Universidad de las Ciencias Informáticas

Facultad 6

Desarrollo de un componente de comunicación en tiempo real sobre XMPP para un Dashboard.

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

Autores: Reynaldo Ernesto Barocela Almaguer.

Marislay Soria Pérez.

Tutores: Ing. Adonis Ricardo Rosales.

Ing. Jorge Bedoya Rusenko.

Ciudad Habana, Cuba junio 2011.

"Año 53 de la Revolución"

¨ Las ciencias no tienen calzadas, los que intenten ascender sus luminosas cumbres tendrán que recorrer caminos escabrosos.¨

Karl Marx.

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Marislay Soria Pérez.

_____

Firma del Autor

Reynaldo Ernesto Barocela Almaguer.

_____

Firma del Autor

Ing. Adonis Ricardo Rosales.

_____

Firma del Tutor

Ing. Jorge Bedoya Rusenko.

_____

Firma del Tutor

Datos de Contacto

Centro de Tecnologías de Gestión de Datos (DATEC). Dpto. de Integración de Soluciones.

http://portal.datec.prod.uci.cu

Tutores:

Ing. Adonis Ricardo Rosales.

Universidad de las Ciencias Informáticas.

La Habana, Cuba.

e-mail: arrosales@uci.cu

Ing. Jorge Bedoya Rusenko.

Universidad de las Ciencias Informáticas.

La Habana, Cuba.

e-mail: jbedoya@uci.cu

Autores:

Marislay Soria Pérez.

Universidad de las Ciencias Informáticas.

La Habana, Cuba.

e-mail: msoriap@estudiantes.uci.cu

Reynaldo Ernesto Barocela Almaguer.

Universidad de las Ciencias Informáticas.

La Habana, Cuba.

e-mail: rebarocela@estudiantes.uci.cu

De Marislay:

Agradezco a la Revolución y en especial a Fidel por darme la oportunidad de formarme como ingeniera.

A mis padres, por ser guardianes de mi bienestar y desarrollo profesional, por ese amor tan infinito que me ha hecho crecer mucho y del cual estoy muy orgullosa, le doy gracias a dios por poder contar con ellos.

A mi hermano, gracias por su ayuda, apoyo y por su amor inigualable de hermano.

A toda mi familia, por preocuparse por mí.

A mi novio por cuidarme y brindarme todo su amor y cariño, porque sin él no hubiese llegado hasta donde estoy, te mereces un pedacito del resultado de esta obra por estar siempre ahí cuando te necesito. Te quiero mucho.

A todos los profesores que en estos cinco años me brindaron los conocimientos y consejos necesarios para crecer profesionalmente.

A mis compañeros y amigos, a los que están y a los que no, por su paciencia, ayuda y dedicación en los momentos más difíciles.

A mi compañero de tesis, por la paciencia y confianza, sin él no hubiera logrado este triunfo.

A mis tutores por guiarme en este trabajo.

A todos los que contribuyeron amable y desinteresadamente con la realización de este trabajo, con su tiempo, conocimiento y experiencia.

A todos....Gracias

De Reynaldo:

Quisiera agradecer primeramente a esta universidad por haber permitido formarme como mejor persona y brindarme tantos conocimientos.

A mi familia por siempre preocuparse por mí y por mis resultados, este es el reflejo de todo el apoyo que me han brindado, de forma especial a mis tías, todas en general que siempre han sido como madres para mí.

A mi mamá por ser una mujer admirable, y por sobre todas las cosas la mejor madre del mundo, por apoyarme siempre, ayudarme y estar pendiente de todo, de no ser por ese apoyo hoy no estaría aquí mima graduándome de ingeniero como siempre quisiste. Eres mi ejemplo e inspiración. Te quiero mucho mima.

A mi abuela del alma que siempre ha estado pendiente de mi pese a su avanzada edad, este es el fruto de todo el cariño que sembraste en mi, te agradezco tu comprensión, amor y preocupación. Te quiero mucho abuelita.

A mi novia Dayné Gutierrez González por ser tan preocupada y ayudarme tanto cuando más lo necesitaba, por siempre estar ahí en los momentos malos y en los buenos, por comprenderme y aceptarme como soy, de no ser por tu ayuda y preocupación hoy no estuviera aquí mi amor. Te amo.

A mis tías Mercy y Marta por su ayuda y estar siempre pendiente de mi, en especial a tía Marcia por ser como una madre, gracias por tanto cariño y dedicación un beso bien grande.

A mi suegra Niurka por haberme acogido como un hijo más en su familia, también has sido parte de este sueño, te lo agradezco mucho, un beso.

A mis hermanas que siempre me han apoyado, las quiero mucho.

A mis compañeros que siempre han estado presentes en todo momento desde el primer día, mencionarlos todos se me hace imposible porque son muchos pero si quisiera mencionar algunos que siempre me han acompañado desde cerca, un agradecimiento especial a: Dino, Juan José, Osdel, Jorgito, Héctor, El Nani, Flavio, Henry, Montano, El Yaspa, Freddy, Gendris, Luigi, Omar, Yasel, Alejo, El tío, Pedrito, Boza, Elio, Yordania, Lili, Annis, Patricia, Yani, Maybel, Elianis y muchos más que han sido parte de este logro. Siempre estarán en mi corazón.

A mi piquete La Krema que más que amigos ya se han convertido en hermanos, a ustedes que siempre han estado en las buenas y en las malas, de los que tanto he aprendido y que siempre me han apoyado un agradecimiento especial para, el Rodo, el Yoyo y el Cabezón (Dariel), siempre estarán en mi corazón los quiero como se quiere a un hermano.

A mi dúo de tesis Marislay por comprenderme y aguantarme durante la realización de este trabajo, créeme que te considero porque yo sé que soy insoportable cuando las cosas son en serio. Te agradezco mucho que hayas compartido tantas horas de trabajo a mi lado.

Al colectivo de trabajo del laboratorio que siempre me ayudó cuando hiso falta, en especial a Garnache, Tico, Lobo, Diana, Diole y Rachid, muchas gracias de corazón he aprendido mucho de ustedes, siempre los voy a recordar.

A mi arquitecto Georvys que construyó el cimiento de este trabajo. Muchas gracias por compartir conmigo tantas horas de sacrificio.

De forma especial también quisiera agradecer a los dos tutores que me han acompañado y guiado en el transcurso de este trabajo por ser siempre tan profesionales, guiarme y estar para todo lo que hiso falta, sin su ayuda hoy no estuviera parado aquí en frente de todos diciendo estas palabras.

Al colectivo de profesores que de una forma u otra tuvieron que ver con mi desempeño como estudiante también lleguen mis agradecimientos, quisiera hacerlo de forma especial a Pedro Elennis, Yusdenis, Yanelis, Yosdenis, Yanet y Asnay.

A mis amigos del barrio y mis vecinos que siempre han estado al tanto de mis estudios, siempre los tuve presente, en especial a Tamara, Robertico, Mariela, Héctor, Norma, Ricardo, Gloria, Isabel, Sonia, Tania, Manolo, Miguelito, Maykel, Alicia, Leonides, Liuver, Osmany, Edilio, Fofi y todos los que de una forma u otra han sido parte de este logro.

De Marislay:

Dedico este trabajo de manera especial a mis padres por todo lo que significan para mí porque sin ellos estoy segura, no habría logrado realizar cada uno de mis sueños, gracias por su amor incondicional y por su incansable apoyo a lo largo de todos estos años.

A mi hermano que ha sido siempre mi ejemplo a seguir y mi inspiración para continuar.
A mi novio por lo especial y maravilloso que ha sabido ser conmigo, por brindarme tanto amor y comprensión, gracias por ser paciente y ayudarme tanto en todo.

A mi familia...

A todos mis amigos...

A todos los que de una forma u otra me han ayudado a llegar hasta aquí, por su confianza, su apoyo, su amor.

De Reynaldo:

A mi padre que hoy no se encuentra entre nosotros pero sé que estaría muy orgulloso de mí.

A mi mamá por ser mi ejemplo a seguir.

A mi abuelita por siempre estar pendiente de mí y quererme tanto.

A mi novia por su apoyo y ser tan especial.

Resumen

A partir de la necesidad que surge en los proyectos desarrollados por el Centro de Tecnologías de Gestión de Datos (DATEC), conjuntamente con la facultad 6, de la presencia de un componente Dashboard para lograr la comunicación y visualización de indicadores en tiempo real, nace la presente investigación; en la misma se propone, fundamenta y describe el desarrollo de una aplicación con el objetivo de lograr un mejoramiento en los procesos de toma de decisiones de diferentes tipos de empresas.

Este trabajo tiene como objetivo fundamental: desarrollar un componente de comunicación en tiempo real y que a su vez cumpla con los requerimientos funcionales y no funcionales exigidos por el usuario. Con el desarrollo de esta aplicación, la cual constituye un sistema que permite mantener conectadas dos entidades a la escucha de cualquier variación de datos en la entidad servidora, se propone mejorar la comunicación en tiempo real en aras de agilizar el proceso de toma de decisiones en cualquier organización.

Para el desarrollo de la solución se emplea el lenguaje PHP, las librerías: Dojo, Strophejs y Sixties; además de los protocolos de comunicación XMPP y BOSH.

PALABRAS CLAVE: Tecnologías, DATEC, indicadores, comunicación, decisiones, XMPP.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Introducción

Actualmente, para la puesta en marcha de cualquier estrategia que facilite la conquista de los objetivos trazados y que agilice el proceso de toma de decisiones, se hace imprescindible el empleo de las Tecnologías de la Información y las Comunicaciones (TIC). Por tal motivo se ha convertido en un elemento fundamental para el desarrollo en las diferentes unidades o departamentos de cualquier organización.

Con el avance y desarrollo que poseen las TIC hoy día, el control de gestión tradicional no es suficiente en los entornos actuales altamente competitivos, ya que los parámetros muestran sólo resultados de actividades pasadas. Ante la necesidad de completar la perspectiva financiera tradicional de medición del éxito de las organizaciones, surge el Cuadro de Mando Integral (CMI) o *Balancead Scorecard* como también es conocido en su versión en inglés. Este posibilita que la visión y la estrategia se conviertan en objetivos, indicadores estratégicos, metas e iniciativas donde su despliegue se lleve a cabo a través de otras perspectivas además de la financiera. Un componente fundamental de un CMI es un *Dashboard* (DB) el cual es un conjunto de gráficos con una interfaz de usuario encargada de mostrar una variación cuantitativa de los diferentes datos utilizados en las empresas.

Cuba, país bloqueado y en vías de desarrollo, lleva a cabo un proceso de informatización de todos los sectores de la nación, sin embargo en la actualidad aún existen un gran número de instituciones que se encuentran envueltas en un gran flujo de información, la que en su mayoría no aparece en formato digital, lo que provoca cierta lentitud en el tratamiento de la misma.

Paralelamente a esto, surge en el año 2002 por idea del Comandante en Jefe Fidel Castro, la Universidad de Ciencias Informáticas (UCI) en la cual se encuentra la facultad 6, que conjuntamente con el Centro de Gestión de Tecnologías de Datos (DATEC) desarrollan diversos proyectos ejemplo de estos son: la Sala Situacional UCI y SiGOB (Sistema de Información de Gobierno). En estos proyectos se necesita de la presencia de un componente DB para lograr la comunicación y visualización de indicadores en tiempo real. Por lo que se ha adoptado la solución que brinda Pentaho BI Suite 3.5 *Community Edition*. Esta versión no cuenta con un diseñador de DB, ni da soporte a la visualización de Indicadores de Rendimiento Clave (KPIs del inglés *Key Performance Indicators*) en tiempo real, lo cual representa un problema para la representación gráfica de la información, y requiere mayor esfuerzo para el diseño y comprensión de la misma.

Cuando se habla de actualización en tiempo real, se refiere a aquellos KPIs de los cuales se necesita conocer sus valores y monitorizar su comportamiento constantemente, sin la intervención del usuario. Esto daría a los responsables de la toma de decisiones, información actualizada de los procesos que se desarrollan en la empresa, y una acertada base sobre la cual tomar las decisiones. Por los motivos explicados surge la necesidad de una tecnología que soporte la comunicación en tiempo real, donde las fuentes gráficas se restauren inmediatamente después que sean actualizados los datos, sin que el usuario intervenga (1).

Por lo antes mencionado se define como **problema científico de la investigación:** ¿Cómo lograr la implementación de comunicación y visualización gráfica de indicadores claves en tiempo real sobre la Web?

Siendo el **objeto de estudio:** La comunicación en tiempo real y representación gráfica sobre la Web, enmarcado en el **campo de acción:** Comunicación de indicadores claves en tiempo real para un *Dashboard* en la Web.

Se define como **objetivo general:** Desarrollar un componente de comunicación en tiempo real.

Definiendo los siguientes **objetivos específicos:**

- Realizar un estudio acerca de las principales tendencias de los *DashBoard.*
- Diseñar un componente de comunicación en tiempo real.
- Implementar el componente diseñado.
- Representar gráficamente la información haciendo uso del componente implementado.
- Validar el funcionamiento del componente de comunicación en tiempo real.

Para resolver el problema planteado y cumplir así los objetivos específicos, se definen las siguientes **tareas de la investigación:**

- Revisión de las librerías manejadoras del protocolo de comunicación seleccionado.
- Análisis del dominio de la solución.
- Definición de los requisitos funcionales del componente de comunicación.
- Definición de los requisitos no funcionales del componente de comunicación.
- Realización de los diagramas de clases del diseño correspondiente al componente de comunicación.
- Realización de los diagramas de secuencia del diseño correspondiente al componente de comunicación.

- ❖ Realización del diagrama de componentes correspondiente al componente de comunicación.
- ❖ Realización de la conexión de los KPI con los gráficos utilizando el componente implementado.
- ❖ Validación de la persistencia de los datos de la conexión haciendo uso de los gráficos.
- ❖ Realización de las pruebas de desarrollador, de sistema y de integración una vez concluido el componente implementado y el análisis de sus resultados.

Estructura de la Tesis

El presente documento se encuentra estructurado de la siguiente manera: resumen, introducción y 3 capítulos, que contienen todo lo relacionado con el trabajo investigativo, la descripción de la solución propuesta, la implementación del sistema, así como la realización de las pruebas y los resultados arrojados por estas. Además presenta también las conclusiones, las recomendaciones, los anexos y el glosario de términos que se incluyen en la realización del mismo.

En el capítulo 1 **Fundamentación Teórica de la Investigación** se presenta el estudio del estado del arte, el cual comprende un análisis de los sistemas que existen en la actualidad a nivel Nacional e Internacional que se vinculan con la investigación. Además se analizan las tecnologías, metodología y herramientas propuestas que apoyan la solución del problema a resolver.

En el capítulo 2 **Análisis y Diseño de la Solución** se abordan aspectos concernientes a la descripción de la solución, se especifican los requerimientos funcionales y no funcionales, se elaboran diagramas como: el Diagrama de Casos de Uso del Sistema, Modelo de Dominio y Clases del Diseño. Además se muestra la interacción entre los actores y el sistema mediante los Diagramas de Secuencia.

Finalmente en el capítulo 3 **Implementación y Pruebas** se describe la implementación de la aplicación, a través de la descripción de algunos elementos importantes del sistema, también se reflejan el diagrama de componentes y el diagrama de despliegue y se realiza la validación general del sistema a través de las pruebas de funcionalidad y rendimiento, utilizando los niveles de Unidad, Integración y Sistema.

Capítulo 1: Fundamentación Teórica de la Investigación

Introducción

Durante el análisis del presente capítulo se realiza un estudio enmarcado en las soluciones existentes tanto a nivel Internacional y Nacional que se vinculan con la investigación. Se realizará además un estudio del estado del arte sobre las tecnologías, metodología y herramientas propuestas por el arquitecto del proyecto que se utilizan para dar solución al problema planteado.

1.1 Surgimiento del Cuadro de Mando Integral

Según Antonio Dávila: “Durante los años sesenta se comenzó a utilizar una herramienta llamada *Tableau de Bord* o Tablero de Mando que incorporaba diversos ratios para el control financiero de la empresa. Con el paso del tiempo, esta herramienta ha evolucionado y combina no solo ratios financieros, sino también indicadores no financieros que permiten controlar los diferentes procesos del negocio. La idea de utilizar un conjunto de indicadores para obtener información de gestión es un antecedente que recoge el CMI” (2).

El concepto de CMI fue presentado en el número de Enero/Febrero de 1992 de la revista *Harvard Business Review*, en base a un trabajo realizado para una empresa de semiconductores *Analog Devices Inc.* Sus autores, Robert S. Kaplan y David P. Norton, plantean que: “El CMI es un sistema de administración o sistema administrativo que va más allá de la perspectiva financiera con la que los gerentes acostumbran evaluar la marcha de una empresa” (3).

El CMI surge ante la necesidad de completar la perspectiva financiera tradicional de medición del éxito de las organizaciones. De tal manera que la visión y la estrategia se conviertan en objetivos, indicadores estratégicos, metas e iniciativas y que el despliegue se lleve a cabo a través de otras perspectivas además de la financiera, habiendo un esquema integrado de seguimiento y mejora.

La idea del CMI es sencilla y transparente, reconoce que la finalidad para conseguir beneficios, es el resultado de una cadena de causas y efectos que suceden en cuatro ámbitos: financiero, cliente, procesos internos, aprendizaje y crecimiento.

El CMI es un sistema de gestión y control de gestión que debe de ser utilizado como un sistema de comunicación, de información y de formación. Constituye además, una herramienta de gerencia que permite traducir la estrategia de una organización en un conjunto completo de medidas de desempeño, tal que, informa a la alta gerencia sobre como la organización avanza hacia el logro de sus objetivos (4).

