Universidad Central de Venezuela

Facultad de Ciencias

Escuela de Computación

Centro de Investigación en Sistemas de Información

Sistema para el Control y Seguimiento de la Implementación de la Metodología de Gestión de Procesos de Negocio sustentada en el Uso de Patrones

Caso de Estudio: Compañía Anónima Nacional de Teléfonos de Venezuela (CANTV)

Tesis de Pregrado Presentada ante la Ilustre

Universidad Central de Venezuela

Por los bachilleres

Deiby Gabriel González Hernández

Mayerling Desiree Mendoza Infante

Como requisito parcial para optar al título de

LICENCIADO EN CIENCIAS DE LA COMPUTACIÓN

MENCIÓN SISTEMA DE INFORMACIÓN

Tutor

Dr. Pedro Bonillo

Febrero,2011

Dedicatoria

Nuestra tesis, la dedico con todo mi amor y cariño:

A ti mi Dios todopoderoso, que nos distes la oportunidad de vivir y regalarnos una familia bonita y me permitiste llevar en mi vientre a un bebé precioso, que hoy día, es mi mayor tesoro. Por él seré una gran profesional, para apoyarlo en todo lo que necesite y darle mucho amor. A mi hijo amado Gabriel Alejandro.

Con mucho cariño para mis padres, en especial mi madre, **Deisy Infante**, la cual siempre ha sido mi guía, mi inspiración, mi amiga, siempre madre, por apoyarme en todas las etapas de mi vida, brindándome su amor, por ser como eres te dedico esta tesis. Gracias papá y mamá por haberme dado la vida, por todo le agradezco de corazón, los quiero muchísimo.

A mi esposo por ser un hombre integro, por amarme como lo hace, por ser mí amigo, mí confidente, mi amante, por su apoyo, en fin por ser quien eres. Te Amo.

A mi hermana querida, por apoyarme en todo momento, por ser mi amiga incondicional, gracias por existir, te quiero mucho.

A mi hermano por ese gran amor que sientes por tus hermanas.

A mi abuela, a mi tía y primos gracias por estar en todos los momentos que he necesitados de ustedes, por ese amor que nos tenemos como familia.

A Alfredo quien ha sido como un padre para mí, por ser mi amigo, por creer en mí y porque junto a mi madre me llevaron a lo que soy hoy día.

A Adrián quien es un niño muy lindo, quien me quiere muchísimo, a ti mi niño, te dedico este nuevo paso de mi vida.

A mi suegra quien se ha portado muy bien conmigo, por ser amiga te dedico esta tesis.

A mis amigos Sergio y Lorena quienes me han apoyado en todo momento a lo largo de la carrera, ya que luchamos juntos con las materias para llegar a este momento tan preciado para nosotros.

A todos mis amigos les dedico esta tesis, porque hemos compartidos momentos bellísimos en cada etapa de nuestras vidas.

A los profesores quienes lucharon con nosotros para que entendiéramos sus clases. En especial, a ti Antonio Silva que desde que nos conocimos fuiste y seguirás siendo un amigo.

Gracias a todos…

Mayerling Mendoza

Dedicatoria

Primero que nada Gracias a Dios por haberme dado, y aún me da, la oportunidad de existir en este mundo. GRACIAS por regalarme el entorno que me rodea, mi familia, mis compañeros y amigos. GRACIAS por las oportunidades que surgen día a día para aprender y desarrollarme. En fin, Gracias por todo lo que me das y por lo que no me haces llegar.

A mis padres, Williams Rafael y Rosa del Valle, por haberme concebido y permitirme que conociera este mundo. Por haberme inculcado valores desde que era niño. Por crear en mí, conciencia para pensar y haberme llenado de confianza para superar todas las etapas de mi vida. A ti Rosa del Valle por invertir en mí, todo el tiempo que fuese necesario, por tus regaños que me ayudaron, por tus enseñanzas, por tus sentimientos, en fin madre todo lo que escriba aquí se queda pequeño ante todo lo que te tengo que agradecer. A ti Williams Rafael por ser mi papá, por llenarme de confianza, por acompañarme a todos mis juegos, por creer en mí, por brindarme todo el apoyo que aún me continúas dando.

A mi único hermano, José María, quien se crió conmigo, quién me ayuda en las buenas y en las malas. Por ser mí amigo, mi compañero, mi guía. En definitiva, te quiero mucho hermano.

