



**Universidad de las Ciencias Informáticas.**

**Sistema de Gestión para el Análisis y Monitoreo de los Procesos de Negocio.**

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

**Autor:**

**Ivan Pérez Arencibia.**

**Tutores:**

**Msc. Jofman Pérez Tarancon.**

**Ing. Renier Sotés Mesa.**

Ciudad de La Habana, Cuba.

Junio del 2010.





*Para triunfar en la vida, no es importante llegar primero. Para triunfar simplemente hay que llegar, levantándose cada vez que se cae en el camino.*



## DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor del trabajo titulado: “Sistema de Gestión para el Análisis y Monitoreo de los Procesos de Negocio”, y autorizo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los \_\_\_\_\_ días del mes de junio del 2010.

Ivan Pérez Arencibia

\_\_\_\_\_  
Firma del Autor

Msc. Jofman Pérez Tarancon

\_\_\_\_\_  
Firma del Tutor

Ing. Renier Sotés Mesa

\_\_\_\_\_  
Firma del Tutor



## DEDICATORIA

*A mi mamá y a mi papá, por ser lo más preciado que tengo en la vida, porque siempre me han apoyado, comprendido y sobre todas las cosas me han dado mucho amor.*

*A mis hermanos, porque son mi ejemplo a seguir desde pequeño.*

*A mis abuelos, por darme tanto cariño.*

*En especial a la memoria de mi abuelo Domingo, porque aún te quiero y te extraño.*



## AGRADECIMIENTOS

*A la Revolución, a Fidel y a la Universidad de las Ciencias Informáticas, por dejarme formar parte de este proyecto.*

*A mis viejitos lindos, por saber guiarme por el camino correcto ante las adversidades de la vida, por hacer de mí un hombre preparado y demostrarme que toda meta puede ser alcanzada, solo hay que proponérselo.*

*A mis hermanos, por ser portadores de experiencia y admiración, por estar siempre ahí para aconsejarme y evitarme los tropezones con los cuales ustedes largaron las suelas de sus zapatos.*

*A mi inmensa familia, por siempre brindarme su apoyo incondicional, por ser un regalo hermoso donde se comparten sentimientos de amor y paz.*

*A mi princesa, por saber quererme y estar siempre a mi lado a pesar de los obstáculos de la vida. Cada día te quiero más que ayer y menos que mañana.*

*A Martha, Armando y Ransel, por darme la oportunidad de formar parte de su bella familia.*

*A mis tutores, por la confianza depositada en mí, en especial a Renier Sotés por brindarme su experiencia, ayuda y sus consejos siempre que los necesité, gracias mi hermano.*

*A todos mis amigos y compañeros del preuniversitario y de la UCI con los cuales he compartido y vivido momentos inolvidables.*

*A mis profesores, gracias por hacer de mí un mejor estudiante cada día.*

*A todos mis compañeros del Centro de Identificación y Seguridad Digital, en especial a todos mis amigos del laboratorio 104.*

*Gracias a todas las personas que me han ayudado a construir este sueño.*



## RESUMEN

La información es uno de los principales recursos que poseen las organizaciones actualmente. Para maximizar la utilidad de la información, se debe manejar de forma correcta y eficiente; ya que su uso es estrictamente estratégico para posicionar de forma ventajosa a la empresa dentro del mercado. Para mantener la competitividad y facilitar un mejor servicio, se necesita analizar en tiempo real los valores acumulados a través de gráficos y tablas, y de esta forma lograr la toma rápida de decisiones que conlleven a aprovechar una oportunidad o solucionar algún problema crítico para el negocio. Esto se logra a través de la Monitorización de las Actividades del Negocio o *Business Activity Monitoring* (BAM).

En este trabajo se propone desarrollar un sistema flexible que tiene como objetivo explotar al límite las tecnologías libres, obteniendo una solución robusta, con una capacidad de adaptarse a los diferentes negocios para los cuales se haga útil su implantación y que pueda ser aplicada en cualquier plataforma. Con esta solución se establecerá un medio de comunicación instantáneo entre los procesos de negocios y los encargados de la gestión de los mismos. Será un complemento a la capacidad de toma de decisiones, además, de presentarse como un factor clave en la detención del caos en los distintos negocios que tengan desplegados esta solución BAM, siempre y cuando exista disponible un medio por donde propagarse la información de las métricas en dichos negocios.

**Palabras claves:** gráficos, información, métricas, monitorización de las actividades de negocio, procesos de negocios, tiempo real.



# ÍNDICE

INTRODUCCIÓN.....	1
CAPÍTULO 1. LA INFORMACIÓN EN LAS ORGANIZACIONES.....	5
1.1 Desenvolvimiento de las empresas.....	6
1.1.1 Gestión de los Procesos de Negocio.....	6
1.1.2 Gestión de Servicios en el Negocio.....	7
1.1.3 Inteligencia Empresarial o Inteligencia de Negocios.....	8
1.1.4 Monitorización de la Actividad de Negocio.....	9
1.1.5 Análisis de soluciones existentes.....	12
1.2 Tecnologías a utilizar.....	13
1.2.1 Arquitectura Cliente Servidor.....	13
1.2.2 Las aplicaciones Web.....	14
1.2.3 Lenguajes de Programación Web.....	15
1.2.4 Metodología de desarrollo de software.....	20
1.2.5 Lenguaje Unificado de Modelado.....	21
1.3 Herramientas para el desarrollo de la solución informática.....	22
1.3.1 Servidor Web Sun GlassFish Enterprise Server v3.....	22
1.3.2 Entornos integrados de desarrollo.....	23
1.3.3 Sistema de Gestión de Base de Datos (SGBD).....	23
1.3.4 Herramienta Case.....	24
1.4 Propuesta de solución.....	25
Conclusiones.....	26
CAPÍTULO 2. PRESENTACIÓN DE LA SOLUCIÓN PROPUESTA.....	27
2.1 Modelo de Dominio.....	28
2.2 Diagrama del modelo de dominio.....	28
2.3 Especificación de los requisitos de software.....	28
2.3.1 Requerimientos funcionales.....	29
2.3.2 Requerimientos no funcionales.....	33



2.4 Modelo del sistema .....	36
2.4.1 Especificación de los casos de uso del sistema. ....	38
2.5 Estimación de Esfuerzo.....	38
2.6 Análisis.....	43
2.6.2 Diagrama de Clases del Análisis.....	43
2.7 Diseño.....	44
2.6.1 Descripción de la arquitectura. ....	44
2.6.2 Patrones de Diseño.....	45
2.6.3 Diagrama de Clases del Diseño.....	48
2.6.4 Diagrama de Interacción (secuencia). ....	48
2.6.5 Diseño de la base de datos.....	49
2.6.6 Definiciones de diseño a aplicar. ....	49
Conclusiones.....	51
CAPÍTULO 3. CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA.....	52
3.1 Diagrama de Despliegue.....	53
3.2 Diagrama de Componentes.....	54
3.3 Prueba.....	56
3.3.1 Método de Caja Blanca.....	56
3.3.2 Método de Caja Negra.....	62
Conclusiones.....	65
CONCLUSIONES GENERALES.....	66
RECOMENDACIONES. ....	68
REFERENCIAS BIBLIOGRÁFICAS.....	69
BIBLIOGRAFÍA. ....	72
GLOSARIO DE TÉRMINOS.....	75





## ÍNDICE DE FIGURAS

Figura 1. Gestión de los Servicios de Negocio.....	8
Figura 2. Componentes Principales de GWT.....	19
Figura 3. Diagrama del Modelo de Dominio.....	28
Figura 4. Diagrama de Casos de Uso del Sistema.....	37
Figura 5. Diagrama de Entidad Relación de la Base de Datos.....	49
Figura 6. Diagrama de Despliegue de los Componentes.....	53
Figura 7. Diagrama de Componentes General.....	54
Figura 8. Diagrama de Componentes por Módulos.....	55
Figura 9. Grafo de Flujo del Caso de Prueba Comprobar Conexión con Oracle.....	59
Figura 10. Grafo de Flujo del Caso de Prueba Comprobar Conexión.....	61



## ÍNDICE DE TABLAS

Tabla 2.1 Descripción de los Actores del Sistema. ....	37
Tabla 2.2 Factor de Peso de los Actores sin Ajustar. ....	39
Tabla 2.3 Factor de Peso de los Casos de Uso sin Ajustar. ....	40
Tabla 2.4 Factor de Complejidad Técnica. ....	41
Tabla 2.5 Factor de Ambiente. ....	42
Tabla 2.6 Esfuerzo en Horas-Hombre. ....	43
Tabla 2.7 Prueba de Caja Negra, Funcionalidad "Adicionar Conexión (Oracle)" ....	63
Tabla 2.8 Prueba de Caja Negra, Funcionalidad "Adicionar Conexión (PostgreSQL)" ....	64
Tabla 2.9 Prueba de Caja Negra, Funcionalidad "Modificar Conexión". ....	65
Tabla 2.10 Prueba de Caja Negra, Funcionalidad "Eliminar Conexión". ....	65



# INTRODUCCIÓN.

Ante la fuerte competencia global, las organizaciones han puesto en marcha estrategias que les permiten operar en tiempo real, necesitan rapidez y agilidad en su toma de decisiones, ya que cualquier demora o falta de visión puede suponer la pérdida de una oportunidad de negocio. El mercado exige hoy a las empresas no solo que produzcan más y mejor que sus competidores, sino también que lo hagan de forma diferente, esto es, respondiendo a las necesidades específicas de una demanda tremendamente variable. Anticiparse y reaccionar a tiempo son anhelos compartidos en todos los sectores.

En la actualidad, la convergencia entre distintas tecnologías está haciendo posible que en las organizaciones exista una recolección instantánea de la información de la mayoría de sus operaciones, la cual puede ser analizada y visualizada en cualquier momento para optimizar la gestión. Para lograrlo, se requiere de indicadores de negocio y tecnologías que permiten capturar e integrar la información de las distintas áreas y sus procesos, para que sus directivos y el personal encargado puedan conocer el estado del negocio con la máxima certeza.

Saber exactamente en qué situación se encuentra un proceso de negocio en tiempo real, permite tener más elementos y tomar decisiones de manera más rápida, lo cual beneficia la gestión asociada a la actividad. La automatización de los procesos de negocio en el mundo condicionan la necesidad de contar con un sistema que permita la recolección de las métricas y la visualización de estas en una interfaz de monitoreo.

La recolección de la información operacional, el seguimiento del curso de las métricas para detectar posibles anomalías y su visualización gráfica, constituyen un aspecto importante para cualquier proceso de negocio. La necesidad de lograr la persistencia y el análisis en tiempo real de los valores acumulados, a través de gráficos y tablas, para lograr la toma rápida de decisiones, conlleva al incremento de la competitividad y una mayor eficiencia en los negocios.

La información generada en las transacciones cotidianas, ya sea en los procesos productivos, logísticos y financieros de las organizaciones, no siempre están disponibles en el momento oportuno



para prevenir anomalías, aprovechar una oportunidad o solucionar algún problema crítico para el negocio.

El monitoreo de los procesos de negocio es una tecnología emergente que se encarga de analizar y supervisar las actividades de una empresa, brinda acceso a la información en tiempo real y genera alertas para reaccionar rápido ante eventuales problemas u oportunidades. En julio del 2001, Gartner<sup>1</sup> acuñó un término que conceptualiza esta nueva ola de soluciones tecnológicas para apoyar la gestión. Se trata de la Monitorización de la Actividad de Negocio o *Business Activity Monitoring* (BAM), el cual ha captado el interés de las organizaciones y ha planteado a la industria de las Tecnologías de la Información (TI) un nuevo desafío.

La tecnología BAM tiene características distintivas dentro del grupo de las TI, una de ellas es que emplea inteligencia de negocios operacional y aplicaciones de tecnologías de integración para monitorizar procesos y mejorarlos continuamente, basándose en la información que proviene directamente del conocimiento de los eventos operacionales.

El presente trabajo se centra en un estudio exhaustivo de las herramientas para la monitorización de las actividades de negocio, sus principales características y funcionalidades. Esta información será la base para el diseño y la implementación de un “Sistema de Gestión para el Análisis y Monitoreo de los Procesos de Negocio”, el cual permitirá analizar y monitorizar las actividades de una empresa, brindando información en tiempo real y generando alertas que permitan reaccionar ante cualquier decisión importante.

Para la realización de la investigación se hace necesario plantear el siguiente **problema científico**: ¿Cómo analizar y monitorizar la información generada en procesos de negocios dentro de una organización?

En aras de dar solución al problema planteado se define como **objetivo general**: desarrollar un sistema capaz de analizar y monitorizar mediante los indicadores claves de desempeño el control de situaciones de riesgo u oportunidades de éxito en los procesos de negocio.

Para dar cumplimiento a lo formulado en el objetivo general se trazan los siguientes **objetivos específicos**.

---

<sup>1</sup> Gartner Group: consultora americana especializada en tecnologías de la información.



- Diseñar una arquitectura que posibilite la inmediatez de disponibilidad de la información y la eficiencia, escalabilidad y confiabilidad en la solución del software.
- Desarrollar una herramienta capaz de analizar y monitorizar el comportamiento de las métricas generadas en procesos de negocio.
- Lograr un producto que pueda ser utilizado en todas las organizaciones.

**Objeto de Estudio:** Administración<sup>2</sup> de los procesos de negocios.

**Campo de Acción:** Análisis y monitoreo en tiempo real de los indicadores claves de desempeño generados en los procesos de negocio dentro de una organización.

**Hipótesis:** El desarrollo del sistema de gestión para el análisis y monitoreo de los procesos de negocio permitirá el acceso en tiempo real a la información para detectar y responder rápidamente a los eventos que están siendo monitorizados.

Se definen como **Tareas** a seguir en la investigación:

- Caracterización de sistemas de monitoreo de los procesos de negocio existentes en la actualidad.
- Especificación de los requisitos del software.
- Definición de funcionalidades del sistema.
- Definición de la propuesta del sistema.
- Definición de las capas lógicas dentro del sistema.
- Identificación de las herramientas y tecnologías a utilizar.
- Diseño del sistema.
- Implementación de las funcionalidades del sistema.
- Aplicación de pruebas de calidad al sistema.

**Métodos de Investigación:**

*Teóricos.*

- *Histórico – lógico:* Se realizará un estudio histórico-lógico de la evolución de las herramientas que permiten el análisis y monitoreo de los procesos de negocio, y de la documentación de los

---

<sup>2</sup> Administración: automatizar, monitorizar, gestionar y optimizar.



sistemas desarrollados anteriormente para encontrar una solución a los problemas y lograr entender la lógica interna de su funcionamiento.

- *Analítico – lógico:* Se podrá realizar un análisis de toda la información obtenida a través de los diferentes medios bibliográficos. Por esta vía se identificarán los principales hitos para lograr el éxito en el desarrollo del sistema.
- *Inducción – deducción:* Este método ayudará a estructurar el conocimiento científico a partir de la revisión bibliográfica.

El documento se encuentra estructurado en tres capítulos.

**Capítulo I:** La información en las organizaciones. En este capítulo se exponen los conceptos fundamentales que sustentan la investigación. Incluye el análisis de la información existente acerca de las tendencias y tecnologías que existen en el mundo de los negocios empresariales en la actualidad.

**Capítulo II:** Presentación de la solución propuesta. Se describe el objeto de estudio en términos de negocio, se especifican los detalles de la construcción del sistema, requerimientos funcionales y no funcionales y los casos de uso del sistema. Se modelan los diagramas de clases del análisis y del diseño para cada caso de uso, con sus correspondientes diagramas de interacción, el modelo de la base de datos y otras restricciones del diseño.

**Capítulo III:** Construcción de la solución propuesta. Se describe la distribución física del sistema mediante el modelo de despliegue y el modelo de componentes. Se realizan pruebas al producto final con el objetivo de garantizar la calidad del sistema.



## **CAPÍTULO 1.**

# **LA INFORMACIÓN EN LAS ORGANIZACIONES.**

---

Para mejorar la competitividad, las organizaciones deben adecuar sus estrategias al entorno dinámico de la economía actual. El éxito de la empresa dependerá, en gran medida, de su rapidez y capacidad para tomar decisiones ante cualquier anomalía que produzca una oportunidad de éxito o ponga en riesgo al negocio.

La información se ha colocado en un lugar cimero como uno de los principales recursos que poseen las empresas para tomar decisiones. Para maximizar la utilidad que posee la información, se debe manejar de forma correcta y eficiente; ya que su uso es estrictamente estratégico para posicionar de forma ventajosa a la empresa dentro del mercado. Es por esto que es necesario conceptualizar elementos claves que permitan analizar y monitorizar la información generada en las organizaciones.



### 1.1 Desarrollo de las empresas.

Las empresas se mueven cada vez a mayor velocidad. Esta necesidad de operar a velocidades cada vez mayores puede tener varias causas. En el sector financiero, se debe a la necesidad de supervisar de forma precisa y en el tiempo requerido las transacciones realizadas. Este periodo no es cuestión de días, ni de semanas, sino que se miden en horas, incluso segundos. En la rama de las telecomunicaciones, la necesidad de operar con rapidez tiene que ver con la posibilidad de aprovecharse de la demanda cada vez mayor de servicios y fijar precios en tiempo real. Es posible, además, considerar la necesidad de actuar con prisa, como un requisito cada vez más universal que las empresas deben satisfacer para poder batir a sus competidores, y al mismo tiempo, construir y consolidar su posición en el mercado.

Muchas empresas se están esforzando por operar con mayor rapidez. Pero esta celeridad es una ventaja sólo en tanto y en cuanto se sepa gestionar, ya que la velocidad descontrolada es tan peligrosa para una empresa como una lentitud exagerada. Es por ello, que las empresas más avanzadas realizan inversiones para poder administrar a mayor velocidad sus operaciones y, de este modo, responder al ritmo de cambio cada vez más rápido que se les impone.

Para lograr sus objetivos las empresas realizan un conjunto de actividades las que pueden estar ciento por ciento relacionadas a la actividad principal, o bien ser procesos auxiliares que permiten el correcto desenvolvimiento de las actividades principales. A este conjunto de actividades que una organización realiza para generar un producto o servicio dado, se le llama procesos de negocio.

#### 1.1.1 Gestión de los Procesos de Negocio.

La competitividad incide en la forma de plantear y desarrollar cualquier iniciativa de negocio. Lograr condiciones que permitan competir con mayores oportunidades, exige de las empresas desarrollar ventajas competitivas en su forma de operar. La fuente de estas ventajas está en las actividades que desarrolla, por lo que la eficiencia en los procesos de negocio representa un foco de acción para sus directivos. (Group, 2008)

Un proceso de negocio es un conjunto estructurado de actividades, diseñado para producir una salida determinada o lograr un objetivo; describe como es realizado el trabajo en la empresa y se caracteriza por ser observable, medible, mejorable y repetitivo. Es la base operativa de una organización y el éxito de la misma depende fuertemente de la eficiencia con que sea gestionado. Una mala gestión de los





procesos de negocios trae aparejados altos costes, baja productividad e inadecuados tiempos de respuesta, tanto frente a las oportunidades como a las amenazas.

En la actualidad, los procesos de negocio requieren ser gestionados independientemente de un dominio específico de un sistema. Ellos constituyen el foco y la unidad primaria de iniciativas de automatización e integración de información. Son necesarios para responder ágilmente a los cambios exigidos por la dinámica del mercado. La administración de procesos de negocio, en estas condiciones, ha dado origen a una nueva etapa en la gestión de procesos, denominada: Gestión de los Procesos de Negocio o *Business Process Management* (BPM).