1.1.2 Soluciones existentes a nivel mundial

A nivel mundial existen, diversos sistemas informáticos que facilitan a las organizaciones cómo implantar un CMI, la mayoría de estas herramientas en la actualidad son de carácter privativo y no utilizan comunicación en tiempo real. Algunos de estos programas tienen como objetivo proporcionar informes claros, concisos y relevantes sobre la marcha de la organización; entre ellos se encuentran:

Decide Soft: conocido como el primer software en español para la completa realización e implantación del CMI. Entre sus principales ventajas se encuentran (5):

- **Flexibilidad:** dispone de varias opciones para visualizar la información (varios modelos de gráficos, informes e impresos).
- **Fácil de utilizar:** sus menús intuitivos permiten que incluso una persona no experta en informática se beneficie de toda la capacidad del mismo.
- **Integración de la información global:** permite integrar la información más específica de áreas, objetivos e indicadores con una visión más global de su empresa, resultados financieros y no financieros.

La principal desventaja del Decide Soft consiste en que es un software privativo el cual es comercializado, creado y diseñado por empresas españolas (5).

Delphos: software de Control de Gestión que permite implementar íntegramente un CMI, un Plan Estratégico, un Plan Anual Operativo o cualquier otro modelo que se requiera para controlar las operaciones de cualquier tipo de organización, inclusive mejorar el desempeño y la productividad (6). Es dirigido principalmente a usuarios no informáticos, que lo convierte en una herramienta muy sencilla de implementar y utilizar (7).

Delphos integra en una sola aplicación (6):

- Administración estratégica (objetivos, indicadores y responsables).
- Administración de Proyectos.

- ❖ Visualización de información mediante cubos o modelos multidimensionales.
- ❖ Graficador, Reportador, Presupuesto Gerencial.

La principal desventaja del mismo es que no utiliza la comunicación en tiempo real.

Pentaho: *Pentaho Dashboards* proporciona información inmediata sobre el rendimiento individual, departamental, o de la empresa. Mediante la entrega de claves métricas en una interfaz visual atractiva e intuitiva, los cuadros de mando Pentaho ofrecen a los usuarios de los negocios la información crítica que necesitan para comprender y mejorar el desempeño organizacional (8).

Además constituye una potente herramienta que cuenta con las siguientes características (9):

- ❖ Identificación de métricas clave KPIs mediante la generación de Monitoreo/Métricas.
- ❖ Realización de investigaciones de detalles subyacentes, con reportes de soportes.
- ❖ Ejecución de seguimientos de excepciones, permitiendo pre-establecer alertas basadas en reglas del negocio.
- ❖ Permite la incorporación de múltiples tipos de gráficos, tablas y velocímetros a un determinado proyecto de Inteligencia de Negocio (del inglés *Business Intelligence*).
- ❖ Portal de la integración para hacer más fácil la entrega de métricas relevantes para el negocio a un gran número de usuarios, perfectamente integrado en su aplicación.
- ❖ Integrado de alerta para controlar continuamente las excepciones y notificar a los usuarios a tomar medidas.

Este software a pesar de ser una herramienta libre en su versión 3.5 actualmente utilizada en la universidad no cuenta con un diseñador de DB, ni da soporte a la visualización de los KPIs en tiempo real, por lo que no se considera una solución óptima.

El CMI es el más popular de los modelos de medición estratégica, debido a que proporciona a los directivos de las empresas u organizaciones un conjunto de herramientas o procedimientos que los ayudan en el sistema de medición del desempeño de la empresa. Con el objetivo de lograr un mejor funcionamiento del mismo, es necesario el empleo de los sistemas de comunicación en tiempo real.

1.2 Surgimiento de los sistemas de comunicación en tiempo real

La comunicación sincrónica o comunicación en tiempo real ha evolucionado a lo largo del tiempo. En un primer momento la herramienta de comunicación instantánea por excelencia era

el chat el cual consiste en un programa o aplicación Web que permite la comunicación con otras personas en tiempo real mediante mensajes de texto (10).

La comunicación sincrónica está caracterizada por una serie de rasgos que la hacen peculiar y que habitualmente no están presentes en la comunicación presencial, estos son:

- **Independiente del lugar**: la comunicación se produce entre dos o más personas que pueden encontrarse físicamente ubicados en contextos distintos, e incluso pueden compartir el mismo espacio. Por ejemplo, los usuarios de un chat podrían estar presentes en un mismo lugar, como por ejemplo en un aula de informática con conexión en red; o bien, por el contrario, podrían estar distanciados al vivir en ciudades diferentes.
- **Temporalmente dependiente**: esto quiere decir que para que este tipo de comunicación tenga lugar, es necesario que los comunicantes coincidan en un mismo tiempo (11).

La principal característica que distingue a los Sistemas de Tiempo Real (STR) de otros, es el tiempo de interacción, y es por esto que Alan Burns y Andy Wellings presentan la siguiente definición de STR: “Un sistema en Tiempo Real es cualquier sistema donde el tiempo en que se produce su salida es significante. Esto es debido a que generalmente la entrada corresponde a algún instante del mundo físico y la salida tiene relación con ese mismo instante. El retraso transcurrido entre la entrada y la salida debe ser lo suficientemente pequeño para considerarse una respuesta puntual” (12).

1.2.1 Actualidad y tendencia de los sistemas de comunicación en tiempo real

Varios son los sistemas que utilizan a nivel mundial la comunicación en tiempo real, ejemplo de estos son:

- **Clientes de mensajería instantánea:**

  La mensajería instantánea requiere el uso de un cliente de mensajería instantánea que realiza el servicio y se diferencia del correo electrónico en que las conversaciones se realizan en tiempo real.

  Los sistemas de mensajería tienen unas funciones básicas aparte de mostrar los usuarios que hay conectados y chatear. Unas son comunes a todos o casi todos los clientes o protocolos y otras son menos comunes entre estas se pueden citar (13):

- ✓ Registrar y borrar usuarios de la lista de contactos propia.
- ✓ Mostrar varios estados como son: disponible, ocupado, lejos del computador, invisible.
- ✓ Se puede usar un avatar: una imagen o fotografía tipo ícono.
- ✓ Compartir recurso: envío de archivos.

Usados muy frecuentemente en el mundo de la informática y particularmente en la UCI se ha convertido en una herramienta fundamental para la comunicación y el trabajo colaborativo, entre estos se pueden mencionar: Pidgin, PSI, Coccinella Messenger, Neos, Pandion.

❖ **Whisbi:**

Es una aplicación empresarial para la realización de Video llamadas Instantáneas que permiten a los usuarios establecer una comunicación más cercana con las empresas en tiempo real, y a estas a su vez, permite conocer de forma directa a los clientes, sus intereses y necesidades (14).

Entre sus principales características se encuentran (14):

- ✓ El usuario no necesita ni altavoces ni micrófonos, ya que la comunicación se realiza a través de su teléfono fijo; tampoco necesita una Webcam porque el operador, del lado de la empresa nunca verá al cliente, ni tendrá que descargar ningún software específico.
- ✓ Según encuestas en las compañías que lo utilizan, se demuestra que el software es completamente seguro y que mejora de manera extraordinaria los ratios de conversión.

❖ **ICR (Internet Relay Chat):**

El IRC conocido popularmente como chat, se trata de un medio de comunicación que permite realizar conversaciones en tiempo real a través de Internet, aunque normalmente se utiliza para conversar en línea, también sirve para transmitir ficheros. Es generalmente utilizado en actividades virtuales de enseñanza y aprendizaje, y para acceder al mismo, es necesario, en primer lugar, ejecutar el programa cliente de IRC y posteriormente conectar con algún servidor que ofrezca dicho servicio (15).

Este software permite entre sus principales ventajas (15):

- ✓ Aplicar metodologías de enseñanza y aprendizaje, por parte de un facilitador.
- ✓ Practicar destrezas o estrategias con el moderador y con otros participantes.
- ✓ Evaluar el aprendizaje de los alumnos.
- ✓ Hacer discusiones de un material dado.
- ✓ Hacer actividades en grupos cooperativos.
- ✓ Propiciar consultas individuales o grupales con el moderador.
- ✓ Realizar una prueba interactiva en línea de un tema determinado.

Estas herramientas no utilizan la comunicación en tiempo real unido al CMI, sus funciones y características están diseñadas para lo que fueron concebidas cada una de ellas, siendo imposible adaptarlas a las necesidades del proyecto. Además son de un carácter privado, siendo esta una de sus principales desventajas. Debido a lo antes expuesto se hace necesario realizar una selección de las tecnologías a utilizar para la realización de la solución a este problema.

1.3 Herramientas, protocolos y estándares

Las herramientas seleccionadas para la realización de este sistema fueron definidas por la arquitectura del proyecto, las cuales están descritas con mayor especificidad en el Trabajo de Diploma “Definición de la arquitectura de un componente de comunicación de datos sobre la Web en tiempo real”, por lo cual en el presente documento solo se hace mención a las mismas.

- ❖ Metodología de desarrollo de software: Proceso unificado abierto (OpenUp/Basic).
- ❖ Lenguaje de modelado: UML (*Unified Modeling Language*).
- ❖ Herramienta de modelado: Visual Paradigm en su versión 6.1.
- ❖ Lenguaje de Programación: PHP (*Professional Home Page Tools*) en su versión 5.0.
- ❖ Sistema de control de versiones: Subversion.
- ❖ Entorno integrado para el desarrollo: NetBeans en su versión 6.9.
- ❖ Protocolo para facilitar la comunicación en tiempo real: XMPP.

- Para establecer la conexión a un servidor XMPP desde HTTP: BOSH (*Bidirectional streams Over Synchronous* HTTP).
- Librerías: Dojo en su versión 1.3, Strophe en su versión 1.0.1 y Sixties.
- Servidor de mensajería instantánea: Ejabberd en su versión 2.1.5.
- Servidor Web: Apache en su versión 2.0.

1.4 Pruebas de software

Para contribuir con la calidad del software es recomendable que el producto software vaya siendo evaluado a medida que se va construyendo. Posteriormente, se hace necesario llevar a cabo, paralelo al proceso de desarrollo, un proceso de evaluación y comprobación de los distintos productos o modelos que se van generando, en el que participarán desarrolladores y clientes.

Las pruebas son una actividad en la cual un sistema o componente es ejecutado, bajo unas condiciones o requerimientos específicos, donde los resultados son observados y registrados, para una posterior evaluación de algún aspecto del sistema o componente (16).

Los niveles en que se prueba, las técnicas y los métodos, no se pueden analizar de forma independiente, ya que dichos términos se encuentran estrechamente vinculados, por lo que no se realiza una distinción específica de las técnicas en un nivel u otro, sino que se ven de manera complementada.

Para la realización de las pruebas al componente desarrollado, se decide utilizar los niveles de Unidad, Integración y Sistema, empleando las técnicas de funcionalidad y rendimiento. Haciendo uso del método de caja negra y empleando la técnica de Partición de Equivalencia, los cuales se explican a continuación. Además, como herramientas son seleccionadas para su uso: los casos de pruebas funcionales y la automatización de las pruebas con el software JMeter.

Niveles de prueba seleccionados

La prueba es aplicada para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo, dentro de los que se selccionaron (16, pág 9) :

- **Prueba de unidad:** enfocada a los elementos testeables más pequeños del software. Es aplicable a componentes representados en el modelo de implementación para verificar

que los flujos de control y de datos están cubiertos, y que ellos funcionen como se espera. La prueba de unidad siempre está orientada a caja blanca.

- **Prueba de integración**: es ejecutada para asegurar que los componentes en el modelo de implementación, operen correctamente cuando son combinados para ejecutar un caso de uso. Estas pruebas, descubren errores o incompletitud en las especificaciones de las interfaces de los paquetes. Es el proceso de combinar y probar múltiples componentes juntos. El objetivo es tomar los componentes probados en unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño.
- **Prueba de sistema:** son las pruebas que se hacen cuando el software está funcionando como un todo. Es la actividad de prueba dirigida a verificar el programa final, después que todos los componentes de software y hardware han sido integrados. En un ciclo iterativo estas pruebas ocurren más temprano, tan pronto como subconjuntos bien formados de comportamiento de caso de uso son implementados.

Técnicas de prueba seleccionada

Cada nivel de prueba engloba una técnica de prueba específica. Por lo que se escogen para su aplicación basado en dimensiones de calidad las siguientes técnicas (16, pág. 13) :

- **Funcionalidad**
  - ✓ **Función:** pruebas fijando su atención en la validación de las funciones, métodos, servicios y caso de uso.
- ***Performance*** **(Rendimiento)**
  - ✓ **Carga:** usada para validar y valorar la aceptabilidad de los límites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo prueba, permanece constante. La variación en carga es simular la carga de trabajo promedio y con picos que ocurren dentro de tolerancias operacionales normales.
  - ✓ ***Performance profile*** **(Perfil de desempeño)**: son pruebas enfocadas a monitorear el tiempo en flujo de ejecución, acceso a datos, en llamada a funciones y sistemas para identificar y direccionar los cuellos de botellas y los procesos ineficientes.
  - ✓ **Contención:** encaminada a la validación de las habilidades del elemento a probar para manejar aceptablemente la demanda de múltiples actores sobre un mismo recurso.

Método de prueba a utilizar

Existen métodos de prueba independientemente de la técnica que se utilice o el nivel en que se enmarquen estas. Estos métodos proporcionan distintos criterios para generar casos de prueba que provoquen fallos en los programas, a continuación se explica el método seleccionado para su aplicación:

- **Método de Caja Negra o Funcional:** se realizan pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas de dicho programa. No es necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar (17). Para confeccionar los casos de prueba de Caja Negra existen distintos criterios tal es el caso de la técnica de partición de equivalencia la cual se explica a continuación:
  - ✓**Técnica de la partición de equivalencia:** consiste en dividir el campo de entrada de un programa en variables con juegos de datos de entrada y salida. En esencia, esta técnica intenta dividir el dominio de entrada de un programa en un número finito de variables de equivalencia. De tal modo que se pueda asumir razonablemente, que una prueba realizada con un valor representativo de cada variable, es equivalente a una prueba realizada con cualquier otro valor de dicha variable (16, pág. 16).

Conclusiones parciales

En este capítulo se han abordado temas relacionados con el objeto de investigación, tendencias, metodologías y tecnologías más utilizadas en el mundo del CMI y los STR, con el objetivo de lograr la comunicación y visualización de indicadores en tiempo real, para el mejoramiento del proceso de toma de decisiones de diferentes tipos de empresas. El estudio de herramientas de Cuadro de Mando Integral y de los sistemas que utilizan comunicación en tiempo real, evidenció que ninguna cumple con las características requeridas, por lo que se hace necesaria la creación de un nuevo sistema para la comunicación y visualización de KPIs en tiempo real. Las herramientas escogidas para su empleo en la solución constituyen una selección del arquitecto del proyecto. Estas apoyan el desarrollo general de la aplicación, brindando facilidades a los desarrolladores en cuanto a rapidez, facilidad de uso y variedad de funciones, sin dejar de mencionar que son sistemas o tecnologías libres. Se decidió además, una vez terminado el componente realizarle pruebas, en los niveles de unidad, integración y

sistema, empleando las técnicas funcional y de rendimiento y aplicando como método el de caja negra. Como herramienta los casos de pruebas funcionales y el JMeter para la realización de las pruebas de carga y stress.

Capítulo 2: Análisis y Diseño de la Solución

Introducción

En este capítulo, con el objetivo de lograr una mejor comprensión de la funcionalidad e integralidad del sistema, se da cumplimiento al objetivo específico correspondiente al diseño del componente a desarrollar. El cual se encuentra vinculado con las siguientes tareas de la investigación: análisis del dominio de la solución; definición de los requisitos funcionales del componente de comunicación; definición de los requisitos no funcionales del componente de comunicación; realización de los diagramas de clases del diseño correspondiente al componente de comunicación; además de la realización de los diagramas de secuencia del diseño correspondiente al componente de comunicación.

2.1 Características del sistema

2.1.1 Descripción del componente a desarrollar

El componente a desarrollar tiene entre sus objetivos fundamentales establecer la comunicación en tiempo real y la visualización gráfica de información. Entre las principales opciones que el mismo brindará se destacan:

- Conectar y desconectar del servidor: permite al usuario realizar diversas operaciones en dependencia del estado en que se encuentre ya sea conectado o desconectado, asociado a su estado se le facilitarán las funcionalidades correspondientes a sus privilegios.
- Gestionar nodos: permite al usuario gestionar toda la información correspondiente a un nodo en el servidor XMPP, entre las que se encuentran crear y eliminar nodos.
- Publicar datos: permite al usuario publicar una serie de datos en un nodo específico para el control de indicadores mediante gráficos actualizados en tiempo real.
- Obtener nodos: permite al usuario obtener el listado de nodos Pubsub existentes en el servidor.
- Suscribir a nodo: permite al usuario realizar una suscripción a un nodo Pubsub en el servidor.

- ❖ Recuperar datos: permite al usuario recuperar los datos publicados en los nodos Pubsub previo a su suscripción.
- ❖ Eliminar suscripción al nodo: permite al usuario eliminar la suscripción a un nodo y a su vez se reciben los datos que sean publicados en dicho nodo.

2.2 Modelo de dominio

Teniendo en cuenta que no se encuentran definidos de forma clara los diferentes procesos del negocio y que los mismos, pueden estar sujetos a cambios, se propone como paso inicial, la realización de un Modelo de Dominio. El cual constituye un artefacto de la disciplina de análisis, construido con las reglas de UML. Representado además como uno o más diagramas de clases conceptuales y que contiene, no conceptos propios de un sistema de software sino de la propia realidad física. Este puede ser tomado como el punto de partida para el diseño del sistema y constituye por así decirlo una primera versión del sistema a desarrollar (18) (Ver Figura 1).