A mi esposa e hijo Gabriel Alejandro, quienes inflan, hoy en día, mis ganas de vivir, mis ganas de crecer como profesional y persona. Por quienes estaré luchando hasta que mi señor Dios me lo permita. A ustedes los amo.

A todos mis compañeros, amigos, entrenadores y técnicos del deporte. Por dejarme sobresalir y hacer que mi autoestima se incentivara para sentirme capaz de hacer las cosas.

A todos los maestros y profesores que invirtieron tiempo en mí. Muchas gracias por sus conocimientos transmitidos, espero poner en práctica todas las enseñanzas que me dejaron en cada una de las etapas de mi educación.

Deiby González

Agradecimiento

Agradecemos de corazón a nuestros padres, quien con sacrificios y esperanzas han logrado hacer de nosotros un Hombre y una Mujer. Rosa, Williams, Deisy y Alfredo los amamos, este triunfo es realmente de ustedes, se lo agradeceremos toda la vida.

Agradecemos a todos nuestros Docentes y Formadores. En especial a los Profesores Pedro Bonillo, por su apoyo y Fe y muy especialmente al Profesor Antonio Silva, por ayudarnos en todo momento en nuestra carrera.

Agradecemos a nuestro tutor por su dedicación, empeño y amistad en la realización de este trabajo, Gracias Profesor Pedro Bonillo.

A todos los que nos apoyaron en este recorrido para cumplir con esta meta.

Deiby González y Mayerling Mendoza

RESUMEN

En la compañías que desarrollan sofware, bajo el paradigma Gestión de Proceso de Negocio, es necesario un sistema que permita el seguimiento y control de los métodos, herramientas y técnicas usados en el Proceso de Desarrollo; a fin de lograr la reutilización en otros proyectos de los componentes, a través, de su adaptación y la asociación de un modelo de calidad, permitiendo la trazabilidad de cada proceso. Este Trabajo Especial de Grado busca implementar un Sistema que permita verificar y asumir las hipótesis mencionadas anteriormente. Una solución capaz de llevar de manera metódica y sistemática el proceso de desarrollo de software, para que con ello se pueda garantizar la calidad del producto final en cada una de sus fases a través de la reutilización de experiencias exitosas y un modelo de calidad establecido.

Como base teórica a la situación planteada, se tomó en cuenta una Metodología de Gestión de Proceso de Negocio sustentada en el uso de Patrones propuesta por el Dr. Pedro Bonillo y para la plataforma de desarrollo se seleccionó Intalio BPMS 6.0 y Netbeans IDE con el framework JSF (Java Server Faces).

El uso de estas herramientas permitió modelar, diseñar y colocar en ejecución el proceso de negocio en una plataforma que permite gestionarlo y adaptarlo a cualquier organización en base a las mejores prácticas.

Palabras Claves: Proceso, Patrones, BPM, Metodología, usabilidad, reutilizar, Servicos Web.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Actualmente, el manejo de la información en las empresas, se ha convertido en un factor de éxito en el desempeño de sus funciones. La Tecnología de Información cada día adquiere mayor importancia debido a que las personas encargadas de tomar decisiones importantes, para el impacto a nivel empresarial, necesitan contar con un verdadero apoyo a nivel de datos para que tengan mayor probabilidad de ser exitosas tanto las decisiones como las empresas como tal.

Según la vigésima segunda edición de diccionario de la Real Academia Española un negocio es "aquello que es objeto o materia de una ocupación lucrativa o de interés", lo que indica que se refiere al desarrollo de una actividad que genera cierto tipo de ambición monetaria para algún ente involucrado.

De todo lo anterior, se entiende por proceso de negocio al conjunto metódico de actividades que buscan de lograr un objetivo común para obtener cierto grado de ganancia, ya sea económica o de cierto status, en el ramo en la cual se ejecuta.

Como todo proceso, un proceso de negocio necesita ser monitoreado constantemente para chequear si se está ejecutando de la manera más eficiente. Todo esto está basado en una gran cantidad de investigaciones en la que se llega a conclusiones que indican que los procesos empresariales o de negocio requieren ser gestionados de forma eficaz para obtener una optimización del rendimiento y control en las empresas.

Por otro lado, los patrones en su sentido más genérico establecen el uso de una solución en una situación específica y, en la cual, se ha comprobado ser exitosa y de dar un desempeño adecuado. Por esta razón, es importante el uso de soluciones que garanticen, con un alto grado de probabilidad, el éxito al momento de diseñar una estrategia o proceso de negocio.