La gestión de los procesos de negocio es la metodología empresarial cuyo objetivo es mejorar la eficiencia a través de la gestión sistemática (...) se deben modelar, automatizar, integrar, monitorizar y optimizar de forma continua. Se enfoca en la administración de los procesos del negocio. (Group, 2008)

Howard Smith por su parte, define la gestión de los procesos de negocio como una nueva aproximación para abordar y gestionar procesos de innovación en las compañías, los cuales construyen el mejoramiento a partir del estado actual de un proceso en un momento determinado y plantea una diferencia radical frente a la reingeniería; la cual construye el mejoramiento desde la redefinición total del proceso. En esta óptica BPM se convierte en una respuesta al caos operativo que presentan las compañías en la actualidad. (Smith, et al., 2003)

BPM es el entendimiento, gestión e innovación de los procesos alineados con la estrategia de negocio para asegurar la efectividad del proceso, y crear valor a la cadena productiva de la organización y a su sector. Constituye un nuevo paradigma para abordar procesos de mejoramiento que aumenten la eficiencia y facilidad de integración entre diferentes compañías. (Smith, et al., 2003)

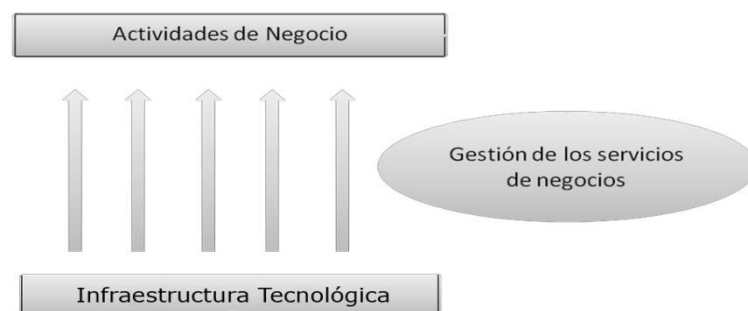
### **1.1.2 Gestión de Servicios en el Negocio.**

Las empresas mantienen la necesidad de seguir creciendo, evolucionando y compitiendo para ofrecer al mercado mejores servicios, infraestructuras más adecuadas y por supuesto alcanzar el rol más alto. Para lograr este objetivo, es fundamental que tecnología y negocio vayan de la mano, es decir, que todas las decisiones tecnológicas respondan a un objetivo claro del mismo proceso. A medida que las arquitecturas tecnológicas se tornan más complejas, es constante y se hace más fuerte la necesidad de utilizar la Gestión de Servicios en el Negocio o *Business Service Management* (BSM).



Según Peter Armstrong la gestión de servicios en el negocio se define como una estrategia de software para administrar la infraestructura tecnológica desde la perspectiva de negocios. Al implementar BSM, las empresas pueden alinear rápidamente sus operaciones para apoyar directamente los objetivos críticos del negocio. Esta estrategia flexible, ayuda a las compañías a identificar y optimizar los procesos claves en sus empresas y es, además, un paso importante, debido a que la calidad de los servicios críticos del negocio puede impactar el éxito del mismo. (Armstrong, 2007)

Esta disciplina juega un papel cada vez más importante en las organizaciones y tiene como objetivo relacionar los servicios de negocio con la infraestructura tecnológica que los soporta. El enfoque que presenta está completamente orientado a maximizar éxitos en el negocio, teniendo una visión clara de la tecnología que se utiliza, así como, de la distribución de los componentes en una arquitectura cada vez más compleja y heterogénea. Al contar con una información precisa de la relación entre la tecnología y el negocio, se pueden tomar decisiones estratégicas conociendo de antemano el impacto que provocarán. Estos servicios del negocio abarcan múltiples aplicaciones, bases de datos y redes en toda la infraestructura tecnológica.



**Figura 1. Gestión de los Servicios de Negocio.**

Con el objetivo de ofrecer conocimientos para respaldar la toma de decisiones estratégicas, las empresas utilizan la información generada en sus procesos de negocio a través de la inteligencia empresarial.

### **1.1.3 Inteligencia Empresarial o Inteligencia de Negocios.**

La información es la clave para obtener una ventaja en el mundo de los negocios. Para mantenerse competitiva una empresa, los gerentes y tomadores de decisiones requieren de un acceso rápido y fácil a la información útil. Si malo es no tener información disponible, peor es tener mucha información



y no saber qué hacer con ella. La solución a este problema es por medio del uso de la Inteligencia Empresarial o Inteligencia de Negocio, su término en inglés es *Business Intelligence* (BI).

La inteligencia de negocio es la combinación de herramientas, técnicas y metodologías que apoyadas en las tecnologías de información, facilitan la explotación y el análisis de información, para convertirla en conocimiento y con ello apoyar a la toma de decisiones. BI nace de la necesidad de poder contar con información relevante de cualquier área de la empresa, proveniente de diferentes fuentes de datos de manera rápida, oportuna y fácil, enfocada a aquellos usuarios cuya responsabilidad es la de dirigir una organización. (Consulting, 2010)

La inteligencia de negocio, asociada directamente a las tecnologías de información, se puede definir como el conjunto de herramientas, metodologías, aplicaciones y tecnologías que permiten reunir, depurar y transformar datos de los sistemas transaccionales e información desestructurada (interna o externa a la compañía) en información estructurada, para su explotación directa o para su análisis y conversión en conocimiento soporte a la toma de decisiones sobre el negocio. (ICES, 2009)

La inteligencia de negocio es la transformación de los datos de la compañía en conocimiento para obtener una ventaja competitiva. Con BI se puede crear una base de datos de clientes, prever ventas y devoluciones, compartir información entre diferentes departamentos y mejorar el servicio al cliente.

### **1.1.4 Monitorización de la Actividad de Negocio.**

En el mundo actual altamente competitivo, las organizaciones requieren la visibilidad de los procesos empresariales, a través de los indicadores de negocio. Para mantener la competitividad y facilitar un mejor servicio, se necesita analizar en tiempo real los valores acumulados a través de gráficos y tablas, y de esta manera lograr la toma rápida de decisiones que conlleven a aprovechar una oportunidad o solucionar algún problema crítico para el negocio. Esto se logra a través de la Monitorización de las Actividades del Negocio o *Business Activity Monitoring* (BAM).

Las soluciones de monitorización de la actividad del negocio, primas de la tecnología de gestión de procesos de negocio (BPM), se incluyen dentro de un campo emergente, que promete incrementar la competitividad y la toma rápida de decisiones en las organizaciones que las implementen, gracias a la visualización de los procesos de negocio en tiempo real. (COMMUNICATIONS, 2009) Con la



monitorización de las actividades de negocio, se podría evitar cuellos de botella<sup>3</sup> en los procesos de entrega y responder rápidamente para suministrar pedidos inesperados o reconocer oportunidades de negocios potenciales antes de que lo haga la competencia. (Microsoft, 2009)

Las métricas y los indicadores claves de desempeño son la base para construir un tablero de mando de gran despliegue visual, ya que son las herramientas más eficaces para alertar a los usuarios de cómo se encuentran situados en relación a los objetivos; por ello es importante contar con una clara definición de estos elementos que constituyen la base del diseño de un tablero de mando.

Una métrica es una medida numérica directa, que representa un conjunto de datos generados en los procesos de negocio en relación a una o más dimensiones. Un ejemplo sería: “las ventas brutas por semana”. En este caso, la medida sería en pesos (ventas brutas) y la dimensión sería el tiempo (semana).

Un Indicador clave de desempeño es un indicador que está vinculado a un objetivo. Posee un estado que indica si está por encima o por debajo de una meta pre determinada. Generalmente se muestran en una tasa o porcentaje, y están diseñados para permitir que un usuario de negocio pueda saber instantáneamente si sus objetivos están dentro o fuera de su plan, sin que tenga que entrar directamente a analizar las métricas. Por ejemplo, de acuerdo a lo planteado por (López, 2008), a fin de lograr nuestro objetivo de ventas trimestrales, tenemos que vender \$10000 de dispositivos por semana. La métrica sería “venta de dispositivos por semana”, la meta sería de \$10000. Si se utiliza un porcentaje de visualización para representar este indicador clave de desempeño y se hubiera vendido \$8000 de dispositivos el día miércoles, el usuario podría ver al instante que estaría en el 80% de su objetivo. (López, 2008)

Un tablero de mando es un tipo de interfaz de usuario interactivo diseñado para proporcionar al usuario información específica relativa al estado de la empresa, representado normalmente a través de indicadores claves de desempeño y enlaces a informes relevantes. Existen señales visuales, gráficos y controles de proceso que centran la atención del usuario en las tendencias, cambios y excepciones importantes.

Monitorizar un proceso de negocio sirve para incrementar la efectividad y la rapidez de análisis de las operaciones. Con la utilización del acceso en tiempo real a los valores acumulados a través de los

---

<sup>3</sup> Cuello de botella: es una operación o proceso dentro de un flujo de procesos (varias operaciones) cuya carga de trabajo a realizar supera en tiempo al resto de las operaciones anteriores y/o posteriores.



tableros de mando, los que tienen como propósito proporcionar al usuario información de la empresa en formato que es a la vez intuitivo y perspicaz, enfatizan datos operativos sobre todo en forma de cifras e indicadores claves de desempeño. De esta forma, permite a los gestores del negocio realizar las acciones correctivas necesarias en el momento oportuno o bien, les permite refinar y optimizar los procesos de negocio.

La monitorización de las actividades de negocio proporciona a los ejecutivos una perspectiva unificada del negocio con rapidez, y les permite combinar en interfaces gráficas de fácil interpretación, datos de diversas fuentes, lo que facilita el análisis de los procesos del negocio que necesitan acometer. Por otro lado, vale la pena destacar que una de las capacidades más interesantes de un sistema BAM reside en que puede detectar situaciones a partir de distintos eventos aparentemente no relacionados entre sí.

BAM requiere establecer reglas precisas que permitan disparar alarmas, cuando exista alguna tendencia o indicio de algún problema potencial. Dichas alarmas pueden tomar forma de correo electrónico, un mensaje a un celular, beeper o cualquier otro mecanismo de notificación, lo que posibilita una reacción inmediata. Incluso, las alertas pueden estar asociadas a otras acciones, como el envío de un e-mail a determinadas personas dentro de la empresa, o bien a proveedores, clientes o socios de negocio. (WebMethods, 2006)

La clave del BAM consiste en disponer de indicadores claves de desempeño apropiados a la actividad del negocio, capturar los datos (métricas) que los componen en tiempo real e integrarlos para construir una visualización clara (tableros de mando) sobre el estado del negocio.

La importancia del BAM radica en que ayuda a la monitorización de los procesos de negocio y al mejoramiento económico de una organización. Provee información en tiempo real acerca del estado y los resultados de diversas operaciones, procesos y transacciones. De esta manera, los usuarios pueden seguir el rendimiento de negocios actuales, comparando los resultados reales con las expectativas y objetivos, lo que permite el análisis de tendencias en el tiempo. BAM también puede ayudar cuando algo sale mal, así como en situaciones cuando las expectativas no se cumplen. Logra ayudar a que una organización sea consciente de posibles problemas a tiempo, lo que permite que una acción dirigida pueda ser planeada y llevada a cabo.



### 1.1.5 Análisis de soluciones existentes.

En el mundo hay muchas empresas que se dedican al desarrollo de soluciones BAM, las que son dirigidas a algunos problemas de integración; pero son propietarias y caras. Estas van desde soluciones costosas para la compra<sup>4</sup> hasta soluciones personalizadas<sup>5</sup>. La desventaja aplastante de estas soluciones son el alto coste y la baja flexibilidad debido a implementaciones no estándares. Entre ellas encontramos:

**Oracle Business Activity Monitoring (Oracle BAM)** es una solución completamente propietaria para la construcción de tableros de mando interactivos, en tiempo real y con alarmas que permiten monitorizar los procesos y servicios de negocio. Oracle BAM permite a los directivos de negocio y directores de operaciones obtener la información que necesitan, para tomar mejores decisiones de negocio y llevar a cabo acciones correctivas si el entorno de negocio cambia. También puede identificar cuellos de botella integrando el BAM con sus sistemas existentes para hacer el seguimiento de los procesos y capturar los eventos de negocio, reconocer de manera inmediata, e incluso predecir los problemas que se dan en las operaciones a través del seguimiento de los eventos apropiados en sus sistemas de tecnologías de la información. (Oracle, 2009)

**TIBCO One™**, solución centrada en la experiencia del usuario para la creación y despliegue de aplicaciones de infraestructura en entornos de TI heterogéneos y distribuidos. Dicho sistema refuerza y amplía las ventajas de la arquitectura orientada a servicios, la gestión de procesos de negocio y las aplicaciones basadas en arquitecturas orientadas a eventos. Tiene la capacidad de analizar y monitorizar los procesos de principio a fin y hacerlo en tiempo real. (TIBCO, 2009)

**BusinessBridge Business Vision** es una solución estática, que propone unos tableros de mandos con un nivel de aceptación grande para el mundo empresarial, pero su fundamento es solo probabilístico, lo que hace una solución poco flexible, sin ninguna perspectiva de intercambio o mejoras por parte de los usuarios finales. Es una solución propietaria y con altos costes por conceptos de licencias.

**WSO2 BAM** es una solución software libre diseñada para monitorizar las actividades de negocio, destinada a cubrir las necesidades de los expertos en tecnologías de la información para controlar y entender las actividades de negocio dentro de una implementación basada en la arquitectura orientada a servicios. (WSO2, 2010)

---

<sup>4</sup> Alto coste, compras cerradas.

<sup>5</sup> Alto mantenimiento, alto coste.



Las empresas interesadas en aprovechar al máximo su infraestructura informática enfrentan numerosos problemas, que se pueden resumir como complejidad, visibilidad, priorización y costes. Mientras la infraestructura crece en complejidad con la integración de múltiples servidores, aplicaciones, base de datos y dispositivos, se hace más difícil comprender las prioridades del negocio y visualizar la forma en que un pequeño fallo en un componente tecnológico puede tener un impacto grave en un servicio de negocio completo.

La infraestructura de una empresa puede ser muy sólida, aún así, la no existencia de un tablero de mando capaz de identificar estados de crisis dentro de los procesos, y la demora en los tiempos de análisis de la información generada en los procesos de negocio, la vuelve deficiente, por lo que se hace necesario la existencia de un sistema automatizado que contribuya a mejorar la calidad de los servicios que pueda servir de solución a cualquier organización.

### **1.2 Tecnologías a utilizar.**

#### **1.2.1 Arquitectura Cliente Servidor.**

La arquitectura cliente-servidor llamada modelo cliente-servidor o servidor-cliente, es una forma de dividir y especializar programas y equipos de cómputo a fin de que la tarea que cada uno de ellos realiza se efectúe con la mayor eficiencia, y permita simplificarlas. (Madrid, 2007)

La arquitectura Cliente/Servidor es la integración distribuida de un sistema en red, con los recursos, medios y aplicaciones que, definidos modularmente en los servidores, administran, ejecutan y atienden las solicitudes de los clientes; todos interrelacionados física y lógicamente, compartiendo datos, procesos e información. Se establece así un enlace de comunicación transparente entre los elementos que conforman la estructura.

El sistema al estar basado en la web responde a este tipo de arquitectura y tiene como características:

- Orientado a servicios. El servidor los ofrece y el cliente los consume.
- Compartición de recursos. Servicios ofrecidos a muchos clientes. Un servidor puede atender muchos clientes que solicitan esos servicios.
- Transparencia de ubicación. El servidor es un proceso que puede residir en el mismo ordenador que el cliente o en uno distinto a lo largo de una red. Un programa puede ser un servidor en un momento y convertirse en un cliente posteriormente.



- Mezcla e igualdad. Esta es una de las más importantes ventajas de este paradigma. Una aplicación cliente/servidor, idealmente es independiente del hardware y de sistemas operativos; mezclando e igualando estas plataformas.
- Interacción a través de mensajes, para envío y respuesta de servicios.
- Servicios encapsulados, exponiendo los servicios a través de interfaces, lo que facilita la sustitución de servidores sin afectar a los clientes; permitiendo a la vez una fácil escalabilidad. (Madrid, 2007)

Ventajas de la arquitectura cliente/servidor:

- El servidor no necesita potencia de procesamiento, parte del proceso se reparte con los clientes.
- Se reduce el tráfico de red considerablemente. Idealmente, el cliente se conecta al servidor cuando es estrictamente necesario, obtiene los datos que necesita y cierra la conexión dejando la red libre.

### 1.2.2 Las aplicaciones Web.

Una aplicación web es aquella que los usuarios, usando un navegador, acceden a un servidor web a través de Internet o de una intranet. Las aplicaciones web son populares debido a la facilidad de uso del navegador web como cliente ligero. La habilidad, para actualizar y mantener aplicaciones web sin distribuir e instalar software en miles de potenciales clientes, es otra razón de su popularidad.

Las aplicaciones web generan dinámicamente una serie de páginas en un formato estándar como HTML o XHTML, soportado por navegadores web comunes. Se utilizan lenguajes interpretados del lado del cliente, tales como JavaScript, para añadir elementos dinámicos a la interfaz de usuario.

Las aplicaciones Web son una especialización y concreción de las aplicaciones cliente/servidor, o sea, su arquitectura general es la de un sistema cliente/servidor, donde tanto el cliente (el navegador) como el servidor (el servidor Web), y el protocolo, mediante el que se comunican (el HTTP: HyperText Transfer Protocol) son estándar, y no han de ser creados por el desarrollador. La parte del cliente de las aplicaciones Web está formada por el código HTML (HyperText Markup Language) que forma la página Web, con opción a código ejecutable, mediante los lenguajes script de los navegadores (JavaScript, VBScript, PerlScript) o mediante pequeños programas (applets) en Java. La parte del





servidor está formada por un programa o script que es ejecutado por el servidor Web, y cuya salida se envía al navegador del cliente. (Información, 2008)

Para el desarrollo del sistema se tiene presente las ventajas de las Aplicaciones Web:

- **Multiplataforma:** Con un solo programa, un único ejecutable, nuestras aplicaciones pueden ser utilizadas a través de múltiples plataformas, tanto de hardware como de software.
- **Actualización instantánea:** Debido a que todos los usuarios de la aplicación hacen uso de un sólo programa que radica en el servidor, los usuarios siempre utilizarán la versión más actualizada del sistema.
- **Suave curva de aprendizaje:** Los usuarios, como utilizan la aplicación a través de un navegador, hacen uso del sistema tal como si estuvieran navegando por Internet, por lo cual su acceso es más intuitivo.
- **Fácil de integrar con otros sistemas:** Debido a que se basa en protocolos estándares, la información manejada por el sistema puede ser accedida con mayor facilidad por otros sistemas.
- **Acceso móvil:** El usuario puede acceder a la aplicación con la única restricción de que cuente con un acceso a la red privada de la organización o a Internet, dependiendo de las políticas de dicha organización; puede hacerlo desde una computadora de escritorio, una laptop o desde una agenda electrónica; desde su oficina, hogar u otra parte del mundo.

Aunque muchas variaciones son posibles, una aplicación web está comúnmente estructurada como una aplicación de tres-capas. En su forma más común, el navegador web es la primera capa, un motor que usa alguna tecnología web dinámica (ejemplo: GGI, PHP, Java Servlets o ASP) es la capa del medio, y una base de datos como última capa. El navegador web manda peticiones a la capa media, que la entrega valiéndose de consultas y actualizaciones a la base de datos y genera una interfaz de usuario.