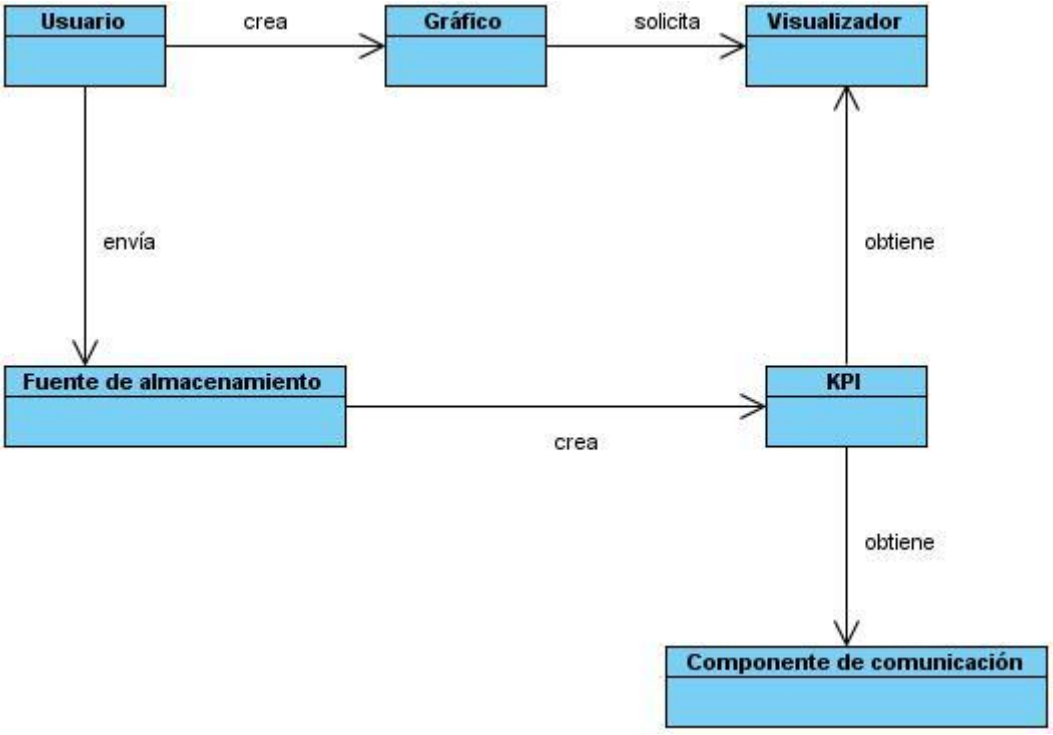


Figura 1: Modelo de Dominio

Conceptos asociados al dominio:

- **Usuario:** su función será realizada por una persona o sistema, que deberá tener en cuenta la entrada de datos indispensable para la creación de los gráficos.
- **Gráfico:** es el encargado de mostrar un conjunto de gráficos, además de realizar la actualización en tiempo real de los datos.
- **Visualizador:** gestiona la información obtenida del proceso de suscripción y la hace llegar a los gráficos, para de esta manera lograr su actualización.
- **Fuente de almacenamiento:** realiza la función de un servidor de datos y contiene además los datos necesarios para la creación y actualización de los gráficos.
- **Componente de comunicación:** es el objeto responsable de gestionar la información para la actualización del servidor Ejabberd.
- **KPI:** realiza un constante monitoreo del comportamiento de los datos y de la actualización en tiempo real, encargándose del envío de los datos.

Descripción del problema del dominio:

El objeto usuario, es el responsable de la creación de los gráficos, por lo que deberá tener en cuenta la entrada de datos a graficar, además de la dirección del KPI y la dirección de la fuente de almacenamiento. También es el encargado de enviar una serie de datos obtenidos de las distintas suscripciones efectuadas a la fuente de almacenamiento para su actualización. Igualmente, el objeto fuente de almacenamiento, constituye un servidor de datos donde se gestiona la información resultante de la creación de los nodos KPI necesarios para la realización y actualización de los gráficos.

El objeto gráfico muestra un paquete de gráficos, que son el resultado de la representación, de un conjunto de datos obtenidos de las suscripciones de los nodos realizadas en la fuente de almacenamiento. También incluye un conjunto de funcionalidades con el fin de lograr un mejor entendimiento por parte del usuario a la hora del proceso de toma de decisiones. Realiza además la actualización en tiempo real de los datos que le son enviados automáticamente de la fuente de almacenamiento haciendo uso del objeto visualizador.

El objeto visualizador obtiene los datos provenientes del KPI para posibilitar la actualización del paquete de gráficos. Por su parte el objeto KPI constituye el nodo de publicación-suscripción creado en la fuente de almacenamiento, el mismo recibe un conjunto de datos de los cuales se requiere su graficación.

El modelo de dominio ocupa un rol protagónico en el desarrollo moderno de software. Además de que puede ser tomado como el punto de partida para el diseño del mismo y utilizado como entrada en la tarea: análisis de los casos de uso. Por tal motivo se relacionan a continuación una serie de artefactos propios de los flujos de trabajo análisis y diseño.

2.3 Requerimientos y modelo del sistema

El flujo de trabajo Requerimientos, se desarrolla durante la fase de inicio, esta abarca el proceso de comunicación con el cliente y las actividades de planeación del proyecto. Al colaborar con los clientes y usuarios finales, se identifican los requisitos necesarios para determinar las funcionalidades que el sistema debe tener. Tomando como punto de partida la especificación de requisitos y la modelación gráfica de estos, agrupados en casos de uso, se brinda una visión de lo que se debe implementar.

Uno de los artefactos que se generan durante la realización de este flujo de trabajo es el Modelo del Sistema, el cual recoge la descripción de los requisitos a través de un conjunto preliminar de casos de uso, que detallan cuáles características y funciones son deseables para cada clase importante de usuarios. En general un caso de uso describe una secuencia de acciones que desarrolla un actor cuando éste interactúa con el software, ayudando además a identificar el ámbito del proyecto y proporcionando una base para la planeación del mismo (19). Este artefacto se utiliza también como entrada esencial para las actividades de análisis, diseño y prueba.

2.3.1 Técnicas de captura de requisitos

La captura de requisitos ayuda al equipo de desarrollo de un sistema a entender mejor el problema en cuya solución trabajarán. Este proceso incluye un conjunto de tareas que conducen a comprender, qué es lo que el cliente quiere y como interactuarán los usuarios finales del software (19, pág. 155). Por la complejidad que esto implica, el equipo de desarrollo empleó técnicas que permitieron hacer este proceso de una forma más eficiente y precisa.

- **Entrevistas:** resultan una técnica muy aceptada dentro de la ingeniería de requisitos y su uso está ampliamente extendido. Las entrevistas le permiten al analista tomar conocimiento del problema y comprender los objetivos de la solución buscada. A través de esta técnica el equipo de trabajo se acerca al problema de una forma natural.

Básicamente, la estructura de la entrevista abarca cuatro pasos: identificación de los entrevistados, preparación de la entrevista, realización de la entrevista y documentación de los resultados (20).

- **Tormenta de ideas:** es una técnica de reuniones en grupo cuyo objetivo es que los participantes muestren sus ideas de forma libre. Consiste en la acumulación de ideas e información sin evaluar las mismas. El grupo de personas que participa en estas reuniones no debe ser muy numeroso (máximo 10 personas), una de ellas debe asumir el rol de moderador de la sesión, pero sin carácter de controlador. Como técnica de captura de requisitos es sencilla de usar y de aplicar (21).
- **Sistemas existentes:** consiste en analizar los sistemas ya desarrollados que estén relacionados con el sistema que va a ser construido. Analizándose las interfaces de usuarios y las entradas y salidas que el mismo produce (22).

Para la realización del proceso de levantamiento de requisitos se utilizaron las técnicas: entrevista y tormenta de ideas, las cuales ayudaron a definir las funcionalidades necesarias para la elaboración del componente a implementar.

2.3.2 Especificación de los requisitos

Un requerimiento se define como un atributo necesario dentro de un sistema, que puede representar una capacidad, una característica o un factor de calidad del sistema de tal manera que le sea útil a los clientes o a los usuarios finales (23).

Los requerimientos funcionales comprenden y especifican las acciones que el sistema debe realizar. Los requerimientos no funcionales detallan las propiedades del sistema, tales como confiabilidad y seguridad (24).

A continuación se exponen los requisitos capturados para la primera versión del componente de comunicación en tiempo real a desarrollar.

Requisitos funcionales:

- **CU Conectar a servidor XMPP:**

  RF 1.1 Autenticar usuario.

  RF 1.2 Conectar al servidor XMPP.

- **CU Obtener nodos:**

  RF 2.1 Listar los nodos existentes.

- **CU Suscribir a nodo:**

  RF 3.1 Insertar nombre del servicio Pubsub.

  RF 3.2 Realizar la suscripción del nodo.

- **CU Recuperar datos:**

  RF 4.1 Recuperar datos publicados antes de la suscripción.

  RF 4.2 Visualizar conjunto de datos.

- **CU Eliminar suscripción al nodo:**

  RF 5.1 Eliminar suscripción del nodo.

  RF 5.2 Actualizar listado de nodos suscritos.

- **CU Desconectar de servidor XMPP:**

  RF 6.1 Desconectar de servidor XMPP.

- **CU Gestionar nodo:**

  RF 7.1 Crear nodo.

  RF 7.2 Editar nodo.

  RF 7.2.1 Actualizar nodo.

  RF 7.3 Eliminar nodo.

- **CU Publicar datos:**

  RF 8.1 Seleccionar los datos a publicar.

  RF 8.2 Publicar datos.

- **CU Gestionar usuario jabber:**

  RF 9.1 Crear usuario jabber.

  RF 9.2 Editar usuario jabber.

  RF 9.2.1 Buscar y visualizar usuario jabber.

  RF 9.2.2 Actualizar usuario jabber.

  RF 9.3 Eliminar usuario jabber.

- **CU Graficar datos en nodo:**

  RF 10.1 Graficar datos.

Requisitos no funcionales:

Los requerimientos no funcionales identificados durante el proceso de levantamiento de requisitos, son propuestos por el arquitecto de software, los cuales se encuentran relacionados y descritos en el Trabajo de Diploma “Definición de la arquitectura de un componente de comunicación de datos sobre la Web en tiempo real”.

2.4 Justificación de los actores del sistema

Un actor del sistema representa cualquier persona, entidad o sistema externo que interactúe con la aplicación (25). Por tal motivo, son seleccionados y descritos a continuación los siguientes actores del sistema.

Tabla 1: Descripción de los actores del sistema

Actor	Descripción
Usuario	Actor que representa a toda aquella persona que gestiona los datos con los cuales interactúa el sistema, es el responsable de las acciones: crear, eliminar ,editar y obtener un nodo; así como realizar la conexión o desconexión del servidor, recuperar los datos y publicarlos en los nodos, suscribir al nodo y eliminar la suscripción a este, además de graficar los datos.
Administrador Jabber	Actor que se vincula con los casos de uso relacionados con la administración del servidor Ejabberd instalado, tales como: crear, editar y eliminar usuario Jabber.
Sistema Externo	Actor que se relaciona con los casos de uso publicar datos y gestionar nodos de los cuales obtiene información.

2.5 Diagrama casos de uso del sistema

Los casos de uso del sistema (CUS) se utilizan para capturar los requisitos de un sistema, describiendo la interacción entre el usuario y el sistema para lograr un objetivo específico. En el desarrollo de la investigación se determinaron un total de 10 casos de uso, que se muestran y describen a continuación (Ver Figura 2):

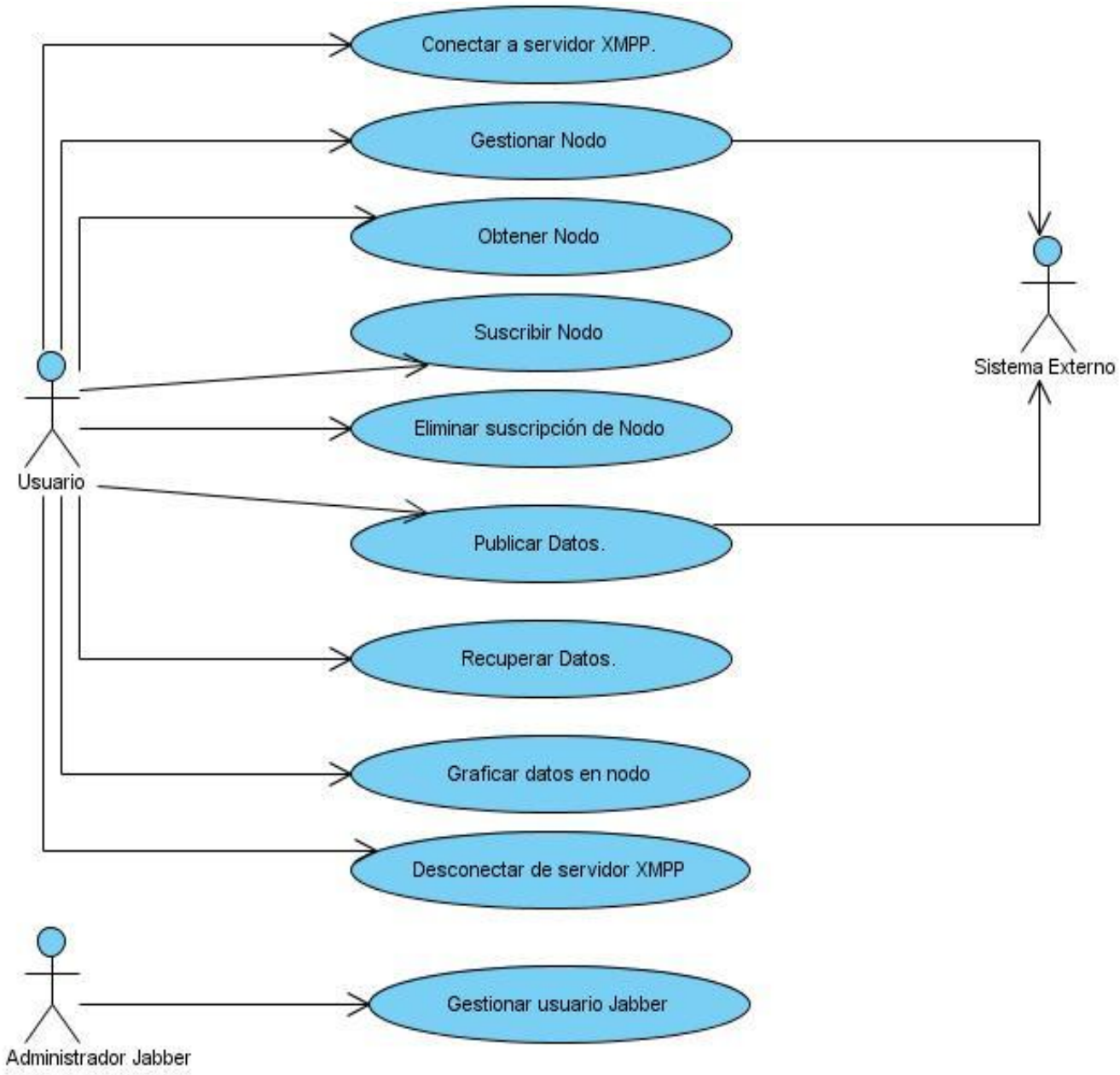


Figura 2: Diagrama de Caso de Uso del Sistema

2.5.1 Especificación de los casos de uso del sistema

A continuación se muestran de manera resumida cada una de las descripciones textuales de los Casos de Uso del Sistema, para lograr una mayor comprensión de las mismas mediante una descripción más detallada ver el expediente de proyecto.