Es necesario destacar que la Gestión de Procesos de Negocio se realiza con el fin de tomar decisiones efectivas que beneficien las estrategias de una empresa, ya que los administradores de negocio requieren entrar a las operaciones de la compañía para realizar un análisis y planear el futuro de la misma, por lo que se requiere tener una buena perspectiva para generar la correcta planeación y administración de recursos.

En la actualidad existen diversos paquetes de software propietario, y también software libre, que permiten modelar, diseñar, simular y monitorear los procesos de negocio de una determinada empresa u organización.

Al momento de seleccionar la herramienta para modelar el proceso de *Metodología de Gerencia de Procesos sustentada en el uso de Patrones*, se realizó una evaluación exhaustiva de los diversos sistemas de gestión de procesos de negocios. La solución seleccionada para su modelado fue Intalio BPMS 6.0.1, ya que este satisface todas las características deseables en una herramienta de gestión de procesos de negocios (uso de licencia libre, diseñador BPMN, servidor BPEL, manejo de servicios web, soporte de diversos sistemas operativos y bases de datos, ambientes colaborativos en línea, desarrollo cero código, basado en código java.

Durante el desarrollo del proceso, surgió la necesidad de generar un conjunto de interfaces relacionadas con el look & feel de la aplicación final, a fin de mostrarle al usuario un conjunto de datos que iban a ser procesados a través de los diagramas de procesos modelados, con el BPMS Intalio. Para diseñar el conjunto de interfaces se seleccionó la herramienta Netbeans IDE 6.7, la cual tiene un uso de licencia software libre, en la que las aplicaciones pueden ser desarrolladas a partir de un conjunto de módulos que pueden ser creados independientemente uno de otros) y se utilizó un framework de nombre Java Server Faces – JSF, el cual es un marco de trabajo de desarrollo ágil que permite la división en capas para implementar, de manera independiente, los componentes web y los de negocio.). Finalmente, para el almacenamiento de la data se seleccionó el manejador de base de datos MySQL 5.1 (software libre, es una base de datos relacional, multihilo y multiusuario).

El propósito de este Trabajo Especial de Grado es presentar ante la Escuela de Computación de la Facultad de Ciencias de la Universidad Central de Venezuela, un Sistema para el Control y Seguimiento de la Implementación de la Metodología de Gestión de Proceso de Negocio sustentada en el Uso de Patrones, propuesta por el Dr. Pedro Bonillo en 2008, y aplicar algunas de sus prácticas en un caso de estudio en la Compañía Anónima Nacional de Teléfonos de Venezuela (CANTV), a fin de mejorar las condiciones de trabajo, reducir esfuerzos y tiempo de producción.

Con tal finalidad, este documento está estructurado en cinco capítulos con el propósito de cumplir con los objetivos planteados.

Capítulo I, El Problema, se inicia la contextualización en espacio y tiempo del problema de investigación, sus manifestaciones y evidencias, luego se presentan los objetivos, la justificación y delimitación.

El Capítulo II se refiere al Marco Teórico, donde se entrega al lector los conocimientos necesarios para el desarrollo del trabajo. Aquí se especifican las definiciones, características, procesos y relaciones que dan sustento a las bases teóricas de la metodología planteada según Bonillo 2008, incluyendo las características para la evaluación de la herramienta de software libre Intalio.

El Capítulo III comprende el Marco Metodológico, en el cual se detallan las unidades de análisis y de investigación; así como los métodos, técnicas y procedimientos de observación y recolección de datos necesarios para llevar a cabo la presente investigación. Además se indica la metodología de desarrollo de software utilizada al momento del modelado.

A continuación, el capítulo IV muestra el ambiente para la implementación del Sistema de Control y Seguimiento para la aplicación de la Metodología de Gestión de Procesos de Negocio sustentada en el uso de Patrones, así como algunos de los resultados de la investigación.

En el Capítulo V, se expone el marco aplicativo de la solución dada para la aplicación.

Para finalizar, se realizarán las conclusiones y recomendaciones.

A continuación se inicia con el planteamiento del objeto de estudio o contexto.

CAPÍTULO I – EL PROBLEMA

En esta sección del Trabajo Especial de Grado se detalla toda la información referente al contexto, delimitación y justificación del problema, así como el objetivo general y los objetivos específicos que determinan el alcance del mismo.

