Por su parte, las metodologías ágiles intentan evitar los tortuosos y burocráticos caminos de las

metodologías tradicionales enfocándose en los clientes y los resultados. Se basan en promover iteraciones en el desarrollo a lo largo de todo el ciclo de vida del proyecto, logrando que se minimicen los riesgos desarrollando software en corto tiempo.(Metodologías Tradicionales Vs. Metodologías Ágiles, 2011)

Esta investigación estudiará las metodologías ágiles y robustas, para decidir cuál es la más adecuada para ser utilizada.

Proceso Unificado de Desarrollo - RUP

RUP es un proceso formal que provee un acercamiento disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que satisfaga los requerimientos de los usuarios finales, respetando cronograma y presupuesto. Puede ser adaptado y extendido para satisfacer las necesidades de la organización que lo adopte. Es guiado por casos de uso y centrado en la arquitectura, iterativo e incremental y utiliza UML como lenguaje de notación.(Figueroa, y otros, 2011)

SCRUM

Es un proceso ágil y liviano que sirve para administrar y controlar el desarrollo de software. El desarrollo se realiza de forma iterativa e incremental (una iteración es un ciclo corto de construcción repetitivo). Cada ciclo o iteración termina con una pieza de software ejecutable que incorpora nuevas funcionalidades. SCRUM se enfoca en priorizar el trabajo en función del valor que tenga para el negocio, maximizando la utilidad de lo que se construye y el retorno de inversión.(Schwaber, y otros, 2011)

XP

XP es una metodología ligera de desarrollo de software que se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado. Esta consiste en una programación rápida o extrema y es utilizada para proyectos de corto plazo, con requisitos imprecisos y muy cambiantes, donde existe un alto riesgo técnico.(Wells, 2009)

SXP

SXP está compuesta por las metodologías SCRUM y XP, ofreciendo una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva. Esta metodología fomenta el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo.

SXP consta de 4 fases principales:

- Planificación-Definición: se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.
- Desarrollo: se realiza la implementación del sistema hasta que esté listo para ser entregado.
- Entrega: es la puesta en marcha.
- Mantenimiento: se realiza el soporte para el cliente.

De cada una de estas fases se realizan numerosas actividades tales como el levantamiento de requisitos, la priorización de la Lista de Reserva del Producto, definición de las Historias de Usuario, Diseño, Implementación, Pruebas, entre otras; de donde se generan artefactos para documentar todo el proceso. Las entregas son frecuentes, y existe una refactorización continua, lo que permite mejorar el diseño cada vez que se le añada una nueva funcionalidad.

SXP está especialmente indicada para proyectos con pequeños equipos de trabajo, un constante cambio de requisitos o requisitos imprecisos, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Fomenta el trabajo en equipo, con un objetivo claro, permitiendo el seguimiento y control de las tareas a realizar.(SXP, Metodología Ágil para el Desarrollo de Software, 2010)

Fundamentos de la Selección

Debido a las grandes ventajas que proporcionan las metodologías ágiles, además de que el proyecto está formado por un equipo de trabajo pequeño, el cliente no

tiene bien definido algunos requisitos y el desarrollo está orientado a una entrega rápida de resultados, se propone para esta investigación el uso de la metodología SXP.

1.4. Herramientas y Tecnologías usadas para el desarrollo de la solución

1.4.1. Marcos de trabajo

Un Marco de Trabajo es un esquema, esqueleto o patrón para el desarrollo y/o la implementación de una aplicación. En otras palabras, un marco de trabajo se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta, puede incluir soporte de programas, bibliotecas y un lenguaje de scripting entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Los marcos de trabajo son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de *software* que tratando con los tediosos detalles de bajo nivel para proveer un sistema funcional. (¿Qué es un 'framework'?, 2006)

1.5 Marco de Trabajo del lado del Servidor

Marco de Trabajo jWebSocket

jWebSocket es un marco de trabajo de código abierto para el desarrollo de aplicaciones web estacionarias y móviles basado en Java en el lado del servidor y en JavaScript del lado del cliente. jWebSocket establece un modelo de token. Los tokens son datos abstractos que a través de una estructura jerárquica y una API proporcionan métodos de acceso a los contenidos. Con el objetivo de realizar una abstracción en la manipulación de los diferentes formatos, el marco de trabajo convierte los paquetes de datos entrantes y salientes en *tokens*. El cliente nativo soporta el intercambio de paquetes en los formatos JSON, XML y CSV, que en entornos específicos se pueden utilizar sin la necesidad de manejarlos a través de *tokens*. El cliente jWebSocket tiene una arquitectura de plug-in que permite aumentar con facilidad sus funcionalidades.

El servidor jWebSocket está diseñado para funcionar como servidor de comunicaciones o como servidor web, brindando total flexibilidad.

jWebSocket como servidor web proporciona un conjunto importante de funcionalidades y su arquitectura extensible mediante plug-in permite añadir fácilmente características adicionales a un sistema independiente. Por otra parte los administradores pueden configurar el servidor exactamente como sea necesario y dejar a un lado todos los módulos que no necesiten. Estas características muestran la fortaleza y flexibilidad del marco de trabajo para el desarrollo de aplicaciones web estacionarias y móviles, multiplataforma, multisectorial y compatible con todos los navegadores.

(Frame Approach for WebSockets, 2011)

jWebSocket es un proyecto del cual un equipo de estudiantes de la Facultad Regional de la UCI “Mártires de Artemisa”, constituye alrededor de un 50% de los integrantes del grupo de desarrollo de este marco de trabajo. La presente investigación es parte del proyecto jWebSocket, precisamente por esta razón la aplicación a desarrollar se lleva a cabo sobre este marco de trabajo.

1.5.1 Lenguajes de Programación y Modelado

Lenguaje Modelado Unificado

Lenguaje Modelado Unificado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. Es importante resaltar que UML es un lenguaje para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. Se decide utilizar UML como lenguaje de modelado en su versión 2.0. (Rumbaugh, y otros, 2007)

Java

Java es un lenguaje moderno, de alto nivel, que recoge los elementos de programación que típicamente se encuentran en todos los lenguajes de programación, permitiendo la realización de programas profesionales. La tecnología Java está compuesta básicamente por 2 elementos: el lenguaje Java y su plataforma. Con plataforma se refiere a la máquina virtual de Java *(*Java *Virtual Machine).* Actualmente este lenguaje es el más utilizado para el desarrollo de aplicaciones por las ventajas que brindan sus características, las cuales serán expuestas a continuación.

- Interpretado y compilado a la vez: genera ficheros de clases compiladas, pero estas clases compiladas, son en realidad interpretadas por la máquina virtual de java. Siendo la máquina virtual de java la que mantiene el control sobre las clases que se estén ejecutando.
- Multiplataforma: El mismo código java que funciona en un sistema operativo, funcionará en cualquier otro sistema operativo que tenga instalada la máquina virtual java. Esta es un de las principales características que favorece el crecimiento y difusión del lenguaje.
- Seguro: La máquina virtual, al ejecutar el código java, realiza comprobaciones de seguridad, además el propio lenguaje carece de características inseguras, como por ejemplo los punteros.
- Distribuido: Java proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets, establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas.
- Orientado a objetos: Java fue diseñado como un lenguaje orientado a objetos desde su creación. Los objetos agrupan en estructuras encapsuladas tanto sus datos como los métodos que manipulan esos datos. Esto le permite crear programas modulares y código reutilizable.
- Robusto: Exhaustivo de la fiabilidad. Java pone mucho énfasis en el control temprano de posibles errores, como compiladores de Java son capaces de

detectar muchos problemas que en primer lugar aparecen durante el tiempo de ejecución en otros idiomas.

- Multiproceso: Hoy día ya se ven como terriblemente limitadas las aplicaciones que sólo pueden ejecutar una acción a la vez. Java soporta sincronización de múltiples hilos de ejecución (multithreading) a nivel de lenguaje, especialmente útiles en la creación de aplicaciones de red distribuidas. Así, mientras un hilo se encarga de la comunicación, otro puede interactuar con el usuario mientras otro presenta una animación en pantalla y otro realiza cálculos.

A pesar de que se pueden utilizar otros lenguajes de programación como C#, PHP o Phyton, se decidió el uso de Java porque era más fácil de integrar, además jWebSocket es una tecnología escrita totalmente en ese lenguaje de programación.

1.5.2 Herramienta CASE

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) propician un conjunto de métodos y técnicas automatizadas que brindan ayuda y dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todo el ciclo de vida del desarrollo de un software, reduciendo el esfuerzo, el costo y el tiempo.

Rational Rose

Rational Rose es una herramienta CASE que da soporte al modelado visual con UML cubriendo todo el ciclo de vida de un proyecto. Es compatible con la metodología RUP que permite crear los diagramas que se van generando durante el proceso de ingeniería en el desarrollo del software. Se enmarca dentro del desarrollo de modelado para fines académicos, investigativos y comerciales. (Herramientas Case, 2011)

Visual Paradigm

Visual Paradigm es una poderosa herramienta CASE que hace uso del lenguaje modelado unificado (UML) con soporte multiplataforma, que proporciona un ciclo de vida completo del desarrollo de software, excelentes facilidades de interoperabilidad con otras aplicaciones, compatibilidad entre versiones, así como

dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Dicha herramienta brinda una serie de facilidades que se mencionan a continuación:

- Soporta un conjunto de estándares entre los que se encuentran UML, SysML, BPMN, XML y XMI.
- Soporte de modelado UML, modelado de procesos de negocios y un generador de mapeo de objetos-relacionales para el lenguaje de programación Java.
- Integración con la herramienta NetBeans IDE.
- Permite la generación de código y la ingeniería inversa para un conjunto de lenguajes entre los que se encuentran Java.
- Ofrece herramientas para la generación de reportes en formatos HTML, pdf y doc.
- Generación de documentación: brinda la posibilidad de documentar todo el trabajo sin necesidad de utilizar herramientas externas.
- Interoperabilidad e integración. Permite la integración con un conjunto de herramientas (Visio drawing, Rational Rose, ERwin Data Modeler Project, Microsoft Excel y Microsoft Word document) e intercambiar diagramas UML y modelos usando representaciones industriales comunes.
- Modelado de base de datos. Proporciona una mayor documentación de la base de datos y diagramas de mapeo de relación de objetos.
- Integración con herramientas para el control de versiones.
- Diseño de prototipo de Interfaz de Usuario. Permite insertar información adicional a los diagramas mediante notas y comentarios para describir sus elementos lo que facilita la revisión de los prototipos así como el trabajo en equipo.

(Visual Paradigm International Ltd, 2011)

Fundamentos de la Selección

Anteriormente se analizaron algunas de las herramientas CASE existentes en el mundo para el modelado de software, y teniendo en cuenta que se desea desarrollar un trabajo bajo las políticas de software libre, se selecciona como

herramienta CASE a Visual Paradigm. Esta herramienta a pesar de no ser libre, cuenta con una licencia comercial la cual posee la Universidad de las Ciencias Informáticas (UCI).

1.5.3 Herramientas de Control de Versiones

Para el desarrollo de un proyecto de software de esta envergadura se hace indispensable la utilización de una herramienta para el control de versiones debido a las necesidades de controlar los cambios realizados al código fuente.

Un sistema de control de versiones es un sistema de gestión de archivos y directorios, cuya principal característica es mantener el historial de cambios y modificaciones que se han realizado sobre dichos archivos a lo largo del tiempo. Es importante decir que estos sistemas no solo se limitan a gestionar archivos de texto sino que también gestionan documentos, imágenes y ficheros de todo tipo. A continuación se hace una caracterización de algunos sistemas de control de versiones con el objetivo de determinar el más idóneo para garantizar mayor seguridad y disponibilidad de los datos.

Git

Git es un sistema de control de versiones, diseñado para el trabajo con proyectos de cualquier tamaño con gran rapidez y eficiencia. Tiene una forma diferente y revolucionaria en su sistema de guardado haciendo que cada copia de trabajo sea un repositorio en sí mismo y contenga todo el historial de modificaciones. Esta es una característica que lo hace muy eficiente porque se puede disponer de toda la información necesaria para trabajar sin conexión y sincronizar los cambios una vez restablecida la conexión.