### **1.2.3 Lenguajes de Programación Web.**

La diferencia fundamental de Internet a otros medios de comunicación es la interacción y personalización de la información con el usuario. Esto se logra por medio de alguno de los diferentes lenguajes de programación para Web que existen hoy en día. Dichos lenguajes se clasifican en dos partes fundamentales: los lenguajes del lado del Servidor y los lenguajes del lado del Cliente.



Entre los lenguajes del lado del servidor podemos encontrar entre los más sobresalientes por el auge que estos han tenido, algunos como PERL, ASP, PHP, Java, entre otros. Estos se caracterizan por desarrollar la lógica de negocio dentro del Servidor, además de ser los encargados del acceso a Bases de Datos, tratamiento de la Información, etc. Del lado del cliente se encuentra principalmente el JavaScript, encargado de aportar dinamismo a la aplicación en los navegadores. Seguidamente se analiza el lenguaje utilizado para el desarrollo de este trabajo.

### **Lenguaje de programación Java.**

Una de las principales características de Java es que es un lenguaje independiente de la plataforma. Eso quiere decir que si se hace un programa en Java podrá funcionar en cualquier ordenador del mercado. Es una ventaja significativa para los desarrolladores de software, pues antes se debía hacer un programa para cada sistema operativo, por ejemplo Windows, Linux, Apple. Esto lo consigue porque se ha creado una Máquina de Java para cada sistema que hace de puente entre el sistema operativo y el programa de Java y posibilita que este último se entienda perfectamente. La independencia de plataforma es una de las razones por las que Java es interesante para Internet, ya que muchas personas deben tener acceso con ordenadores distintos.

Java es un lenguaje potente, seguro y universal gracias a que lo puede utilizar todo el mundo y es gratuito. Uno de los primeros triunfos de Java fue que se integró en el navegador Netscape y permitía ejecutar programas dentro de una página web, hasta entonces impensable con el HTML. (Oracle, 2009)

Se decide utilizar este lenguaje de programación por las virtudes que posee. Como son:

- Brinda facilidad de aprendizaje del lenguaje.
- Gran cantidad de funciones desarrolladas (Java incorpora más de 1000 funciones).
- Tiene una amplia disponibilidad de secuencia de comandos regeneradas en Java.
- Tiene la capacidad de incrustar código JavaScript en las páginas HTML, además se vincula fácilmente con los principales gestores de bases de datos.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad.
- Es multiplataforma y multisistema.
- Posee una amplia documentación.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.



- Permite las técnicas de Programación Orientada a Objetos.
- Miles de ejemplos y código fuente disponible.
- No depende de un único proveedor de servicios.
- El código fuente es abierto y gratuito.

### **Lenguaje de programación JavaScript.**

JavaScript es un lenguaje interpretado, con una sintaxis semejante a la de los lenguajes Java y C. Su diseño no le permite considerarse un lenguaje puramente orientado a objetos como es el caso de Java.

Fue desarrollado por la empresa *Netscape Communications*. Es muy utilizado para controlar la apariencia y manipular los eventos dentro de la ventana del navegador Web, así como para validar datos de entrada en las interfaces de las aplicaciones.

Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado. Con JavaScript se pueden crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones JavaScript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único con que cuenta este lenguaje, es el propio navegador. (Valdés, 2009)

Para interactuar con una página Web se provee al lenguaje JavaScript de una implementación del Modelo de Objetos de Documento, frecuentemente abreviado DOM, es una interfaz de programación de aplicaciones (API) para documentos HTML (páginas Web) y XML. Define la estructura lógica de los documentos y el modo en que se accede y manipula un documento. En la especificación del DOM, el término documento se utiliza en un sentido amplio ya que el documento es el contenedor que soporta los demás elementos. A través del DOM los programadores pueden construir documentos, navegar por su estructura, añadir, modificar o eliminar elementos y contenido. Se puede acceder a cualquier elemento que se encuentre en un documento HTML o XML, y se puede modificar, eliminar o añadir usando DOM, salvo algunas excepciones.

### **AJAX: Una técnica de desarrollo Web.**

JavaScript y XML Asíncronos (*Asynchronous JavaScript And XML*, en lo adelante AJAX) es una técnica de desarrollo Web para crear aplicaciones interactivas. Estas se ejecutan en el cliente y mantienen comunicación asíncrona con el servidor en segundo plano. De esta forma es posible



realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma. (Eguíluz Pérez, 2008)

AJAX no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes. Las tecnologías que conforman AJAX son:

- XHTML (o HTML) y CSS para el diseño que acompaña a la información.
- DOM accedido con un lenguaje de scripting por parte del usuario, como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.
- El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor Web.
- XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML preformateado, texto plano.

Propiedades:

- Peticiones y respuestas HTTP.
- Código de lado del cliente usando JavaScript.
- Programación de lado del servidor en Java.
- Transferir y procesar datos mediante el uso de XML.

Las aplicaciones que utilizan técnicas AJAX no necesitan refrescar la página completa para actualizar la información, pueden simplemente actualizar parte de la página en cualquier momento, dándoles a los usuarios una respuesta instantánea a sus ingresos y consultas. Esto les permite a los usuarios ver continuamente con lo que están trabajando y reaccionar a cualquier cambio, error, o actualización que la interfaz les notifique.

En la web, es fácil chequear los campos en el lado del cliente utilizando JavaScript. Esto produce un efecto inmediato, y replica el comportamiento de una aplicación de escritorio. Sin embargo, por razones de seguridad, es necesario chequear todos los campos también del lado del servidor. Afortunadamente, AJAX permite también que esto suceda. (Eguíluz Pérez, 2008)

### **Google Web Toolkit.**

Google Web Toolkit es un marco de trabajo (*Framework*) de desarrollo Java, basado en software libre y que permite escribir aplicaciones AJAX fácilmente. GWT, permite escribir las aplicaciones en el lenguaje de programación Java, y luego se encarga de compilarlo, traduciendo la parte del cliente al



lenguaje de programación JavaScript + HTML + CSS, generando código JavaScript más eficiente que el escrito a mano. (Google, 2009)

Las aplicaciones generadas por GWT ejecutan código Java del lado del servidor, utilizando las llamadas a procedimientos remotos (RPC) para la comunicación entre el Cliente y el Servidor, llevando a cabo llamadas asíncronas. (Google, 2009)

Una aplicación GWT puede ser ejecutada en dos modos:

- Modo host (Hosted mode): La aplicación se ejecuta como código bytecode de Java dentro de la Máquina Virtual de Java (JVM). Este modo es el más usado para desarrollo, soportando el cambio de código en caliente y el depurado.
- Modo web (Web mode): La aplicación se ejecuta como código JavaScript y HTML puro, compilado a partir del código Java. Este modo se suele usar para el despliegue de la aplicación. (Dewsbury, 2008)

Posee cuatro componentes principales: un compilador Java-a- JavaScript, un navegador web “hosted” y dos librerías de clases.

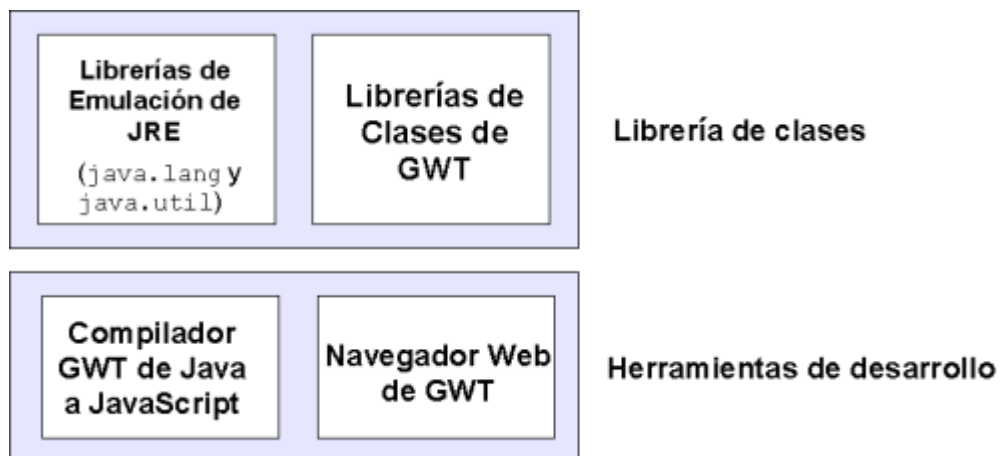


Figura 2. Componentes Principales de GWT.

Descripción de los componentes.

- **Compilador GWT Java-a-JavaScript.**
  - El Compilador GWT Java-a-JavaScript traduce del lenguaje de programación Java a JavaScript.
  - El compilador se utiliza cuando se necesita correr la aplicación en modo web.



- Navegador web “Hosted” de GWT.

El Navegador web “Hosted” de GWT te permite correr y ejecutar aplicaciones GWT en modo hosted, donde lo que estás corriendo son bytecodes de Java sobre una máquina virtual sin compilarlos a JavaScript. Para lograr esto, el navegador GWT incrusta un controlador de browser especial (un control del Internet Explorer sobre Windows o un control de Gecko/Mozilla sobre Linux) dentro de la máquina virtual de Java.

- Emulación de librerías JRE.

GWT contiene implementaciones en JavaScript de las librerías de clases más usadas en Java, incluyendo la mayoría de las clases del paquete `java.lang` y un subconjunto de clases del paquete `java.util`. El resto del estándar de librerías de Java no es soportado nativamente con GWT. Por ejemplo, las clases de los paquetes como `java.io` no se utilizan en aplicaciones web ya que estas acceden a recursos en la red y al sistema de archivos local.

- Librería de clases de interfaz de usuario de GWT.

Las librerías de clases de interfaz de usuario de GWT son un conjunto de interfaces y clases personalizadas que te permiten crear "widgets" para el navegador, como botones, cajas de texto, imágenes. Este es el núcleo de las librerías de interfaz de usuario para crear aplicaciones GWT. (Dewsbury, 2008)

### 1.2.4 Metodología de desarrollo de software.

Para desarrollar, se decidió utilizar como metodología el Proceso Unificado de Desarrollo, por sus características y las facilidades que aporta a todo el proceso. El Proceso Unificado de Desarrollo o *Rational Unified Process* (RUP), es un proceso de desarrollo de software que en unión con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Es un proceso de software genérico que puede ser utilizado para una gran cantidad de tipos de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de competencia y diferentes tamaños de proyectos. Provee un enfoque disciplinado en la asignación de tareas y responsabilidades dentro de una organización de desarrollo. Su meta es asegurar la producción de software de muy alta calidad que satisfaga las necesidades de los usuarios finales, dentro de un calendario y presupuesto predecible. (Jacobson, et al., 2000)

Las características del ciclo de vida de RUP:



- Guiado/Manejado por casos de uso: La razón de ser de un sistema de software es servir a usuarios ya sean humanos u otros sistemas; un caso de uso es una facilidad que el software debe proveer a sus usuarios. Los casos de uso reemplazan la antigua especificación funcional tradicional y constituyen la guía fundamental, establecida para las actividades a realizar durante todo el proceso de desarrollo incluyendo el diseño y la implementación.
- Centrado en la arquitectura: La arquitectura involucra los elementos más significativos del sistema y está influenciada entre otros por plataformas de software, sistemas operativos, manejadores de bases de datos, protocolos, consideraciones de desarrollo como sistemas heredados y requerimientos no funcionales. Los casos de uso guían el desarrollo de la arquitectura y la arquitectura se realimenta en los casos de uso; los dos juntos permiten conceptualizar, gestionar y desarrollar adecuadamente el software.
- Iterativo e Incremental: Para hacer más manejable un proyecto se recomienda dividirlo en ciclos. Para cada ciclo se establecen fases de referencia, cada una de las cuales debe ser considerada como un mini proyecto, cuyo núcleo fundamental está constituido por una o más iteraciones de las actividades principales básicas de cualquier proceso de desarrollo.

RUP se basa en casos de uso para describir lo que se espera del software y está muy orientado a la arquitectura del sistema, documentándose lo mejor posible, basándose en UML (*Unified Modeling Language*) como herramienta principal. (Jacobson, et al., 2000)

### 1.2.5 Lenguaje Unificado de Modelado.

El modelado sirve no solamente para los grandes sistemas, aun en aplicaciones de pequeño tamaño se obtienen beneficios del modelado; sin embargo, es un hecho que cuanto más grande y más complejo es el sistema, más importante es el papel que juega el modelado por una simple razón: "El hombre hace modelos de sistemas complejos porque no puede entenderlos en su totalidad". (Larman, 1999)

El Lenguaje de Modelado Unificado o *Unified Modeling Language* (UML) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad, es un lenguaje estándar para escribir planos de software. UML; puede utilizarse para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra gran cantidad de software.

UML ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como



expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas y proporciona un estándar que permite al analista de sistemas generar un anteproyecto de varias facetas que sean comprensibles para los clientes, desarrolladores y todos aquellos que estén involucrados en el proceso de desarrollo. Un modelo UML indica qué es lo que supuestamente hará el sistema pero no cómo lo hará. (Larman, 1999)

El desarrollo de sistemas con UML, siguiendo el proceso unificado, incluye actividades específicas, cada una de ellas, a su vez, contiene otras subactividades las cuales sirven como una guía de cómo deben ser las actividades desarrolladas y secuenciadas con el fin de obtener sistemas exitosos. Consecuentemente, el desarrollo de los sistemas puede variar, de desarrollador en desarrollador, de proyecto en proyecto, de empresa en empresa, adoptando siempre un Proceso de Desarrollo.

### **1.3 Herramientas para el desarrollo de la solución informática.**

Las herramientas de desarrollo de software diseñan y construyen aplicaciones y dan soporte al desarrollo e implantación de las mismas. Se analizarán una serie de herramientas necesarias para la implementación del sistema.

#### **1.3.1 Servidor Web Sun GlassFish Enterprise Server v3.**

*Sun GlassFish Enterprise Server* es un servidor de aplicaciones de código abierto compatible con la plataforma Java para el desarrollo de aplicaciones Java EE y los servicios web en entornos de producción a gran escala.

*Sun GlassFish Enterprise Server v3* es el primer servidor de aplicaciones que soporta la nueva plataforma Java EE 6 y que se destaca por su mayor flexibilidad y la nueva funcionalidad de Perfil Web. Brinda la posibilidad de realizar un despliegue más rápido, un descenso de los costes de desarrollo y una reducción en los tiempos de salida al mercado. Los tiempos de ejecución de *Sun GlassFish Enterprise Server v3* se encuentran por encima del doble de la versión v2 y en el caso del Perfil Web, es casi tres veces más rápido. (Rojas, 2010) Está basado en el código fuente, donado por *Sun y Oracle Corporation*. Este último proporcionó el módulo de persistencia *TopLink*, que posee como base al servidor *Sun Java System Application Server* de *Sun Microsystems*, un derivado de *Apache Tomcat*, y que usa un componente adicional llamado *Grizzly* para escalabilidad y velocidad.





*Sun GlassFish Enterprise Server v3* ofrece funcionalidades que mejoran el tiempo de puesta en marcha y reduce la utilización de recursos, además de añadir nuevas capacidades de monitorización que ofrecen una vista mejorada tanto para los desarrolladores como para los operadores. Debe considerarse que tiene un tiempo de ejecución que permite desarrollar de forma rápida y repetitiva con múltiples lenguajes de programación. (Oracle, 2010)

Como *Sun GlassFish Enterprise Server* es gratuito si se utiliza para el desarrollo e implementación, y conjuntamente posee un tiempo de ejecución rápido, se propone la utilización de este servidor web para el despliegue del sistema a desarrollar.

### **1.3.2 Entornos integrados de desarrollo.**

Entre los entornos de desarrollo libres más acreditados se encuentran Eclipse, NetBeans, entre otros. Se trabajará con Netbeans por tener soporte para la utilización del marco de trabajo (*Framework*) GWT y permitir crear widget de forma más rápida, además, de tener un ambiente amigable y ser muy potente.

#### **NetBeans.**

Es un entorno de desarrollo integrado IDE (*Integrante Development Environments*) fundado como código abierto en junio del 2000 por *Sun Microsystems*. Este entorno de desarrollo integrado es una herramienta para programadores, pensada para escribir, compilar, depurar y ejecutar programas. Su misión consiste en evitar tareas repetitivas, facilitar la escritura de código correcto, disminuir el tiempo de depuración e incrementar la productividad del desarrollador. El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Es un producto libre y gratuito sin restricciones de uso y está escrito completamente en Java usando la plataforma NetBeans. (Microsystems, 2009)

NetBeans es una plataforma modular y extensible que permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo". (Sun Microsystems, 2010)

### **1.3.3 Sistema de Gestión de Base de Datos (SGBD).**

Un Sistema de Gestión de Bases de Datos (SGBD) consiste en un conjunto de programas, procedimientos y lenguajes que nos proporcionan las herramientas necesarias para trabajar con una



base de datos. Incorporar una serie de funciones que nos permita definir los registros, sus campos, sus relaciones, insertar, suprimir, modificar y consultar los datos.

### **PostgreSQL.**

PostgreSQL es un sistema de gestión de base de datos relacional y libre, publicado bajo la licencia BSD basado en el proyecto POSTGRES, de la universidad de Berkeley. Este proyecto, liderado por Michael Stonebraker, fue uno de los primeros intentos en implementar un motor de base de datos relacional.

PostgreSQL es una derivación libre (Open Source) de este proyecto y utiliza el lenguaje de SQL92/SQL99. Fue el pionero en muchos de los conceptos existentes en el sistema objeto relacional actual, incluido más tarde en otros sistemas de gestión comerciales. Incluye características de la orientación a objetos, como pueden ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. (PostgreSQL, 2010)

#### Características de PostgreSQL.

- Atomicidad (Indivisible) es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.
- Consistencia es la propiedad que asegura que sólo se empieza aquello que se puede acabar. Por lo tanto se ejecutan aquellas operaciones que no van a romper la reglas y directrices de integridad de la base de datos.
- Aislamiento es la propiedad que asegura que una operación no puede afectar a otras. Esto asegura que dos transacciones sobre la misma información nunca generarán ningún tipo de error.
- Durabilidad es la propiedad que asegura que una vez realizada la operación, esta persistirá y no se podrá deshacer aunque falle el sistema.

### **1.3.4 Herramienta Case.**

#### **Visual Paradigm.**

El Visual Paradigm es una poderosa herramienta CASE de modelación visual. Utiliza UML para el modelado, permitiendo crear tipos diferentes de diagramas en un ambiente totalmente visual. Es muy sencillo de usar, fácil de instalar y actualizar. Genera código para varios lenguajes. Tiene disponible



distintas versiones: Enterprise, Professional, Standard, Modeler, Personal y Community. Facilita licencias especiales para fines académicos.

Debido a la magnitud del proyecto a desarrollar se propone Visual Paradigm como una herramienta para ser utilizada en la creación de software libre, siendo esta una de las características que dio lugar a que se seleccionara para el desarrollo del sistema. Se utilizó la versión Enterprise de Visual Paradigm.

Por otra parte, posibilita la representación gráfica de los diagramas permitiendo ver el sistema desde diferentes perspectivas, como el de componentes, despliegue, secuencia, casos de uso, clase, actividad, estado, entre otros. Además, identifica requisitos y comunica información, se centra en cómo los componentes del sistema interactúan entre ellos, sin entrar en detalles excesivos, además, permite ver las relaciones entre los componentes del diseño y mejora la comunicación entre los miembros del equipo usando un lenguaje gráfico. (Paradigm, 2009)

### **1.4 Propuesta de solución.**

El seguimiento de los procesos de negocios y de las métricas se realizan típicamente, empujando información desde múltiples aplicaciones y fuentes de datos para presentarla al usuario en tiempo real. La información puede ser filtrada para que cada usuario reciba o visualice datos de especial interés en su área.