1.1 Contexto

La nueva economía, denominada "digital" e inteligencia en red por Tapscott (1998), se mueve en un mundo cada vez más volátil, global y competitivo. En las últimas décadas se puede apreciar cómo han surgido distintos modos gerenciales para afrontar los problemas de las organizaciones en función de sus entornos (interno/externo), pero fundamentalmente en pro de mejorar el manejo de la información, la cual cada día es, en todas sus formas, más gestionada a través de Internet y en las propias redes internas de las organizaciones.

Las empresas actuales, requieren de modelos de negocios complejos con una estructura organizacional, procesos y sistemas que deben ser diseñados explícitamente, incluyendo:

- Diseño del modelo de negocio: estructura del negocio y diseño del producto y el valor agregado –en relación a soluciones tradicionales- que éste aportará, estrategia de captura de mercado y distribución.
- Diseño del sitio Web a través del cual se canalizará la oferta.
- Diseño de los procesos de negocios -procesamiento de pedidos, generación del producto o servicio, logística, etc. –que implementen la oferta, a través de una cierta estructura y funcionamiento organizacional.
- Diseño de los sistemas computacionales que manejen las transacciones que se capturan del sitio, mantienen las bases de datos (usuarios, productos, contable/financieras) necesarias para procesar eficientemente tales transacciones y automatizar/apoyar con información apropiada los procesos mencionados en el punto tres (Barros, 2002).

El trabajo de diseñar estos modelos de negocio es claramente interdisciplinario, ya que requiere conocimientos de: desarrollo de negocios, los diferentes procesos que ocurren en la empresa, las Tecnologías de la Información y la Comunicación (TIC), diseño organizacional, manejo de recursos humanos, habilidades de innovación, entre otros. Por lo tanto, sería difícil que una sola persona pudiera tener todo el conocimiento y formación para hacer tal trabajo.

Esto genera la necesidad de trabajar con equipos de diseño en los cuales se encuentren los talentos enunciados. Sin embargo, los profesionales que puedan facilitar el trabajo de un equipo de este tipo deben tener conocimientos profundos de una metodología de diseño que oriente todo el trabajo y conocimientos apropiados en todos los otros temas que son relevantes. Lo anterior confirma la necesidad y conveniencia de realizar diseños explícitos de los negocios en base a los procesos, sus prácticas de gestión y a las TIC disponibles, es decir realizar una Ingeniería de Negocio (Barros, 2002).

Los cambios producidos en las organizaciones para lograr esta adaptación han sido acciones continuas y rápidas, a través del desarrollo de nuevas interrelaciones entre las TIC's, con base en los casos de éxito surgidos en cada uno de los dominios específicos. A partir de la década de los ochenta se presenta en las Ciencias de la Computación un cambio de paradigma hacia la computación centrada en red, surgiendo un conjunto de técnicas, en la Ingeniería de Software, que enfocan al software como un producto de ingeniería que debe definir, crear y aplicar: (1) una metodología bien definida dirigida a un ciclo de vida de planeamiento, desarrollo, y mantenimiento; (2) un conjunto establecido de componentes de software que documenta cada paso en el ciclo de vida y muestra un seguimiento, y (3) un conjunto de hitos predecibles que pueden ser revisados a intervalos regulares a través del ciclo de vida del software. (Pressman, 1997).

La Ingeniería de Software ha evolucionado en todas sus áreas: análisis de requisitos, procesos de desarrollo de software, interacción humano-humano asistida por el computador, estrategias de implementación, modelos de costos, entre otros. Sin embargo, aún está por debajo de las necesidades de calidad demandadas por las organizaciones y los sistemas cada vez más complejos; además, estos esfuerzos han surgido de forma independiente y no se integran (en muchos casos se oponen y se complementan a la vez,

pretendiendo crear el software desde varias perspectivas al mismo tiempo), se hace necesario un marco referencial que las una y que facilite el seguimiento y la evaluación necesaria para garantizar su eficacia, desde una perspectiva integral.

En este contexto, existen considerables esfuerzos de investigación y desarrollo con el objetivo de perfeccionar el proceso de producción de software unificado, tanto a través de estudios teóricos, como de estudios aplicados.

La construcción de aplicaciones ha sido siempre una tarea compleja, pero esta complejidad, lejos de reducirse, es cada vez mayor: la aparición de nuevos y sofisticados dispositivos periféricos, la computación sin cable, los grupos de usuarios cada vez más heterogéneos y exigentes, la velocidad con que se deben construir los productos debido a la competencia y exigencias del mercado; son todos factores que conllevan a la necesidad de definir nuevos métodos o adaptaciones de métodos existentes (reflexión sobre los métodos hasta lograr una metodología), a fin de abordar de manera integrada, clara y precisa la construcción de software centrado en los usuarios.