Subversion

Subversion es un sistema de control de versiones completamente equipado que fue originalmente diseñado para reemplazar a CVS. Desde entonces se ha expandido más allá de su objetivo original, pero su modelo básico, el diseño y la interfaz fueron fuertemente influenciados por CVS por lo que debido a estas particularidades los usuarios de CVS se sienten muy cómodos al interactuar con Subversion. (Apache Software Foundation, 2011)

Este sistema presenta varias características importantes. Los directorios son versionados. La resolución de conflictos es de forma interactiva. Gestiona de manera eficaz los archivos binarios. Bloquea los archivos. Vincula los lenguajes de programación. Vincula varios repositorios. Posee soporte para desarrolladores y desarrollo paralelo.

Fundamentos de la Selección

Teniendo en cuenta las características expuestas, se selecciona Subversion como herramienta de control de versiones para ser utilizado en el desarrollo de la aplicación, ya que permite la integración con NetBeans y la posibilidad de desarrollar en paralelo. También influyó en la decisión, la experiencia de trabajo por parte del equipo de desarrollo con esta herramienta y su fácil uso. Además también se tiene en cuenta que la Universidad de las Ciencias Informáticas hace uso de esta herramienta.

1.5.4 Cliente de Control de Versiones

Los clientes permiten la gestión de cambios que se realizan sobre los elementos de algún producto. Permitiendo una conexión entre él como cliente y el servidor, para un mejor manejo de los archivos locales.

TortoiseSVN

Es un cliente gratuito de código abierto para el sistema de control de versiones Subversion. TortoiseSVN maneja ficheros y directorios a lo largo del tiempo. Los ficheros se almacenan en un repositorio central. El repositorio es prácticamente lo mismo que un servidor de ficheros ordinario, salvo que recuerda todos los cambios que se hayan hecho a sus ficheros y directorios. Esto permite que pueda recuperar versiones antiguas de sus ficheros y examinar la historia de cuándo y cómo cambiaron sus datos, y quién hizo el cambio. (team, 2011)

RapidSVN

Es una plataforma de interfaz gráfica de usuario, para el sistema de revisión de Subversion. Este proyecto también incluye un cliente de Subversion C++. RapidSVN está licenciado bajo la v3 de GNU General Public License.

Utiliza las mejores características de los clientes de otras arquitecturas de control de versiones. Si bien es bastante fácil para los nuevos usuarios de Subversion trabajar con él, también debe ser lo suficientemente potente como para que los usuarios con experiencia sean aún más productivos.(RapidSVN, 2011)

Características:

- Simple: Proporciona una interfaz fácil de usar para las características de Subversion.
- Eficiente: Simple para los principiantes pero lo suficientemente flexible como para aumentar la productividad para los usuarios de Subversion.
- Portátil: Se ejecuta en cualquier plataforma: Linux, Windows, Mac OS / X, Solaris.
- Rápido: Completamente escrito en C++.

Fundamentos de la Selección

Se analizaron algunas de las herramientas, como por ejemplo Tortoise y Rapidsvn, a pesar de todas las ventajas y funcionalidades que ofrece Tortoise, se decide usar Rapidsvn ya que ésta es libre, y fácil de usar, tanto por quienes ya conocen Subversion como para quienes empiezan, pudiendo acceder a direcciones SVN, subir y descargar contenido, sincronizarlo con el servidor original, comprobar su estado, crear y fusionar direcciones.

1.5.5 Entorno integrado de desarrollo - IDE

Un IDE (acrónimo en inglés de Integrated Development Environment) es un entorno de programación que integra varias herramientas con el objetivo de facilitar el desarrollo de software sobre uno o varios lenguajes de programación. La mayoría de los IDEs cuentan con herramientas tales como: editor de código, herramientas para el rastreo de código, compilador, depurador y constructor de interfaz gráfica (Nourie, 2005). A continuación se hace una caracterización de varios IDEs que permitirá adquirir los elementos necesarios para determinar cuál es el más idóneo para el desarrollo de solución propuesta.

Eclipse

Eclipse es un entorno integrado de desarrollo de código abierto y multiplataforma. En un principio Eclipse fue desarrollado por IBM y posteriormente su desarrollo fue llevado a cabo por Eclipse Fundation, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios. Eclipse basa su funcionalidad en módulos (en inglés plug-in) que se adaptan a las necesidades del programador. Este mecanismo de módulos es una plataforma ligera para componentes de software que permite el uso de diferentes lenguajes de programación como son Java, C/C++ y Python (Eclipse, 2011).

NetBeans

NetBeans IDE es una herramienta desarrollada por Sun Microsystems. Está completamente escrito en Java, por lo que puede ser utilizado desde cualquier sistema operativo compatible con la máquina virtual de Java. Permite el desarrollo de aplicaciones de escritorio, web y móviles. Brinda soporte a varios lenguajes de programación como Java, PHP, C/C++, Groovy, Python, JavaScript, entre otros. Es un producto libre y gratuito sin restricciones de uso. Este entorno integrado de desarrollo es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Su misión consiste en evitar tareas repetitivas, facilitar la escritura correcta de código, disminuir el tiempo de depuración e incrementar la productividad del desarrollador. Cuenta con un depurador, perfilador de integración, herramientas para refactorizaciones, completamiento de código y control de versiones de archivos.(NetBeans, 2011)

Fundamentos de la Selección

Las características anteriormente expuestas sobre NetBeans IDE y Eclipse SDK demuestran las potencialidades de ambos para el desarrollo de aplicaciones Java. Sin embargo, para el desarrollo de la solución propuesta se selecciona como entorno integrado de desarrollo a NetBeans IDE debido a que se tiene una mayor experiencia y familiarización con esta herramienta. Además su versión 7.0.1 introduce un soporte para el desarrollo con la especificación JavaSE7 (Java Standard Edition) con las características de JDK7 (Java Development Kit).

1.5.6 Sistema Gestor de Base de Datos

Los sistemas de gestión de bases de datos son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Su propósito es manejar de forma clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante para una organización.

SimpleDB

Tiene algunas limitaciones. La primera es que una consulta no puede durar más de 5 segundos. La segunda, que no hay más tipos de datos que cadenas de texto. Cada almacenamiento, recuperación o comparación se realiza en formato de cadena de texto (string), por lo que las comparaciones de fechas no funcionan a menos que se éstas se conviertan al formato ISO8601. La tercera, el tamaño máximo de cada cadena de texto es de 1024 caracteres, lo cual limita mucho los textos (como descripciones de productos, etc.) que puedes almacenar un atributo simple. Pero debido a que el esquema es dinámico y flexible, puedes superar esta limitación añadiendo atributos “DescripcionProducto1″, “DescripcionProducto2″, etc. El límite por cada elemento es de 256 atributos. Mientras SimpleDB sea Beta, los dominios no pueden ser más grandes de 10GB, y las bases de datos enteras no pueden superar 1TB.

CouchDB

CouchDB es una base de datos orientada a documentos, open source y gratuita. Derivada del almacenamiento clave/valor, utiliza JSON para definir el esquema de un elemento. CouchDB está concebida para tender un puente al hueco que existe entre bases de datos relacionales y bases de datos orientadas a documentos gracias a las “vistas”, que pueden ser creadas dinámicamente a través de JavaScript. Estas vistas mapean los datos del documento en estructuras similares a tablas que pueden ser indexadas y consultadas.

Cassandra

Este sistema de base de datos fue desarrollado por Facebook, abandonando el prestigioso MySQL. Este sistema ya forma parte de la incubadora de proyectos de

Apache. Cassandra es open source, altamente distribuido, y tiene un modelo de datos similar al de BigTable (Google). Entre sus características cabe destacar la alta disponibilidad, consistencia eventual, escalabilidad incremental, replicación optimista, administración mínima. Posee API para acceder desde lenguajes de programación tales como C++, C#, Java, Perl o PHP (entre otros).

MongoDB

Mongo es el sistema de base de datos desarrollada en 10gen por Geir Magnusson y Dwight Merriman. Mongo es una base de datos orientada a documentos JSON, salvo que está diseñada para ser una verdadera base de datos de objetos, más que para un almacenamiento de clave/valor puro. Originalmente, 10gen enfocó poner juntos una pila completa de servicios web, aunque sin embargo, más recientemente ha tenido que ser reenfocado principalmente en la base de datos Mongo.(10gen, Inc., 2011)

MongoDB es escalable, de alto desempeño, de código abierto, base de datos orientada a documentos. Escrito en C++ ofrece las siguientes características:

- Almacenamiento orientado a documentos: Documentos estilo JSON con esquemas dinámicos ofrecen simplicidad y poder.
- Soporte Full index: Índices sobre cualquier atributo, tal y como estamos acostumbrados.
- Replicación y alta disponibilidad: Espejos entre LANs y WANs
- Auto-Sharding: Escalabilidad horizontal sin comprometer la funcionalidad.
- Consultas: Ricas y basadas en documentos
- Rápidas actualizaciones en el contexto.
- Mapeo y reducción: Agregación flexible y procesamiento de datos.
- GridFS: Almacena archivos de cualquier tamaño sin complicar tu “stack”.
- Soporte comercial: Soporte comercial, capacitación y consultoría disponibles.

(10gen, Inc., 2011)

Fundamentos de la Selección

Basándose en las características expuestas anteriormente se hace uso de la base de datos documental MongoDB. Esta supera en gran medida a las restantes bases de datos ya que es el enlace perfecto entre el almacenamiento clave/valor (que son rápidos y altamente veloces) que proporcionan consultas ricas y una profunda funcionalidad.

Conclusiones del capítulo

En el presente capítulo, luego de realizar una profunda investigación, se expusieron las definiciones más significativas relacionadas con las principales temáticas abordadas en diferentes fuentes bibliográficas acerca del estado del arte que presenta el tema que se investiga. Se definió la metodología que se va a poner en práctica, así como los lenguajes de programación y tecnologías a utilizar, se seleccionaron las herramientas acordes a este tipo de proyecto para el desarrollo de la solución.

CAPÍTULO 2. CARACTERÍSTICAS, ANÁLISIS Y DISEÑO DEL SISTEMA

Introducción

En este capítulo se describe cómo debe funcionar la aplicación de escritorio para la gestión remota de servidores jWebSocket, dándose una propuesta de solución, mostrando sus principales características. Se realiza el modelado del dominio con el objetivo de comprender su contexto. Se detallan brevemente los artefactos de la metodología de desarrollo SXP, propuesta en el capítulo anterior. Describiéndose los requisitos funcionales y no funcionales de la aplicación, las historias de usuario y las tareas de ingeniería asociadas a las mismas.

2.1. Propuesta de Solución

jWebSocket Watchdog Client es una aplicación de escritorio que permite monitorizar al servidor jWebSocket para llevar a cabo una acción cuando este falla. Es un mecanismo de control que toma ciertas acciones a ejecutar como por ejemplo: notificar a los administradores del sistema vía e-mail o SMS. Permitiéndoles saber el estado del servidor en el instante que este falla, contribuyendo a la eficiencia y la productividad de las empresas informáticas así como a la alta disponibilidad de las aplicaciones desarrolladas con el marco de trabajo jWebSocket.

Esta solución es dirigida al marco de trabajo jWebSocket por tanto es diferente a las demás herramientas de gestión remota de servidores existentes, porque utiliza el protocolo websocket, para realizarle las pruebas al servidor. Este protocolo proporciona que se logre mantener el control en tiempo real del estado de los servidores jWebSocket a altos niveles de velocidad y escalabilidad.

jWebSocket Watchdog Client puede ser utilizado principalmente en los nodos donde se requiera tener el control de las aplicaciones que se están ejecutando sobre el servidor jWebSocket. Es una aplicación muy útil si se tiene varios servidores jWebSocket ejecutándose, pues evitaría pérdidas de tiempo a los administradores del sistema que tendrían que leer manualmente los log de todos los servidores.

2.2. Planificación del Proyecto por Roles

Un Proyecto es el conjunto de actividades intelectuales, básicamente estructuradas y ordenadas, que establece (mediante descripciones y prescripciones) lo que hay que hacer y cómo hacerlo para resolver un problema complejo, descomponible en subproblemas relacionados entre sí. El proyecto, además, persigue la satisfacción de determinadas necesidades humanas, no siempre percibidas previamente

A partir de esta definición podemos establecer los siguientes roles para el personal que participa en el proyecto. Se muestra a continuación en la Tabla 1 los roles planificados para el desarrollo de la aplicación, en ella se describe el rol, responsabilidad en el proyecto y nombre de la persona que ocupa ese rol.