Los tableros de mando permitirán obtener mayor beneficio a la organización. Al crear un conjunto común de indicadores, habrá una comunicación más eficaz con los miembros del consejo y con los involucrados en la mejora del rendimiento empresarial. El seguimiento de los progresos, y la utilización de las métricas obtenidas permitirán ayudar a la organización y a su personal a reconocer su fuerza relativa e identificar oportunidades para mejorar aún más la creación de valor.

Teniendo en cuenta que las soluciones propietarias presentan altos costes por concepto de licencias y baja flexibilidad debido a implementaciones no estándares se decide no recurrir a ninguna de estas. No se utiliza la solución de software libre WSO2 BAM porque no permite la conexión con diferentes gestores de base de datos para la obtención de métricas. Es por ello, que se propone en este trabajo desarrollar un sistema flexible que tiene como objetivo explotar al límite las tecnologías mencionadas y hacer uso de las herramientas nombradas anteriormente en la industria de la producción de software



libre, obteniendo una solución robusta, con una capacidad de adaptarse a los diferentes negocios para los cuales se haga útil su implantación y que pueda ser aplicada en cualquier plataforma.

La aplicación permitirá conectarse a cualquier sistema de gestión de base de datos para la obtención de métricas en tiempo real, además, de permitir el diseño de tableros de mando interactivos que posibilitará relacionar los objetivos de diferentes procesos de negocio. Se establecerá un medio de comunicación instantáneo entre los procesos de negocios y los encargados de la gestión de los mismos. Será un complemento a la capacidad de toma de decisiones, además, de presentarse como un factor clave en la detención del caos en los distintos negocios que tengan desplegados esta solución BAM, siempre y cuando exista disponible un medio por donde propagarse la información de las métricas en dichos negocios.

### **CONCLUSIONES.**

La creación de una herramienta BAM, permitirá monitorizar el comportamiento de los procesos de negocio en tiempo real, será posible la agregación, análisis y presentación de la información en el momento oportuno, mostrando datos estadísticos que garanticen una rápida identificación de cuellos de botella.

Permitirá a mandos altos, expertos en dominios, corresponsales de negocio y jefes de operaciones controlar el funcionamiento de una empresa en tiempo real. Podrán recabar, de forma rápida, información procedente de gran cantidad de fuentes (tanto internas como externas) para obtener una panorámica amplia, completa y de extremo a extremo de las operaciones que se realizan.



## CAPÍTULO 2.

### PRESENTACIÓN DE LA SOLUCIÓN PROPUESTA.

---

La solución a desarrollar, debe ser fruto de un correcto análisis y una amplia comprensión de todos los elementos que se relacionan en correspondencia con el tema de monitoreo de procesos de negocio, profundizándose en el estudio de las características que posibilitan desarrollar los mismos.

En este capítulo se interpretarán las necesidades del sistema, especificándolas mediante los requerimientos funcionales y los no funcionales. Además, se hará un estudio del negocio en que se enmarca el problema donde se debe realizar una modelación del dominio e identificar para esto las entidades principales que se tendrán y las relaciones entre ellas.

El capítulo, al mismo tiempo, expondrá el diagrama de casos de uso del sistema con sus respectivas descripciones, los diagramas de clases del análisis, el diseño de la solución; los diagramas de clases del diseño y sus respectivos diagramas de interacción (secuencias).



## 2.1 Modelo de Dominio.

No se logró encontrar procesos de negocio que enmarcaran el problema, sin embargo se identificaron conceptos tecnológicos que definen correctamente cada eslabón de la solución, y que determinan entonces, la creación de un modelo de dominio el cual deja bien claro cómo funciona el entorno en el cual está enmarcado el problema. El modelo del dominio representa cosas del mundo real y para poder identificar los conceptos se hace necesario investigar el dominio del problema. (Proenza, 2005) A continuación, se muestra el modelo de dominio que conceptualiza los elementos principales y sus relaciones.

## 2.2 Diagrama del modelo de dominio.

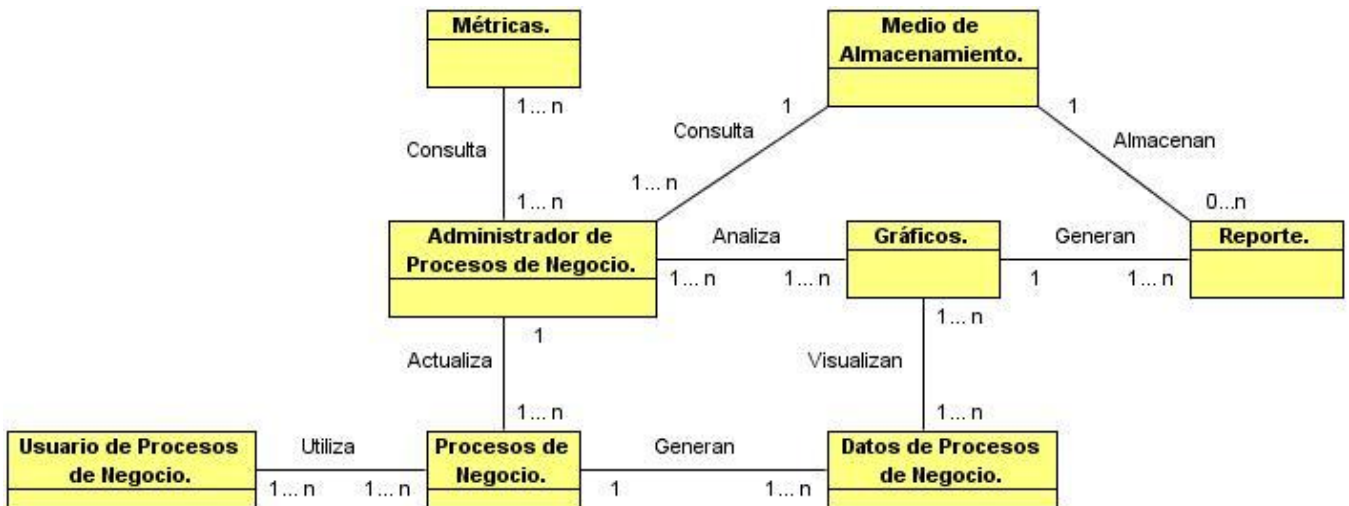


Figura 3. Diagrama del Modelo de Dominio.

## 2.3 Especificación de los requisitos de software.

La obtención de requerimientos es un paso muy importante para el posterior desarrollo de las siguientes etapas, pues un error en estas fases iniciales puede dar al traste con un sistema que no cumpla las expectativas de los usuarios y difícilmente aporte valor agregado al negocio para el que fue concebido. (Software, 2004)



### 2.3.1 Requerimientos funcionales.

Los requerimientos funcionales especifican acciones que el sistema debe ser capaz de realizar, sin tomar en consideración ningún tipo de restricción física. Es decir, especifican el comportamiento de entrada y salida del sistema y surgen de la razón fundamental de la existencia del producto. (Software, 2004) A continuación, se muestran las funcionalidades que el sistema debe cumplir agrupándolos mediante módulos:

Módulo Monitoreo.

#### RF 1. Gestionar conexión con el servidor externo.

##### 1.1 Adicionar conexión con el servidor externo.

###### 1.1.1 Conectar con Oracle.

###### 1.1.1.1 Introducir los datos de la conexión.

- Nombre de la conexión.
- Nombre del host del Servidor.
- Puerto.
- SID de la base de datos.
- Usuario.
- Contraseña.
- Tipo de conexión.

###### 1.1.2 Conectar con PostgreSQL.

###### 1.1.2.1 Introducir los datos de la conexión.

- Nombre de la conexión.
- Nombre del host del Servidor.
- Puerto.
- SID de la base de datos.
- Usuario.
- Contraseña.

###### 1.1.3 Registrar los datos de la conexión.

##### 1.2 Modificar conexión con el servidor externo.

###### 1.2.1 Seleccionar conexión.

###### 1.2.2 Mostrar los datos de la conexión.



- 1.2.3 Introducir los datos a modificar.
- 1.2.4 Registrar los datos modificados de la conexión.
- 1.3 Eliminar conexión con el servidor externo.
  - 1.3.1 Seleccionar conexión.
  - 1.3.2 Eliminar conexión.

### **RF 2. Gestionar las métricas.**

- 2.1 Adicionar métricas.
  - 2.1.1 Seleccionar la conexión a utilizar.
  - 2.1.2 Introducir la información referente a la métrica.
    - Intervalo en que se pedirán los datos al servidor externo.
    - Consulta a realizar en el servidor externo.
  - 2.1.3 Mostrar la opción para poner nombre a cada métrica.
  - 2.1.4 Registrar la información referente a la métrica.
- 2.2 Modificar métricas.
  - 2.2.1 Seleccionar la métrica.
  - 2.2.2 Mostrar la información referente a la métrica.
  - 2.2.3 Introducir los datos a modificar.
  - 2.2.4 Registrar la información modificada de la métrica.
- 2.3 Eliminar métricas.
  - 2.3.1 Seleccionar la métrica.
  - 2.3.2 Eliminar la métrica.

### **RF 3. Gestionar tableros de mando.**

- 3.1 Adicionar tablero de mando.
  - 3.1.1 Seleccionar componente(s).
  - 3.1.2 Asociar métrica(s) al(a los) componente(s) seleccionado(s).
  - 3.1.3 Generar tablero de mando.
- 3.2 Modificar tablero de mando.
  - 3.2.1 Seleccionar tablero de mando.
  - 3.2.2 Modificar tableros de mando (compontes y métricas).
  - 3.2.3 Regenerar tableros de mando.





3.2.3.1 Sobrescribir tablero de mando existente.

3.2.3.2 Obtener un nuevo tablero de mando.

3.3 Eliminar tablero de mando.

3.3.1 Seleccionar tablero de mando.

3.3.2 Eliminar tablero de mando.

### **RF 4 Generar alertas.**

4.1 Verificar estado de la información proveniente de los servidores externos.

4.1.1 Generar alertas en caso necesario con su comentario asociado.

4.1.2 Enviar alertas a los usuarios suscritos.

### **RF 5 Monitorizar las métricas de negocio representadas en tableros de mando.**

5.1 Seleccionar tablero de mando.

5.2 Supervisar el contenido del tablero de mando.

5.3 Tomar foto en una determinada situación.

5.4 Convertir a formato Word para después imprimir.

Módulo Administrativo.

### **RF 6. Gestionar roles del sistema.**

6.1 Adicionar rol al sistema.

6.1.1 Introducir datos del rol.

- Nombre.
- Descripción.
- Permisos.

6.1.2 Registrar los datos del rol.

6.2 Modificar rol del sistema.

6.2.1 Seleccionar rol.

6.2.2 Mostrar los datos del rol.

6.2.3 Introducir los datos a modificar.

6.2.4 Registrar los datos modificados del rol.

6.3 Eliminar rol del sistema.



6.3.1 Seleccionar rol.

6.3.2 Eliminar rol.

### **RF 7 Gestionar usuarios del sistema.**

7.1 Adicionar usuario del sistema.

7.1.1 Introducir datos del usuario.

- Nombre.
- Apellidos.
- Usuario.
- Correo.
- Contraseña.

7.1.2 Registrar los datos del usuario.

7.2 Modificar usuario del sistema.

7.2.1 Seleccionar usuario.

7.2.2 Mostrar los datos del usuario.

7.2.3 Introducir los datos a modificar.

7.2.4 Registrar los datos modificados del usuario.

7.3 Eliminar usuario del sistema.

7.3.1 Seleccionar usuario.

7.3.2 Eliminar usuario.

### **RF 8. Asignar rol al usuario del sistema.**

8.1 Seleccionar rol.

8.2 Seleccionar usuarios a asignar.

8.3 Registrar la asignación.

### **RF 9. Autenticar usuario.**

9.1 Comprobar coincidencias entre el usuario y la contraseña.

9.2 Permitir al usuario el cambio de contraseña.



### 2.3.2 Requerimientos no funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido y confiable. Los requerimientos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con toda la funcionalidad requerida, las propiedades no funcionales, como cuán usable, seguro, conveniente y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación. (Software, 2004)

#### Requisitos de Software.

Estaciones de Trabajo. (PC Cliente)

- Sistema operativo: Multiplataforma.
- Navegador web: Internet Explorer, Mozilla, NetScape.

Servidor de Aplicaciones.

- Sistema operativo: Multiplataforma.
- Servidor web Sun GlassFish Enterprise Server v3.
- Máquina Virtual de Java 6.0.

Servidor de Base de Datos.

- Sistema operativo: Multiplataforma.
- Servidor Gestor de Base de Datos PostgreSQL 8.2.

#### Requisitos de Hardware.

Estaciones de trabajo. (PC Cliente)

- Periféricos: Mouse y Teclado.
- Tarjeta de Red.
- 256 MB de Memoria RAM.
- Procesador Pentium IV.
- HDD SCCI 40 GB de espacio en disco.

Impresora.

- Conexión USB.



- Controladores multiplataforma.

Servidor de Aplicaciones.

- Tarjeta de Red Gigabit Ethernet<sup>6</sup> 1 Gbps.
- 1GB de Memoria RAM.
- Pentium IV 2.4 GHz.
- HDD SCCI 40 GB de espacio en disco.

Servidor de Base de Datos.

- Tarjeta de Red Gigabit Ethernet 1 Gbps.
- 512 GB de Memoria RAM.
- Pentium IV 2.4 GHz.
- HDD SCCI 160 GB de espacio en disco. (Usando RAID 1)

### **Restricciones en el diseño y la implementación.**

Estándares requeridos.

- Ethernet.
- Java.

Uso obligatorio de ciertas herramientas.

- Sun GlassFish Enterprise Server v3.
- PgManager.
- NetBeans.
- Visual Paradigm.

Restricciones en la arquitectura o el diseño.

- Se utiliza Ethernet para la comunicación con otros servidores.
- El protocolo de comunicación es http.
- Los usuarios deben autenticarse para acceder al BAM.

Bibliotecas de clases.

- Google Web Toolkit.

---

<sup>6</sup> Ethernet es un estándar de transmisión de datos para redes de área local.



### Requisitos de apariencia o interfaz externa.

- El sistema posee una interfaz sencilla, intuitiva, amigable. Mantiene el formato en páginas similares. Es fácil de usar y agradable a la vista del usuario.
- Establecer mecanismos de barrido visual para el contenido de la página, y distribuir los elementos de información y navegación según su importancia, en zonas de mayor o menor jerarquía visual. Las zonas superiores de la interfaz poseen más jerarquía visual que las inferiores.
- Se utilizan colores frescos y se ofrece suficiente contraste entre texto y fondo para no dificultar la lectura y que resulte agradable a la vista del usuario.
- Se muestran las alertas resaltadas con colores fuertes.
- Los enlaces están detallados con el contenido temático, que permiten acceder a cualquiera de las partes o secciones, con un número mínimo de clic.

### Requisitos de seguridad.

- Existencia de distintos roles que establezcan las acciones que pueden realizar los usuarios.
- Identificar al usuario antes de que pueda realizar cualquier acción sobre el sistema.
- Verificación sobre acciones irreversibles (por ejemplo las eliminaciones).
- El sistema constará de varios niveles de acceso. Un primer nivel o nivel básico donde están las funciones asociadas al especialista o ejecutivo, que requieren poca responsabilidad en cuanto al sistema. El segundo nivel está compuesto por funciones de mayor complejidad y que pueden destruir información relacionada a las entidades del sistema. A este nivel pertenecen los arquitectos y diseñadores de tableros de mando. El tercer nivel está conformado con las funciones administrativas del sitio y del sistema.
- Confidencialidad: La información manejada por el sistema estará protegida de acceso no autorizado y divulgación.
- Integridad: Garantía de un tratamiento adecuado de las excepciones y validación de las entradas del usuario para evitar entradas inadecuadas.
  - El sistema debe tener soporte para la recuperación ante fallos y errores.
  - Los servidores tendrán respaldo de los datos (*Backup*).



### Requisitos de Usabilidad.

- La aplicación debe ser implementada de modo que resulte fácil su uso para los usuarios que posean poca experiencia, posean conocimientos básicos en el manejo de la computadora, de un ambiente Web en sentido general y que no sepan el funcionamiento de la aplicación. Para lograr este propósito el sistema cuenta con una ayuda que será capaz de brindar la información sobre el sistema que el usuario necesita.
- Ofrecer señalizaciones que agilicen el aprendizaje del usuario a trabajar con el sistema.

### Ayuda y documentación en línea.

- Desarrollar un manual de usuario y procedimientos relativos a la utilización de la aplicación y tecnología a utilizar.
- Entregar documentos técnicos y las guías de usuario, que incluyen presentaciones realizadas en cada tema.
- Entregar carpeta del proyecto con la documentación técnica generada en el desarrollo para la especificación del sistema.

### Requisitos de Portabilidad.

- El software está construido con fines de multiplataforma, de modo que pueda ser ejecutado tanto en Linux como en Windows.

## 2.4 Modelo del sistema.

El diagrama de casos de uso del sistema representa la forma en cómo un actor opera con el sistema en desarrollo, además de la forma, tipo y orden en cómo los elementos interactúan. Un diagrama de casos de uso muestra, por tanto, los distintos requisitos funcionales que se esperan de una aplicación o sistema y cómo se relaciona con su entorno (usuarios u otras aplicaciones). Se muestra el diagrama de casos de uso del sistema.

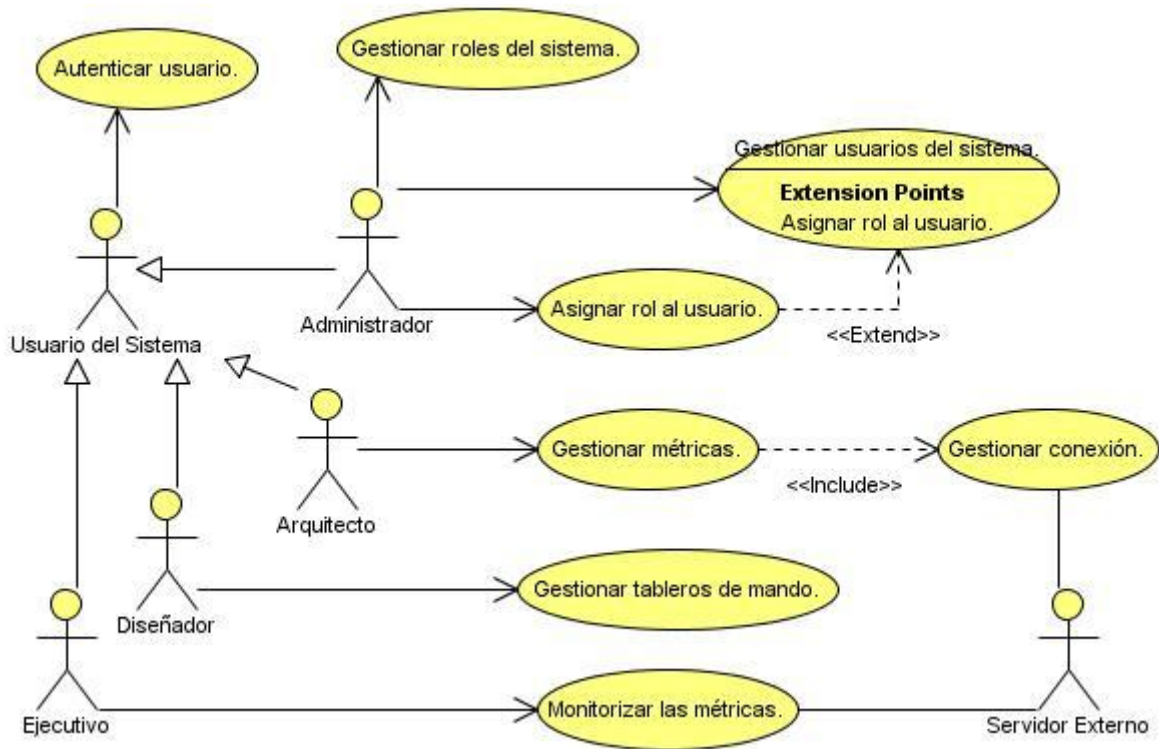


Figura 4. Diagrama de Casos de Uso del Sistema.