Tabla 2: Descripción textual resumida del CU Conectar a servidor XMPP

Caso de Uso:	Conectar a servidor XMPP
Actores:	Usuario
Resumen:	El caso de uso inicia cuando se requiere establecer la conexión al servidor XMPP, por lo que se realiza un proceso de autenticación, donde el sistema muestra una interfaz en la que el usuario introduce sus datos, una vez realizado este proceso, se conecta la aplicación cliente al servidor, finalizando así el caso de uso.
Precondiciones:	El usuario debe de encontrarse registrado en el servidor XMPP.
Referencias:	RF 1.1, RF 1.2.
Prioridad:	Crítica

Prototipo de Interfaz

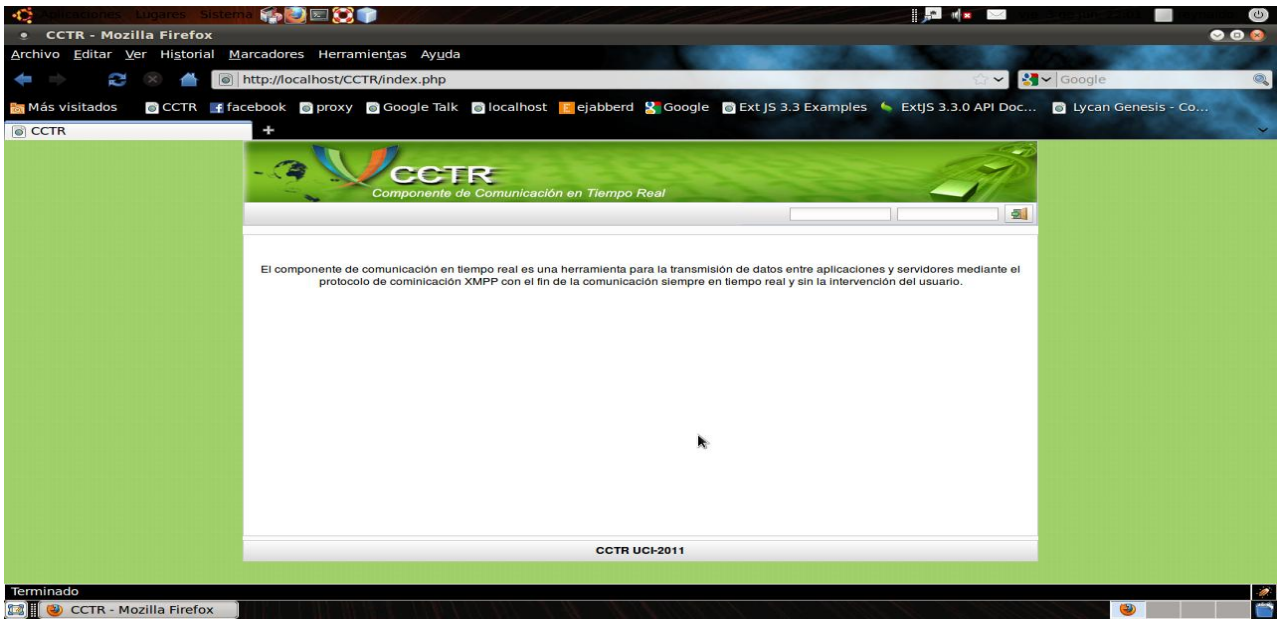


Tabla 3: Descripción textual resumida del CU Obtener Nodos

Caso de Uso:	Obtener nodos

Actores:	Usuario
Resumen:	El caso de uso inicia cuando el usuario conectado, solicita el listado de los nodos Pubsub existentes en el servidor XMPP, luego de gestionada la operación, se muestra el listado solicitado, finalizando así el caso de uso.
Precondiciones:	El usuario deberá estar conectado al servidor XMPP.
Referencias:	RF 2.1.
Prioridad:	Crítica

Prototipo de Interfaz

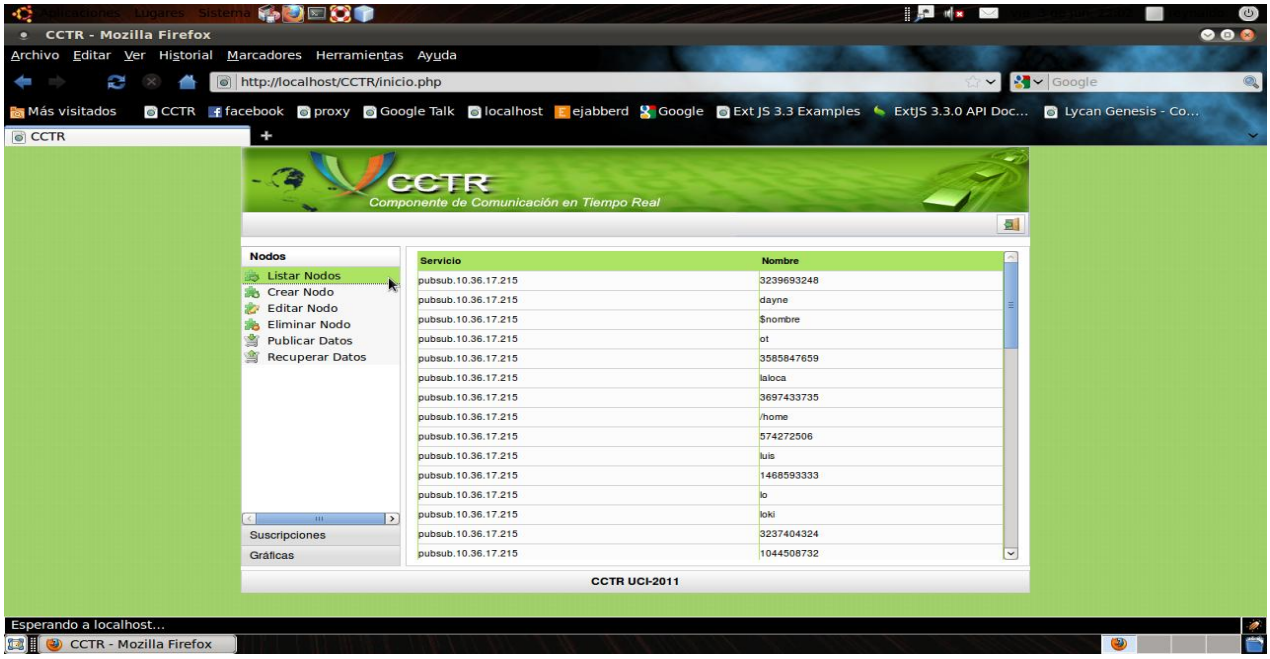


Tabla 4: Descripción textual resumida del CU Suscribir a Nodo

Caso de Uso:	Suscribir a nodo
Actores:	Usuario
Resumen:	El caso de uso inicia cuando se requiere realizar la suscripción a un nodo Pubsub, por lo que se necesita, el nombre del servicio Pubsub y el nombre del nodo, para comenzar a recibir los datos que sean publicados en el mismo, finalizando así el caso de uso.
Precondiciones:	El usuario deberá estar conectado al servidor XMPP.
Referencias:	RF 3.1, RF 3.2.

Prioridad:	Crítica

Prototipo de Interfaz

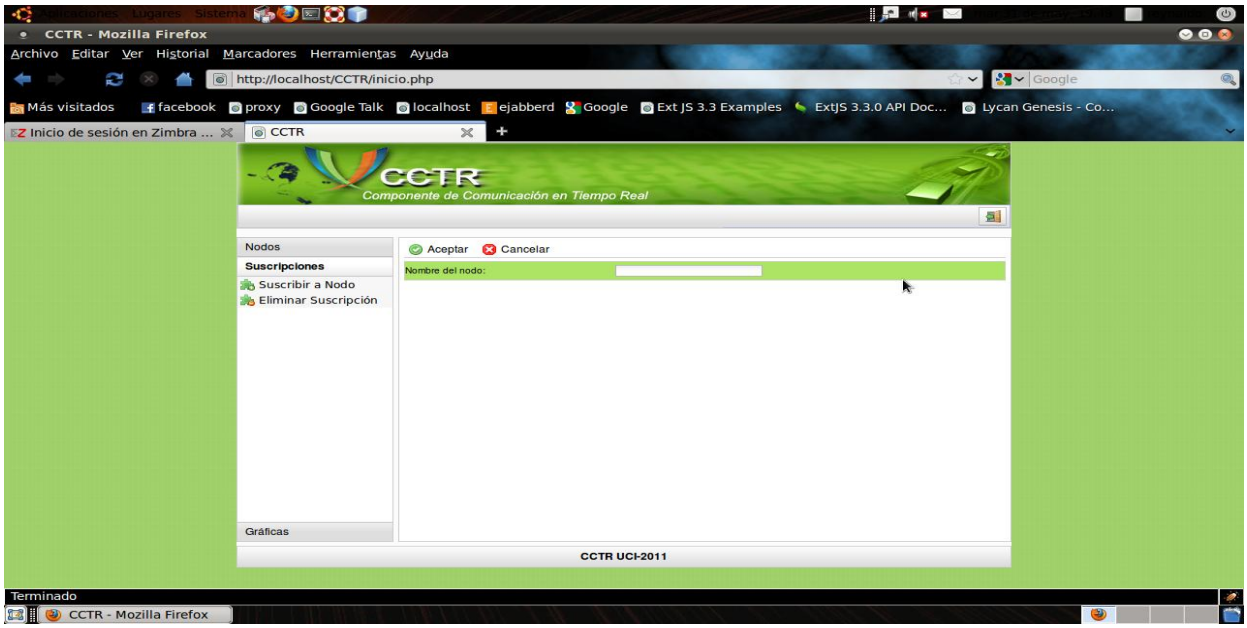


Tabla 5: Descripción textual resumida del CU Recuperar Datos

Caso de Uso:	Recuperar datos
Actores:	Usuario
Resumen:	El caso de uso inicia cuando se solicita al servidor, un conjunto de datos que hayan sido publicados con anterioridad a la suscripción de un nodo, para realizar una recuperación de los mismos, una vez atendida la solicitud, el sistema mostrará los datos solicitados por el usuario finalizando así, el caso de uso.
Precondiciones:	El usuario deberá estar conectado al servidor XMPP.
Referencias:	RF 4.1, RF 4.2.
Prioridad:	Normal

Prototipo de Interfaz

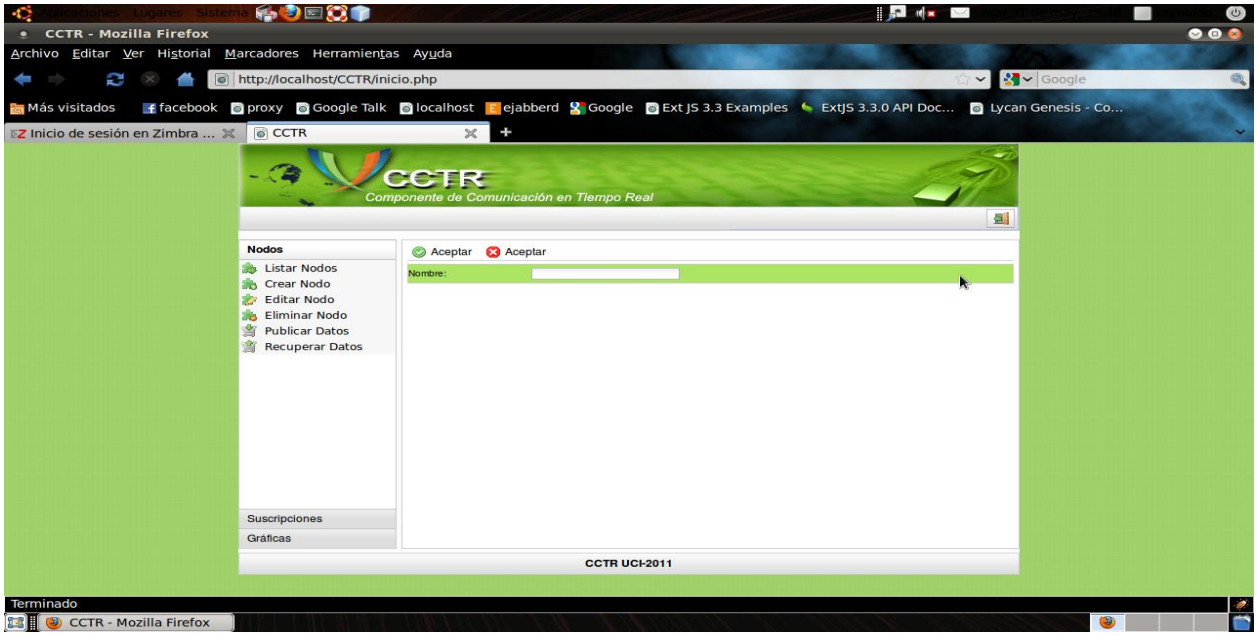


Tabla 6: Descripción textual resumida del CU Eliminar Suscripción al Nodo

Caso de Uso:	Eliminar suscripción al nodo
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el usuario desea eliminar la suscripción a un nodo, una vez efectuada la eliminación por parte del sistema, no se continuarán recibiendo los datos que sean publicados en dicho nodo, de esta forma finaliza el caso de uso.
Precondiciones:	El usuario debe encontrarse suscrito al nodo al que se quiere eliminar la suscripción.
Referencias:	RF 5.1, RF 5.2.
Prioridad:	Crítica

Prototipo de Interfaz

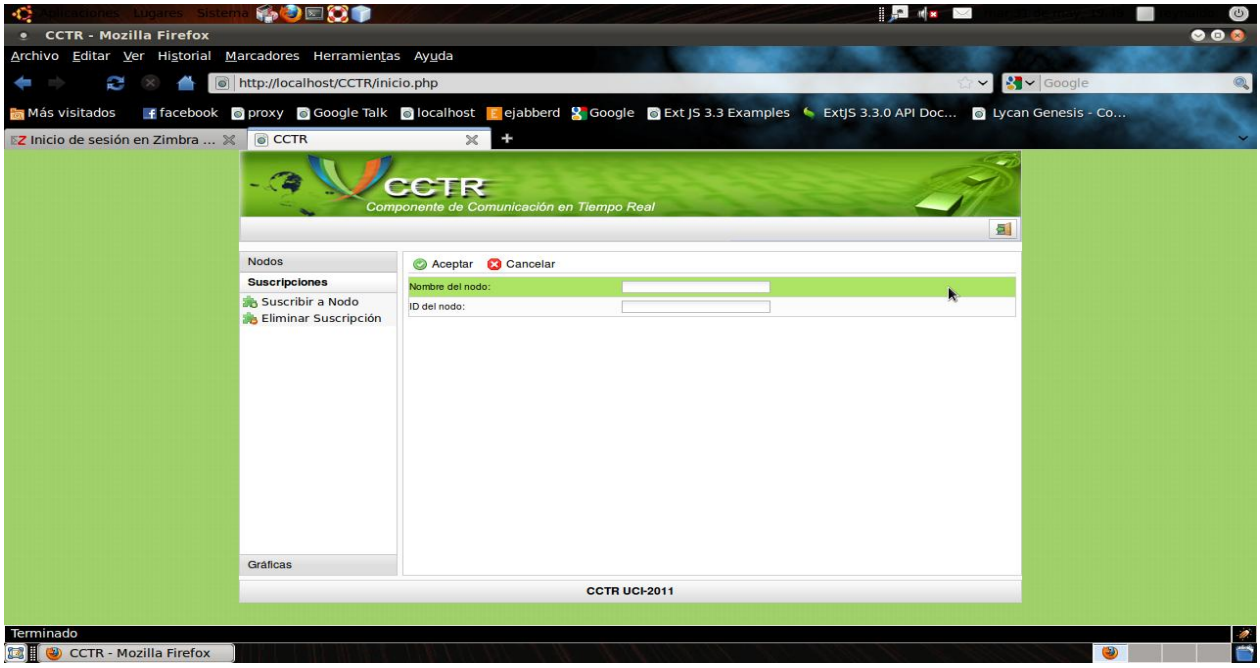


Tabla 7: Descripción textual resumida del CU Desconectar de servidor XMPP

Caso de Uso:	Desconectar de servidor XMPP
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el usuario solicita la opción de desconectarse del servidor XMPP y finaliza, una vez realizada la desconexión por parte del sistema.
Precondiciones:	El usuario deberá estar conectado al servidor XMPP.
Referencias:	RF 6.1.
Prioridad:	Normal

Prototipo de Interfaz

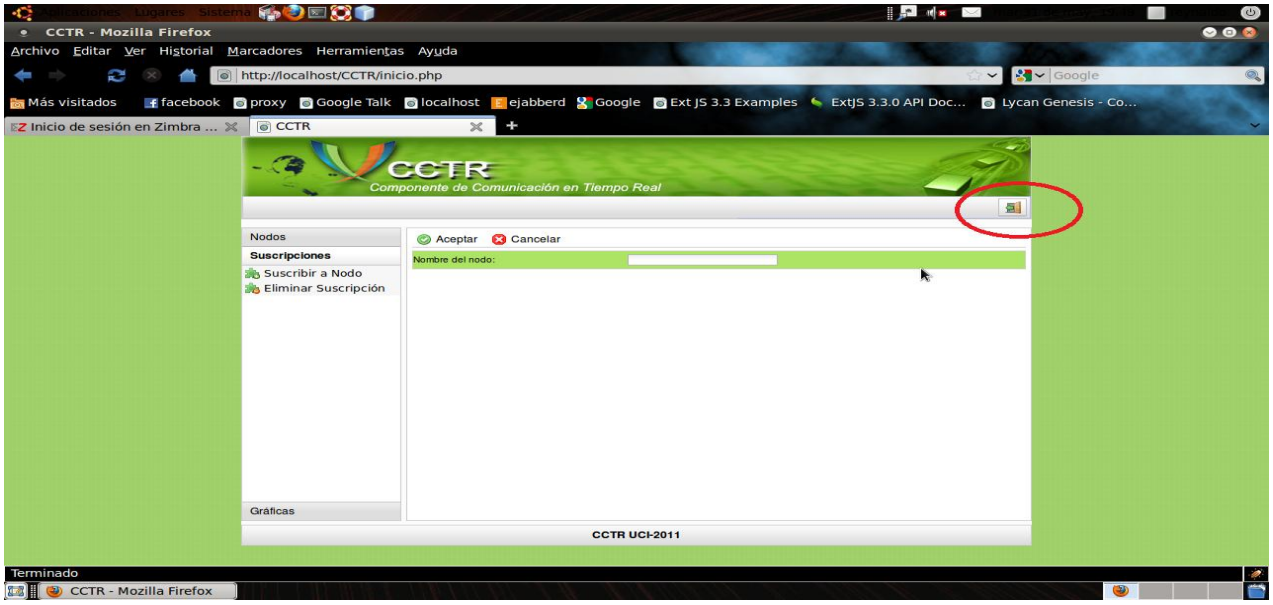


Tabla 8: Descripción textual resumida del CU Gestionar Nodo

Caso de Uso:	Gestionar nodo
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el usuario decide seleccionar una de las siguientes opciones: **Crear nodo:** el usuario decide crear un nodo para lo cual introduce los datos correspondientes, el sistema guarda los datos y crea el nodo, terminando así el caso de uso. **Editar nodo:** el usuario introduce los cambios en los campos que desea editar, el sistema guarda los cambios actualizando la información correspondiente a dicho nodo y de esta forma finaliza el caso de uso. **Eliminar nodo:** el usuario selecciona la opción para eliminar un nodo, el sistema muestra una interfaz con los campos para eliminar el nodo y el usuario introduce los datos del nodo a eliminar, finalizando así el caso de uso.
Precondiciones:	El usuario debe estar autenticado.
Referencias:	RF 7.1, RF 7.2, RF 7.2.1, 7.3.