Tabla 1: Planificación del proyecto por roles

Rol	Responsabilidad	Nombre
Líder del Proyecto	Debe asegurar que el proyecto se está llevando a cabo de acuerdo con las prácticas y que todo funciona según lo planeado. Su principal trabajo es remover impedimentos y reducir riesgos del producto. Coordina y facilita las reuniones. Asegura que se consiguen los objetivos de cada iteración.	Alexander Schulze
Gerente	Es el responsable de tomar las decisiones finales, acerca de estándares y convenciones a seguir durante el proyecto. Participa en la selección de objetivos y requerimientos. Tiene la responsabilidad de controlar el progreso y trabaja junto con el Jefe de Proyecto en	Alexander Schulze

	la reducción de la Lista de Reserva del Producto.	
Consultor	Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas, además aportan ideas y experiencias para el beneficio del sistema en desarrollo. Esta es una especialización menos activa, quien la ejecuta funciona en este rol por un corto período de tiempo.	Alexander Schulze
Cliente	Participa en las tareas que involucran la lista de reserva del producto.	Alexander Schulze
Programador	Elabora el código de las nuevas funcionalidades a implementar. Escribe las pruebas unitarias. Debe existir una comunicación y coordinación adecuada entre los programadores y el resto del equipo.	Lester Alfonso Zaila Viejo
Analista	Escribe las historias de usuario y las pruebas funcionales para validar su implementación.	Lester Alfonso Zaila Viejo
Diseñador	Encargado del diseño del sistema y de los prototipos de interfaces, son los máximos responsables de la realización del diseño de las metáforas y	Lester Alfonso Zaila Viejo

	supervisan el proceso de construcción.	
Encargado de Pruebas	Es el encargado de ayudar al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.	Lester Alfonso Zaila Viejo
Arquitecto	Se vincula con el analista y el diseñador ya que su trabajo tiene que ver con la estructura y el diseño del sistema. Ayuda en el diseño de las metáforas.	Lester Alfonso Zaila Viejo

2.3. Modelo de Dominio

Dentro de los artefactos que genera la metodología SXP se encuentra la plantilla del Modelo Historias de usuario del negocio, donde se definen las características específicas del negocio, así como la forma en que interactúa el sistema con los clientes y viceversa. Pero si dicho negocio no está bien definido entre los clientes y los ejecutores del proyecto; entonces es generado el Modelo de Dominio que se propone se presenta a continuación en la Fig. 1:

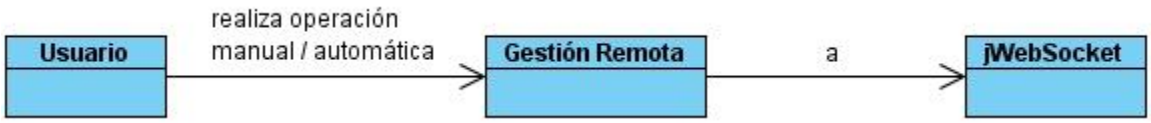


Fig. 1: Modelo de dominio.

A continuación se describen brevemente los elementos del diagrama del modelo de dominio presentado.

Usuario: Persona interesada en gestionar remotamente el servidor jWebSocket.

Gestión Remota: Permite conectarse al servidor jWebSocket y comprobar su estado actual.

jWebSocket: Es un puro servidor Java basado en websocket que brinda soluciones de transmisión de flujo en tiempo real y controlado.

2.4. Lista de Reserva del Producto (LRP)

Almacena en una lista priorizada todo el trabajo a desarrollar en el proyecto. Esta lista puede crecer y modificarse a medida que se obtienen más conocimientos acerca del producto y del cliente. Tiene la limitación de que sólo puede cambiarse entre iteraciones. Tendrá como objetivo asegurar que el producto definido al terminar la lista es el más correcto, útil y competitivo posible. Puede estar conformada por requerimientos técnicos y del negocio, funciones, errores a reparar, defectos, mejoras y actualizaciones tecnológicas. La presente investigación cuenta con 10 requisitos funcionales y 21 requisitos no funcionales, los cuales se muestran en la Tabla 2:

Tabla 2: Lista de reserva del producto

Prioridad	Ítem *	Descripción	Estimación	Estimado por
Muy Alta				
	1	Añadir Prueba.	1 Semana	Analista
	2	Eliminar Prueba.	1 Semana	Analista
	3	Actualizar Prueba	1 Semana	Analista
	4	Listar Prueba	1 Semana	Analista
	5	Añadir Tarea	1 Semana	Analista
	6	Eliminar Tarea	1 Semana	Analista
	7	Actualizar Tarea	1 Semana	Analista
	8	Listar Tarea	1 Semana	Analista
	9	Ejecutar Tarea: ejecuta la tarea y obtiene resultados.	3 Semanas	Analista
	10	Notificar: Notifica al administrador del sistema el estado	2 Semana	Analista

		actual del servidor.		
Alta				
Media				
Baja				
RNF (Requisitos No Funcionales)				
	11	La aplicación está dirigida a usuarios con altos conocimientos de informática y un correcto dominio del negocio en cuestión, pero la herramienta debe estar diseñada para ser de fácil manejo por el cliente.		
	12	El Sistema es una aplicación de escritorio		
	13	El sistema se mantendrá disponible 24 horas diarias durante los 7 días de la semana.		
	14	Garantizar la integridad y consistencia de los datos.		
	15	Identificar al usuario		

		antes de que pueda realizar cualquier acción sobre el sistema.		
	16	Se debe realizar la aplicación de forma versionable que permita darle mantenimientos al sistema a fin de aumentar las funcionalidades y/o corregir los errores del mismo a través de versiones posteriores.		
	17	Establecer pautas para la codificación y el diseño de la BD que permita mantener un código uniforme a la hora de realizar modificaciones en el mismo.		
	18	Se documentará la aplicación con diferentes manuales con el objetivo de explicar el uso de la aplicación para garantizar el soporte de la misma.		
	19	En el lado del		

		servidor el lenguaje de programación a utilizar es Java, empleando el Framewok jWebSocket y el IDE NetBeans 7.0.1.		
	20	En el lado del cliente se hará uso del lenguaje Java y el IDE NetBeans 7.0.1.		
	21	La metodología de desarrollo a seguir es SXP y para la modelación se utilizará la herramienta Visual Paradigm 3.4.		
	22	El gestor de BD a utilizar es MongoDB v2.0.		
	23	Todos los textos y mensajes en pantalla aparecerán en idioma inglés.		
	24	La interfaz deberá ser consistente con el mundo real, de manera que los conceptos manejados sean conocidos por los usuarios, para que		

		les sea fácil su uso y aprendizaje.		
	25	Lograr un diseño amigable y de fácil uso.		
	26	Todas las interfaces de usuario que se definan para el sistema respetarán los patrones de diseño establecidos para el proyecto.		
	27	Los requisitos mínimos de hardware para el correcto funcionamiento de la aplicación son: 512 MB de memoria RAM, microprocesador Pentium IV, tarjeta red 100 Mbps.		
	28	Los requisitos mínimos de hardware para el servidor de Base de Datos son: 2GB de RAM.		
	29	El servidor de aplicaciones debe tener instalado la máquina virtual de Java OpenJDK 7 y el servidor de		

		jWebSocket.		
	30	El servidor de base de datos requiere un sistema operativo de 64 bits.		
	31	El código de la aplicación será liberado bajo la Licencia Pública General Reducida de GNU (LGPL)		

2.5. Historias de Usuario y Tareas de Ingeniería

Las historias de usuarios en la metodología de desarrollo SXP tienen como objetivo describir las tareas que el sistema debe realizar, depende en gran medida de las especificaciones realizadas por el cliente. Se escriben con un lenguaje natural y con palabras concisas. Serán la guía para la construcción posterior de las pruebas de aceptación demostrando de esta manera la correcta implementación de las historias de usuario.

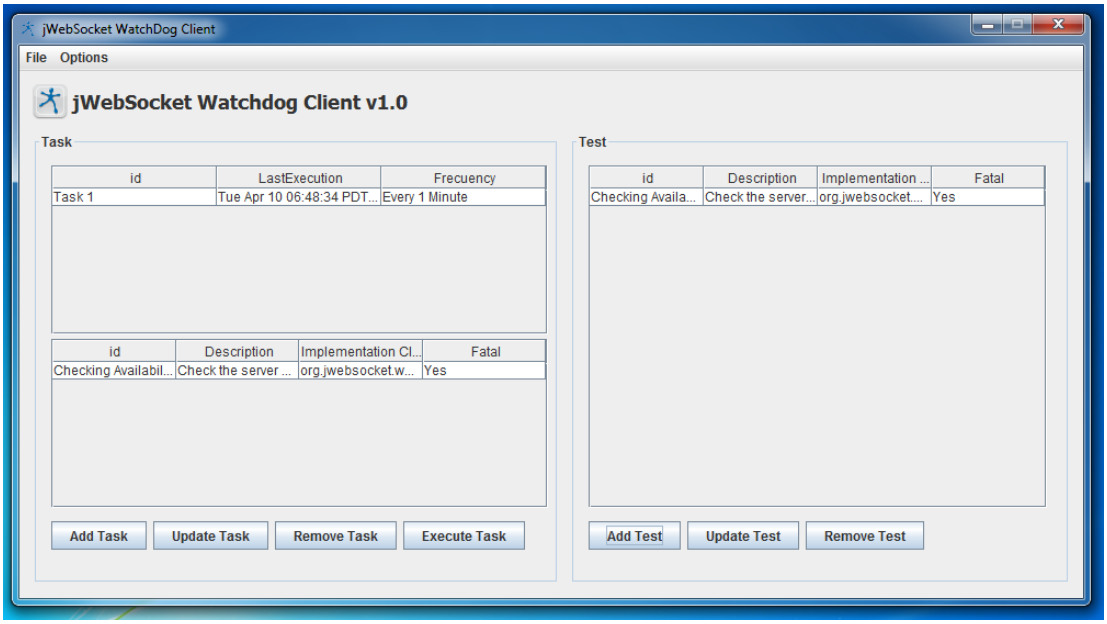
Tabla 3: Gestionar prueba de la HU_1

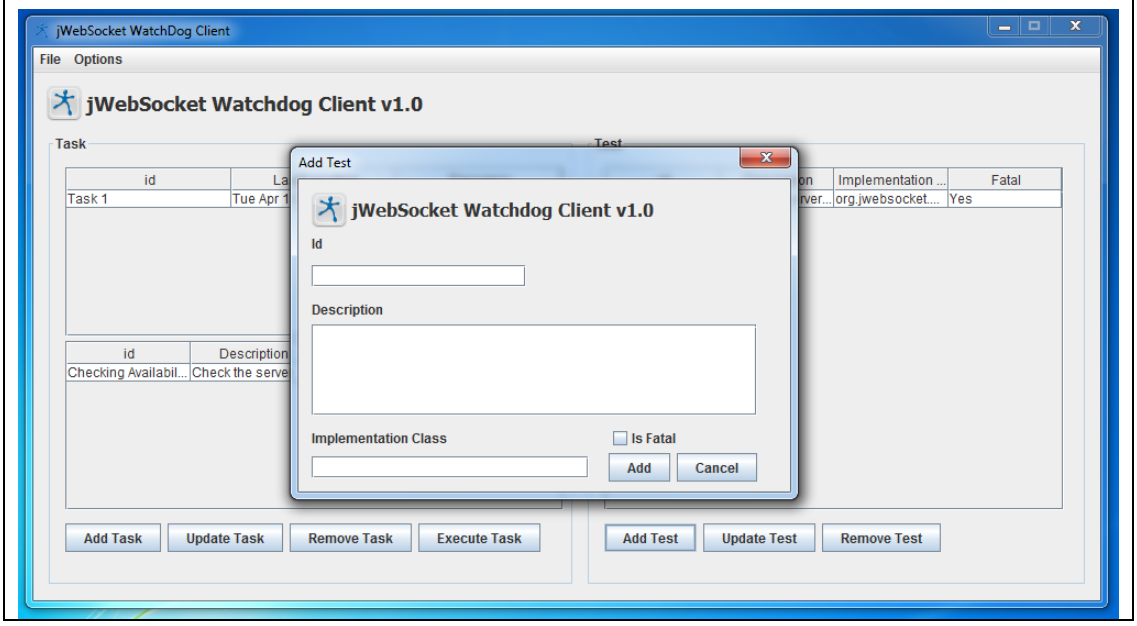
Historia de Usuario	
Número: HU_1	**Nombre Historia de Usuario:** Gestionar prueba
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Lester Alfonso Zaila Viejo	**Iteración Asignada:** 2
Prioridad en Negocio: Alta	**Puntos Estimados:**3 semanas
Riesgo en Desarrollo: Bajo	**Puntos Reales:** 2
Descripción: Esta Historia de Usuario tiene como objetivo añadir, eliminar, actualizar y listar una prueba para realizarle al servidor jWebSocket. Las	

pruebas tienen como objetivo comprobar la disponibilidad e integridad del servidor jWebSocket.