En muchos casos la complejidad tecnológica provoca que el ingeniero de software dedique un mayor esfuerzo a conocer aspectos técnicos, que a entender el problema que debe resolver el software a desarrollar. Una de las tareas críticas en la Ingeniería del Software y que tiene una mayor repercusión sobre la productividad y la calidad final del software es el modelado conceptual.

Estudios relacionados con la construcción de software en ambientes Usuario/Servidor y Orientados a Objetos sostienen la tesis de que las tareas más críticas siguen estando en la especificación y en el análisis de requisitos. Los errores introducidos en esta etapa del proceso de producción de software pueden tener un impacto considerable en la fiabilidad, el costo y la seguridad del sistema. (Toval, 2005).

La falta de rigor que presentan los métodos y entornos actuales en la definición de los elementos de modelado dificultan la obtención de software que sea funcionalmente equivalente a la descripción capturada en el esquema conceptual. (Insfrán, 2002).

Si se analizan los métodos de desarrollo más utilizados en la industria del software, se puede observar que muchas veces el esfuerzo invertido en la realización de algunas

tareas (como puede ser el modelado conceptual) y en la producción de documentación, no tiene ningún reflejo directo en el producto final.

Estos problemas invitan a replantearse la forma de abordar los procesos de desarrollo de software. Estos procesos necesitan métodos y herramientas que: (1) aborden el proceso de desarrollo con rigor y de forma sistémica, proporcionando los modelos y las etapas exclusivamente necesarias para la producción de software; (2) se centren en la especificación precisa del espacio del problema; (3) ayuden al analista a tomar decisiones en el proceso de modelado conceptual; (4) faciliten la generación asistida de código tanto de estructura como de comportamiento a partir de esquemas conceptuales; (5) ofrezcan independencia tecnológica, ocultando la tecnología subyacente a los desarrolladores; y (6) produzcan software de calidad funcionalmente equivalente a su especificación. (Ridao, 2001).

Uno de los esfuerzos asociados al tratamiento de estos problemas es presentado en agosto de 2002, cuando el OMG (*Object Management Group*) hizo pública la iniciativa MDA (*Model Driven Architecture*), una variante de una nueva tendencia extendida por todo el mundo denominada Ingeniería de Modelos. Las ideas básicas de la ingeniería de modelos están relacionadas con otras propuestas como la programación generativa, lenguajes para dominios concretos, computación integrada con modelos, factorías de software, entre otros.

MDA puede definirse como la consecución de los principios de la Ingeniería de Modelos haciendo uso de un conjunto de estándares del OMG como MOF (*Meta Object Facility)*, XMI (XML *Metadata Interchange*), OCL (*Object Constraint Languaje*), UML (*Unified Modeling Languaje*), CWM (*Commond Warehouse Metamodel*), SPEM (*Software Process Engineering Metamodel*), entre otros.

De un modo similar a como el principio básico “todo es un objeto” resultó fundamental para establecer la tecnología orientada a objetos durante la década de los 80, MDA en la actualidad se concentra en los modelos, sin embargo, no ha evolucionado lo suficiente, a fin de demostrar su utilidad en la práctica.

Los métodos de desarrollo han proliferado proporcionando lenguajes para describir esquemas conceptuales y diseños; además de esto, deberían proporcionar diseños prácticos

y efectivos. Una alternativa para esto es la utilización de los patrones de software, los patrones de software han surgido y evolucionado a partir de varias iniciativas: la de Kent Beck y Ward Cunningham, dos de los pioneros de Smalltalk, también a partir de las ideas de Christopher Alexander, que desarrolló una teoría y una colección de patrones para el mundo de la arquitectura urbanística (Alexander et al., 1977), y de las ideas de Peter Coad (Coad et al., 1999) en las cuales presenta distintos patrones de análisis y diseño como una serie de relaciones entre elementos (dos o tres) de bajo nivel (clases).