Un actor del sistema es un rol que un usuario juega con respecto al sistema. Es importante destacar que un actor no necesariamente representa a una persona en particular, sino más bien la labor que realiza frente al sistema, es decir, que un equipo o un sistema automatizado, son parte del sistema, y pueden intercambiar información con él o ser recipientes pasivos de información.

Actores	Descripción
<b>Ejecutivo</b>	Cualquier persona que esté capacitada para tener la función de supervisión y toma de decisiones a partir de la información representada en los tableros de mando.
<b>Diseñador</b>	Persona encargada de adicionar, modificar o eliminar los tableros de mando.
<b>Arquitecto</b>	Persona encargada de adicionar, modificar o eliminar las métricas.
<b>Administrador</b>	Persona encargada de gestionar a los usuarios y recursos del sistema.
<b>Usuario del Sistema</b>	Es un usuario general del sistema encargado de autenticarse para recibir los privilegios según los roles que tiene asignados.
<b>Servidor Externo</b>	Cualquier servidor externo al sistema que pueda intercambiar información con él.

Tabla 2.1 Descripción de los Actores del Sistema.



### 2.4.1 Especificación de los casos de uso del sistema.

La representación gráfica del diagrama de casos de uso del sistema no es suficiente para entender las funcionalidades asociadas a los casos de uso. La descripción extendida brinda los detalles de la secuencia de acciones, las precondiciones como estado inicial, los posibles estados finales como poscondiciones, además, de cuándo comienza y cuándo termina el caso de uso. Describe explícitamente qué debe hacer el sistema, al separar las responsabilidades del sistema y la de los actores. La descripción de los casos de uso del sistema referente a ambos módulos. Ver Anexo 1.

## 2.5 Estimación de Esfuerzo.

Una vez determinados los casos de usos que guiarán el desarrollo del software, se puede predecir una estimación del tiempo de duración del proyecto mediante el análisis de Puntos de Casos de Uso. La estimación mediante el análisis de Puntos de Casos de Uso se trata de un método de estimación del tiempo de desarrollo de un proyecto mediante la asignación de “pesos” a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores. (Peralta, 2007) Se detallan los pasos a seguir para la realización de este método.

### Paso 1. Cálculo de Puntos de Casos de Uso sin ajustar.

El cálculo de Puntos de casos de Uso sin ajustar se calcula mediante la siguiente ecuación:

$$UUCP = UAW + UUCW$$

Donde:

**UUCP:** Puntos de casos de uso sin ajustar.

**UAW:** Factor de peso de los actores sin ajustar.

**UUCW:** Factor de peso de los casos de uso sin ajustar.

### Factor de Peso de Actores sin ajustar (UAW).

El valor del Factor de Peso de los Actores sin ajustar se calcula mediante un análisis de la cantidad de Actores presentes en el sistema y la complejidad de cada uno de ellos. Los criterios se muestran en la siguiente tabla:

Tipo de Actor	Descripción	Factor de peso	Actores	Total
<b>Simple</b>	Otro sistema que interactúa con el	1	0	0





	sistema a desarrollar mediante una interfaz de programación.			
<b>Medio</b>	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto.	2	1	2
<b>Complejo</b>	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3	4	12
<b>Total</b>				14

**Tabla 2.2 Factor de Peso de los Actores sin Ajustar.**

En el sistema propuesto van a interactuar 5 actores.

- El servidor externo, es un actor de tipo medio porque se comunica con el sistema mediante un protocolo de comunicación, al cual se le asigna como peso 2.
- El ejecutivo, el diseñador, el arquitecto y el administrador, son cuatro actores de tipo complejo ya que son personas que interactúan con el sistema mediante una interfaz gráfica, a los cuales se le asigna como peso 3.

Luego el factor de peso de los actores sin ajustar se calcula mediante la siguiente ecuación:

$$UAW = \Sigma \text{ cantidad actores} * \text{peso}$$

$$UAW = 14$$

**Factor de Peso los Casos de Uso sin ajustar (UUCW).**

El valor del Factor de Peso de los Casos de Uso sin ajustar se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción es una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia.

Los criterios se muestran en la siguiente tabla:



Tipo de Casos de Uso	Descripción	Factor de peso	Casos de Uso	Total
<b>Simple</b>	El caso de uso contiene de 1 a 3 transacciones.	5	3	15
<b>Medio</b>	El caso de uso contiene de 4 a 7 transacciones.	10	5	50
<b>Complejo</b>	El caso de uso contiene más de 8 transacciones.	15	0	0
<b>Total</b>				65

Tabla 2.3 Factor de Peso de los Casos de Uso sin Ajustar.

Para la realización de la aplicación en cuestión existen 3 casos de uso que tienen de 1 a 3 transacciones y 5 casos de uso que tienen de 4 a 7 transacciones, por lo que se tiene entonces 3 casos de uso de tipo simple (peso 5) y 5 casos de uso de tipo medio (peso 10), con lo cual el factor de peso de los casos de uso sin ajustar resulta:

$$UUCW = \Sigma \text{cantidad casos de uso} * \text{peso}$$

$$UUCW = 65$$

Finalmente, los Puntos de Casos de Uso sin ajustar resultan:

$$UUCP = UAW + UUCW$$

$$UUCP = 14 + 65$$

$$UUCP = 79$$

**Paso 2. Cálculo de Puntos de Casos de Uso ajustados.**

Ya obtenidos los Puntos de Casos de Uso sin ajustar, se debe ajustar este valor mediante la siguiente ecuación:

$$UCP = UUCP * TCF * EF$$

Donde:

**UCP:** Puntos de Casos de Usos Ajustados.

**UUCP:** Puntos de Casos de Usos Sin Ajustar.

**TCF:** Factor de Complejidad Técnica.

**EF:** Factor de Ambiente.



El factor de complejidad técnica (TCF) se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada factor se cuantifica en un valor desde 0 (aporte irrelevante) hasta 5 (aporte muy relevante).

Factor	Descripción	Peso	Valor Asignado	Total
T1	Sistema distribuido.	2	0	0
T2	Tiempo de respuesta.	1	4	4
T3	Eficiencia del usuario final.	1	4	4
T4	Procesamiento interno complejo.	1	1	1
T5	El código debe ser reutilizable.	1	3	3
T6	Facilidad de instalación.	0.5	3	1.5
T7	Facilidad de uso.	0.5	3	1.5
T8	Portabilidad.	2	4	8
T9	Facilidad de cambio.	1	4	4
T10	Concurrencia.	1	2	2
T11	Incluye objetivos especiales de seguridad.	1	2	2
T12	Provee acceso directo a terceras partes.	1	2	2
T13	Se requieren facilidades especiales de entrenamiento.	1	3	3
<b>Total</b>				<b>36</b>

Tabla 2.4 Factor de Complejidad Técnica.

$$TCF = 0.6 + 0.01 * \Sigma (\text{peso} * \text{valor asignado})$$

$$TCF = 0.6 + 0.01 * 36$$

$$TCF = 0.6 + 0.36$$

$$TCF = 0.96$$

El factor de ambiente (EF) está relacionado con las habilidades y entrenamiento del grupo de desarrollo que realiza el sistema. Cada factor se cuantifica con un valor desde 0 (aporte irrelevante) hasta 5 (aporte muy relevante).

Factor	Descripción	Peso	Valor Asignado	Total
E1	Familiaridad con el modelo de proyecto utilizado.	1.5	4	6
E2	Experiencia en la aplicación.	0.5	3	1.5
E3	Experiencia en trabajo orientado a objetos.	1	5	5
E4	Capacidad de analista líder.	0.5	5	2.5



<b>E5</b>	Motivación.	1	5	5
<b>E6</b>	Estabilidad de los requerimientos.	2	4	8
<b>E7</b>	Personal a tiempo completo.	-1	0	0
<b>E8</b>	Dificultad del lenguaje de programación.	-1	0	0
<b>Total</b>				<b>28</b>

Tabla 2.5 Factor de Ambiente.

$$EF = 1.4 - 0.03 * \Sigma (\text{peso} * \text{valor asignado})$$

$$EF = 1.4 - 0.03 * 28$$

$$EF = 1.4 - 0.84$$

$$EF = 0.56$$

$$UCP = UUCP * TCF * EF$$

$$UCP = 79 * 0.96 * 0.56$$

$$UCP = 42$$

### Paso 3. Estimación de esfuerzo a través de los puntos de casos de uso.

$$E = UCP * CF$$

Donde:

**E:** Esfuerzo estimado en horas hombres.

**UCP:** Punto de casos de usos ajustados.

**CF:** Factor de conversión.

Para obtener el factor de conversión (CF) se cuentan cuántos valores de los que afectan al factor ambiente (E1...E6) están por debajo de la media (3), y los que están por encima de la media para los restantes (E7, E8).

- Si el total es 2 o menos, se utiliza el factor de conversión 20 Horas-Hombre / Punto de Casos de uso.
- Si el total es 3 ó 4, se utiliza el factor de conversión 28 Horas-Hombre / Punto de Casos de uso.
- Si el total es mayor o igual que 5, se recomienda efectuar cambios en el proyecto ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

En este caso se puede decir que:

CF = 20 Horas-Hombre / Puntos de casos de uso.



$$E = UCP * CF$$

$$E = 42 * 20$$

$$E = 840$$

**Paso 4. Calculando el esfuerzo de todo el proyecto.**

En la siguiente tabla se muestra el esfuerzo total en horas-hombre.

Actividad	Porcentaje	Valor esfuerzo
Análisis	10%	210 Horas-Hombre
Diseño	20%	420 Horas-Hombre
Implementación	40%	840 Horas-Hombre
Prueba	15%	315 Horas-Hombre
Sobrecarga(otras actividades)	15%	315 Horas-Hombre
<b>Total</b>	<b>100%</b>	<b>2100 Horas-Hombre</b>

Tabla 2.6 Esfuerzo en Horas-Hombre.

El sistema requiere de **2100 horas-hombre** para su desarrollo completo, y se estima que cada mes tiene 24 días laborales y se trabaja 10 horas diarias como promedio, se trabajarían en un mes 240 horas laborables. Eso daría un **ET = 8.7 mes-hombre**. Esto quiere decir que el sistema se desarrollará en aproximadamente 8 meses y medio. Haciendo un balance costo-beneficio de la investigación y teniendo en cuenta que la construcción del sistema no supone grandes gastos, ya sea monetario o de tiempo, y que su desarrollo es utilizando aplicaciones basadas en software libre, se puede concluir que es factible su realización.

## 2.6 Análisis.

En el análisis se analizan los requisitos que fueron descritos en la captura de requerimientos, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos, que sea fácil de mantener, y que ayude a estructurar todo el sistema, incluyendo su arquitectura.

### 2.6.2 Diagrama de Clases del Análisis.

El diagrama de clases del análisis está estructurado por clases y paquetes estereotipados que proporcionan la estructura de la vista interna del sistema. Es utilizado fundamentalmente por los desarrolladores para comprender cómo debería darse forma al sistema, es decir, cómo debería ser



diseñado e implementado. Este modelo define realizaciones de casos de uso, y cada una de ellas representa el análisis de un caso de uso del modelo de casos de uso. Los diagramas de clases del análisis de los casos de uso referente a ambos módulos. Ver Anexo 2.

### **2.7 Diseño.**

El diseño es un refinamiento del análisis que tiene en cuenta los requisitos no funcionales, es decir, cómo cumple el sistema sus objetivos. El diseño debe ser suficiente para que el sistema pueda ser implementado sin ambigüedades, cuando la precisión del diseño es muy grande, la implementación puede ser hecha por un generador automático de código. En el diseño se modela el sistema incluyendo la arquitectura, para que soporte los requisitos, incluyendo los no funcionales y las restricciones que se le suponen.

#### **2.6.1 Descripción de la arquitectura.**

La aplicación está diseñada mediante la implementación del patrón de arquitectura n capas, el cual tiene como objetivo primordial proveer escalabilidad, disponibilidad, seguridad e integración al sistema. Este patrón permite construir componentes físicos a partir de los niveles lógicos. De esta forma, podemos tomar decisiones sobre qué parte lógica de la aplicación vamos a encapsular en cada uno de los componentes, de igual modo que encapsulamos los componentes en varios niveles. Un nivel está conformado por varios componentes, por tanto puede suplir varios servicios.

La aplicación está estructurada en tres niveles fundamentales:

1. Capa de Interfaz de Usuario.

Los componentes de este nivel proporcionan la interfaz visual que los clientes utilizarán para ver la información y los datos. En este nivel, los componentes son responsables de solicitar y recibir servicios de otros componentes del mismo nivel o de la capa de negocio.

Aunque las funciones del negocio residen en otro nivel, para el usuario es transparente la forma de operar.

2. Capa de negocio.

Como los servicios de la capa de interfaz de usuario no pueden contactar directamente con el nivel de la capa de acceso a datos, es responsabilidad de la capa de negocio hacer de puente entre estos. Los objetos de negocio proporcionan servicios que completan las tareas de



negocio tales como verificar los datos enviados por el cliente antes de llevar a cabo una transacción a la base de datos.

Los componentes de la capa de negocio también nos sirven para evitar que el usuario tenga acceso directo a la base de datos, lo cual proporciona mayor seguridad en la integridad de ésta.

### 3. Capa de acceso a datos.

Este nivel se encarga de las típicas tareas que se realizan con los datos: Inserción, modificación, consulta y borrado. La clave de la capa de acceso a datos es que los servicios de negocio no sean implementados aquí, aunque un componente de la capa de acceso a datos es responsable de la gestión de las peticiones realizadas por un objeto de negocio.

La importancia de esta arquitectura consiste en aislar la lógica de la aplicación. Esta separación entre la lógica de la aplicación y la interfaz de usuario añade una enorme flexibilidad al diseño, permitiendo construirse y desplegarse múltiples interfaces de usuario sin cambiar en absoluto la lógica; siempre que esté presente una interfaz claramente definida en la capa de presentación.

### 2.6.2 Patrones de Diseño.

El patrón es una descripción de un problema y su solución, que recibe un nombre y que puede emplearse en otros contextos; en teoría, indica la manera de utilizarlo en circunstancias diversas<sup>7</sup>. Cada patrón describe un problema que ocurre una y otra vez en nuestro ambiente y luego describe el núcleo de la solución a ese problema, de tal manera, que puedes usar esa solución muchas veces más, sin hacer jamás la misma cosa dos veces. (Larman, 1999)

Por su parte, un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular, identifica: clases, instancias, roles, colaboraciones y la distribución de responsabilidades. Se mencionan los patrones de diseño utilizados para el desarrollo del sistema.

- Patrones de Software para la Asignación General de Responsabilidad (GRASP por sus siglas en inglés). El nombre se eligió para indicar la importancia de captar (*grasping*) estos principios, si se quiere diseñar eficazmente el software orientado a objetos. Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos,

---

<sup>7</sup> La notación formal de los patrones nació con los patrones arquitectónicos de Christopher Alexander.



expresados en forma de patrones. (Larman, 1999) Estos patrones son utilizados en el diseño para el desarrollo del sistema, los cuales son:

- Experto.

Problema: ¿Cuál es el principio fundamental en virtud del cual se asignan las responsabilidades en el diseño orientado a objetos?

Solución: Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

El patrón experto se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Expresa simplemente la “intuición” de que los objetos hacen cosas relacionadas con la información que poseen.

Beneficios.

Se conserva el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. Esto soporta un bajo acoplamiento, lo que favorece al hecho de tener un sistema más robusto y de fácil mantenimiento.

El comportamiento se distribuye entre las clases que cuentan con la información requerida, alentando con ello definiciones de clases sencillas y más cohesivas que son más fáciles de comprender y de mantener, de esta forma brinda una alta cohesión.

- Creador.

Problema: ¿Quién debería ser responsable de crear una nueva instancia de alguna clase?

Solución: Asignarle a la clase B la responsabilidad de crear una instancia de la clase A.

El patrón creador guía la asignación de responsabilidades relacionadas con la creación de objetos. Su propósito fundamental es encontrar un creador que debemos conectar con el objeto producido en cualquier evento.

Beneficios.

Se brinda soporte a un bajo acoplamiento, lo cual supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización. Es probable que el acoplamiento no aumente, pues la clase creada tiende a ser visible a la clase creador, debido a las asociaciones actuales que nos llevaron a elegirla como el parámetro adecuado.

- Bajo Acoplamiento.





Problema: ¿Cómo dar soporte a una dependencia escasa y a un aumento de la reutilización?

Solución: Asignar una responsabilidad para mantener bajo acoplamiento.

El Bajo Acoplamiento es un principio que debemos recordar durante las decisiones de diseño: es la meta principal que es preciso tener presente siempre. Es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño.

Beneficios.

No se afectan por cambios en otros componentes.

Fáciles de entender por separado.

Fáciles de reutilizar.

- Alta Cohesión.

Problema: ¿Cómo mantener la complejidad dentro de límites manejables?

Solución: Asignar una responsabilidad de modo que la cohesión siga siendo alta.

Como el patrón Bajo Acoplamiento, también Alta Cohesión es un principio que debemos tener presente en todas las decisiones de diseño: es la meta principal que ha de buscarse en todo momento. Es un patrón evaluativo que el desarrollador aplica al valorar sus decisiones de diseño.

Beneficios.

Mejorar la claridad y la facilidad con que se entiende el diseño.

Se simplifican el mantenimiento y las mejoras en funcionalidad.

A menudo se genera un bajo acoplamiento.

La ventaja de una gran funcionalidad soporta una mayor capacidad de reutilización, porque una clase muy cohesiva puede destinarse a un propósito muy específico.

- Controlador.

Problema: ¿Quién debería encargarse de atender un evento del sistema?

Solución: 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, los cuales generalmente incluyen una interfaz gráfica para el usuario. En estos casos, en los que se recurre a un diseño orientado a objetos, hay que elegir los controladores que manejan esos eventos de entrada.

Beneficios.

Mayor potencial de los componentes reutilizables.



- Agente Remoto y Agente.

El sistema debe comunicarse con un servicio externo (servidor externo). En tal situación, el patrón Agente Remoto sugiere crear una clase de software local que represente el servicio externo y asignarle la responsabilidad de contactar el servicio real.

Problema: El sistema debe comunicarse con un componente situado en el espacio de otra dirección. ¿Quién deberá asumir la responsabilidad?

Solución: Crear una clase de software local que represente al componente externo y asignarle la responsabilidad de contactar al componente real.

El Agente Remoto es un caso especial del patrón general Agente (Proxy) y sugiere utilizar un sustituto del componente (servidor externo) en algunos contextos.

Problema: No se desea o no es posible acceder directamente a un componente. ¿Qué hacer?