Observaciones: Ninguna

Prototipo de interface:





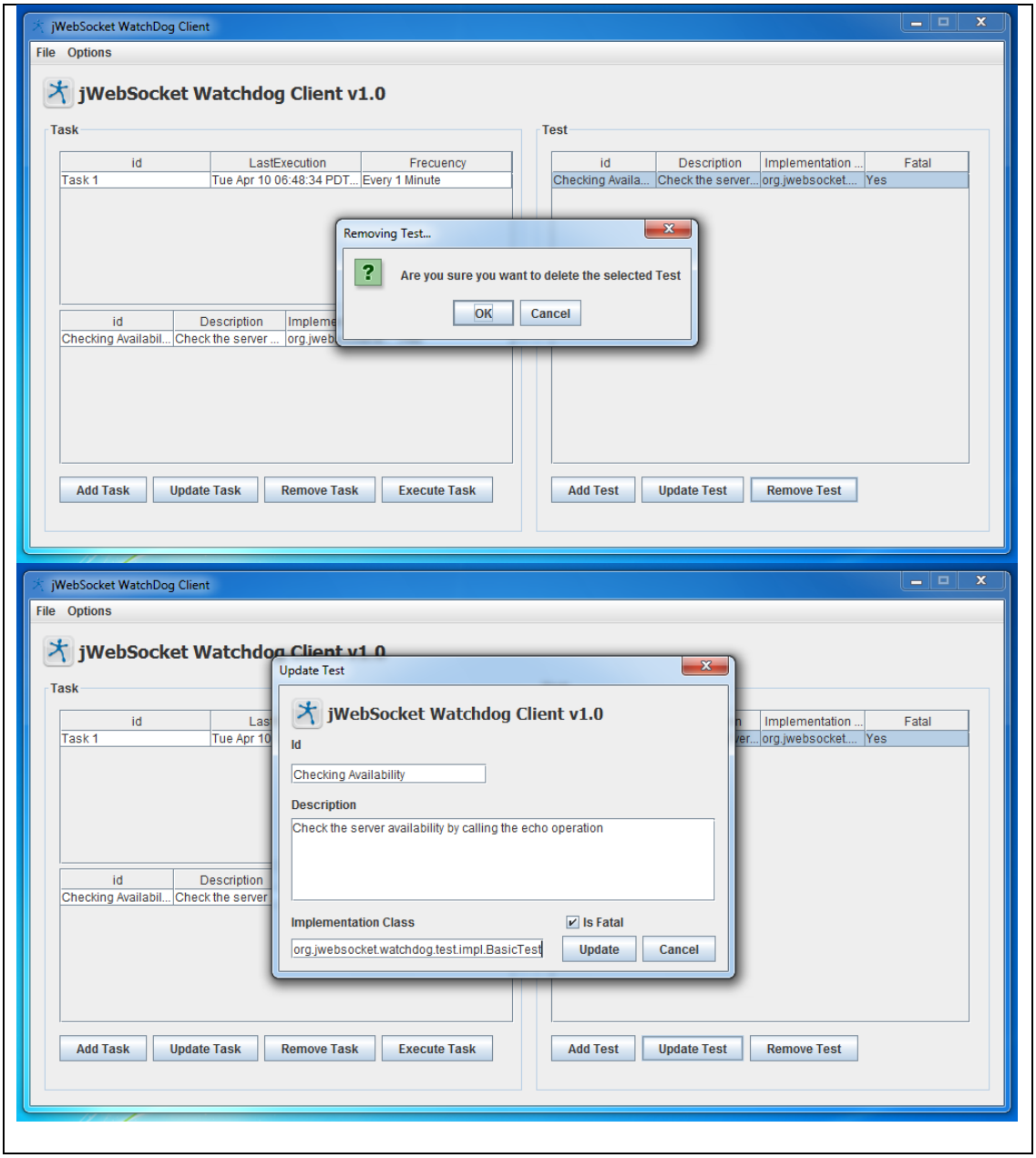


Tabla 4: HU_1 de la tarea de ingeniería adicionar prueba

	Tarea de Ingeniería
Número Tarea: 1.1	**Número Historia de Usuario:** HU_1
Nombre Tarea: Adicionar prueba	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 5 días

Fecha Inicio: 1/11/2011	**Fecha Fin:** 7/11/2011
Programador Responsable:: Lester Alfonso Zaila Viejo	
Descripción: Adicionar una prueba para permitir para que sea luego ejecutada dentro de una tarea. Una prueba consta, de un id, una descripción, una clase de implementación y si es fatal o no.	

Tabla 5: HU_1 de la tarea de ingeniería actualizar prueba

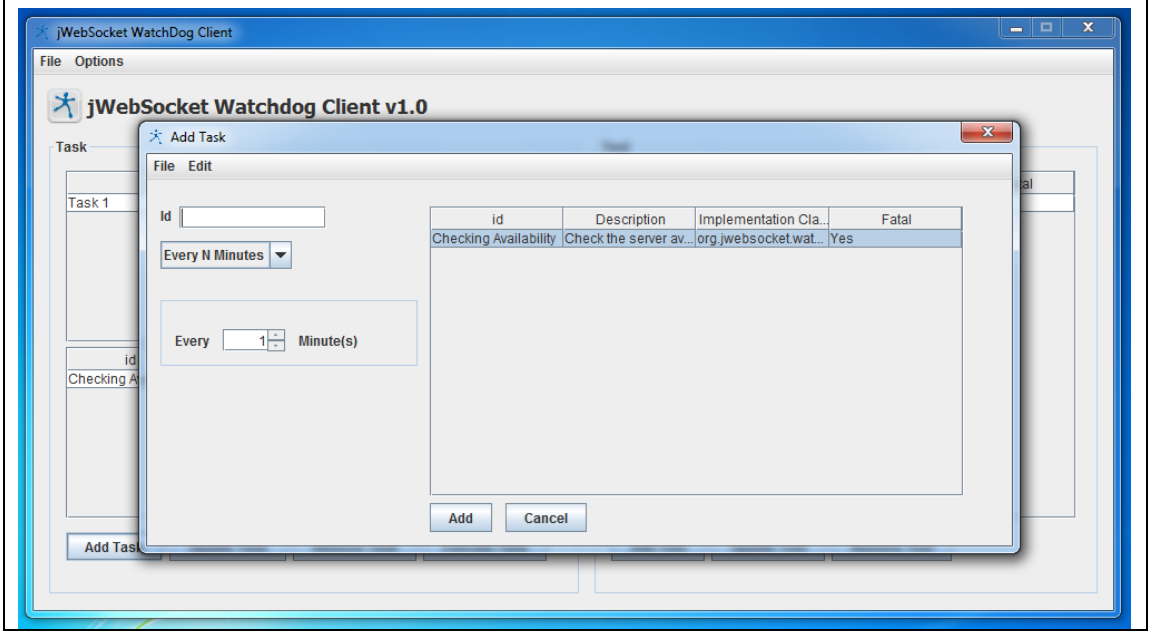
		Tarea de Ingeniería
Número Tarea: 1.2	**Número Historia de Usuario:** HU_1	
Nombre Tarea: Actualizar prueba		
Tipo de Tarea : Desarrollo		**Puntos Estimados:** 5 días
Fecha Inicio: 8/11/2011		**Fecha Fin:** 14/11/2011
Programador Responsable:: Lester Alfonso Zaila Viejo		
Descripción: Actualizar los datos de las pruebas o corregir errores. Los campos a actualizar son id, descripción, clase de implementación, y si es fatal o no.		

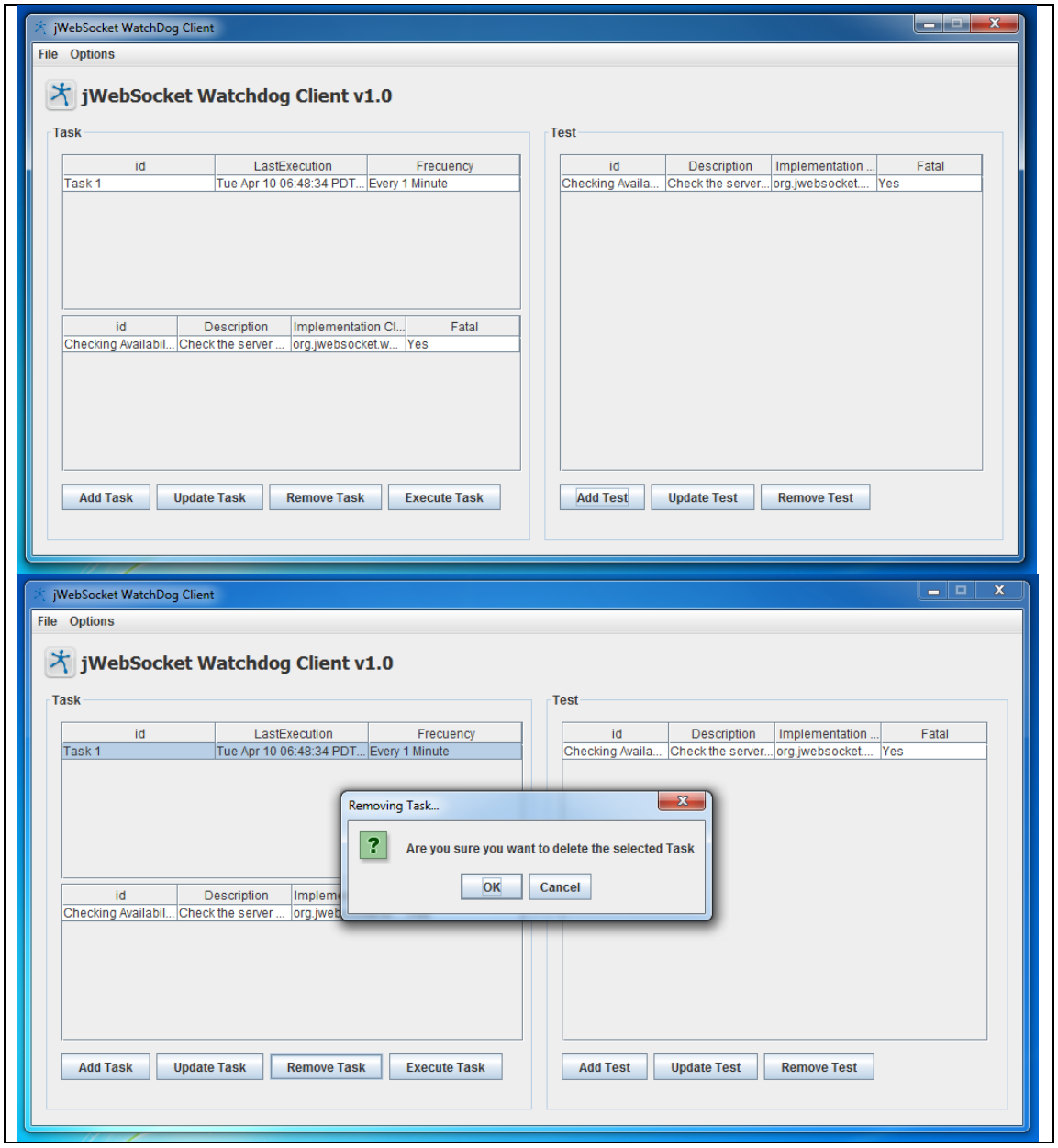
Tabla 6: HU_1 de la tarea de ingeniería eliminar prueba

		Tarea de Ingeniería
Número Tarea: 1.3	**Número Historia de Usuario:** HU_1	
Nombre Tarea: Eliminar prueba		
Tipo de Tarea : Desarrollo		**Puntos Estimados:** 5 días
Fecha Inicio: 15/09/2011		**Fecha Fin:** 21/09/2011
Programador Responsable:: Lester Alfonso Zaila Viejo		

Descripción: Eliminar una prueba hace que ya no forme parte de una tarea y no podrá ser recuperada.