Prioridad:	Crítica

Prototipo de Interfaz

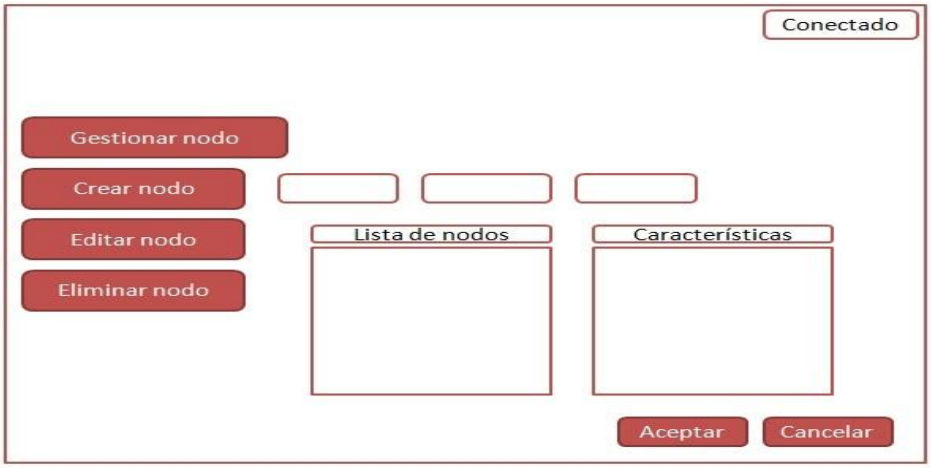


Tabla 9: Descripción textual resumida del CU Publicar Datos

Caso de Uso:	Publicar datos
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el usuario selecciona los datos y el nodo en el que se realizará la publicación de los mismos, el sistema actualiza los datos finalizando así el caso de uso.
Precondiciones:	El nodo para realizar la publicación debe de encontrarse creado.
Referencias:	RF 8.1, RF 8.2.
Prioridad:	Crítica

Prototipo de Interfaz

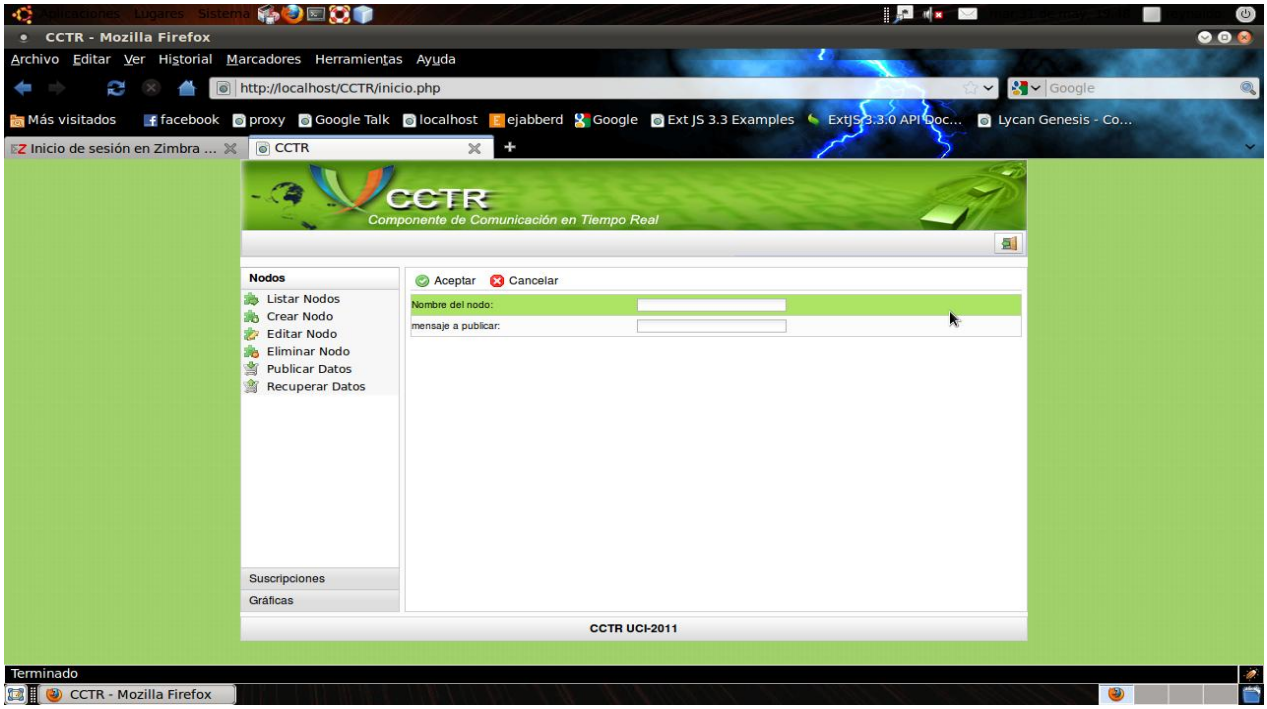


Tabla 10: Descripción textual resumida del CU Gestionar Usuario Jabber

Caso de Uso:	Gestionar usuario jabber
Actores:	Administrador Jabber
Resumen:	El caso de uso inicia cuando el administrador de jabber, decide seleccionar una de las siguientes opciones: **Crear usuario jabber:** el administrador de jabber, decide crear un usuario para el cual introduce los datos correspondientes, el sistema guarda los datos y crea el usuario, terminando así el caso de uso. **Editar usuario jabber:** el administrador de jabber, introduce los cambios en los campos que desea editar, el sistema guarda los cambios actualizando la información correspondiente al usuario, de esta forma finaliza el caso de uso. **Eliminar usuario jabber:** el administrador del jabber elimina el usuario deseado, finalizando así el caso de uso.

Precondiciones:	El usuario debe estar autenticado.
Referencias:	RF 9.1, RF 9.2, RF 9.2.1, RF 9.2.2, RF 9.3.
Prioridad:	Crítica

Prototipo de Interfaz

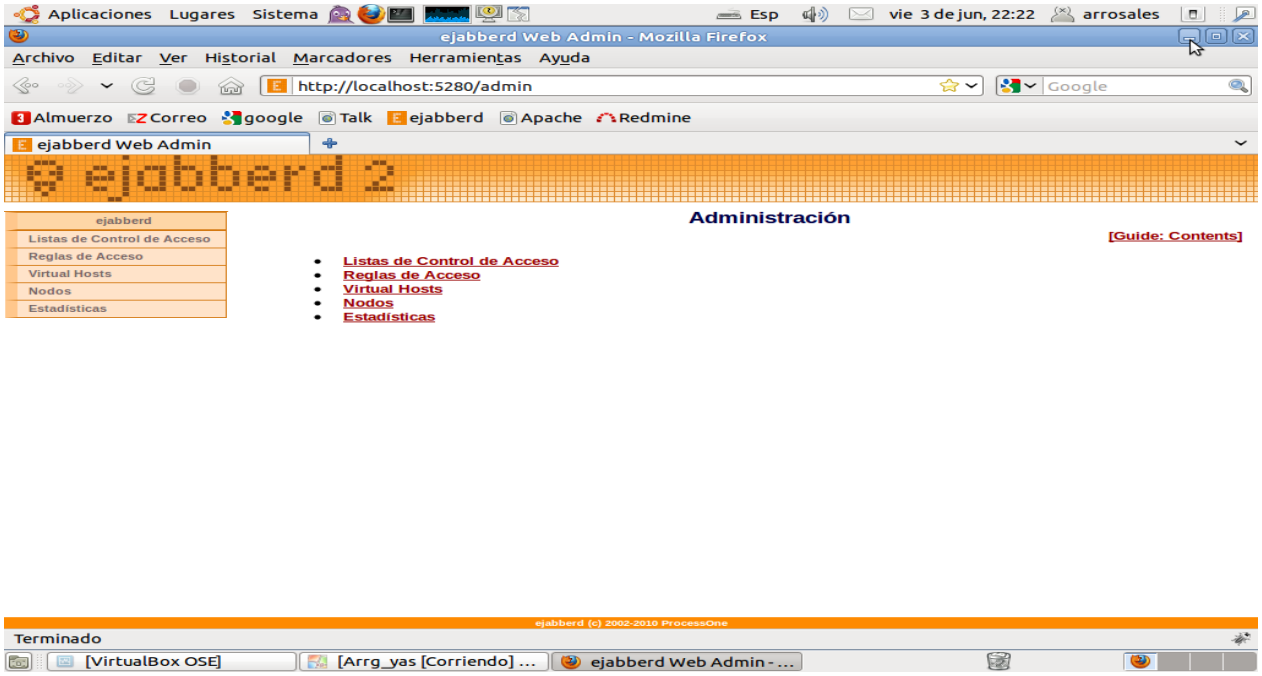
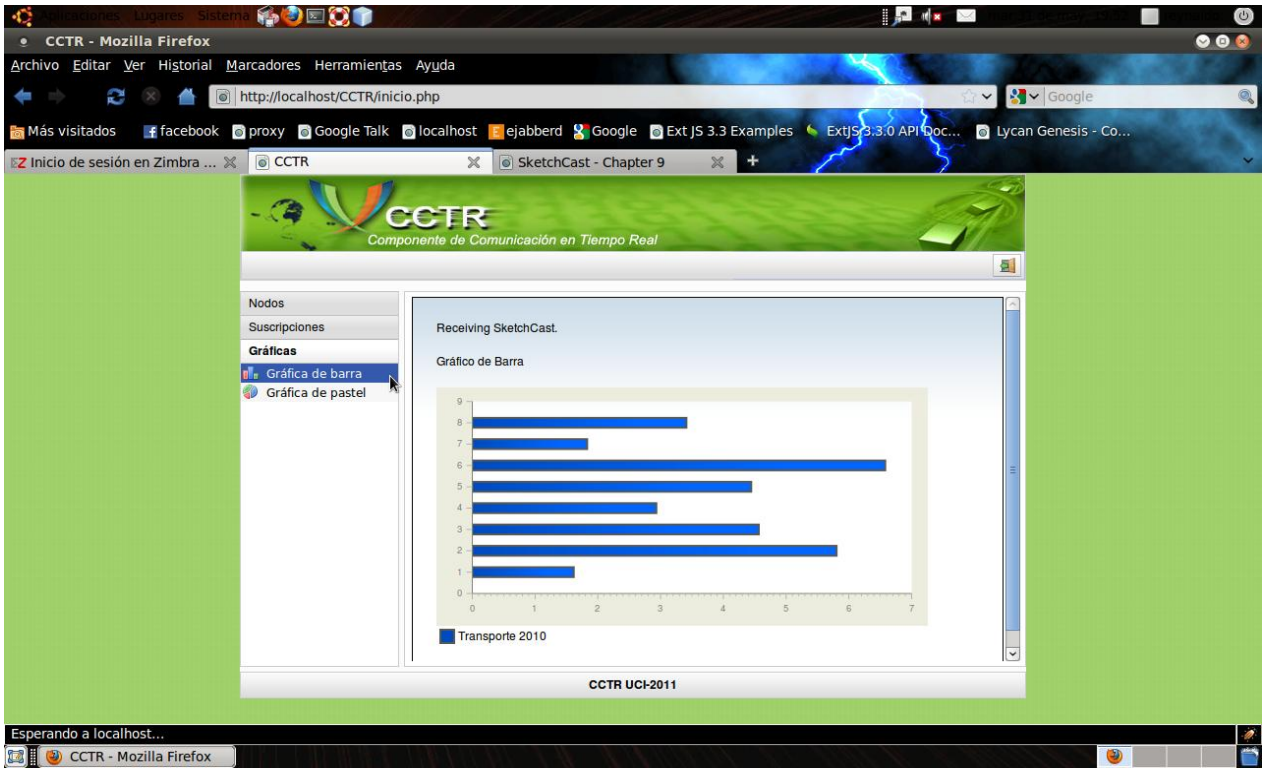


Tabla 11: Descripción textual resumida del CU Graficar Datos en Nodo

Caso de Uso:	Graficar datos en nodo
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el usuario decide seleccionar la opción: Graficar datos. El sistema muestra una interfaz y el usuario introduce los datos necesarios para la graficación, finalizando así el caso de uso.
Precondiciones:	El usuario debe estar autenticado.
Referencias:	RF 10.1
Prioridad:	Crítica

Prototipo de Interfaz



2.6 Diseño de la solución

El diseño tiene el objetivo de formular los modelos que se centran en los requisitos no funcionales y en el dominio de la solución. Constituye el punto inicial para la fase de implementación y prueba del sistema, por lo que se puede definir como propósito del mismo (25):

- Adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución y concurrencia y tecnologías de interfaz de usuario.
- Crear una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases.

- Descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo.

2.6.1 Patrones de diseño

Un patrón describe un problema que ocurre una y otra vez en el entorno de trabajo. Posteriormente describe el núcleo de la solución a ese problema, de tal forma que puede usar esa solución un millón de veces más, sin haberlo hecho de la misma forma dos veces (26).

Este a su vez representa (26, pág. 160):

- Una solución estándar para un problema común de programación.
- Una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios.
- Un proyecto o estructura de implementación que logra una finalidad determinada.
- Un lenguaje de programación de alto nivel.
- Una manera más práctica de describir ciertos aspectos de la organización de un programa.
- Conexiones entre componentes de programas.
- La forma de un diagrama de objeto o de un modelo de objeto.

Para la realización del diseño del componente se tienen en cuenta los patrones GRASP (*Patrones Generales de Software para Asignación de Responsabilidades*). Los cuales describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones (26, pág. 160).

A continuación se muestra una selección de estos patrones los cuales serán utilizados durante la realización del diseño de la aplicación (26, pág. 164):

- **Experto:** es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Expresa que siempre se debe asignar una responsabilidad al experto en información, o sea, a la clase que cuenta con la información necesaria para llevar a cabo la funcionalidad.
- **Creador:** este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos. La intención básica del mismo es encontrar un creador que necesite conectarse al objeto creado en alguna situación, eligiéndolo como el creador. Se favorece el bajo acoplamiento.

- **Bajo Acoplamiento:** el patrón bajo acoplamiento impulsa la asignación de responsabilidades de manera que su localización no incremente el acoplamiento hasta un nivel que lleve a los resultados negativos que puede producir un acoplamiento alto. No soporta el diseño de clases que son más independientes, lo que reduce el impacto del cambio. El mismo no se puede considerar de manera aislada a otros patrones como el Experto o el de Alta Cohesión, sino que necesita incluirse como uno de los diferentes principios de diseño que influyen en una elección, al asignar una responsabilidad**.**
- **Alta Cohesión:** este patrón es un principio a tener en mente durante todas las decisiones de diseño. Constituye un objetivo subyacente a tener en cuenta continuamente. Es un principio evaluativo que aplica un diseñador mientras evalúa todas las decisiones de diseño. Como beneficios que este aporta se pueden mencionar:
  - ✓ Se incrementa la claridad y facilita la comprensión del diseño.
  - ✓ Se simplifican el mantenimiento y las mejoras.
  - ✓ Se soporta a menudo bajo acoplamiento.
- **Controlador**: es el encargado de asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. La mayor parte de los sistemas reciben eventos de entrada externa, en cualquiera de los casos que puedan existir, si se recurre a un diseño orientado a objetos, hay que elegir los controladores que manejen esos eventos de entrada.

En búsqueda de un diseño claro, flexible y reutilizable también se utilizan patrones de diseño GOF (*Gang-Of-Four*). Estos se clasifican según su propósito en creacionales, estructurales y de composición, mientras que respecto a su ámbito se clasifican en clases y objetos, de los mismos fueron seleccionados para su uso (26, pág. 187).

- ***Observer*:** el patrón observador o publicación-suscripción proporciona un modo de acoplar débilmente los objetos en términos de la comunicación. Los emisores conocen a los suscriptores sólo a través de una interfaz, y los suscriptores pueden registrarse o darse de baja de los emisores dinámicamente. Se basa en el Polimorfismo, y proporciona variaciones protegidas en cuanto a que protege al emisor del conocimiento de la clase específica de objetos, y número de objetos, con los que se comunica cuando genera un evento.
- ***Decorator* (Envoltorio):** Añade dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender la funcionalidad.

2.6.2 Diagramas de clases del diseño

Los diagramas de clases del diseño, representan las relaciones entre clases que mantienen la información manipulada por el sistema, constituyen además, la interacción de las clases del diseño y sus objetos en la realización de los casos de uso, de forma que de cada uno se desprende un diagrama de clases (29). A continuación, se muestra el diagrama de clases del diseño del caso de uso Gestionar Nodo (Ver Figura 3), en el cual son empleados los patrones antes definidos seguidos de una breve descripción del mismo. Para lograr un mayor conocimiento de estos ver el expediente del proyecto: Diagramas de Clases del Diseño.