Solución: Definir una clase sustituta de software que represente al componente y asignarle la responsabilidad de comunicarse con el componente real. (Larman, 1999)

### 2.6.3 Diagrama de Clases del Diseño.

Los diagramas de clases del diseño muestran un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Gráficamente, un diagrama de clases es una colección de nodos y arcos. Los subsistemas que contienen las clases de diseño a menudo participan en la realización de varios casos de uso. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema principalmente. Esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas. Los diagramas de clases son importantes no sólo para visualizar, especificar y documentar modelos estructurales, sino también para construir sistemas ejecutables, aplicando ingeniería directa e inversa. Los diagramas de clases del diseño de los casos de uso referente a ambos módulos. Ver Anexo 3.

### 2.6.4 Diagrama de Interacción (secuencia).

Un diagrama de interacción consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Un diagrama de secuencia es un diagrama de interacción que destaca la disposición temporal de los mensajes. Por cada realización de caso de uso se ha realizado un diagrama de interacción (específicamente diagrama de secuencia), donde se expone el flujo principal de información entre los objetos del diseño, con sus métodos y parámetros. Los diagramas de interacción de los casos de uso referente a ambos módulos. Ver Anexo 4.



### 2.6.5 Diseño de la base de datos.

Se muestra el diagrama de Entidad Relación de la base de datos.

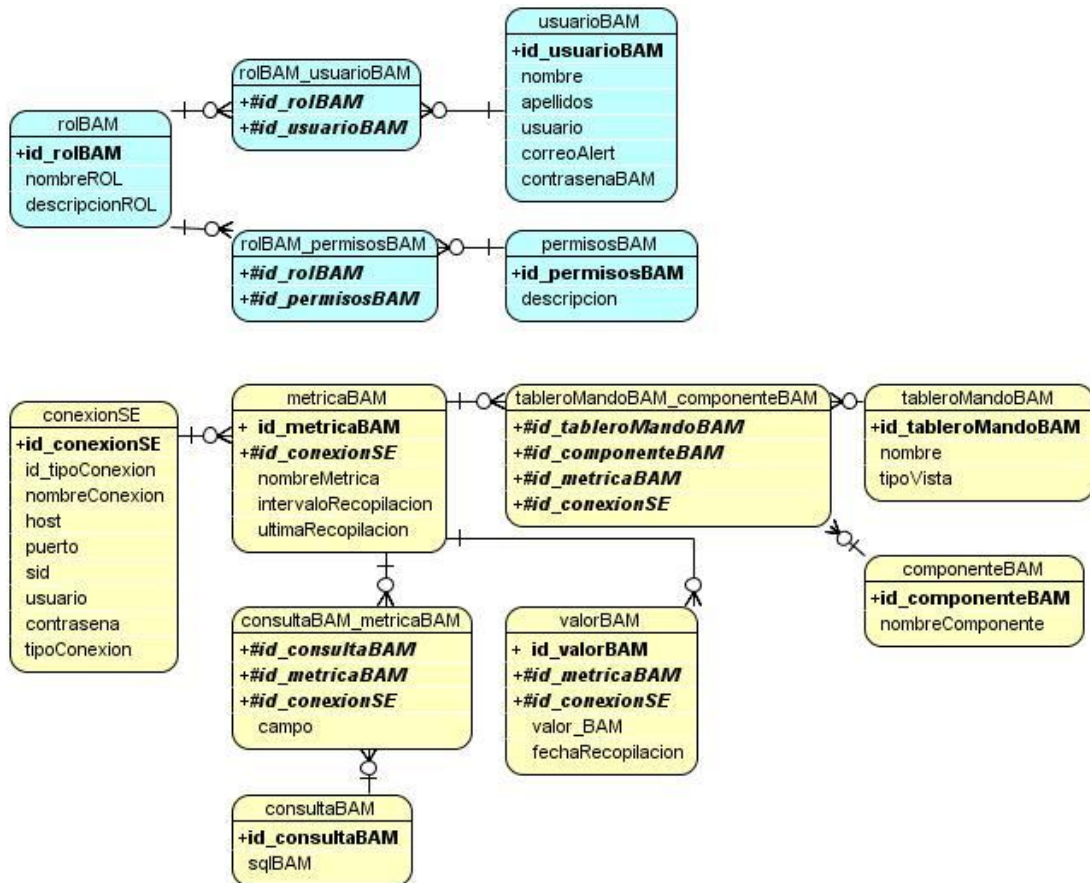


Figura 5. Diagrama de Entidad Relación de la Base de Datos.

El diagrama entidad relación de la base de datos se encuentra compuesto por 13 tablas. Las que están resaltadas en color amarillo pertenecen al módulo monitoreo y las demás al módulo administrativo. Las tablas se componen de dos estructuras. Los campos, que corresponden al nombre de la columna, deben ser únicos y tener un tipo de dato asociado, y los registros que corresponden a cada fila que compone la base de datos, allí se componen los datos y los registros. Las tablas serán las encargadas de almacenar la información recogida desde el sistema.

### 2.6.6 Definiciones de diseño a aplicar.

La interfaz gráfica del usuario constituye el medio de interacción con el sistema. Por tales razones debe ser lo más agradable y clara posible, no sólo para lograr uniformidad entre sus componentes, también para que el cliente se sienta cómodo y logre adaptarse a su ambiente de trabajo.



La calidad de la interfaz de usuario puede ser uno de los motivos que conduzca a un sistema al éxito o al fracaso. Es por eso que uno de los aspectos más relevantes de la usabilidad de un sistema es la consistencia de su interfaz de usuario.

### 2.6.6.1 Aspectos definidos para las interfaces de usuario.

Para el desarrollo de las interfaces de usuario se tuvieron en cuenta los siguientes aspectos:

- Obtener información de retroalimentación.
- Diseño de diálogos que conducen a una conclusión.
- Previsión de errores y manejo de errores simples.
- Que el usuario sintiera la sensación de control.
- Todos los botones deben tener imágenes sugerentes de manera que el usuario memorice la imagen y no necesite buscar mucho mientras se adapte al entorno.

Una de las premisas fundamentales de la aplicación es la ventaja que proporciona las interfaces Web sobre las interfaces de comando, ya que las interfaces Web:

- Proporcionan un ambiente amigable.
- Conducen a un aprendizaje más natural.
- Establecen un “sentimiento” (sobre todo en la uniformidad del ambiente) al usuario que enriquece su experiencia en el uso de la aplicación.

Además de estos principios, se tuvieron en cuenta las siguientes características:

- Utilizar una misma tipografía, forma y estilo en todas las páginas.
- La facilidad del usuario de poder navegar desde cualquier punto a otro dentro de la aplicación.
- La simplicidad y consistencia, favoreciendo la usabilidad de la aplicación.
- Estabilidad y uniformidad del diseño, para así poder ubicar al usuario dentro del mismo y hacerlo sentir parte de él.

Se realizan múltiples procedimientos en cada página, de forma que el usuario no tenga que moverse tanto dentro de la aplicación, para completar una operación.



### 2.6.6.2 Tratamiento de Errores.

Para que el sistema tenga buenos resultados en su funcionamiento, es preciso informarle al usuario cuándo es incorrecta la operación que realizan. Se necesita una buena validación en sus controles a nivel de interfaz y tratamiento de excepciones a nivel de código.

Siempre se debe:

- Detectar las excepciones.
- Mostrar la información sobre la excepción detectada.

El sistema propuesto permitirá evitar, minimizar y tratar los posibles errores, con el fin de garantizar la integridad y confiabilidad de la información que en este se registra y muestra.

Los mensajes de error que emite el sistema se muestran en un lenguaje de fácil comprensión para los usuarios. Cuando se introduce información en un formulario y faltan datos, sale un cuadro de alerta indicando el campo o dato que falta.

## CONCLUSIONES.

Se define el modelo de dominio que conceptualiza los principales elementos y sus relaciones, se describen los requisitos funcionales y no funcionales del sistema propuesto. Se muestra la descripción de los actores del sistema, así como el diagrama de casos de uso del sistema y sus respectivas descripciones de casos de uso del sistema. Quedando planteadas de esta forma, las condiciones y características del sistema propuesto.

Se describe la propuesta de solución a partir de la realización de forma detallada de los diagramas de clases del análisis y los diagramas de clases del diseño de cada caso de uso, así como los diagramas de interacción específicamente los de secuencia y el modelo entidad relación de la base de datos. Fueron descritos los principios de diseño en los que se basa la solución propuesta.

El sistema deberá renderizar gráficos para mostrar los cambios en los procesos de negocio en tiempo real, además el usuario con el rol de arquitecto estará obligado a conocer la estructura de la base de datos de donde obtendrá las métricas.



### CAPÍTULO 3.

## CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA.

---

Durante la fase del diseño, se refinan las clases y sus relaciones para dar paso a la implantación del sistema. También en esta etapa, se toman las medidas necesarias para lograr un diseño del sistema consistente con el entorno en el que se implementará. De una correcta implementación, que responda y dé solución a los requerimientos planteados, depende en gran medida que de las pruebas que se realizarán, una vez implementado el sistema, se obtengan los resultados esperados por los desarrolladores.



### 3.1 Diagrama de Despliegue.

Los Diagramas de Despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria. (Jacobson, et al., 2000)

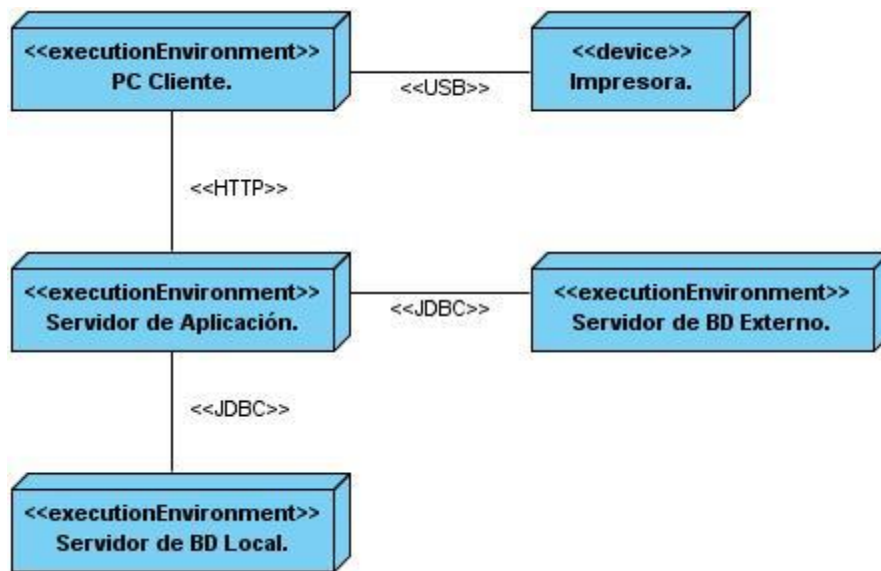


Figura 6. Diagrama de Despliegue de los Componentes.

- Un nodo representa la PC Cliente en la cual se utilizará el sistema para monitorizar procesos de negocio, comunicándose mediante USB con el nodo impresora y mediante el protocolo de comunicación HTTP con el servidor de aplicaciones.
- El nodo Impresora se utiliza para imprimir fotos tomadas de la aplicación en un determinado instante de tiempo.
- El nodo Servidor de Aplicación representa un servidor web que se utiliza para desplegar la solución, donde se encuentran los componentes que implementan las funcionalidades del sistema, cada componente desarrolla como uno de sus niveles la capa de acceso a datos, utilizando así el servidor de base de datos local. Para la obtención de las métricas se obtiene información a través de servidores externos mediante el protocolo de comunicación JDBC.
- El nodo Servidor de Base de Datos Local, representa un servidor utilizado para el almacenamiento de los datos del sistema de análisis y monitoreo de los procesos de negocio, y

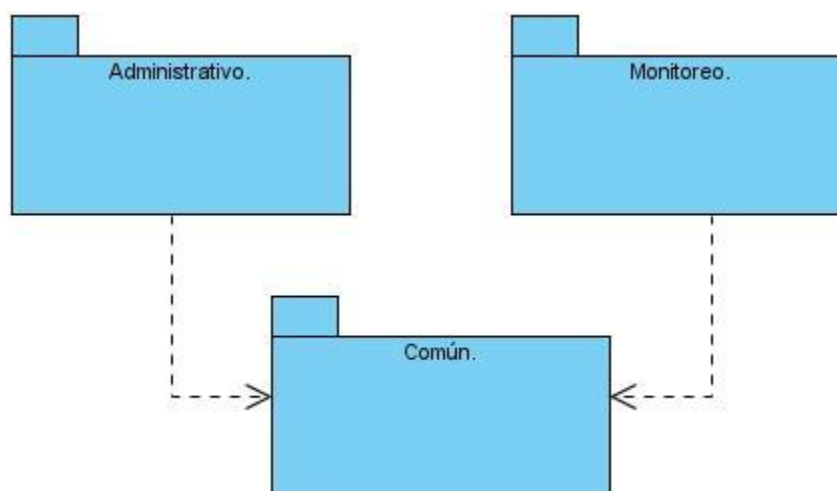


para lograr la conexión del sistema con la base de datos se utiliza JDBC como protocolo de comunicación.

- El nodo Servidor de Base de Datos Externo representa cualquier servidor de base de datos que utilice el sistema para la obtención de métricas.

### 3.2 Diagrama de Componentes.

El diagrama de componentes representa cómo el sistema de software es dividido en componentes y muestra las dependencias entre estos. Los componentes pueden agruparse en paquetes siguiendo un criterio lógico y con vista a simplificar la implementación. Son paquetes estereotipados en subsistemas (o módulos). El diagrama de componentes representa la forma en que el sistema se ha desarrollado basado en el patrón de arquitectura n capas.



**Figura 7. Diagrama de Componentes General.**

El Diagrama de componentes general está dividido en módulos:

- El módulo administrativo en el cual se encuentran las funcionalidades gestionar usuarios del sistema, gestionar roles del sistema, asignar rol al usuario y autenticar usuario.
- El módulo monitoreo donde se localizan las funcionalidades gestionar conexión con el servidor externo, gestionar métricas, gestionar tablero de mando y monitorizar métricas.
- El módulo común en el que se localizan todos los componentes que son utilizados por los dos módulos anteriores.



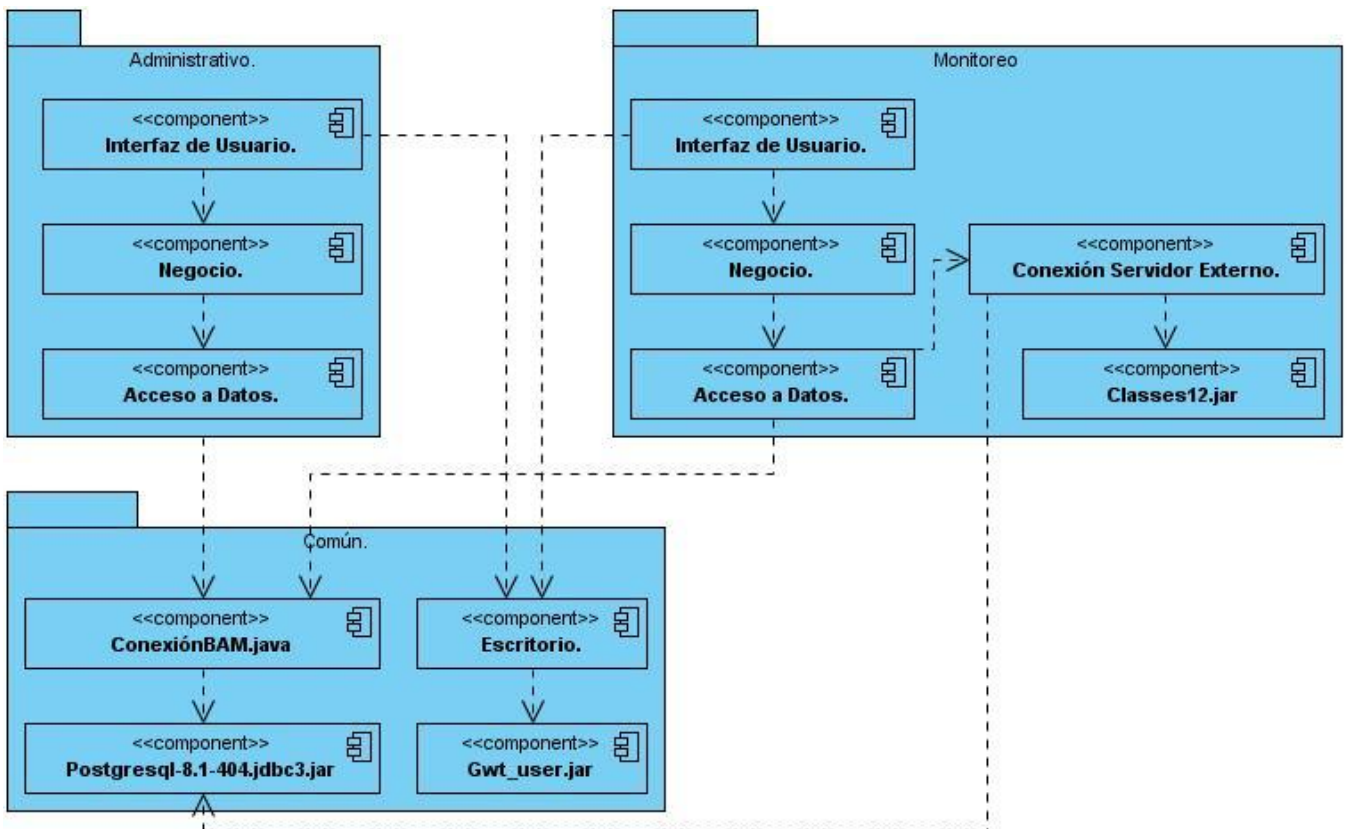


Figura 8. Diagrama de Componentes por Módulos.

El Diagrama de componentes por módulos:

- En el módulo administrativo y en el módulo de monitoreo se encuentra un componente **Interfaz de Usuario** que se relaciona con el componente **Escritorio** del módulo común encargado de la creación de la interfaz de usuario. El componente **Interfaz de Usuario** de cada módulo se comunica con su respectivo componente de **Negocio**.
- En el módulo administrativo y en el módulo de monitoreo se encuentra un componente **Negocio** que se relaciona con su respectivo componente de **Acceso a Datos**.
- En el módulo administrativo y en el módulo de monitoreo se encuentra un componente **Acceso a Datos** que se relaciona con el componente **ConexionBAM.java** del módulo común encargado de establecer la conexión con servidor de base de datos local. También el componente de **Acceso a Datos** del módulo de monitoreo se comunica con el componente **Conexión Servidor Externo** encargado de crear la conexión con los servidores externos a través de los componentes **Classes12.jar** y **Postgresql-8.1-404-jdbc3.jar**.



- En el módulo común se encuentra el componente **ConexionBAM.java** que se relaciona con el componente **Postgresql-8.1-404-jdbc3.jar** que provee el driver para crear la conexión con el servidor de base de datos local. Se localiza, además, el componente **Escritorio** que se comunica con el componente **Gwt\_user.jar**.

### 3.3 Prueba.

El desarrollo de sistemas de software implica una serie de actividades de producción en las que las posibilidades de que aparezca el fallo humano son enormes. Debido a la imposibilidad humana de trabajar y comunicarse de forma perfecta, el desarrollo del software ha de ir acompañado de una actividad que garantice la calidad.

Las pruebas de software son una actividad en la cual el sistema es ejecutado bajo ciertas condiciones o requerimientos específicos. Los resultados son observados y registrados, y se realiza una evaluación de algún aspecto determinado del sistema. Las pruebas son un elemento crítico como garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación. (Pressman, 2002)

Las pruebas de software que se efectuaron:

- Pruebas unitarias: Para asegurar que en cada uno de los módulos definidos el control de flujos y de datos funcione correctamente. Como parte de estas pruebas se aplicará el método de Caja blanca.
- Pruebas de caja negra: Para demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y se produce un resultado correcto.