Tabla 7: Gestionar tarea de la HU_2

Historia de Usuario	
Número:HU_2	**Nombre Historia de Usuario:** Gestionar tarea.
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Lester Alfonso Zaila Viejo	**Iteración Asignada:** 2
Prioridad en Negocio: Alta	**Puntos Estimados:**3 semanas
Riesgo en Desarrollo: Bajo	**Puntos Reales:** 2
Descripción: Esta Historia de Usuario tiene como objetivo añadir, eliminar, actualizar y listar una tarea. Las tareas tienen como objetivo agrupar pruebas para que puedan ser ejecutadas en tiempo dado.	
Observaciones: Ninguna	
Prototipo de interface: 	


jWebSocket WatchDog Client
File Options
jWebSocket Watchdog Client v1.0
Task
id
LastExecution
Frecuency
Task 1
Tue Apr 10 06:48:34 PDT...
Every 1 Minute
id
Description
Implementation Cl...
Fatal
Checking Availabil...
Check the server ...
org.jwebsocket.w...
Yes
Add Task
Update Task
Remove Task
Execute Task
Test
id
Description
Implementation ...
Fatal
Checking Availa...
Check the server...
org.jwebsocket....
Yes
Add Test
Update Test
Remove Test
jWebSocket WatchDog Client
File Options
jWebSocket Watchdog Client v1.0
Task
Test
Removing Task...
Are you sure you want to delete the selected Task
OK
Cancel

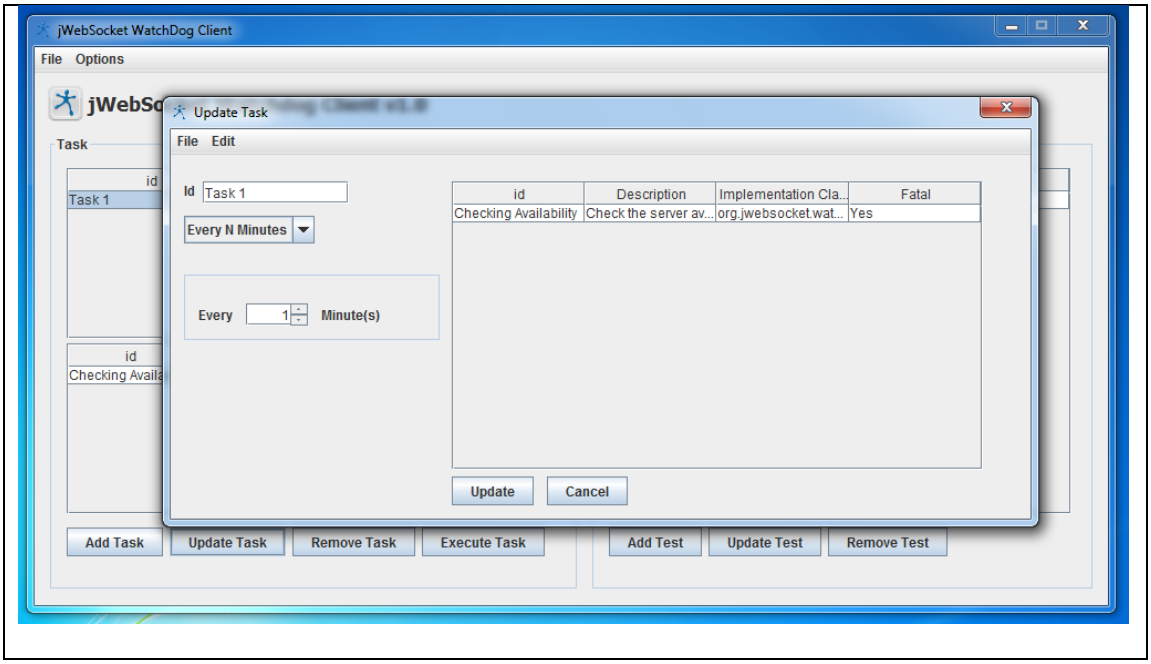


Tabla 8: HU_2 de la tarea de ingeniería adicionar tarea.

	Tarea de Ingeniería
Número Tarea: 2.1	**Número Historia de Usuario:** HU_2
Nombre Tarea: Adicionar Tarea	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 5 días
Fecha Inicio: 22/11/2011	**Fecha Fin:** 28/11/2011
Programador Responsable:: Lester Alfonso Zaila Viejo	
Descripción: Adicionar una tarea consiste en adicionarle pruebas para que éstas puedan ser ejecutadas en una hora determinada o manualmente. Una tarea contiene un id, una frecuencia, un intervalo de tiempo y una o varias tareas.	

Tabla 9: HU_2 de la tarea de ingeniería actualizar tarea.

Tarea de Ingeniería	
Número Tarea: 2.2	**Número Historia de Usuario:** HU_2
Nombre Tarea: Actualizar Tarea	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 5 días
Fecha Inicio: 29/11/2011	**Fecha Fin:** 5/12/2011
Programador Responsable:: Lester Alfonso Zaila Viejo	
Descripción: Actualizarle los datos a las tareas consiste en cambiarle el id de la tarea, cambiarle la frecuencia de ejecución o cambiarle las tareas que contiene.	

Tabla 10: HU_2 de la tarea de ingeniería eliminar tarea.

Tarea de Ingeniería	
Número Tarea: 2.3	**Número Historia de Usuario:** HU_2
Nombre Tarea: Eliminar tarea	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 5 días
Fecha Inicio: 6/12/2011	**Fecha Fin:** 13/12/2011
Programador Responsable:: Lester Alfonso Zaila Viejo	
Descripción: Al eliminar una tarea solo desaparece la tarea, no las pruebas que contiene. Una prueba eliminada no podrá ser recuperada.	

Tabla 11: Ejecutar tarea de la HU_3

Historia de Usuario	
Número:HU_3	**Nombre Historia de Usuario:** Ejecutar tarea
Modificación de Historia de Usuario Número: Ninguna	

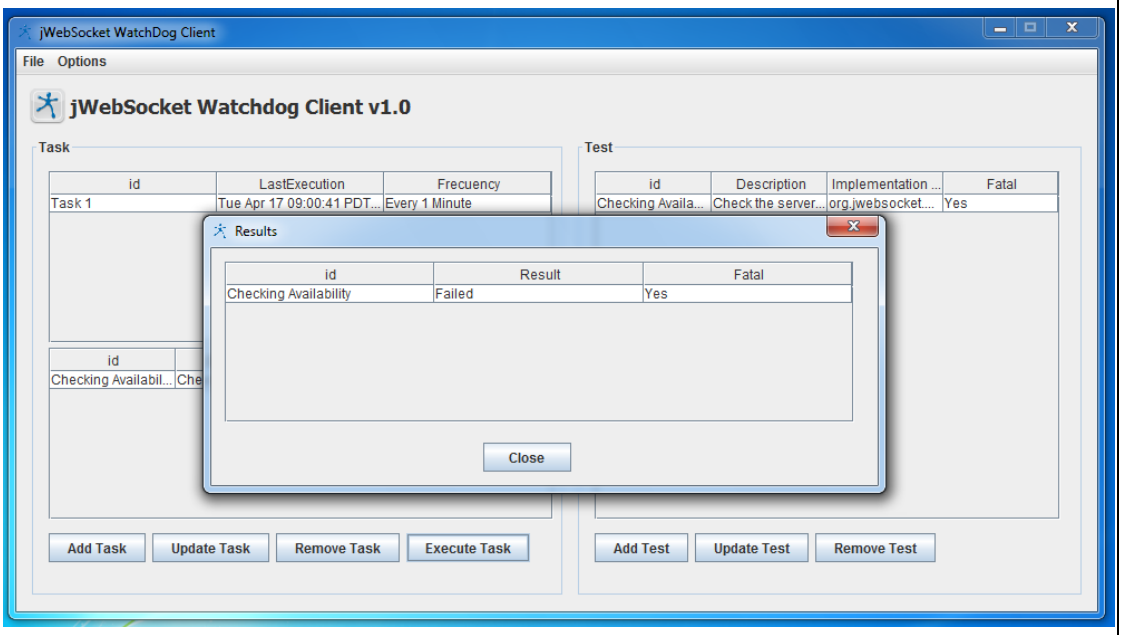
Usuario: Lester Alfonso Zaila Viejo	**Iteración Asignada:** 2
Prioridad en Negocio: Muy Alta	**Puntos Estimados:**4 semanas
Riesgo en Desarrollo: Alta	**Puntos Reales:** 3
Descripción: Esta historia de usuario tiene como objetivo ejecutar todas las tareas.	
Observaciones: Ninguna	
Prototipo de interface: 	

Tabla 12: HU_3 de la tarea de ingeniería ejecutar tarea.

	Tarea de Ingeniería
Número Tarea: 3.1	**Número Historia de Usuario:** HU_3
Nombre Tarea: Ejecutar tarea	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 20 días
Fecha Inicio: 14/12/2011	**Fecha Fin:** 30/1/2012

Programador Responsable:: Lester Alfonso Zaila Viejo
Descripción: Ejecutar una tarea puede hacerse de forma manual o de forma automática, haciendo un reporte al administrador de pruebas que este le envía los datos a los notificadores a través de los escuchadores.

Tabla 13: Notificar de la HU_4

Historia de Usuario	
Número:HU_4	**Nombre Historia de Usuario:** Notificar
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Lester Alfonso Zaila Viejo	**Iteración Asignada:** 2
Prioridad en Negocio: Muy Alta	**Puntos Estimados:**3 semanas
Riesgo en Desarrollo: Media	**Puntos Reales:** 3
Descripción: Esta Historia de Usuario tiene como objetivo notificar al administrador del sistema vía correo electrónico y SMS. Los resultados de las pruebas ejecutadas si éstas fallasen. En el caso de que los resultados sean satisfactorios no se harán notificaciones.	
Observaciones: Ninguna	
Prototipo de interface: Ninguna	

Tabla 14: HU_4 de la tarea de ingeniería notificar.

Tarea de Ingeniería	
Número Tarea: 4.1	**Número Historia de Usuario:** HU_4
Nombre Tarea: Notificar	

Tipo de Tarea : Desarrollo	**Puntos Estimados:** 15 días
Fecha Inicio: 31/1/2012	**Fecha Fin:** 20/2/2012
Programador Responsable:: Lester Alfonso Zaila Viejo	
Descripción: Los notificadores solo serán usados en el caso que una tarea falle, enviando una descripción de la tarea fallida a los administradores del sistema.	

2.6. Plan de Releases

El plan de releases permite realizar las entregas intermedias y la entrega final. Tiene como entrada la relación de historias de usuario definidas previamente. Para colocar una historia en cada iteración se tiene en cuenta la prioridad que definió el cliente para dicha historia. Como resultado de la priorización de historias de usuario se llegó a la siguiente planificación:

Tabla 15: Plan de releases.

Release	Descripción de la iteración	Orden de la HU a implementar	Duración total
Iteración 2	En esta iteración se va a desarrollar una primera versión de la aplicación para la cual se desarrollarán todas las historias de usuario ya que las mismas tienen prioridad alta.	HU_1 HU_2 HU_3 HU_4	*13 semanas*

2.7. Descripción de la Arquitectura

La arquitectura de una aplicación es la vista conceptual de su estructura. Toda aplicación contiene código de presentación, procesamiento de datos y almacenamiento de datos. La arquitectura de las aplicaciones difiere según como este distribuido su código. Para desarrollar la aplicación jWebSocket Watchdog Client se utilizará una arquitectura en capas. La programación por capas es una arquitectura cliente-servidor que tiene como objetivo primordial la separación de la lógica del negocio de la lógica del diseño. La ventaja principal de este estilo es que

el desarrollo se puede llevar a cabo en varios niveles y en caso de que sobrevenga algún cambio, sólo se afecta al nivel requerido sin tener que revisar entre código mezclado. Además distribuye el trabajo de creación de una aplicación por niveles, permitiendo que cada grupo de trabajo esté totalmente abstraído del resto de los niveles.