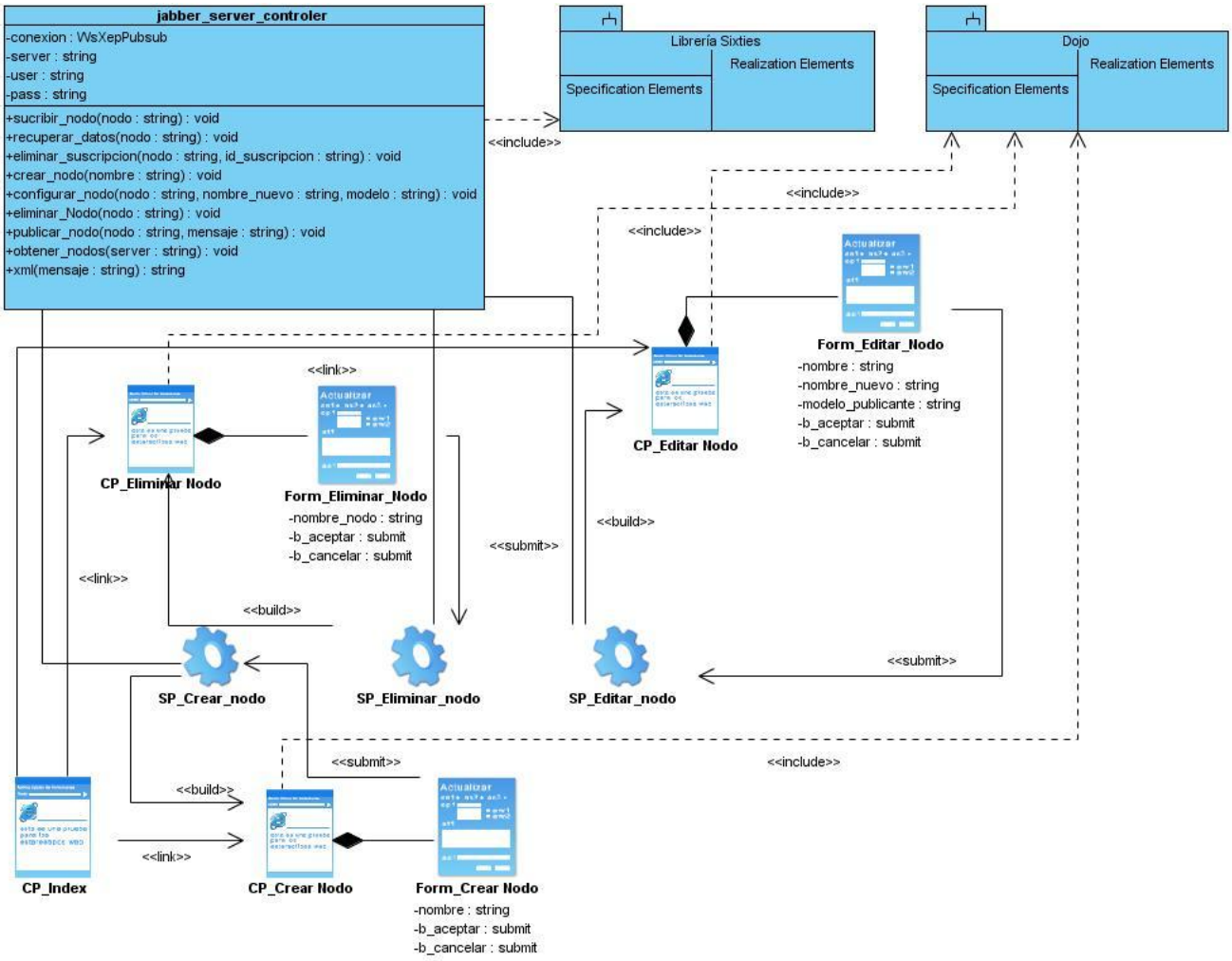


Figura 3: Diagrama de Clases del Diseño del CU Gestionar nodo

El diagrama de clases del caso de uso Gestionar Nodo, está constituido por la clase controladora ***jabber_server_controler***, la cual contiene todas las funciones necesarias para administrar el funcionamiento del servidor Ejabberd por tal motivo se convierte en el objeto fundamental del sitema, esta además, incluye al subsistema ***Libreria Sixties*** mediante una relación de *include*. Además se modela una *server page* para cada *client page,* sus funciones son manejar la información que le concierne dependiendo de cual *client page* controle, extraerla del servidor mediante ***jabber_server_controler*** e informar a quien le solicite alguna acción por tal motivo, las mismas son: ***SP_Editar_Nodo***, ***SP_Crear_Nodo*** y ***SP_Eliminar Nodo***. Dojo presenta una relación de include con las *client page* presentes en el diagrama puesto a que este subsietema es quien brinda estilos y sirve como apoyo para el trabajo con la tecnología *ajax* (***A****synchronous* ***J****avaScript* ***A****nd* ***X****ML*).

Cada una de estas páginas servidoras o *server pages* están asociadas a una *client page* y éstas, a su vez, generan un formulario mediante una relación de composición, los formularios son los encargados del envío de los datos recibidos de la interacción con el usuario a cada una de las *server pages.*

2.6.3 Diagramas de secuencia

Los diagramas de secuencia, describen las interacciones entre un grupo de objetos mostrando de forma secuencial los envíos de mensajes entre estos (28). A continuación, se presenta el diagrama de secuencia correspondiente al caso de uso Gestionar Nodo en su escenario Eliminar Nodo (Ver Figura 4), seguido de una descripción detallada de este. Con el objetivo de lograr una mayor comprensión de estos diagramas ver el expediente del proyecto: Diagramas de Secuencia.

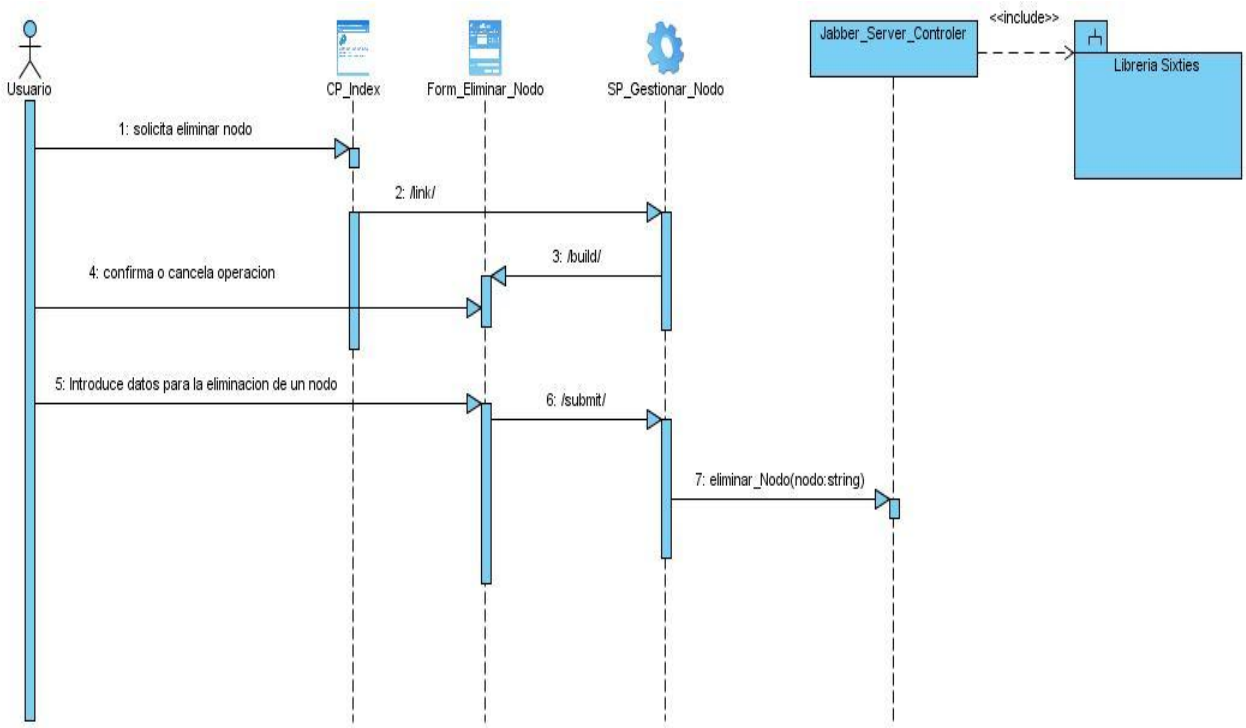


Figura 4: Diagrama de Secuencia del CU Gestionar nodo. Escenario Eliminar nodo

El actor usuario envía a **la *CP_Index*** un mensaje para solicitar la eliminación de un nodo. Esta tramita la operación solicitada enviando un *link* a la ***Server Page Gestionar Nodo*** (SP_Gestionar_Nodo). Luego de esto, la ***Server Page Gestionar Nodo*** (SP_Gestionar_Nodo) construye un formulario (***Form_Eliminar_Nodo***) mediante un mensaje *build*, el que se muestra al usuario, quien es el responsable de la desición de continuar o no con la operación, de eliminar el nodo.

Posteriormente el formulario envía un mensaje de tipo *submit* a la ***Server Page Gestionar Nodo*** (SP_Gestionar_Nodo). Esta envía el mensaje para la eliminación del nodo a la ***Clase jabber_server_controler*** para que esta elimine el nodo del servidor, quien presenta además una relación de inclusión con la librería ***Sixties***.

2.6.4 Aplicación de patrones de diseño:

Experto: La clase jabber_server_controler funciona como experta en información para llevar a cabo la administración del servidor Ejabberd (Ver Figura 5).

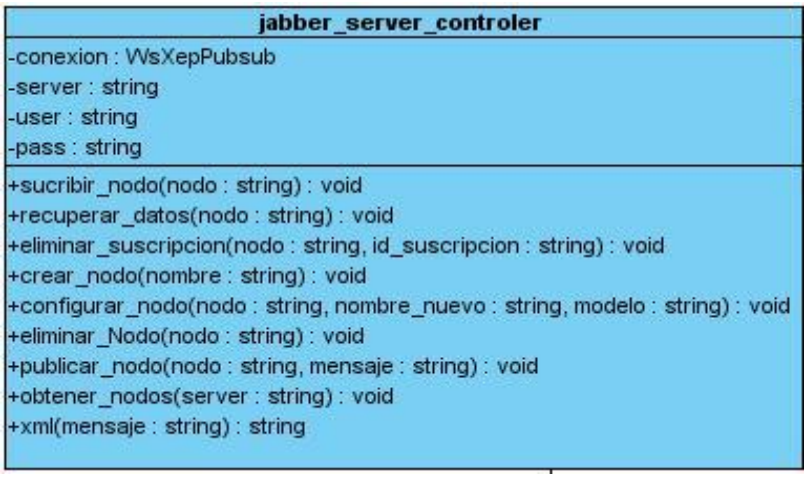


Figura 5: Aplicación del patrón de diseño Experto

Creador: En la sección que se muestra en la figura 6 se evidencia el patrón mencionado ya que en la clase jabber_server_controler se crea un objeto conexión del tipo clase WsXepPubsub contenida esta dentro del paquete librería Sixties.

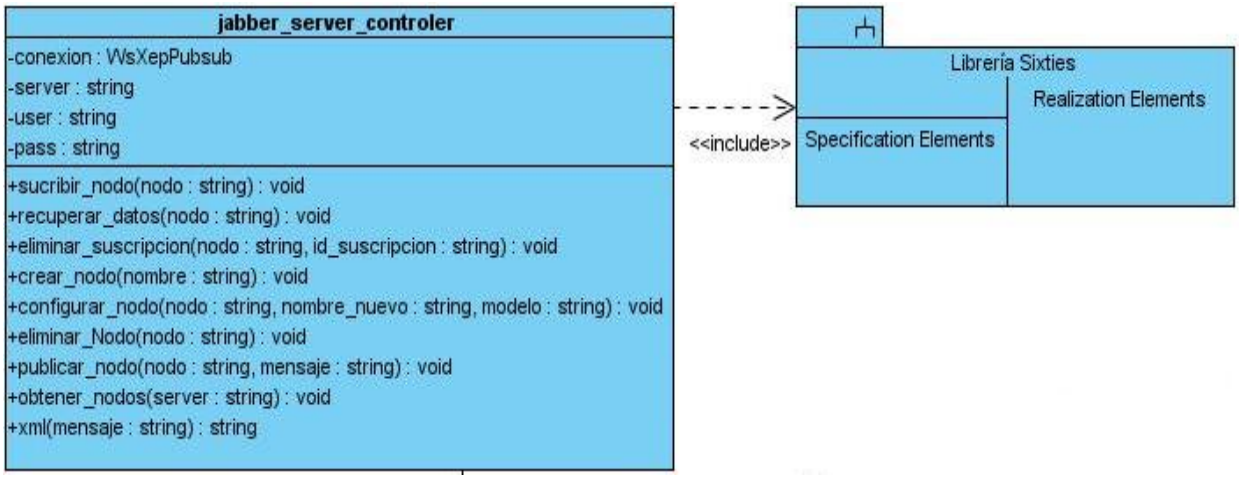


Figura 6: Aplicación del patrón de diseño Creador

Bajo acoplamiento: En la sección que se muestra en la figura 7 se evidencia el patrón de diseño bajo acoplamiento, donde la clase WsXepPubsub utiliza una instancia de la clase XepPubsub en función de crear una conexión demostrándose así poca dependencia de jabber_server_controler con el resto de las clases para realizar sus responsabilidades .

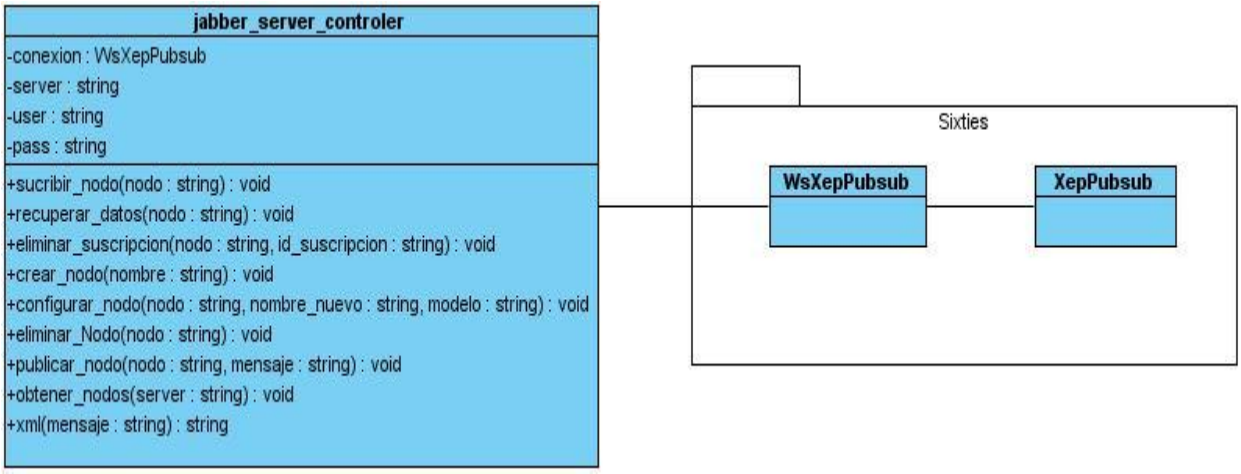


Figura 7: Aplicación del patrón de diseño Bajo acoplamiento

Alta cohesión: A continuación se muestra como cada entidad se ajusta a manejar solo la responsabilidad para la que fue creada, donde para cada página cliente existe una página servidora encargada de manejar sus solicitudes poniéndose de manifiesto el patrón de diseño alta cohesión (Ver Figura 8).

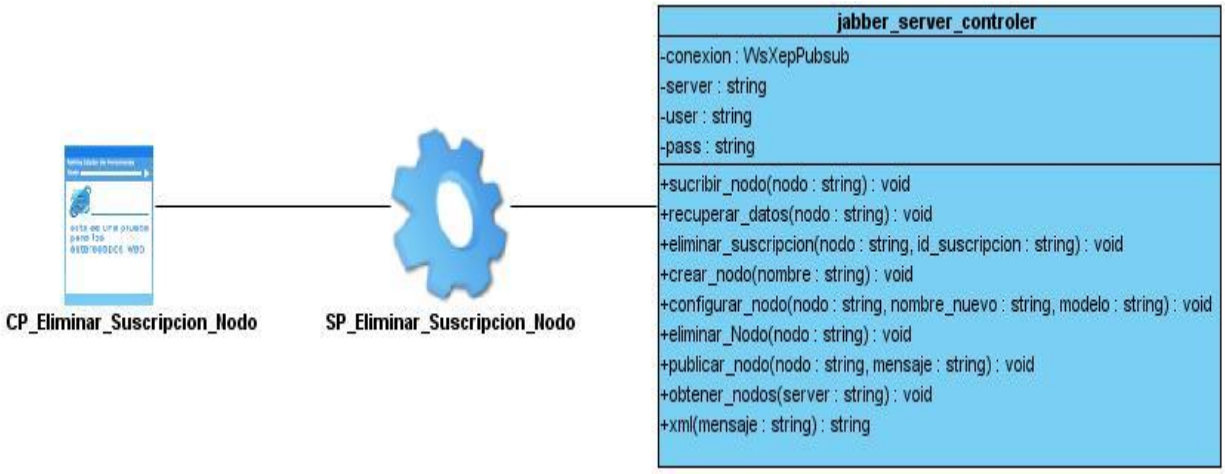


Figura 8: Aplicación del patrón de diseño Alta cohesión

Controlador: En la figura 9 se evidencia el patrón de diseño controlador, poniéndose de manifiesto cuando la página servidora SP_Eliminar_Suscripcion_Nodo crea una instancia de la clase jabber_server_controler, en este caso con el fin de ejecutar la acción de eliminar la subscripción a un nodo.

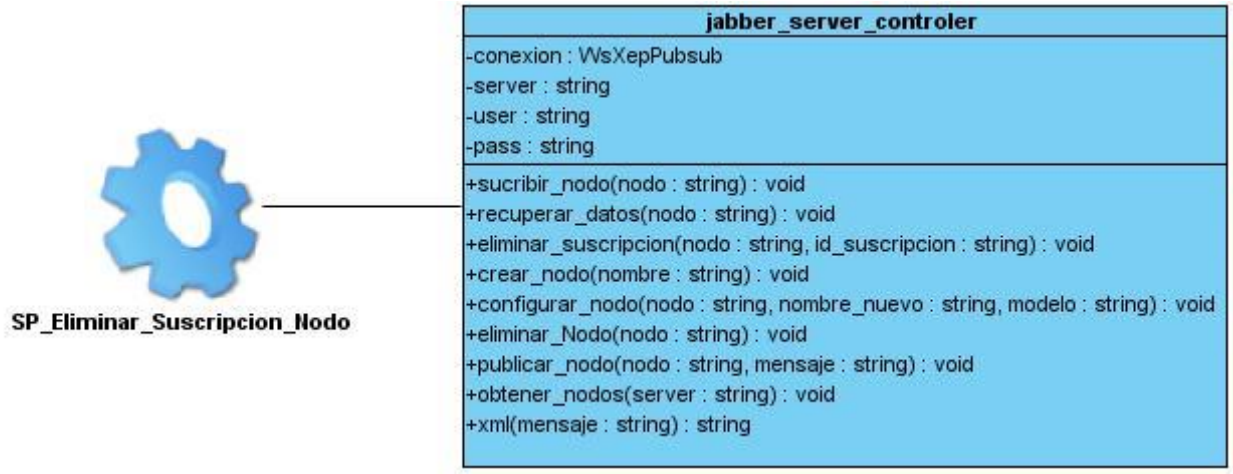


Figura 9: Aplicación del patrón de diseño Controlador

Observer: A continuación se muestra como se pone de manifiesto el patrón de diseño antes mencionado viéndose como el componente gráfico se queda a la espera de los datos que se les hacen llegar a través del componente JavaScript, este último obteniendo la información del servidor Ejabberd (Ver Figura 10).