#### 3.3.1 Método de Caja Blanca.

Las pruebas de caja blanca se basan en el minucioso examen de los detalles procedimentales. Se comprueban los caminos lógicos del software proponiendo casos de prueba que examinen que están correctas todas las condiciones para determinar si el estado real coincide con el esperado.

Mediante las pruebas de caja blanca se pueden obtener casos de prueba que:

- Garanticen que se ejerciten por lo menos una vez todo los caminos independientes de cada módulo, programa o método.
- Ejerciten todas las decisiones lógicas en las vertientes verdaderas y falsas.



- Ejecuten todos los bucles en sus límites operacionales.
- Ejerciten las estructuras internas de datos para asegurar su validez.

Es por ello que se considera a la prueba de caja blanca como uno de los tipos de pruebas más importantes que se les aplica al software, lo que logra que disminuya en un gran porcentaje el número de errores existentes en los sistemas y, por ende, una mayor calidad y confiabilidad.

Para la realización de las pruebas de caja blanca se utilizará la prueba del camino básico, la cual permite obtener una medida de la complejidad lógica del diseño y usar esta medida como guía para la definición de un conjunto básico de caminos de ejecución. La idea es derivar casos de prueba a partir de un conjunto dado de caminos independientes por los cuales puede circular el flujo de control.

Para obtener dicho conjunto de caminos independientes se construye el grafo de flujo asociado y se calcula su complejidad ciclomática. Los pasos que se siguen para aplicar esta técnica son:

- A partir del diseño o del código fuente, se dibuja el grafo de flujo asociado.
- Se calcula la complejidad ciclomática del grafo.
- Se determina un conjunto básico de caminos independientes.
- Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

Los casos de prueba derivados del conjunto básico garantizan que durante la prueba se ejecuta, por lo menos, una vez cada sentencia del programa.

### **Complejidad ciclomática:**

**V (G):** Número de regiones del grafo.

$$V(G) = A - N + 2$$

**A:** Número de aristas del grafo.

**N:** Número de nodos.

$$V(G) = P + 1$$

**P:** Número de nodos predicados.



De acuerdo a la porción de código correspondiente a la función **ComprobarConexionOracle**, perteneciente al componente Interfaz de Usuario, del módulo Monitoreo, se realizó la prueba de caja blanca:

```

public void handleEvent(BaseEvent be) { 1
    if (ValidardatosEntrada()) { 2
        MessageBox.info("Información.", "Todos los campos de la conexión son requeridos.", null); 3
    }
    else{ 4
        Conexion conec = new Conexion(); 4
        conec.setld_tipoConex("1"); 4
        conec.setHost(host.getValue()); 4
        conec.setPuerto(puerto.getValue().toString()); 4
        conec.setSid(sid.getValue()); 4
        conec.setUser(usuario.getValue()); 4
        conec.setPassword(contrasena.getValue()); 4
        conec.setInternal_logon(conectar.getValue().get("conectar").toString()); 4
        getService().comprobarConexion(conec, new AsyncCallback<Integer>() { 5
            public void onFailure(Throwable caught) { 6
                MessageBox.info("Información.", "Conexión fallida.", null); 6
            }
            public void onSuccess(Integer result) { 7
                MessageBox.info("Información.", "Conexión correcta.", null); 7
            }
        });
    }
}

```



```

    }
    });
}
}

```

8

9

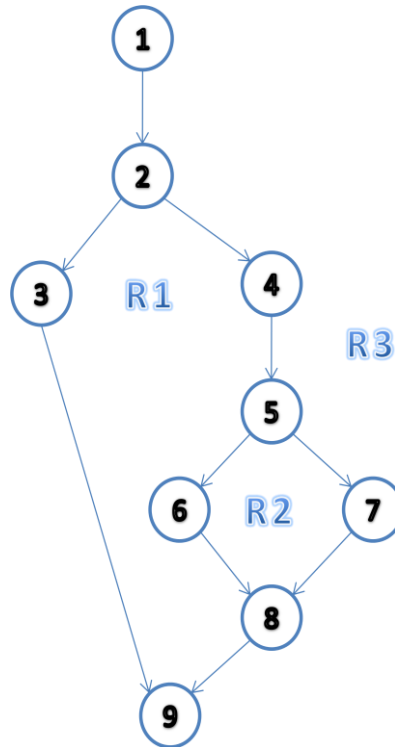


Figura 9. Grafo de Flujo del Caso de Prueba Comprobar Conexión con Oracle.

**V (G) = 3**

**Caminos:** 1-2-3-9, 1-2-4-5-6-8-9, 1-2-4-5-7-8-9.

**Camino:** 1-2-3-9.

**Caso de prueba:** Comprobar Conexión con Oracle.

**Entrada:** Recibe parámetros de la conexión con Oracle no válidos o nulos.



**Resultado:** Se verifican todos los parámetros de la conexión con Oracle. Como existen parámetros no válidos o nulos, el sistema muestra un mensaje donde informa que todos los campos son requeridos para la conexión con el servidor externo.

**Condiciones:** Haber pasado los parámetros de la conexión con Oracle no válidos o nulos.

**Camino:** 1-2-4-5-6-8-9.

**Caso de prueba:** Comprobar Conexión con Oracle.

**Entrada:** Recibe parámetros de la conexión con Oracle incorrectos.

**Resultado:** Se verifican todos los parámetros de la conexión con Oracle. Se comprueba la conexión, pero como existen parámetros incorrectos, el sistema muestra un mensaje donde informa que la conexión con el servidor externo ha fracasado.

**Condiciones:** Haber pasado los parámetros de la conexión con Oracle incorrectos.

**Camino:** 1-2-4-5-7-8-9.

**Caso de prueba:** Comprobar Conexión con Oracle.

**Entrada:** Recibe parámetros de la conexión con Oracle correctos.

**Resultado:** Se verifican todos los parámetros de la conexión con Oracle, se comprueba la conexión y el sistema muestra un mensaje donde informa que la conexión con el servidor externo ha sido satisfactoria.

**Condiciones:** Haber pasado los parámetros de la conexión con Oracle correctos.

De acuerdo a la porción de código correspondiente a la función **ComprobarConexion**, perteneciente al componente Acceso a Datos, del módulo Monitoreo, se realizó la prueba de caja blanca:

```
public int comprobarConexion(Conexion conec) throws Exception {           1
    if (conec.getId_tipoConex().equals("1")) {                             2
        ConexionOracle con = new ConexionOracle(conec);                   3
```



con.ComprobarConexion();	<b>3</b>
con.CloseConexion();	<b>3</b>
return 1;	<b>3</b>
}	
else{	
ConexionPostGreSQL con = new ConexionPostGreSQL(conec);	<b>4</b>
con.ComprobarConexion();	<b>4</b>
con.CloseConexion();	<b>4</b>
return 1;	<b>4</b>
}	
}	<b>5</b>

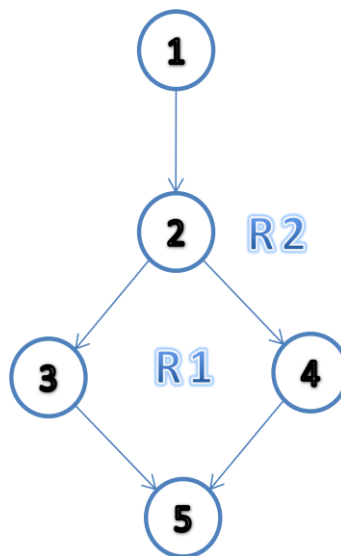


Figura 10. Grafo de Flujo del Caso de Prueba Comprobar Conexión.

**V (G) = 2**

**Caminos:** 1-2-3-5, 1-2-4-5.



**Camino:** 1-2-3-5

**Caso de prueba:** Comprobar Conexión.

**Entrada:** Recibe una conexión con Oracle.

**Resultado:** Se verifica el identificador del tipo de conexión, si es igual a 1 se crea el objeto conexión para el gestor de base de datos Oracle, se comprueba la conexión con el servidor externo, y luego se informa al usuario si la conexión realizada fue satisfactoria o no.

**Condiciones:** Haber pasado por parámetro una conexión con Oracle.

**Camino:** 1-2-4-5

**Caso de prueba:** Comprobar Conexión.

**Entrada:** Recibe una conexión con PostgreSQL.

**Resultado:** Se verifica el identificador del tipo de conexión, si es distinto de 1 se crea el objeto conexión para el gestor de base de datos PostgreSQL. Se comprueba la conexión con el servidor externo, y luego se informa al usuario si la conexión realizada fue satisfactoria o no.

**Condiciones:** Haber pasado por parámetro una conexión con PostgreSQL.

### 3.3.2 Método de Caja Negra.

Las pruebas de caja negra se centran principalmente en los requisitos funcionales del software. Son un enfoque complementario que intenta descubrir diferentes tipos de errores a los encontrados en los métodos de caja blanca.

Estos métodos permiten encontrar:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de inicialización y terminación.

Dentro del método de caja negra se utilizará la técnica de partición de equivalencia la cual permite examinar valores válidos o inválidos de las entradas existentes en el sistema.





Entrada	Resultados	Condiciones
<p>El usuario selecciona la opción Adicionar conexión (Oracle). Se muestra un formulario para que el usuario introduzca los datos de la conexión que desea crear:</p> <p><b>Nombre:</b> Prueba1  <b>Host:</b> 192.168.104.224  <b>Puerto:</b> 1521  <b>SID:</b> master  <b>Usuario:</b> identidad  <b>Contraseña:</b> 1qazxsw2  <b>Conectar como:</b> Normal</p> <p>Selecciona la opción Comprobar conexión.</p>	<p>Verifica que no existan campos inválidos o nulos.</p> <p>Si los datos introducidos están correctos pasa a comprobar la conexión.</p> <p>Se muestra un mensaje indicando que la conexión fue correcta.</p> <p><b>Alertas.</b></p> <ul style="list-style-type: none"> <li>• Todos los campos son requeridos.</li> <li>• Conexión Fallida.</li> <li>• Conexión Correcta.</li> </ul>	<p>El usuario debe estar autenticado en el sistema y haber obtenido los permisos necesarios para gestionar conexión.</p> <p>Haber introducido los datos correctamente y que el servidor a donde desea conectarse esté ejecutándose.</p> <p><b>Prueba satisfactoria.</b>  <b>Se cumplieron los objetivos.</b></p>
<p><b>Nombre:</b> Prueba1  <b>Host:</b> 192.168.104.224  <b>Puerto:</b> 1521  <b>SID:</b> master  <b>Usuario:</b> identidad  <b>Contraseña:</b> 1qazxsw2  <b>Conectar como:</b> Normal</p> <p>El usuario selecciona la opción Adicionar conexión.</p>	<p>Verifica que no existan campos inválidos o nulos.</p> <p>Si los datos introducidos están correctos pasa a adicionar la conexión.</p> <p>Se muestra un mensaje indicando que la conexión fue insertada.</p> <p><b>Alertas.</b></p> <ul style="list-style-type: none"> <li>• Todos los campos son requeridos.</li> <li>• Conexión Fallida.</li> <li>• Conexión insertada.</li> </ul>	<p>El usuario debe estar autenticado en el sistema y haber obtenido los permisos necesarios para gestionar conexión.</p> <p>Haber introducido los datos correctamente y que el servidor a donde desea conectarse esté ejecutándose.</p> <p><b>Prueba satisfactoria.</b>  <b>Se cumplieron los objetivos.</b></p>

Tabla 2.7 Prueba de Caja Negra, Funcionalidad "Adicionar Conexión (Oracle)".

Entrada	Resultados	Condiciones
<p>El usuario selecciona la opción Adicionar conexión (PostgreSQL). Se muestra un formulario para que el usuario introduzca los datos de la conexión que desea crear:</p>	<p>Verifica que no existan campos inválidos o nulos.</p> <p>Si los datos introducidos están correctos pasa a comprobar la conexión.</p>	<p>El usuario debe estar autenticado en el sistema y haber obtenido los permisos necesarios para gestionar conexión.</p>



<p><b>Nombre:</b> Prueba2  <b>Host:</b> 192.168.104.224  <b>Puerto:</b> 5432  <b>SID:</b> prueba  <b>Usuario:</b> prueba  <b>Contraseña:</b> 1qazxsw2</p> <p>Selecciona la opción Comprobar conexión.</p>	<p>Se muestra un mensaje indicando que la conexión fue correcta.</p> <p><b>Alertas.</b></p> <ul style="list-style-type: none"> <li>• Todos los campos son requeridos.</li> <li>• Conexión Fallida.</li> <li>• Conexión Correcta.</li> </ul>	<p>Haber introducido los datos correctamente y que el servidor a donde desea conectarse esté ejecutándose.</p> <p><b>Prueba satisfactoria.</b>  <b>Se cumplieron los objetivos.</b></p>
<p><b>Nombre:</b> Prueba2  <b>Host:</b> 192.168.104.224  <b>Puerto:</b> 5432  <b>SID:</b> prueba  <b>Usuario:</b> prueba  <b>Contraseña:</b> 1qazxsw2</p> <p>El usuario selecciona la opción Adicionar conexión.</p>	<p>Verifica que no existan campos inválidos o nulos.</p> <p>Si los datos introducidos están correctos pasa a adicionar la conexión.</p> <p>Se muestra un mensaje indicando que la conexión fue insertada.</p> <p><b>Alertas.</b></p> <ul style="list-style-type: none"> <li>• Todos los campos son requeridos.</li> <li>• Conexión Fallida.</li> <li>• Conexión Insertada.</li> </ul>	<p>El usuario debe estar autenticado en el sistema y haber obtenido los permisos necesarios para gestionar conexión.</p> <p>Haber introducido los datos correctamente y que el servidor a donde desea conectarse esté ejecutándose.</p> <p><b>Prueba satisfactoria.</b>  <b>Se cumplieron los objetivos.</b></p>

**Tabla 2.8 Prueba de Caja Negra, Funcionalidad "Adicionar Conexión (PostgreSQL)".**

Entrada	Resultados	Condiciones
<p>El usuario selecciona la opción Editar conexión. Se muestra un formulario con todas las conexiones realizadas.</p> <p>El usuario selecciona la conexión que desea modificar.</p> <p>Selecciona la conexión que tiene por nombre Prueba1.</p> <p>El usuario modifica los datos.</p> <p><b>Nombre:</b> Prueba1  <b>Host:</b> 192.168.104.224  <b>Puerto:</b> 1521</p>	<p>Se buscan todas las conexiones realizadas y luego son mostradas al usuario para su modificación.</p> <p>Verifica que se haya seleccionado una conexión.</p> <p>Se muestran los datos de la conexión seleccionada.</p> <p>Si los datos introducidos están correctos, verifica que los datos se cambiaron, pasa a comprobar la conexión y luego la modifica.</p> <p>Se muestra un mensaje indicando</p>	<p>El usuario debe estar autenticado en el sistema y haber obtenido los permisos necesarios para gestionar conexión.</p> <p>Haber seleccionado una conexión, luego de introducidos los datos correctamente y que el servidor en el que desea modificar la conexión esté ejecutándose.</p> <p><b>Prueba satisfactoria.</b></p>



<b>SID:</b> master <b>Usuario:</b> sys <b>Contraseña:</b> 1qazxsw2 <b>Conectar como:</b> Sysdba Selecciona la opción Modificar conexión.	que la conexión fue modificada.  <b>Alertas.</b> <ul style="list-style-type: none"> <li>• Debe seleccionar una conexión.</li> <li>• Debe modificar la conexión.</li> <li>• Conexión Modificada.</li> </ul>	<b>Se cumplieron los objetivos.</b>
--	--	-------------------------------------

Tabla 2.9 Prueba de Caja Negra, Funcionalidad "Modificar Conexión".

Entrada	Resultados	Condiciones
El usuario selecciona la opción Editar conexión. Se muestra un formulario con todas las conexiones realizadas. El usuario selecciona la conexión que desea eliminar. Selecciona la conexión que tiene por nombre Prueba2. Selecciona la opción Eliminar conexión.	Se buscan todas las conexiones realizadas y luego son mostradas al usuario para su eliminación. Verifica que se haya seleccionado una conexión y se elimina. Se muestra un mensaje indicando que la conexión fue eliminada.  <b>Alertas.</b> <ul style="list-style-type: none"> <li>• Debe seleccionar una conexión.</li> <li>• Conexión Eliminada.</li> </ul>	El usuario debe estar autenticado en el sistema y haber obtenido los permisos necesarios para gestionar conexión. Haber seleccionado una conexión.  <b>Prueba satisfactoria.</b> <b>Se cumplieron los objetivos.</b>

Tabla 2.10 Prueba de Caja Negra, Funcionalidad "Eliminar Conexión".

## CONCLUSIONES.

Se define el diagrama de despliegue y se describe cómo quedará el sistema una vez que haya sido desplegado. Se muestra la forma en que se desarrolló la aplicación haciendo uso de los diagramas de componentes. Se realizan pruebas al software con el objetivo de verificar y relevar la calidad del sistema.

El servidor de aplicación debe tener mayores prestaciones que el servidor de base de datos. Se puede concluir que el sistema a partir de los resultados obtenidos durante la fase de prueba, se encuentra listo para desplegarse.



# CONCLUSIONES GENERALES.

Los sistemas de “Monitorización” presentan soluciones eficaces al procesamiento de la información. Ha sido objetivo de este trabajo mostrar una solución a esta problemática, enfocada fundamentalmente al procesamiento de información de las métricas asociadas a los procesos de negocio.

El resultado es la obtención de una solución BAM, la cual permitirá monitorizar el comportamiento de los procesos de negocio en tiempo real. Se logró su implementación con herramientas de desarrollo de software libre y la posibilidad de su despliegue en diferentes Sistemas Operativos.

El desarrollo de esta investigación está orientado a la concepción de un sistema para el análisis y monitoreo de la información durante un período de tiempo deseado, el cual podrá obtener de forma eficiente y rápida la información procedente de un gran número de fuentes (tanto internas como externas a la organización). El valor fundamental de esta herramienta se expresa en la contribución a simplificar el trabajo y la demora que produce la gestión y el análisis en texto o verbal de la información.

Con el estudio realizado a lo largo de todas las etapas de desarrollo, el sistema que se ha implementado cumple con los objetivos generales del trabajo de diploma: se desarrolló una solución con funcionalidades para el procesamiento, análisis y monitoreo de los indicadores de negocio, además, se lograron representar gráficamente distintas informaciones de las métricas que se procesaron.

Con la implementación del sistema de análisis y monitoreo de los procesos de negocio se logra:

- Posibilitar el almacenamiento de la información, lograr disponibilidad a la hora de las representaciones y con esto la velocidad de la toma de decisiones.
- Eliminar todo tipo de documentos en la comunicación entre los encargados de la monitorización y toma de decisiones.
- Portabilidad del sistema y su adaptabilidad a cambios que puedan producirse en los negocios dentro de una organización.
- Demostrar la eficiencia de los lenguajes y tecnologías empleadas en el desarrollo del sistema.



- Diseñar y construir una base de datos donde se almacena toda la información de las métricas asociadas a los procesos de negocio, para de esta forma garantizar la veracidad y centralización de la misma.
- Realizar el análisis, diseño e implementación del sistema. Se ha seguido los principios de diseño descritos para el desarrollo del mismo y se ha logrado una seguridad y protección de la información consecuente con el nivel de seguridad requerido.
- Renderizar gráficos para mostrar los cambios en los procesos de negocio en tiempo real.