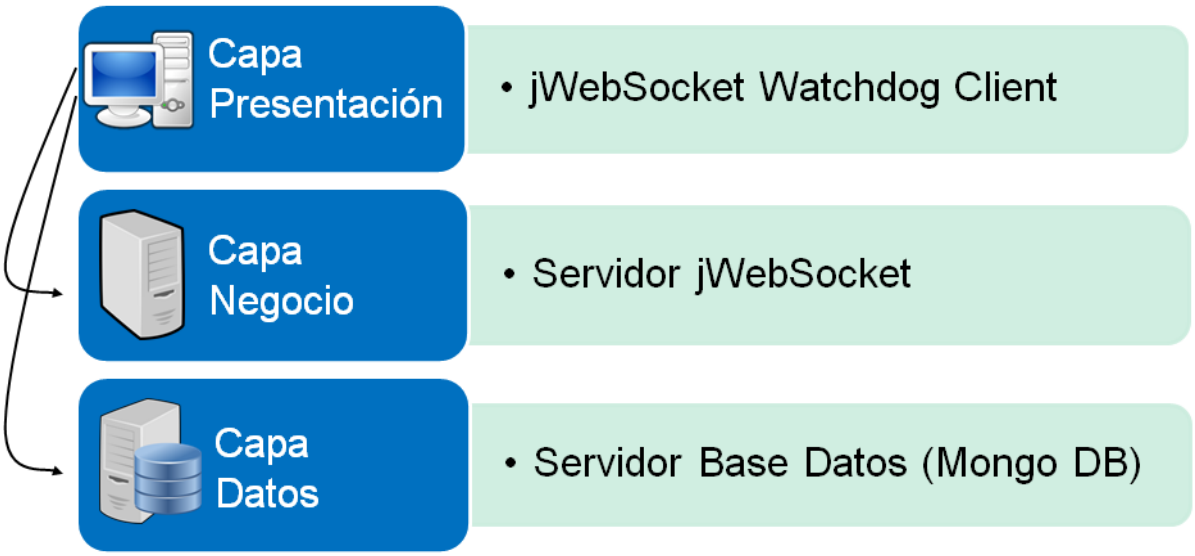


Fig. 2: Descripción de la arquitectura.

A continuación se muestra una breve descripción de las distintas capas utilizadas en la confección de la aplicación:

Capa de presentación: Está representada por la interfaz gráfica de la aplicación de escritorio jWebSocket Watchdog Client. Debe ser amigable, entendible y fácil de usar por el usuario. Esta capa se comunica con la capa de negocio y con la de datos. Muestra el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso.

Capa de negocio: Está representada por el servidor de jWebSocket. Es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados.

Capa de datos: Está representada por el servidor de base de datos MongoDB. En esta capa es donde residen los datos y es la encargada de acceder a los mismos. EL gestor de base de datos por el cual está formado realiza todo el almacenamiento de datos y recibe solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

2.8. Diseño con Metáforas

Debido a que SXP está basada en XP, y dicha metodología define un término llamado metáfora, lo cual según Martin Fowler es una historia compartida que describe como debería funcionar el sistema y define que la práctica de la metáfora consiste en formar un conjunto de nombres que actúen como vocabulario para hablar sobre el dominio del problema.

El Diseño con metáforas es sencillamente el diseño de la solución más simple que pueda funcionar y ser implementado en un momento dado del proyecto; lo cual genera el artefacto conocido como Modelo de Diseño, que a su vez está compuesto por un diagrama de paquetes, el cual expone dicho diseño.

A continuación se representa el diagrama de paquetes para el sistema que se propone:

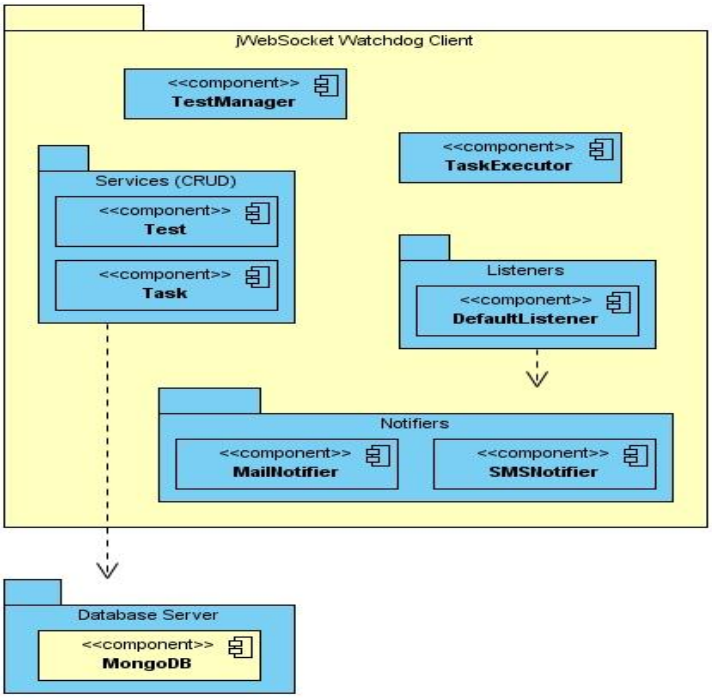


Fig. 3: Diagrama de paquetes de jWebSocket Watchdog Client.

Se presenta una breve descripción de los paquetes contenidos en el diagrama:

- **jWebSocket Watchdog Client:** Contiene los componentes TestManager que se encarga de administrar las pruebas que se le realizan al servidor jWebSocket y TaskExecutor que ejecuta las pruebas almacenadas en las tareas. Además contiene los paquetes Listeneres, Notifiers y Services.
- **Listeners:** Contiene el componente DefaultListener que verifica el resultado de las pruebas, además este paquete se relaciona con el paquete Notifiers.
- **Notifiers:** Contiene a los componentes MailNotifier y SMSNotifier que se encargan de enviar un correo electrónico o un SMS al administrador del sistema en caso de fallos en el servidor.
- **Services:** Contiene a los componentes Test que almacena las pruebas que se le realizarán al servidor jWebSocket y Task almacena las tareas que serán ejecutadas. Además este paquete se relaciona con el paquete Database Server.
- **Database Server:** Contiene el componente MongoDB, que permite persistir o consultar los datos de las pruebas y las tareas.

2.9. Diagrama de Componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo código fuente, binario y ejecutable. Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, entre otros.

A continuación se presenta el diagrama de componentes para el sistema que se propone:

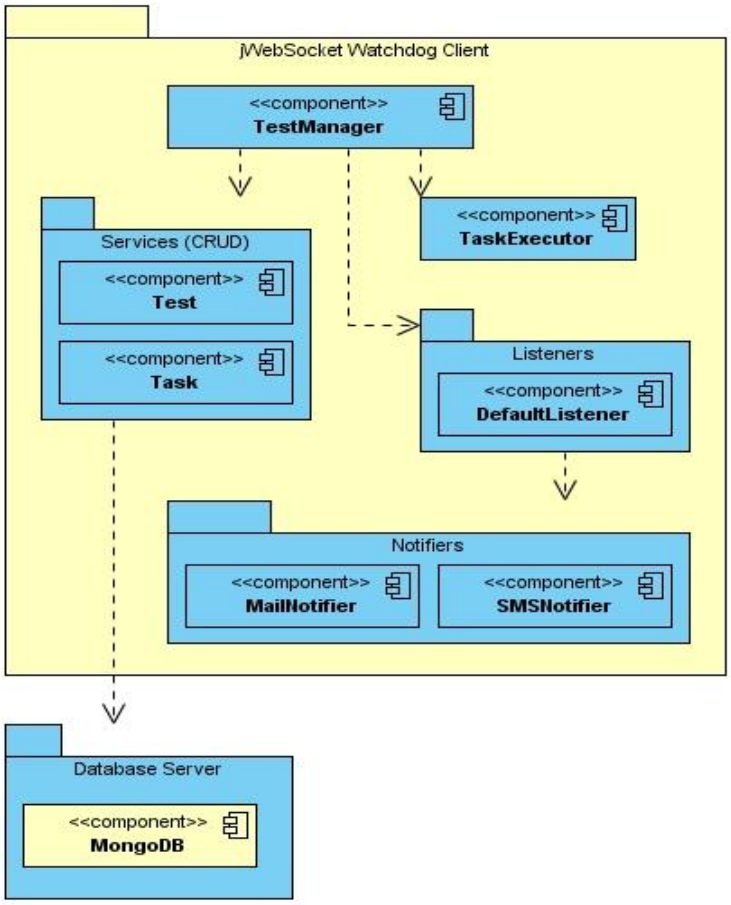


Fig. 4: Diagrama de componentes de jWebSocket Watchdog Client.

Se presenta una breve descripción de los componentes contenidos en el diagrama:

- **jWebSocket Watchdog Client:** Contendrá TestManager (El administrador de pruebas) que se encarga de interactuar con el servidor jWebSocket ejecutándole las pruebas, los listeneres (Escuchadores) sabrán si las pruebas han fallado, para así poder notificar o no en dependencia de los resultados.
- **Database Server:** El servidor de base de datos es solo para el cliente, para que puedan persistir y consultar los datos de las pruebas y las tareas. No tiene nada que ver con jWebSocket Server ya que no es parte de la solución propuesta.

Conclusiones del Capítulo

En este capítulo se definieron las características y funcionalidades de la aplicación de escritorio que se necesita desarrollar para la gestión remota de servidores jWebSocket. Se realizó el modelado de dominio para una mejor comprensión del proceso de desarrollo de la aplicación. Todos los requisitos funcionales y no funcionales fueron aprobados, permitiendo obtener un sistema eficiente. Se describieron las historias de usuario y tareas de ingeniería que se deben implementar, el plan de releases que permitió la organización del trabajo y se modeló el diagrama de paquetes y de componentes para lograr una mejor comprensión de la solución propuesta.

Capítulo 3. Implementación y Validación del Sistema

Introducción

En el presente capítulo se documenta la implementación de la solución propuesta en el capítulo anterior, generándose el diagrama de despliegue del sistema desarrollado. Además se exponen los casos de pruebas o test de aceptación a las que fue sometida la aplicación en cada una de las iteraciones. El cumplimiento de estos casos de pruebas fue el hito para avanzar hacia la próxima iteración. En este capítulo además de las pruebas se dan a conocer los resultados obtenidos hasta el momento.

3.1. Implementación

jWebSocket Watchdog Client es el guardián del servidor jWebSocket, permite hacer una conexión por WebSocket al servidor para realizarle pruebas tanto al servidor, como a las aplicaciones jWebSocket corriendo en el mismo. A su vez analiza el resultado obtenido de las pruebas y notifica al administrador del sistema en caso de fallos. La Interfaz de Programación de Aplicaciones (API) de jWebSocket Watchdog Client es sencilla e intuitiva.

La aplicación de escritorio jWebSocket Watchdog Client se encuentra separado del servidor jWebSocket. Es una aplicación independiente, que controlará el estado del servidor jWebSocket. En el siguiente diagrama podrán observar la estructura de la API de jWebSocket Watchdog Client.

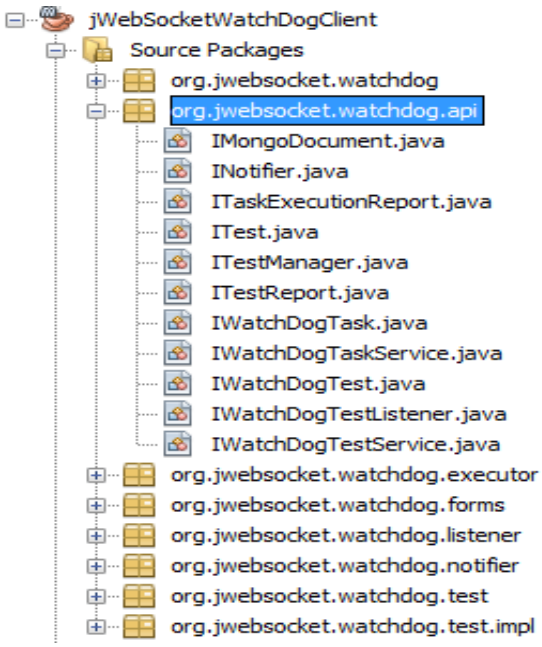


Fig. 5: Estructura de la API

Los métodos más importantes con los que cuenta la aplicación de escritorio jWebSocket Watchdog Client son:

- En la clase **TaskExecutor**: el método **run**:
  Obtiene las tareas que están programadas para ser ejecutadas y las ejecuta en el tiempo correspondiente.