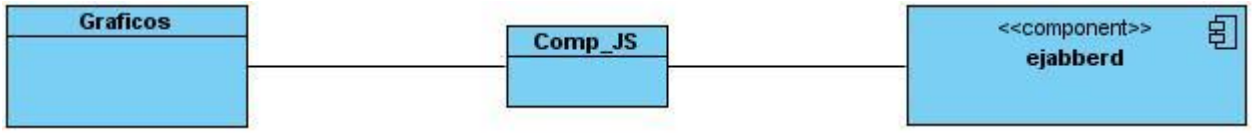


Figura 10: Aplicación del patrón de diseño Observer

Conclusiones parciales

Se identificaron 22 requisitos funcionales con los cuales se llegó a una definición formal de lo que el sistema debe hacer. Se emplearon técnicas de validación de requisitos, lo que trajo como resultado la aprobación por parte del cliente de los requerimientos identificados y aseguró la calidad de los mismos. A partir de la utilización de los patrones de casos de uso se determinaron 10 casos de uso y la relación con los actores del sistema. Finalmente se aplicaron los patrones de diseño, Creador, Experto, Controlador, Alta cohesión y Bajo acoplamiento dentro de los patrones GRASP y dentro de los GOF el *Observer* y *Decorator*. Se especificó la estructura del sistema a través del modelamiento de diagramas de clases del diseño por cada caso de uso y su diagrama de secuencia correspondiente.

Capítulo 3: Implementación y Pruebas.

Introducción

En el desarrollo de este capítulo se abordará todo lo referente a la implementación del sistema y a las pruebas que se realizarán para la validación del mismo, dando cumplimiento a los siguientes objetivos específicos: implementar el componente diseñado, representar gráficamente la información haciendo uso del componente implementado, además de validar el funcionamiento del componente de comunicación en tiempo real. Los cuales a su vez se encuentran asociados a tareas de la investigación tales como: realización de la conexión de los KPI con los gráficos utilizando el componente implementado, validación de la persistencia de los datos de la conexión haciendo uso de los gráficos y realización de las pruebas definidas en el capítulo 1.

3.1 Diagrama de componentes

La vista de implementación toma en cuenta los requerimientos que facilitan la programación, los niveles de reutilización y las limitaciones impuestas por el entorno de desarrollo. Para modelarla se dispone de dos elementos, los paquetes que representan una partición física del sistema y los componentes que representan la organización de los módulos de código fuente(29).

El diagrama de componentes, describe cómo los elementos del modelo de diseño se implementan en términos de componentes, representa además, la estructura y organización de estos, así como las relaciones presentes de unos con otros. Además constituyen la vista de implementación estática de un sistema, modelan los elementos físicos de este, tales como: ejecutables, tablas, librerías, archivos, documentos y las relaciones entre estos(29).

A continuación se representa un diagrama detallado de los componentes correspondientes al sistema a desarrollar (Ver Figura 11), así como una breve descripción del mismo, con el objetivo de lograr una descripción más detallada de nodos del siguiente diagrama, referirse al Trabajo de Diploma “Definición de la arquitectura de un componente de comunicación de datos sobre la Web en tiempo real”.

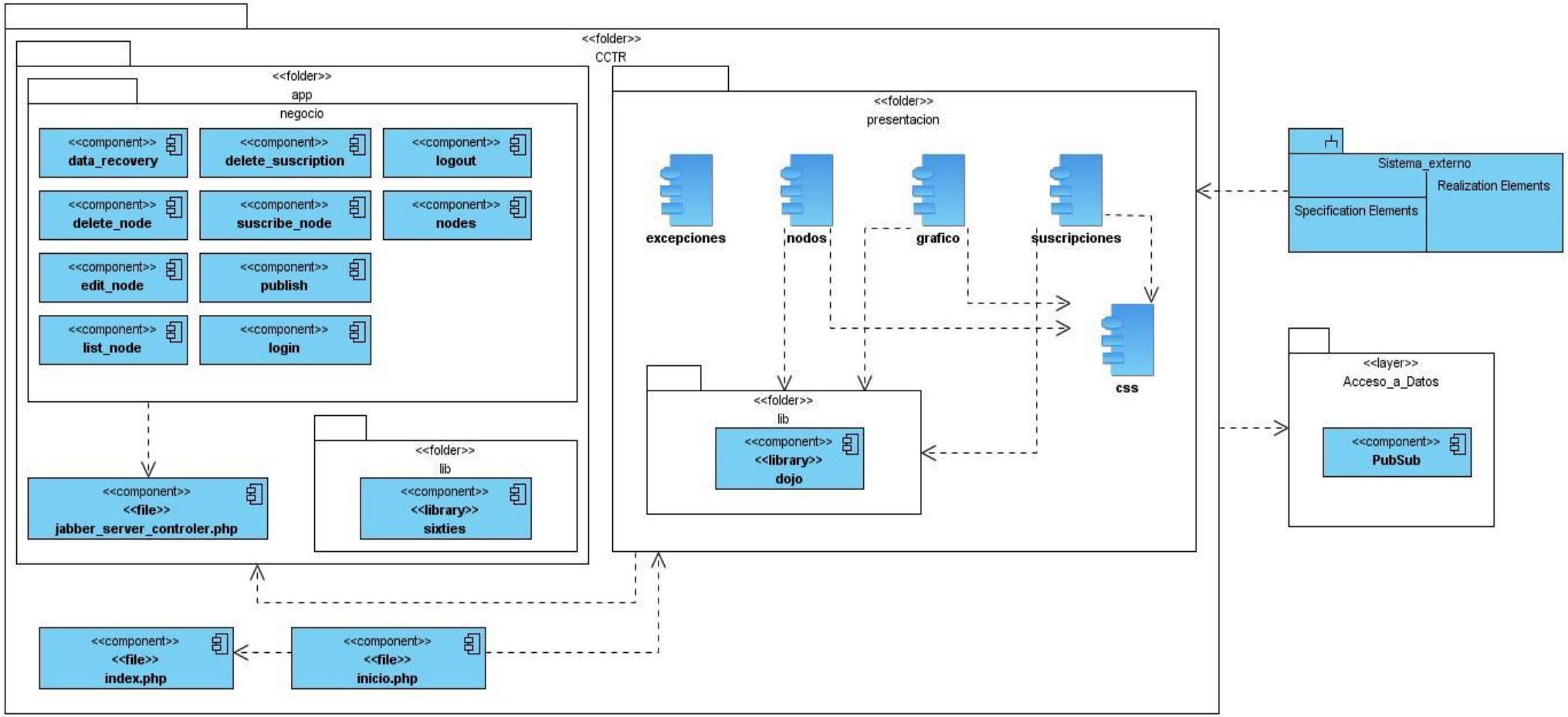


Figura 11: Diagrama de componentes del sistema

- **CCTR:** representa el paquete contenedor de todos los subsistemas, librerías y componentes que forman parte de la aplicación.
- **Index.php:** representa la interfaz principal del sistema.
- **Inicio.php:** representa la interfaz donde el usuario tiene una serie de opciones dependiendo de su estado en el sistema.
- **presentación:** representa el paquete contenedor de todas las interfaces del sistema.
- **nodos:** representa el paquete de componentes en el cual se encuentran las interfaces necesarias para la administración de los nodos.
- **suscripciones:** representa el paquete de componentes en el cual se encuentran las interfaces encargadas de manejar las suscripciones a los nodos del servidor Ejabberd.
- **gráfico:** representa el paquete de componentes encargado de graficar y representar la información previamente seleccionada.
- **excepciones:** engloba y contiene las diferentes interfaces para mostrar en caso de error en el sistema.
- **lib:** representa el paquete contenedor de las diferentes librerías utilizadas en la implementación del componente en ambos casos tanto para la capa de app como para la de presentación.
- **sixties:** representa el componente Sixties el cual es una librería con las funcionalidades encargadas de manejar las distintas funciones del servidor Ejabberd.
- **dojo:** representa el componente Dojo el cual es una librería utilizada para manejar el diseño de las interfaces y representar gráficamente indicadores publicados en los nodos del servidor Ejabberd.
- **app:** representa el paquete contenedor de las clases y componentes encargados de controlar el flujo de datos entre el usuario y el servidor Ejabberd.
- **negocio:** representa el paquete de componentes donde se engloban todos los archivos php que funcionan de intermediario entre las interfaces del sistema y la clase manejadora del servidor: jabber_server_controler, los mismos son: data_recovery.php, delete_node.php, delete_suscription.php, edit_node.php, list_nodes.php, login.php, logout.php, nodes.php, publish.php y suscribe_node.php

- **jabber_server_controler:** representa a la clase controladora principal de la aplicación donde se encuentran todas las funcionalidades implementadas para acceder mediante la librería Sixties al servidor Ejabberd.
- **Acceso_a_Datos:** representa la capa donde se encuentran los datos, en este caso el servidor Ejabberd.
- **PubSub:** representa un servicio dentro del servidor XMPP utilizado para dar solución al problema, el mismo se encarga de manejar las operaciones de publicación y suscripción en el servidor así como de dar la posibilidad de gestionar los nodos pubsub.
- **Sistema_externo:** representa un sistema que se nutrirá de los casos de uso gestionar nodos y publicar datos, dicho sistema sería una fuente externa de información la cual estaría en comunicación constante con el componente.

Teniendo en cuenta de que los diagramas de implementación muestran los aspectos físicos del sistema e incluyen la estructura del código fuente y la implementación y que estos, constituyen además una base para el modelo de despliegue, en el siguiente epígrafe se aborda acerca de la realización de este artefacto.

3.2 Diagrama de despliegue

Un Diagrama de Despliegue modela la arquitectura en tiempo de ejecución de un sistema. Esto muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del software se trazan en esos nodos (30) (Ver Figura 12).

La aplicación será desplegada de la siguiente manera: la PC_cliente que se podrá encontrar en cualquier ubicación utilizará el protocolo BOSH para emular una conexión persistente entre si misma y el servidor XMPP, con estado y conexión bidireccional a dicho servidor. Este servidor a través del protocolo de comunicación XMPP es el encargado de gestionar las conexiones y los envíos de paquetes XML entre el cliente y el servidor pudiendo este último interpretar dichos paquetes, de esta manera se garantiza la persistencia de los datos siempre y cuando las conexiones entre los nodos presentes en el despliegue cumplan con lo establecido a la hora de materializar el proceso.

Para lograr una descripción más detallada de los nodos del siguiente diagrama, remitirse al Trabajo de Diploma “Definición de la arquitectura de un componente de comunicación de datos sobre la Web en tiempo real”.

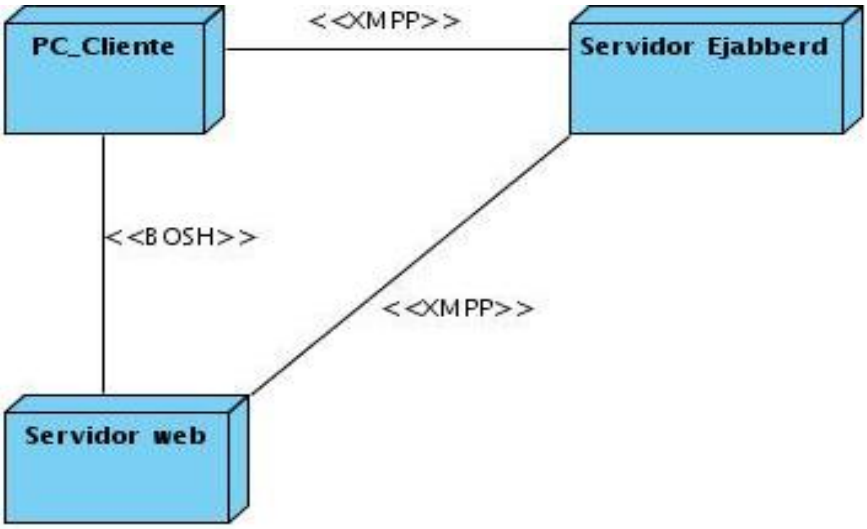


Figura 12: Diagrama de Despliegue

3.3 Validación y pruebas

La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación.(16, pág. 1)

3.3.1 Realización de pruebas de unidad

En aras de lograr la aprobación y verificación del sistema implementado, se aplicaron las pruebas funcionales. Para dar cumplimiento a lo antes referido, se diseñaron casos de pruebas basados en los casos de uso y a su vez en los requisitos funcionales, analizando cada funcionalidad implementada, para verificar que se cumplan las necesidades del cliente. Además se comprobó la correcta validación de los campos, verificando que solamente se aceptarán los caracteres válidos para los mismos, aplicando la Técnica de Partición de Equivalencia de Caja Negra.

Diseño de casos de prueba y registro de no conformidades

Partiendo de la descripción de los casos de uso del sistema, se diseñó un caso de prueba asociado a cada caso de uso. Para detallar el mismo se utiliza la plantilla de diseño de casos de prueba que aparece en el expediente de proyecto propuesto por la Universidad como resultado del proceso de mejoras. Este documento consta de los siguientes aspectos:

- **Nombre de la sección:** se especifica el nombre de la sección a probar.

- **Escenarios de la sección:** se especifican los escenarios de cada sección [EC 1.1: Nombre del Escenario].
- **Descripción de la funcionalidad:** se describe brevemente la funcionalidad del escenario.

Tabla 11: Secciones a probar en el CU Gestionar Nodo. Editar Nodo

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1 Editar nodo	EC 1.1 Editar nodo	En este escenario se editan los datos del nodo seleccionado.
	EC 1.2 Cancelar la operación.	Este escenario consiste en la cancelación de la acción de editar un nodo.

A partir de esta descripción se detallan también en el documento las variables que se encuentran en las interfaces asociadas al caso de uso que se le está diseñando el caso de prueba. Esta descripción está compuesta por campos tales como:

- **No:** se enumeran todos los campos o variables descritas en el Caso de Uso.
- **Nombre de campo:** nombre del campo de entrada.
- **Clasificación:** la clasificación es según el componente de diseño utilizado por ejemplo: campo de texto, lista desplegable o campo de selección.
- **Valor Nulo:** se especifica si el campo puede ser nulo o no. Para ello solo se pone Sí o No.
- **Descripción:** se realiza una breve descripción de los datos que deben introducirse, además se especifican las reglas que tiene que cumplir el campo.

En la siguiente tabla se muestra un ejemplo de la descripción de variables del caso de uso Gestionar Nodo en su escenario Editar Nodo.

Tabla 12: Descripción de variables del CU Gestionar Nodo. Editar Nodo

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	nombre del nodo	Campo de texto	No	Se introduce por el usuario el nombre del nodo que desea editar.

2	Nuevo nombre	Campo de texto	Si	Se introduce por el usuario el nuevo nombre para el nodo.
3	Tipo de nodo	selector	No	Se selecciona por el usuario el nuevo tipo para el nodo.
4	Número Máximo de Ítems	Campo de texto	Si	Se introduce por el usuario el nuevo número máximo de ítems que puede almacenar el nodo.
5	Tamaño Máximo	campo de texto	Si	Se introduce por el usuario el nuevo tamaño máximo del *payload* del nodo.
6	Modelo del publicante	selector	No	Se selecciona por el usuario el nuevo modelo del publicante del nodo.

La ejecución de este caso de prueba, permitió detectar algunos de los defectos y dificultades del componente, logrando así que el mismo cuente con una mayor calidad una vez corregidos los errores encontrados. Estos son considerados como no conformidades y quedan recogidas dentro del documento" Registro de Defectos y Dificultades Detectados", el cual cuenta con los siguientes campos:

- **Elemento:** especifica el nombre del elemento a probar (Documento o Aplicación).
- **No:** un número que identifique al error o no conformidad. No conformidad: se especifica el error detectado, la no conformidad encontrada.
- **Aspecto correspondiente:** se describe la no conformidad, cualquier otro aspecto relevante y se especifica su localización.
- **Significativa:** se marca con una <X> si la no conformidad se considera significativa.
- **No Significativa:** se marca con una <X> si la no conformidad se considera no significativa.
- **Recomendación:** se describe lo que el probador recomienda para dar solución a la no conformidad encontrada.
- **Respuesta del Equipo de Desarrollo:** describe la respuesta del equipo de desarrollo a la no conformidad.

En la siguiente tabla se muestra un ejemplo de Registro de no conformidades detectadas del caso de uso Gestionar Nodo en su escenario Editar Nodo.

Tabla 13: Registro de defectos y dificultades detectados del CU Gestionar Nodo. Editar Nodo

Para lograr un mayor conocimiento de estos casos de prueba con sus respectivos resultados ver el expediente de proyecto.