## RECOMENDACIONES.

El objetivo general de este trabajo ha sido logrado, pero en el transcurso de su desarrollo, fueron surgiendo ideas que podrían implementarse en un futuro, de forma que se logre una aplicación más útil y efectiva, por lo cual se recomienda:

- Seguir trabajando en el desarrollo modular de la solución en próximas iteraciones.
- Ampliar el modelo actual de configuración y diseñar e implementar herramientas que den la posibilidad de gestionar dicha configuración.
- Desarrollar más funcionalidades al sistema, por ejemplo la gestión de respuestas a las alertas emitidas por el sistema.
- Encontrar una forma mediante la cual el sistema pueda interactuar con los distintos medios de información para los que se puede configurar un sistema BAM, ya sea correo electrónico, correo de voz u otro.
- Profundizar en el estudio del marco de trabajo empleado (*Google Web Toolkit*) con el objetivo de optimizar la utilización del mismo, y así obtener mayores rendimientos.



## REFERENCIAS BIBLIOGRÁFICAS.

**Armstrong, Peter. 2007.** *Muévase a Business Service Management*. USA : s.n., 2007.

**COMMUNICATIONS, IDG. 2009.** [En línea] 2009. [Citado el: 23 de Enero de 2010.]  
<http://www.idg.es/computerworld/articulo.asp?id=175254>.

**Consulting, Business Intelligence. 2010.** BI Consulting. [En línea] 2010. [Citado el: 27 de Enero de 2010.]  
[http://www.biconsulting.com.mx/index.php?option=com\\_content&view=article&id=51&Itemid=58](http://www.biconsulting.com.mx/index.php?option=com_content&view=article&id=51&Itemid=58).

**Dewsbury, Ryan. 2008.** *Google Web Toolkit Applications*. USA : Prentice Hall, 2008. ISBN-13: 978-0-321-50196-7, ISBN-10: 0-321-50196-9.

**Eguíluz Pérez, Javier. 2008.** Introducción a AJAX. [En línea] 7 de Junio de 2008. [Citado el: 13 de Enero de 2010.]  
<http://librosweb.es/ajax/index.html>.

**Gartner. 1 Abril 2002.** *Business Activity Monitoring: Calm Before the Storm*. 1 Abril 2002. LE-15-9727.

**Google. 2009.** Google Web Toolkit. [En línea] 2009. [Citado el: 20 de Enero de 2010.]  
<http://code.google.com/webtoolkit/overview.html>.

**Group, Computing. 2010.** Barcelona04. [En línea] 2010. [Citado el: 20 de Enero de 2010.]  
<http://www.barcelona04.com/soluciones/bsm/index.php>.

**Group, Information Technology. 2008.** inteGreat. [En línea] 2008. [Citado el: 19 de Enero de 2010.]  
<http://integreat.com.mx/index.html>.

**ICES. 2009.** Ingeniería, Consultoría y Estrategias de Sistemas. [En línea] 2009. [Citado el: 23 de Enero de 2010.]  
<http://www.icensa.net/>.

**Información, SPL Sistemas de. 2008.** SPL Sistemas de Información, S.L. [En línea] 2008. [Citado el: 3 de Marzo de 2010.] [http://www.spl-ssi.com/?sec=articulos&subsec=descripcion&v=aplicaciones\\_web](http://www.spl-ssi.com/?sec=articulos&subsec=descripcion&v=aplicaciones_web).

**Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000.** *El Proceso Unificado de Desarrollo del Software*. España, Madrid : Pearson Educación, 2000. 84-7829-036-2.

**Larman, Craig. 1999.** *UML y Patrones: Introducción al análisis y programación orientada a objetos*. México : Prentice Hall, 1999.

**López, Calos. 2008.** GestioPolis. [En línea] 2008. [Citado el: 24 de Enero de 2010.]  
<http://www.gestiopolis.com/administracion-estrategia/dashboard-kpi-metricas.htm>.



**Madrid, Universidad Carlos III de. 2007.** Introducción a los sistemas de información: El modelo Cliente/Servidor. [En línea] 2007. [Citado el: 21 de Enero de 2010.] [http://www.it.uc3m.es/mcftp/docencia/si/material/1\\_cliente\\_ser\\_mcfp.pdf](http://www.it.uc3m.es/mcftp/docencia/si/material/1_cliente_ser_mcfp.pdf).

**Microsoft. 2009.** [En línea] 2009. [Citado el: 25 de Enero de 2010.] <http://www.microsoft.com/spain/bi/SolutionScenarios/bam.msp>.

**Microsystems, Sun. 2009.** NetBeans. [En línea] 2009. [Citado el: 27 de Enero de 2010.] <http://netbeans.org/features/>.

**Oracle. 2010.** ForGlassFishV3. [En línea] 2010. [Citado el: 22 de Abril de 2010.] <http://wiki.glassfish.java.net/Wiki.jsp?page=PlanForGlassFishV3#section-PlanForGlassFishV3-Documentation>.

—. **2009.** Java. [En línea] 2009. [Citado el: 20 de Febrero de 2010.] [http://java.com/es/download/whatis\\_java.jsp](http://java.com/es/download/whatis_java.jsp).

—. **2009.** Oracle WebSites. [En línea] 2009. [Citado el: 27 de Enero de 2010.] <http://www.oracle.com/appserver/business-activity-monitoring.html>.

**Paradigm, Visual. 2009.** Boost Productivity with Innovative and Intuitive Technologies. [En línea] 2009. [Citado el: 3 de Marzo de 2010.] <http://www.visual-paradigm.com/shop/vpuml.jsp>.

**Peralta, M. 2007.** *Estimación del Esfuerzo Basada en Casos de Usos*. 2007.

**PostgreSQL. 2009.** PostgreSQL. [En línea] 2009. [Citado el: 14 de Enero de 2010.] <http://www.postgresql.org/about/>.

**Pressman, Roger S. 2002.** *Ingeniería de Software. Un enfoque práctico*. Madrid : McGraw-Hill, 2002. 8448132149, 9788448132149.

**Proenza, Y. 2005.** *Introducción al modelo conceptual*. Ciudad de La Habana, Universidad de las Ciencias Informáticas : s.n., 2005.

**Rojas, Elisabeth. 2010.** La Web del Profesional de la Tecnología. [En línea] 2010. [Citado el: 22 de Abril de 2010.] [http://muycomputerpro.com/FrontHome/\\_wE9ERk2XxDAWDxuZ0-93wnenIQnRqNcvIxPjZ7pG7AxSEbL2nCu2\\_2L5xBX\\_wEYK](http://muycomputerpro.com/FrontHome/_wE9ERk2XxDAWDxuZ0-93wnenIQnRqNcvIxPjZ7pG7AxSEbL2nCu2_2L5xBX_wEYK).

**Smith, Howard y Fingar, Peter. 2003.** *Business Process Management: The Third Wave*. Tampa, Florida, USA : Meghan-Kiffer Press, 2003. ISBN 0929652339.

**Software, Departamento Central de Ingeniería de. 2004.** *Flujo de Trabajo Captura de Requisitos. Modelo de Negocio*. Ciudad de La Habana, Universidad de las Ciencias Informáticas : s.n., 2004.

**Systar. 2010.** Manage with vision. [En línea] 2010. [Citado el: 24 de Enero de 2010.] <http://www.systar.com/>.





**TIBCO. 2009.** The Power of Now. [En línea] 2009. [Citado el: 16 de Enero de 2010.]  
<http://www.tibco.com/solutions/soa/tibco-solutions/default.jsp>.

**Valdés, Damián Pérez. 2009.** Maestros del Web. [En línea] 2009. [Citado el: 28 de Enero de 2001.]  
<http://www.maestrosdelweb.com/editorial/%C2%BFque-es-javascript/>.

**WebMethods, Inc. 2006.** *Business Activity Monitoring (BAM) The New Face of BPM.* 2006.

**WSO2, Inc. 2010.** Oxygen Tank THE DEVELOPER PORTAL FOR SOA. [En línea] 2010. [Citado el: 3 de Febrero de 2010.] <http://wso2.org/projects/bam>.



## BIBLIOGRAFÍA.

**Armstrong, Peter. 2007.** *Muévase a Business Service Management*. USA : s.n., 2007.

**COMMUNICATIONS, IDG. 2009.** [En línea] 2009. [Citado el: 23 de Enero de 2010.]  
<http://www.idg.es/computerworld/articulo.asp?id=175254>.

**Consulting, Business Intelligence. 2010.** BI Consulting. [En línea] 2010. [Citado el: 27 de Enero de 2010.]  
[http://www.biconsulting.com.mx/index.php?option=com\\_content&view=article&id=51&Itemid=58](http://www.biconsulting.com.mx/index.php?option=com_content&view=article&id=51&Itemid=58).

**Dewsbury, Ryan. 2008.** *Google Web Toolkit Applications*. USA : Prentice Hall, 2008. ISBN-13: 978-0-321-50196-7, ISBN-10: 0-321-50196-9.

**Eguíluz Pérez, Javier. 2008.** Introducción a AJAX. [En línea] 7 de Junio de 2008. [Citado el: 13 de Enero de 2010.]  
<http://librosweb.es/ajax/index.html>.

**Gartner. 1 Abril 2002.** *Business Activity Monitoring: Calm Before the Storm*. 1 Abril 2002. LE-15-9727.

**Google. 2009.** Google Web Toolkit. [En línea] 2009. [Citado el: 20 de Enero de 2010.]  
<http://code.google.com/webtoolkit/overview.html>.

**Group, Computing. 2010.** Barcelona04. [En línea] 2010. [Citado el: 20 de Enero de 2010.]  
<http://www.barcelona04.com/soluciones/bsm/index.php>.

**Group, Information Technology. 2008.** inteGreat. [En línea] 2008. [Citado el: 19 de Enero de 2010.]  
<http://integreat.com.mx/index.html>.

**ICES. 2009.** Ingeniería, Consultoría y Estrategias de Sistemas. [En línea] 2009. [Citado el: 23 de Enero de 2010.]  
<http://www.icensa.net/>.

**Información, SPL Sistemas de. 2008.** SPL Sistemas de Información, S.L. [En línea] 2008. [Citado el: 3 de Marzo de 2010.] [http://www.spl-ssi.com/?sec=articulos&subsec=descripcion&v=aplicaciones\\_web](http://www.spl-ssi.com/?sec=articulos&subsec=descripcion&v=aplicaciones_web).

**Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000.** *El Proceso Unificado de Desarrollo del Software*. España, Madrid : Pearson Educación, 2000. 84-7829-036-2.

**Larman, Craig. 1999.** *UML y Patrones: Introducción al análisis y programación orientada a objetos*. México : Prentice Hall, 1999.

**López, Calos. 2008.** GestioPolis. [En línea] 2008. [Citado el: 24 de Enero de 2010.]  
<http://www.gestiopolis.com/administracion-estrategia/dashboard-kpi-metricas.htm>.



**Madrid, Universidad Carlos III de. 2007.** Introducción a los sistemas de información: El modelo Cliente/Servidor. [En línea] 2007. [Citado el: 21 de Enero de 2010.] [http://www.it.uc3m.es/mcftp/docencia/si/material/1\\_cliente\\_ser\\_mcfp.pdf](http://www.it.uc3m.es/mcftp/docencia/si/material/1_cliente_ser_mcfp.pdf).

**Microsoft. 2009.** [En línea] 2009. [Citado el: 25 de Enero de 2010.] <http://www.microsoft.com/spain/bi/SolutionScenarios/bam.mspix>.

**Microsystems, Sun. 2009.** NetBeans. [En línea] 2009. [Citado el: 27 de Enero de 2010.] <http://netbeans.org/features/>.

**Oracle. 2010.** ForGlassFishV3. [En línea] 2010. [Citado el: 22 de Abril de 2010.] <http://wiki.glassfish.java.net/Wiki.jsp?page=PlanForGlassFishV3#section-PlanForGlassFishV3-Documentation>.

—. **2009.** Java. [En línea] 2009. [Citado el: 20 de Febrero de 2010.] [http://java.com/es/download/whatis\\_java.jsp](http://java.com/es/download/whatis_java.jsp).

—. **2009.** Oracle WebSites. [En línea] 2009. [Citado el: 27 de Enero de 2010.] <http://www.oracle.com/appserver/business-activity-monitoring.html>.

**Paradigm, Visual. 2009.** Boost Productivity with Innovative and Intuitive Technologies. [En línea] 2009. [Citado el: 3 de Marzo de 2010.] <http://www.visual-paradigm.com/shop/vpuml.jsp>.

**Peralta, M. 2007.** *Estimación del Esfuerzo Basada en Casos de Usos*. 2007.

**PostgreSQL. 2009.** PostgreSQL. [En línea] 2009. [Citado el: 14 de Enero de 2010.] <http://www.postgresql.org/about/>.

**Pressman, Roger S. 2002.** *Ingeniería de Software. Un enfoque práctico*. Madrid : McGraw-Hill, 2002. 8448132149, 9788448132149.

**Proenza, Y. 2005.** *Introducción al modelo conceptual*. Ciudad de La Habana, Universidad de las Ciencias Informáticas : s.n., 2005.

**Rojas, Elisabeth. 2010.** La Web del Profesional de la Tecnología. [En línea] 2010. [Citado el: 22 de Abril de 2010.] [http://muycomputerpro.com/FrontHome/\\_wE9ERk2XxDAWDxuZ0-93wnenIQnRqNcvIxPjZ7pG7AxSEbL2nCu2\\_2L5xBX\\_wEYK](http://muycomputerpro.com/FrontHome/_wE9ERk2XxDAWDxuZ0-93wnenIQnRqNcvIxPjZ7pG7AxSEbL2nCu2_2L5xBX_wEYK).

**Smith, Howard y Fingar, Peter. 2003.** *Business Process Management: The Third Wave*. Tampa, Florida, USA : Meghan-Kiffer Press, 2003. ISBN 0929652339.

**Software, Departamento Central de Ingeniería de. 2004.** *Flujo de Trabajo Captura de Requisitos. Modelo de Negocio*. Ciudad de La Habana, Universidad de las Ciencias Informáticas : s.n., 2004.

**Systar. 2010.** Manage with vision. [En línea] 2010. [Citado el: 24 de Enero de 2010.] <http://www.systar.com/>.



**TIBCO. 2009.** The Power of Now. [En línea] 2009. [Citado el: 16 de Enero de 2010.]  
<http://www.tibco.com/solutions/soa/tibco-solutions/default.jsp>.

**Valdés, Damián Pérez. 2009.** Maestros del Web. [En línea] 2009. [Citado el: 28 de Enero de 2001.]  
<http://www.maestrosdelweb.com/editorial/%C2%BFque-es-javascript/>.

**WebMethods, Inc. 2006.** *Business Activity Monitoring (BAM) The New Face of BPM*. 2006.

**WSO2, Inc. 2010.** Oxygen Tank THE DEVELOPER PORTAL FOR SOA. [En línea] 2010. [Citado el: 3 de Febrero de 2010.] <http://wso2.org/projects/bam>.



# GLOSARIO DE TÉRMINOS.

**Alertas:** Aviso instantáneo generado por el sistema en busca de una respuesta inmediata a un suceso inesperado o que no está siendo supervisado, pudiera ser un mensaje de correo o un SMS al ejecutivo encargado al respecto.

**API (*Application Programming Interface*):** es el conjunto de funciones y procedimientos o métodos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

**Arquitectura Cliente/Servidor:** es un modelo para el desarrollo de sistemas de información, en el que las transacciones se dividen en elementos independientes que cooperan entre sí para intercambiar información, servicios o recursos.

**Asíncrona:** referencia al suceso que no tiene lugar en total correspondencia temporal con otro suceso. Por ejemplo podemos hablar de motor asíncrono a aquel cuya velocidad de rotación no corresponde con la frecuencia de corriente alterna que lo hace funcionar.

**Bajo acoplamiento (*loose coupling*):** Característica de las soluciones referentes a la mínima dependencia que existe entre servicios, lo que permite una rápida adaptación a los cambios de ambiente.

**BAM (*Business Activity Monitoring*):** orientado a brindar acceso en tiempo real a los indicadores claves de desempeño de la empresa, con el objetivo de mejorar la velocidad y efectividad de sus operaciones y procesos. Se trata de una solución que apunta específicamente a que los encargados de la gestión dispongan de información para tomar mejores decisiones y puedan ser alertados automáticamente ante cualquier problemática que ponga en riesgo al negocio.

**BI (*Business Intelligence*):** utiliza los indicadores claves de desempeño para asistir o ayudar al estado actual de un negocio a prescribir una línea de acción futura.

**BPM (*Business Process Management*):** es un conjunto estructurado de actividades, diseñado para producir una salida determinada o lograr un objetivo. Los procesos describen cómo es realizado el trabajo en la empresa y se caracterizan por ser observables, medibles, mejorables y repetitivos.



**BSM (*Business Service Management*):** es una metodología para el seguimiento y la medición de tecnología de la información desde una perspectiva empresarial

**Compilador:** Programa informático que traduce un programa escrito en un lenguaje de programación a otro lenguaje de programación, generando un programa equivalente que la máquina será capaz de interpretar.

**DOM (*Document Object Model*):** es una forma de representar documentos estructurados (tales como una página web HTML o un documento XML) que es independiente de cualquier lenguaje orientado a objetos.

**Escalabilidad:** Propiedad de un sistema, una red o un proceso, que indica su habilidad para manejar el continuo crecimiento de trabajo de manera fluida y para estar preparado, para hacerse más grande sin perder calidad en los servicios ofrecidos.

**Framework:** Estructura de soporte definida, mediante la cual otro proyecto de software puede ser organizado y desarrollado.

**HTML (*HyperText Markup Language*):** es un lenguaje de marcas. Los lenguajes de marcas no son equivalentes a los lenguajes de programación aunque se definan igualmente como "lenguajes".

**HTTP (Protocolo de Transferencia de Hipertexto):** Protocolo usado en las transacciones de la Web. Define la sintaxis y la semántica que utilizan los elementos software de la arquitectura web (clientes, servidores) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.

**KPIs (*Key Performance Indicators*):** son métricas utilizadas para cuantificar objetivos que reflejan el rendimiento de una organización, y que generalmente se recogen en su plan estratégico.

**Procedimientos de Llamadas Remoto (RPC):** interfaz de programación desarrollada por *Sun Microsystems* que permite el desarrollo de aplicaciones distribuidas. Mediante un conjunto de funciones, este protocolo permite que los programas llamen a subrutinas que se ejecutan en un sistema remoto, incluyendo códigos de retorno y variables predefinidas para soportar el procesamiento distribuido.



**Proceso de negocio:** Ordenación lógicamente interrelacionada de tareas desarrolladas en tiempo y espacio (con comienzo y fin, con entradas y salidas definidas) y que se orienta al logro de un objetivo de negocio, generando una salida de valor (total o parcial) para el cliente del proceso.

**Tablero de mando:** Solución web que da respuesta a la gestión de monitoreo de los indicadores claves de desempeño generados en los procesos de negocio.

**XHTML (*Extensible HyperText Markup Language*):** es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web. XHTML es la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML.

**XML (*Extensible Markup Language*):** Metalenguaje extensible de etiquetas desarrollado por el *World Wide Web Consortium* (W3C). Es una manera de definir lenguajes para diferentes necesidades. XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

**XMLHttpRequest:** es un conjunto de APIs que pueden ser usadas por JavaScript y otros lenguajes para transferir XML u otra información textual hacia o desde un servidor web usando HTTP, mediante el establecimiento de un canal de comunicación independiente entre páginas web del lado del cliente y del lado del servidor. El mayor avance de XMLHTTP es la habilidad de actualizar dinámicamente una página web sin recargar la página entera.