- En la clase **TestManager:** el método **execute:**
  Luego de ser ejecutada la prueba el Test manager crea el reporte de ejecución notificándoles a los escuchadores los resultados.

- En la clase **WatchDogMailListener**: el método **process:**
  Se encarga de notificar vía e-mail a los administradores del sistema, los resultados fallidos obtenidos del reporte de ejecución.

El hecho de que jWebSocket hoy posea esta nueva herramienta garantiza mayor disponibilidad y una asegurada ejecución de sus aplicaciones sin temor a fallos prolongados, puesto que será notificado al instante de cualquier suceso adverso.

3.2. Validación de la solución propuesta

Para validar la presente investigación que tiene como objetivo desarrollar una aplicación para gestionar remotamente los servidores jWebSocket a través de Internet, que logre mayor nivel de disponibilidad en las aplicaciones desarrolladas con este marco de trabajo. Se decidió trazar una estrategia de validación basada en las siguientes etapas:

Se realizan casos de pruebas de aceptación alrededor de las historias de usuario que incluyen los requisitos funcionales que debe cumplir la aplicación desarrollada.

El grupo encargado de la certificación de calidad de software del Centro de Desarrollo de la Facultad Regional "Mártires de Artemisa" genera una certificación basándose principalmente en la funcionalidad, la estandarización, organización y limpieza del código de la aplicación. Además revisa los artefactos documentales que son generados por la metodología de desarrollo SXP seleccionada para la presente investigación.

Por último, Alexander Schulze, líder de la comunidad internacional, principal impulsor del proyecto jWebSocket y cliente ofrece una certificación sobre la aplicación desarrollada.

Casos de Pruebas

Las pruebas de aceptación son definidas por el cliente y preparadas por el equipo de desarrollo, aunque la ejecución y aprobación final corresponden al cliente. La utilización de estas, proporcionan grandes ventajas, permitiendo a los programadores medir la calidad de su trabajo y garantizar la entrega de un producto con calidad y en correspondencia con las necesidades del cliente. Se definieron casos de prueba para todas las historias de usuario, a continuación se dan a conocer las pruebas que se realizaron a cada una de las historias de usuario con las que cuenta la aplicación jWebSocket Watchdog Client.

Tabla 16: Casos de pruebas de aceptación HU_Gestionar prueba.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-1-1	**Nombre Historia de Usuario:** Gestionar prueba.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	
Descripción de la Prueba: Esta prueba tiene objetivo añadir una prueba para permitir para que sea luego ejecutada dentro de una tarea. Una prueba consta, de un id, una descripción, una clase de implementación y si es fatal o no.	
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB.	
Entrada / Pasos de ejecución: Para poder añadir una prueba se debe dar clic en el botón añadir prueba en la sección “Prueba”. Para poder añadir una prueba es necesario llenar todos los campos del formulario. Se dejan algunos campos vacíos, para comprobar que no se puede añadir una prueba con los campos vacíos.	
Resultado Esperado: El usuario puede añadir una prueba.	
Evaluación de la Prueba: Satisfactoria.	

Tabla 17: Casos de pruebas de aceptación HU_Gestionar prueba.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-1-2	**Nombre Historia de Usuario:** Gestionar prueba.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	
Descripción de la Prueba: Esta prueba tiene objetivo actualizar los datos de las pruebas o corregir errores. Los campos a actualizar son id, descripción, clase de	

implementación, y si es fatal o no.
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB.
Entrada / Pasos de ejecución: Para poder actualizar una prueba, ésta debe estar previamente seleccionada, y luego dar clic en el botón actualizar prueba. Para poder actualizar una prueba es necesario llenar todos los campos del formulario. Se deja sin seleccionar la prueba a actualizar para comprobar que no se puede actualizar sin estar previamente seleccionada. Se dejan algunos campos vacíos, para comprobar que no se puede actualizar una prueba con los campos vacíos.
Resultado Esperado: El usuario puede actualizar una prueba.
Evaluación de la Prueba: Satisfactoria.

Tabla 18: Casos de pruebas de aceptación HU_Gestionar prueba.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-1-3	**Nombre Historia de Usuario:** Gestionar prueba.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	
Descripción de la Prueba: Eliminar una prueba hace que ya no forme parte de una tarea y no podrá ser recuperada.	
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB.	
Entrada / Pasos de ejecución: Para poder eliminar una prueba, ésta debe estar previamente seleccionada, y luego dar clic en el botón eliminar prueba. Para poder eliminar una prueba es necesario seleccionar una prueba del formulario principal en la sección de pruebas.	

Se deja sin seleccionar una prueba para comprobar que no se puede eliminar una prueba sin ser previamente seleccionada.
Resultado Esperado: El usuario puede eliminar una prueba.
Evaluación de la Prueba: Satisfactoria.

Tabla 19: Casos de pruebas de aceptación HU_Gestionar prueba.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-1-4	**Nombre Historia de Usuario:** Gestionar prueba.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	
Descripción de la Prueba: Al listar una prueba se muestran en el orden que fueron añadidas. La aplicación es la encargada de listar las pruebas.	
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB.	
Entrada / Pasos de ejecución: Para poder listar las pruebas éstas deben haber sido añadidas previamente.	
Resultado Esperado: La aplicación puede listar las pruebas.	
Evaluación de la Prueba: Satisfactoria.	

Tabla 20: Casos de pruebas de aceptación HU_Gestionar tarea.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-2-1	**Nombre Historia de Usuario:** Gestionar tarea.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	

Descripción de la Prueba: Consiste en adicionarle pruebas para que éstas puedan ser ejecutadas en una hora determinada o manualmente. Una tarea contiene un id, una frecuencia, un intervalo de tiempo y una o varias tareas.
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB.
Entrada / Pasos de ejecución: Para poder añadir una tarea se debe dar clic en el botón añadir tarea en la sección “tarea”. En este caso los campos requeridos son: Id de la tarea, frecuencia y tiempo, y las pruebas a ejecutarse. Para poder añadir una tarea es necesario llenar todos los campos del formulario. Se dejan algunos campos vacíos, para comprobar que no se puede añadir una tarea con los campos vacíos.
Resultado Esperado: El usuario puede añadir una tarea.
Evaluación de la Prueba: Satisfactoria.

Tabla 21: Casos de pruebas de aceptación HU_Gestionar tarea.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-2-2	**Nombre Historia de Usuario:** Gestionar tarea.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	
Descripción de la Prueba: Consiste en actualizarle los datos a las tareas consiste en cambiarle el id de la tarea, cambiarle la frecuencia de ejecución o cambiarle las tareas que contiene.	
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB.	
Entrada / Pasos de ejecución: Para poder actualizar una tarea se debe tener previamente seleccionada una tarea, luego dar clic en el botón actualizar tarea.	

Para poder actualizar una tarea es necesario que esta esté previamente seleccionada y luego llenar todos los campos del formulario. Se deja sin seleccionar la tarea a actualizar para comprobar que no se puede actualizar sin estar previamente seleccionada. Se dejan algunos campos vacíos, para comprobar que no se puede añadir una tarea con los campos vacíos.
Resultado Esperado: El usuario puede actualizar una tarea.
Evaluación de la Prueba: Satisfactoria.

Tabla 22: Casos de pruebas de aceptación HU_Gestionar tarea.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-2-3	**Nombre Historia de Usuario:** Gestionar tarea.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	
Descripción de la Prueba: Al eliminar una tarea solo desaparece la tarea, no las pruebas que contiene. Una prueba eliminada no podrá ser recuperada.	
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB.	
Entrada / Pasos de ejecución: Para poder eliminar una tarea debe ser previamente seleccionada, luego dar clic en el botón eliminar tarea. Se deja sin seleccionar la tarea a eliminar para comprobar que no se puede eliminar sin estar previamente seleccionada.	
Resultado Esperado: El usuario puede eliminar una tarea.	
Evaluación de la Prueba: Satisfactoria.	

Tabla 23: Casos de pruebas de aceptación HU_Gestionar tarea.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-2-4	**Nombre Historia de Usuario:** Gestionar tarea.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	
Descripción de la Prueba: Para poder listar la tarea esta debe haber sido previamente añadida, la aplicación es la encargada de listar las tareas.	
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB.	
Entrada / Pasos de ejecución: Para poder listar las tareas éstas deben haber sido añadidas previamente.	
Resultado Esperado: La aplicación podrá listar las pruebas.	
Evaluación de la Prueba: Satisfactoria.	

Tabla 24: Casos de pruebas de aceptación HU_Ejecutar tarea.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-3-1	**Nombre Historia de Usuario:** Ejecutar tarea.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	
Descripción de la Prueba: Esta prueba tiene como objetivo ejecutar una tarea, puede hacerse de forma manual o de forma automática, pero en este caso lo haremos de forma manual.	
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB.	
Entrada / Pasos de ejecución: Para poder ejecutar una tarea manualmente ésta debe estar previamente seleccionada, luego se debe hacer a través del	

botón “Ejecutar Tarea”. Se deja sin seleccionar la tarea a ejecutar para comprobar que no se puede ejecutar una tarea sin estar previamente seleccionada.
Resultado Esperado: El usuario podrá ejecutar una tarea al servidor jWebSocket.
Evaluación de la Prueba: Satisfactoria.

Tabla 25: Casos de pruebas de aceptación HU_Notificar.

Caso de Prueba de Aceptación	
Código Caso de Prueba: jWSWC-4-1	**Nombre Historia de Usuario:** Notificar.
Nombre de la persona que realiza la prueba: Lester Alfonso Zaila Viejo	
Descripción de la Prueba: Esta Historia de Usuario tiene como objetivo notificar al administrador del sistema vía correo electrónico, los resultados de las pruebas ejecutadas si éstas fallasen. En el caso de que los resultados sean satisfactorios no se harán notificaciones.	
Condiciones de Ejecución: Para poder realizar la prueba debe estar instalado el servidor de base datos MongoDB y el servidor de jWebSocket al que se le realizará la prueba debe estar detenido previamente.	
Entrada / Pasos de ejecución: La aplicación es quien analizará los resultados de las pruebas realizadas, en este caso las pruebas fueron fallidas puesto que se detuvo el servidor jWebSocket previamente. Y se ejecutó una tarea manualmente.	
Resultado Esperado: La aplicación debe notificar al administrador del sistema que los resultados de las pruebas fueron fallidos.	
Evaluación de la Prueba: Satisfactoria.	

Certificación de Calidad de software

La aplicación desarrollada y los artefactos generados por la metodología SXP fueron puestos a consideración del Grupo de Calidad del Centro de Desarrollo de la Facultad Regional "Mártires de Artemisa". Los artefactos que explican todo el proceso de análisis, diseño e implementación de la aplicación utilizando SXP son: Plantilla Lista de Reserva del Producto (LRP), Historias de Usuario, Tareas de Ingeniería, Arquitectura de Software, Modelo de Diseño, Plan de Releases, Modelo de Historias de Usuario del Negocio, Estándar de Código, Casos de Prueba de Aceptación y los Manuales de Usuario, Administración y Desarrollador. Evaluaron además la estandarización y organización del código fuente de la aplicación.

Valoración del cliente

La presente investigación se puso a consideración del cliente Alexander Schulze, con el objetivo de demostrar un correcto funcionamiento de la aplicación desarrollada. Se le entregó no solo el código fuente, sino también la documentación necesaria para apoyar el proceso de valoración por parte del cliente. Se elaboraron varios documentos como el manual de desarrollador, manual de instalación y manual de usuario en el idioma inglés para que la comunicación ocurriera de forma fluida y entendible entre ambas partes.

Algunas imprecisiones fueron señaladas en el proceso de revisión por parte del cliente, las cuales fueron solucionadas y puestas a consideración por el cliente nuevamente. Al finalizar la investigación, el cliente estuvo de acuerdo con la calidad de la documentación y el las funcionalidades de la aplicación.