Elemento	No	No Conformidad	Aspecto Correspondiente	Etapa de Detección	Significativa	No Significativa	Recomendación	Estado NC	Respuesta del Equipo de Desarrollo
Número Máximo de Ítems	1	Mal escrita la palabra: Ítems	Cuando se muestra el campo Número Máximo de Ítems en las propiedades de la funcionalidad Editar Nodo.	Pruebas	Ortografía.		Se recomienda arreglar la ortografía de esta palabra poniendo la tilde necesaria.	PD: Pendiente 10/5/2011 RA: Resuelta 25/5/2011	Quedó corregida la no conformidad referente a la ortografía en el sistema.
Cancelar la operación.	2	El sistema no redirecciona a la página de inicio de acorde a la	Cuando se realiza la acción de cancelar la operación de edición del nodo.	Pruebas	Funcionalidad.		Se recomienda completar la implementación de esta	PD: Pendiente 10/5/2011 RA:	Quedó corregida la no conformidad referente a la implementación de la funcionalidad de cancelar la edición de un nodo.

		descripción.					funcionalidad.	Resuelta 25/5/2011	

Resultado de la prueba de unidad

Después de haberse aplicado este tipo de prueba y de analizar cada uno de los resultados obtenidos durante la ejecución de la misma, se evidencia de que de un total de 10 casos de pruebas que fueron aplicados, se encontraron 3 no conformidades, lo que representa un 30% de funcionalidades defectuosas, como se muestra en el siguiente gráfico (Ver Figura 13):

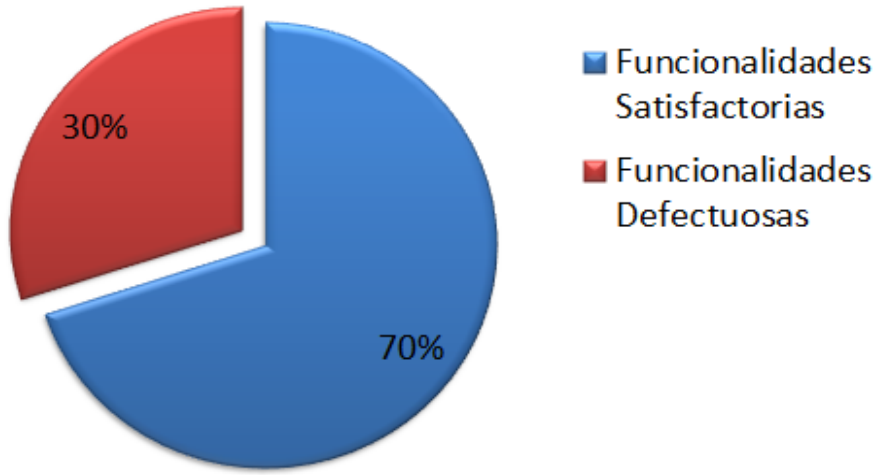


Figura 13: Resultado de las pruebas

Estas no conformidades fueron cubiertas para dejar en un 100% de funcionamiento el sistema.

3.4.3 Realización de pruebas de integración y sistema

Para la realización de estas pruebas se diseñó un caso de prueba donde se encuentran presentes las funcionalidades crear nodo y graficar datos, esta última para llegar a la representación de los gráficos necesita de la integración con un componente gráfico integrado con el componente que crea el nodo, el cual se muestra en las siguientes tablas:

Tabla 14: Secciones a probar mediante la ténica de integración

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad

SC 1 Integración de crear nodo y graficar datos.	EC 1.1 Integración de las funcionalidades crear nodo y graficar datos.	En este escenario se valida que el proceso de integración de las funcionalidades crear nodo y graficar datos con el componente necesario para la realización de los gráficos sea correcto.

Tabla 15: Descripción de variables a probar mediante la ténica de integración

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	nombre del nodo	Campo de texto	No	Se introduce por el usuario en nombre de nodo del que desea recuperar los datos previos a la suscripción.
2	jid	Campo de texto	No	Se introduce el identificador jabberd (jid) del usuario que se desea realizar la representación gráfica de los datos.
3	contraseña	Campo de texto	No	Se introduce la contraseña del usuario.
4	servicio	Campo de texto	No	Se introduce el servicio Pubsub del usuario que se desea realizar la representación gráfica de los datos.

Pruebas de carga y stress

Para la realización de estas pruebas en la herramienta JMeter se utilizó un juego de datos de 1,100 y 200 usuarios para cada caso de prueba diseñado como se muestra a continuación. Para lograr un mayor conocimiento de estos casos de prueba con sus respectivos resultados ver el expediente de proyecto.

Tabla 16: Resultado de la prueba para un usuario del Caso de Uso Crear Nodo

Label	#Muestras	Media	Mediana	Línea de 90%	Min.	Max.	% error	Rendimiento	Kb/sec
/CCTR/app/nodes.php	1	644	644	644	644	644	0.0	0,651ms	0.0
TOTAL	1	644	644	644	644	644	0.0	0,651ms	0.0

Tabla 17: Resultado de la prueba para 100 usuarios del Caso de Uso Crear Nodo

Label	#Muestras	Media	Mediana	Línea de 90%	Min.	Max.	% error	Rendimiento	Kb/sec
/CCTR/app/nodes.php	100	5822	4853	14126	768	15861	0.0	0,815ms	0.0
TOTAL	100	5822	4853	14126	768	15861	0.0	0,815ms	0.0

Tabla 18: Resultado de la prueba para 200 usuarios del Caso de Uso Crear Nodo

Label	#Muestras	Media	Mediana	Línea de 90%	Min.	Max.	% error	Rendimiento	Kb/sec
/CCTR/app/nodes.php	200	14509	12303	30146	653	41890	0.0	0,967ms	0.0
TOTAL	200	14509	12303	30146	653	41890	0.0	0,967ms	0.0

Resultado de la prueba de integración y sistema

Después de aplicarse las pruebas de integración, carga y stress se obtuvo como resultado que en el componente desarrollado las funcionalidades se integran de manera satisfactoria, además que tras la ejecución de los casos de prueba con la herramienta JMeter los tiempos obtenidos para las diferentes

funcionalidades se encuentran dentro del rango esperado, sin mostrar errores en el proceso de ejecución de las mismas.

Conclusiones parciales

En este capítulo se estructuró el modelo de implementación a partir de los resultados del diseño, y se obtuvo el diagrama de componentes. Se realizó el diagrama de despliegue en el cual se evidencia cómo estará distribuido el sistema físicamente una vez liberado. Gracias al uso de un estándar de codificación se logró que el código fuera extensible, verificable, reparable y comprensible. Mediante las pruebas aplicadas se validó que el sistema cumpliera con las funcionalidades definidas.

Conclusiones Generales

- Se realizó un análisis crítico acerca de los sistemas de comunicación en tiempo real, así como su aplicación a diferentes áreas. Se evidenció la necesidad de implementar un componente para dar solución al problema planteado.
- A partir de una adecuada captura de requisitos, se definieron las principales funcionalidades del componente a desarrollar en su primera versión. Se diseñó el componente con la representación de los diagramas de clases de diseño y de secuencia correspondientes.
- Se implementó el componente diseñado, haciendo uso del lenguaje PHP y JavaScript.
- Se integró el componente con un módulo de gráficas, mediante el cual se grafican los datos enviados.
- Se realizaron las pruebas en los niveles de unidad, integración y sistema, empleando las técnicas funcional y de rendimiento y aplicando el tipo de prueba de caja negra. Como herramienta los casos de pruebas funcionales y el JMeter para la realización de las pruebas de carga y stress.

Recomendaciones

Se recomienda:

- Integrar la aplicación al sistema *Dashboard* actualmente en desarrollo en el Centro de Tecnologías y Gestión de Datos (DATEC). Además de continuar el estudio del protocolo XMPP con el fin de integrarlo a otros módulos del proyecto.
- Generalizar el uso del componente en otros proyectos que necesiten la comunicación en tiempo real.
- Continuar el estudio de la especificación XEP_0030: *Service Discovery* con el fin de mejorar las funcionalidades de listar nodo y recuperar datos.

Referencias Bibliográficas

1. **García, A. R.** (s.f.). COMPONENTE DE COMUNICACIÓN EN TIEMPO REAL DE VALORES DE LOS KPIS DE UN DASHBOARD. (s.f.). Obtenido de http://semanatecnologica.fordes.co.cu
2. **Dávila, A.** (s.f.). Nuevas herramientas de control: El Cuadro de Mando Integral.
3. **Kaplan, N.** (Enero/Febrero de 1992). Harvard Business Review.
4. **Bonnefoy, Juan Cristóbal.** Indicadores de Gestión del Desempeño y el Cuadro de Mando Integral. (s.f.). Obtenido de Portal de la CEPAL: http://www.cepal.org/
5. Decide Soft. (s.f.). Obtenido de Decide Soft: http://www.decidesoft.net/
6. Delphos. (s.f.). Obtenido de Delphos: http://www.delphosinv.com.
7. Delphos. (s.f.). Obtenido de Denisa: http://www.deinsa.com/delphos.
8. Características.(s.f.).Obtenido de Pentaho: http//www.gravitar.biz/
9. Características de Pentaho. (s.f.). Obtenido de Dataprix: http://www.dataprix.com/
10. Conceptos básicos de internet 6 - Comunicación en tiempo real (IRC, IM, y más). (s.f.). Obtenido de Blog tecnológico: http://www.asoma.es.
11. Características de la comunicación asíncrona y síncrona. Obtenido de http://herramientastutor.blogspot.com
12. **Burns A. and, W. A.** (1996). Real-time systems and programming languages. University of York, Addison Wesley.
13. Mensajería Instantánea. (s.f.). Obtenido de Centro Costadigital Observatorio: http://www.costadigital.cl
14. Obtenido de http://www.whisbi.com.
15. **Ríos, H. G.** (s.f.). El IRC (Internet Relay Chat). Obtenido de http://www.cabinas.net.
16. Conferencia # 7 Disciplina Prueba. Ingeniería del Software II. (s.f.). Obtenido de: http://eva.uci.cu.
17. **Juristo Natalia, Moreno Ana M., Vegas Sira.** (s.f.). Técnicas de evaluación de software. Obtenido de: http://eva.uci.cu.
18. **Garcerant, Iván.** (s.f.). Modelo de Dominio. Obtenido de http://synergix.wordpress.com.

19. **Pressman, R.** (s.f.). Ingeniería de Software.
20. **Pan, D, Zhu, D y Johnson, K.** (s.f.). Requierements Engeenering Thecniques.
21. **Raghavan, Sridhar, Zelesnik, Gregory y Ford, Gary.** (s.f.). Lecture Notes on Requirements Elicitation.
22. **Gallego, G. J.** (s.f.). Ingeniería de Requerimientos. Obtenido de http://www.scribd.com.
23. **G, L. E.** (s.f.). Ingeniería de Software con UML Unified Modeling Language – Lenguaje Unificado de Modelado. Obtenido de http://www.eduardoleyton.com/
24. Introducción a la Disciplina de Requisitos de RUP. (s.f.). Obtenido de: http://eva.uci.cu.
25. Conferencia # 1 Disciplina de Análisis y Diseño. Ingeniería del Software II. (s.f.). Obtenido de: http://eva.uci.cu.
26. **Larman, C**. (s.f.). UML y Patrones. Introducción al análisis y diseño orientado a objetos y al proceso unificado.2da Edición.
27. **Visconti, M. Y.** (s.f.). Fundamentos de Ingeniería de Software. Obtenido de http://portal.inf.utfsm.cl.
28. Obtenido de http://www.editions-eni.fr/
29. Diagrama de Componentes UML 2. (s.f.). Obtenido de Sparxsystems: http://www.sparxsystems.com.ar.
30. Diagrama de Despliegue UML 2. (s.f.). Obtenido de Sparxsystems: http://www.sparxsystems.com.ar.

Bibliografía

- **Ambler, S. W.** (s.f.). Metodo de Pruebas Orientada a Objetos para el Ciclo de Vida Completo (FLOOT). Obtenido de http://www.ambysoft.com
- Arquitectura de software II Diagramas de Componente y Despliegue. (s.f.). Obtenido de http://es.scribd.com
- Balanced Scorecard : Cuadro de Mando Integral. (s.f.). Obtenido de http://www.cuadrodemandointegral.net
- **Blank Isabel**, H. L. (s.f.). Pruebas de funcionalidad. Obtenido de http://carolina.terna.net/ingsw3/datos/Pruebas_Funcionales.pdf
- **Bonnefoy, Juan Cristóbal**. Indicadores de Gestión del Desempeño y el Cuadro de Mando Integral. (s.f.). Obtenido de Portal de la CEPAL: http://www.cepal.org
- **Burns A. and, W. A.** (1996). Real-time systems and programming languages. University of York, Addison Wesley.
- Características de la comunicación asíncrona y síncrona. Obtenido de http://herramientastutor.blogspot.com
- Características de Pentaho. (s.f.).Obtenido de Pentaho: http//www.gravitar.biz
- Comunicación en tiempo real. (s.f.). Obtenido de http://iworld.com
- Conferencia # 7 Disciplina Prueba. Ingeniería del Software II. (s.f.). Obtenido de: http://eva.uci.cu
- Conceptos básicos de internet 6 - Comunicación en tiempo real (IRC, IM, y más). (s.f.). Obtenido de Blog tecnológico: http://www.asoma.es
- Cuadro de Mando Integral. (s.f.). Obtenido de Sinnexus: http://www.sinnexus.com
- CUADRO DE MANDO INTEGRAL: metodología Kaplan y Norton. (s.f.). Obtenido de http://www.e-visualreport.com
- **Dávila, A.** (s.f.). Nuevas herramientas de control: El Cuadro de Mando Integral.
- Decide Soft. (s.f.). Obtenido de Decide Soft: http://www.decidesoft.net
- Delphos. (s.f.). Obtenido de Delphos: http://www.delphosinv.com
- Delphos. (s.f.). Obtenido de Denisa: http://www.deinsa.com/delphos

- Diagrama de componentes – UML. (s.f.). Obtenido de El CoDiGo K: http://elcodigok.blogspot.com
- Diagramas de componentes de UML: Referencia. (s.f.). Obtenido de http://msdn.microsoft.com
- Diagramas de Despliegues. (s.f.). Obtenido de http://es.scribd.com
- Diagrama de secuencia. (s.f.). Obtenido de http://www.chuidiang.com
- Diagrama de Secuencia. UML 2. (s.f.). Obtenido de Sparxsystem: http://www.sparxsystems.com.ar
- **E. P. J.** (s.f.). Cuadro de Mando Integral, CMI. Obtenido de http://www.mercadeo.com
- **Gallego, G. J.** (s.f.). Ingeniería de Requerimientos. Obtenido de http://www.scribd.com
- **Garcerant, Iván**. (s.f.). Modelo de Dominio. Obtenido de http://synergix.wordpress.com
- **García, A. R.** (s.f.). COMPONENTE DE COMUNICACIÓN EN TIEMPO REAL DE VALORES DE LOS KPIS DE UN DASHBOARD. (s.f.). Obtenido de http://semanatecnologica.fordes.co.cu
- **Group, S. C.** (s.f.). El diseño del Dashboard: cómo incluir los KPI (indicadores clave de desempeño) y sus métricas. Obtenido de http://www.gestiopolis.com
- Introducción a la Disciplina de Requisitos de RUP. (s.f.). Obtenido de: http://eva.uci.cu
- Introducción a UML. (s.f.). Obtenido de http://www.programacion.com
- **Javier, G.** (s.f.). Google Wabe,comuinicacion en tiempo real. Obtenido de Direccion: http://elsobredelosblogs.com
- **Juristo Natalia, Moreno Ana M., Vegas Sira**. (s.f.). Técnicas de evaluación de software. Obtenido de: http://eva.uci.cu
- **Kaplan, N**. (Enero/Febrero de 1992). Harvard Business Review
- **Larman, C.** (s.f.). UML y Patrones. Introducción al análisis y diseño orientado a objetos y al proceso unificado.2da Edición.
- Mensajería Instantánea. (s.f.). Obtenido de Centro Costadigital Observatorio: http://www.costadigital.cl
- Modelado de Sistemas con UML. (s.f.). Obtenido de http://www.ibiblio.org

- Modelo de Clases. (s.f.). Obtenido de http://www.dcc.uchile.cl
- Importancia de la captura de requerimientos en el proceso productivo del proyecto CISCOP de la facultad regional de artemisa. (s.f.). Obtenido de ArticuloZ: http://www.articuloz.com
- Modelos de dominio. Emplear el patrón de modelo de dominio. (s.f.). Obtenido de http://msdn.microsoft.com
- **Pan, D, Zhu, D y Johnson, K**. (s.f.). Requierements Engeenering Thecniques
- **Pressman, R.** (s.f.). Ingeniería de Software
- Probando software y números de versión. (s.f.). Obtenido de http://www.mundoprogramacion.com
- Pruebas de Stress/Jmeter. (s.f.). Obtenido de http://scrum-qa.blogspot.com
- **Raghavan, Sridhar, Zelesnik, Gregory y Ford, Gary**. (s.f.). Lecture Notes on Requirements Elicitation.
- **Ríos, H. G**. (s.f.). El IRC (Internet Relay Chat). Obtenido de http://www.cabinas.net.
- **Roxana. M. C.** (s.f.). El cuadro de mando integral: MEJORES PRÁCTICAS. Obtenido de http://www.gestiopolis.com
- UML. (s.f.). Obtenido de http://usuarios.multimania.es
- **Visconti, M**. (s.f.). Fundamentos de Ingeniería de Software. Obtenido de http://portal.inf.utfsm.cl

Glosario de Términos

BOSH: *Bidirectional streams Over Synchronous HTTP.*

CMI: Cuadro de Mando Integral del inglés *Balanced Scorecard* (BSC).

Indicador: Magnitud utilizada para medir o comparar los resultados efectivamente obtenidos, en la ejecución de un proyecto, programa o actividad. Resultado cuantitativo de comparar dos variables.

Misión: Constituye la razón de ser de la Organización. Representa la identidad y personalidad de la Organización, en el momento actual y de cara al futuro. Es definir el negocio desde una óptica actual y desde una perspectiva futura.

Nodo: Un lugar virtual en el que la información puede ser publicada y del que las notificaciones de eventos se pueden recibir (en los sistemas de Pubsub, esto puede ser etiquetado como un "tema").

Ratio: Una razón financiera (o la relación de la contabilidad) es una magnitud relativa de los dos valores numéricos seleccionados tomados de los estados financieros de una empresa.

Servicio Pubsub: Un servidor XMPP o componente que se adhiere al protocolo XMPP.

Visión: Es el estado futuro apremiante y deseable para la empresa. Es una declaración de lo que usted y su organización (o grupo) deberán considerar como de mayor importancia para poder crear y hacer realidad ese futuro. Usted definirá el futuro de su organización y consecuentemente trabajará para lograrlo.

XMPP: *Extensible Messaging and Presence Protocol.* Proporciona una vía para enviar piezas pequeñas de XML de una entidad a otra en tiempo real.

XML: Siglas en inglés de *eXtensible Markup Language* (lenguaje extensible de marcas).