3.3. Resultados Obtenidos

Por el resultado satisfactorio de los casos de pruebas realizados a la aplicación jWebSocket Watchdog Client queda disponible su versión 1.0. Se obtuvo un sistema que cumple con todas las especificaciones para permitir la gestión remota de servidores jWebSocket.

Por parte del grupo de Certificación de Calidad se obtuvo un buen resultado, debido a la correcta estandarización y limpieza del código fuente y los artefactos están

bien generados. Con estos resultados se garantiza que la aplicación y su documentación poseen la calidad requerida. (Ver Anexo 1)

El criterio emitido por el cliente Alexander Schulze permite que la aplicación y su documentación en inglés sean utilizadas por la Comunidad jWebSocket e integradas al proyecto internacional jWebSocket. (Ver Anexo 2)

3.4. Funcionalidades Obtenidas

Entre las principales funcionales que posee la aplicación jWebSocket Watchdog Client en su versión 1.0 se pueden mencionar:

- Permite insertar, actualizar, eliminar y listar pruebas.
- Admite actualizar, eliminar, actualizar y listar tareas.
- Ejecución de pruebas manuales y automáticas.
- Notificación vía e-mail de los resultados fallidos de las pruebas.

3.5. Diagrama de Despliegue

El diagrama de despliegue es un artefacto que modela la arquitectura en tiempo de ejecución de un sistema. En él se indica la situación física de los componentes lógicos desarrollados, lo que significa que sitúa el software en el hardware que lo contiene. A continuación se muestra el diagrama de despliegue, que representa la distribución física del sistema en términos de cómo se distribuirán las funcionalidades entre los nodos, donde cada nodo representa un recurso de cómputo, siendo estos procesadores o dispositivos hardware que se necesitan para el despliegue del sistema:

Fig. 6: Diagrama de despliegue de jWebSocket Watchdog Client.

Se presenta una breve descripción de los componentes contenidos en el diagrama:

Cliente: El cliente se conectará por medio del protocolo websocket al Servidor jWebSocket para realizarle pruebas y notificará a los administradores del servidor jWebSocket en caso de existir un fallo. El cliente se conectará al servidor de base de datos de MongoDB a través del protocolo TCP/IP para persistir o consultar los datos.

Servidor jWebSocket: Será el servidor ejecutando las aplicaciones y que será monitoreado por el cliente jWebSocket Watchdog.

Servidor MongoDB: El servidor MongoDB almcenará los datos referentes a las pruebas y tareas a ejecutarle al servidor jWebSocket.

3.6. Aporte Social y Económico

jWebSocket Watchdog Client es una aplicación desarrollada para controlar el estado de los servidores jWebSocket y sus aplicaciones. Brindará apoyo fundamentalmente a los administradores de sistemas, favoreciendo la estricta vigilancia de los servicios, para mantener aplicaciones críticas en línea, con un control y monitorización eficiente,

Además, permite la gestión de las aplicaciones desarrolladas con el marco de trabajo jWebSocket. Es una herramienta libre, sencilla e intuitiva, por lo tanto no se emplean recursos en comprar la aplicación ni es necesario enseñar cómo utilizarla. Cualquier empresa informática que haga uso de jWebSocket Watchdog Client podrá ganar valioso tiempo puesto que detecta problemas en cuestión de minutos sin necesidad de viajar al sitio físico donde se encuentran alojados los servidores y podrán ser notificados instantáneamente todos los administradores del sistema de cualquier situación adversa.

Conclusiones del Capítulo

En este capítulo se realizó el diagrama de despliegue que indica la situación física de los componentes lógicos desarrollados. Además se realizaron casos de pruebas que guiaron la calidad de la aplicación desarrollada. De esta manera los casos de prueba demostraron que las funciones de la aplicación son operativas y que se produce un resultado correcto. Demostrándose así que el sistema se encuentra en óptimas condiciones para el proceso de colaboración en línea en tiempo real.

Conclusiones

Con la realización del presente trabajo de diploma se dio respuesta a cada una de las preguntas científicas planteadas, obteniéndose de manera general las siguientes conclusiones:

Se realizó la fundamentación teórico-metodológica de la investigación permitiendo conceptualizar la gestión remota de servidores a través de Internet. Las soluciones actuales para la gestión remota de servidores no utilizan el protocolo websocket. Se desarrolló una aplicación para la gestión remota de servidores jWebSocket. Se realizaron pruebas de aceptación a la aplicación de escritorio desarrollada sobre la solución propuesta, permitiendo demostrar las funcionalidades para la cual fue diseñada y se validó la capacidad y potencialidad de la aplicación de escritorio desarrollada mediante la certificación de calidad de software y la valoración del cliente para comprobar su funcionamiento.

Recomendaciones

Desarrollar jWebSocket Watchdog Client versión 2.0, incluyéndole nuevas funcionalidades para brindar más potencialidades al proceso de gestión remota de servidores jWebSocket a través de Internet utilizando esta aplicación.

Referencias Bibliográficas:

¿Qué es un 'framework'? **Sánchez, Jordi. 2006.** Valencia : http://jordisan.net/blog/2006/que-es-un-framework, 2006.

10gen, Inc. 2011. Agile and Scalable . *mongoDB.* [En línea] 2011. [Citado el: 16 de 02 de 2012.] http://www.mongodb.org/.

—. 2011. What is MongoDB? *10gen.* [En línea] 2011. [Citado el: 16 de 02 de 2012.] http://www.10gen.com/why-mongodb.

Apache Software Foundation. 2011. Apache Subversion. *Apache Subversion.* [En línea] 10 de 12 de 2011. [Citado el: 10 de 12 de 2011.] http://subversion.apache.org/.

Application of Software Watchdog as a Dependability Software Service for Automotive Safety Relevant Systems. **Chen, Xi, y otros. 2007.** Berlin : s.n., 2007.

Conjecture Corporation. 2012. What Is Remote Management? *WiseGEEK.* [En línea] 2012. [Citado el: 16 de 03 de 2012.] http://www.wisegeek.com/what-is-remote-management.htm.

Control de servidores en sistemas de tiempo real. **Velasco, Manuel y Marti, Paul. 2006.** Cataluña : s.n., 2006.

Figueroa, Roberth G, Solís, Camilo J y Cabrera, Armando A. 2011. Entorno Virtual de Aprendizaje. [En línea] 2011. [Citado el: 13 de 12 de 2011.] http://eva.uci.cu/mod/resource/view.php?id=9303&subdir=/Metodologias/RUP.

Frame Approach for WebSockets. **Schulze, Alexander. 2011.** 2011. http://dl.globalstf.org/index.php?page=shop.product_details&flypage=flypage_images.tpl&product_id=528&category_id=42&option=com_virtuemart&Itemid=4&vmcchk=1&Itemid=4, 2011..

Herramientas Case. **Alfaro, Félix Murillo. 2011.** 2011, COLECCION CULTURA INFORMATICA.

Lamb, Charles, Hair, Joseph y McDaniel, Carl.*Marketing.*

Metodologías Tradicionales Vs. Metodologías Ágiles. **Figueroa, Roberth G., Solís, Camilo J. y Cabrera, Armando A. 2011.** Loja : http://www.docstoc.com/docs/102328746/articulo-metodologia-de-sw-formato, 2011.

National Instrument. National Instrument. [En línea] http://zone.ni.com/devzone/cda/tut/p/id/4487.

NetBeans. 2011.*NetBeans.* [En línea] Oracle Corporation, 2011. http://netbeans.org/features/index.html.

Nourie, Dana. 2005.*Oracle Sun Developer Network (SDN).* [En línea] Oracle Corporation, 24 de Mayo de 2005. http://java.sun.com/developer/technicalArticles/tools/intro.html.

NTRglobal. 2011. NTRglobal. [En línea] 2011. [Citado el: 23 de abril de 2012.] http://ww2.ntrglobal.com/gestion-del-servidor-remoto.asp.

Oracle. 2010. Watchdog Process. [En línea] 2010. http://docs.oracle.com/cd/E19316-01/820-2761/aarat/index.html.

—. 2010. Watchdog Timer Application Mode . [En línea] 2010. http://docs.oracle.com/cd/E19350-01/820-3010-12/A_watchdog.html.

Pohronská, Mária y Krajcovic, Tibor. 2010.*FAULT-TOLERANT EMBEDDED SYSTEMS WITH MULTIPLE FPGA IMPLEMENTED WATCHDOGS.* Slovak Republic : s.n., 2010.

RapidSVN. 2011. RapidSVN. [En línea] 2011. [Citado el: 13 de 12 de 2011.] http://www.rapidsvn.org/.

Rumbaugh, James y Jacobson, Ivar. 2007.*El Lenguaje Unificado de Modelado. Manual de Referencia.* s.l. : ADDISON-WESLEY, 2007. 9788478290871.

Schwaber, Ken y Beedle, Mike. 2011.*Agile Software Development with SCRUM.* s.l. : Prentice Hall, 2011. 0130676349.

Sherif, Adnan. 2006.*A FRAMEWORK FOR SPECIFICATION AND VALIDATION OF REAL TIME SYSTEMS USING CIRCUS ACTION.* Brasil : s.n., 2006.

Stanton, William, Etzel, Michael y Walker, Bruce. 2004.*Fundamentos de Marketing.* s.l. : Mc Graw Hill, 2004.

SXP, Metodología Ágil para el Desarrollo de Software. **Peñalver, G, Meneses, A y García, S. 2010.** Chile : 1er congreso Iberoamericano de Ingeniería de Proyectos, 2010.

Visual Paradigm International Ltd. 2011. Visual Paradigm for UML. *Visual Paradigm.* [En línea] 2011. [Citado el: 17 de 02 de 2012.] http://www.visual-paradigm.com/product/vpuml/.

WAINERMAN, EFRAIM. 2002.*SERVIDORES VIRTUALES IMPLEMENTADOS EN EL SISTEMA OPERATIVO LINUX.* Luján, Argentina : s.n., 2002.

Wells, Don. 2009. Extreme Programming: A gentle introduction. *Extreme Programming.* [En línea] 28 de 07 de 2009. [Citado el: 16 de 02 de 2012.] http://www.extremeprogramming.org/.

ANEXOS

Anexo 1: Certificación de Calidad de Software

Universidad de las Ciencias Informáticas
Facultad Regional "Mártires de Artemisa"

Centro de Desarrollo

Aval de Calidad de Software

El grupo de Calidad de Software del Centro de Desarrollo de la Facultad Regional "Mártires de Artemisa" conformado por:

- **Asesora de Calidad**: Ing. Maidel Ojeda Cruz
- **Asesor de Tecnología**: Ing. Domma Moreno Dager
- **Especialista de Calidad**: Ing. Yenisleydi Rodríguez Martínez

emite el presente **Aval de Calidad de Software** en colaboración con los especialistas del Centro de Desarrollo a: Lester Alfonso Zaila Viejo, como resultado satisfactorio de su desempeño en la tareas asociadas al proyecto: JWebSocket.

Para emitir el presente aval se valoraron un conjunto de elementos evaluados de manera individual teniendo en cuenta los parámetros de calidad de software del proyecto. A continuación se presenta los resultados en cada uno de los aspectos valorados:

Elementos evaluados	Resultado
Estandarización del código fuente del proyecto	Satisfactorio
Limpieza, organización y estructuración del código fuente	Satisfactorio
Funcionalidad e integración del sistema	Satisfactorio
Validación y seguridad de la información gestionada	Satisfactorio
Generación de todos los artefactos de la metodología SXP	Satisfactorio
Cumplimiento de las plantillas establecidas para cada artefacto	Satisfactorio
Ortografía, concordancia y redacción de cada uno de los artefactos	Satisfactorio
Modelación de diagramas con el uso de UML	Satisfactorio
Trazabilidad de los requisitos funcionales y no funcionales	Satisfactorio
Efectividad de los casos de pruebas definidos en el proyecto	Satisfactorio

Ing. Maidel Ojeda Cruz
Asesora de Calidad de SW

UCI Universidad de las Ciencias Informáticas
CENTRO DE DESARROLLO
Facultad Regional ARTEMISA

Msc. Yamila Vigil Regalado
Directora Centro de Desarrollo

Anexo 2: Validación del Cliente