Bruce Anderson lideró talleres en OOPSLA (*Object-Oriented Programming, Systems, Languages & Applications*) a principios de los 90 en los cuales se investigó y se desarrolló un libro sobre arquitecturas software. Jim Coplien en su libro (Coplien et al., 1995) describió programas útiles en C++. Basados en la mayoría de estos trabajos se constituyó el grupo de Hillside para investigar estas ideas en un futuro. El empuje más grande para su conocimiento público fue la publicación del libro Design Patterns, Elements of Reusable Object-Oriented Software (Gamma et al., 1995) de la banda o grupo de los cuatro "Gand or Group of Four" (GoF), y la conferencia PLoP (*Pattern Language of Programming*) que inició en 1994 el grupo de Hillside.

Todavía existen discrepancias entre los investigadores a la hora de definir qué es un patrón, por lo que es bastante difícil encontrar definiciones de patrón que sean idénticas. En el libro de Fowler (1997) se encuentra una definición genérica interesante: "Un patrón es una idea que ha sido utilizada en un contexto práctico y que probablemente será útil en otros". El término idea expresa que un patrón puede ser cualquier "cosa". La expresión contexto práctico refleja el hecho de que se desarrollan (algunos autores prefieren: descubren) gracias a la experiencia práctica en proyectos reales.

Por otro lado, los patrones de software no son más que un conjunto de soluciones a problemas habituales en el diseño de software orientado a objetos. Una definición más formal podría ser: "Un patrón es una solución a un problema, aceptada como correcta, a la que se ha dado un nombre y que puede ser aplicada en otros contextos".

El concepto de patrón en la Ingeniería de Software, desde sus inicios fue planteado de forma general para todas las disciplinas, sin embargo surgieron como patrones de diseño (patrones de un nivel de abstracción menor y están por lo tanto más próximos a lo que sería

el código fuente final), pero actualmente los patrones constituyen un concepto más general, representando estructuras conceptuales aplicables durante las fases de análisis, diseño, construcción de la interfaz de usuario e implantación.

La aplicación de patrones a los métodos de desarrollo de software aporta beneficios interesantes ya que, entre otras muchas aplicaciones, permiten: (1) definir guías de modelado mediante lenguajes de patrones (Coplien et al., 1995); (2) estructurar el proceso de generación asistida de código, y; (3) ofrecer soluciones reutilizables y probadas, que sean lo suficientemente abstractas cómo para ser utilizadas en cualquier lenguaje de programación. (García, 2003).

La utilización de patrones en las nuevas tecnologías está evolucionando a pasos agigantados gracias a los estándares y las aplicaciones surgidas en estos últimos años. Aunque la contribución de las aplicaciones tradicionales de flujo de trabajo, "ad hoc", administrativos y colaborativos, es aún notable, hay una nueva generación de aplicaciones que son un híbrido que reúne lo mejor de todos los sistemas de flujo de trabajo y otras tecnologías basadas en patrones, estos son los sistemas de gerencia de procesos de negocio (BPMS).

Actualmente muchas empresas se están orientando hacia los procesos, destacándose principalmente por el e-Business, esta nueva generación de tecnología BPMS (*Business Process Management System*) está siendo actualmente más utilizada y perfeccionada. "La gerencia de los procesos del negocio se refiere a diseñar, ejecutar y optimizar los procesos funcionales del negocio a través de toda la organización incorporando los sistemas, a los procesos y a la gente" (Vollmer, 2004).

BPM es la "integración caracterizada por flujo de trabajo orquestado, orientado a aplicaciones a través de usos internos múltiples y/o entre los socios externos" (Harris, 2003). La puesta en práctica de las soluciones de BPM ha emergido como estrategia dominante, que las organizaciones están adoptando para mejorar la eficacia de sus procesos del negocio.

BPM y MDA utilizan modelos formales, se esfuerzan para conseguir la máxima independencia computacional de la plataforma y están basados en estándares. BPM realiza transformaciones entre el lenguaje de modelado BPMN y los lenguajes BPEL y BPML. Se

utiliza BPM y MDA a fin de relacionarlos con MDE, pero no se usan formalmente los patrones, ni se mide su calidad.

BPM es una arquitectura dirigida por diseños (DDA) y no una arquitectura dirigida por modelos (MDA). En BPM, no hay un paso requerido entre el proceso de diseño y el de despliegue. Una alternativa, es la reusabilidad por extensión o adaptación de las mejores prácticas en los distintos dominios: el uso de patrones de software. (Pérez et. Al, 2007).

Los BPMS son capaces de apoyar este tipo de actividad debido a su capacidad de coordinar interacciones entre: (1) los sistemas de información; (2) los procesos del negocio, y; (3) la gente que los utiliza. Esto da visibilidad de las empresas en el estado de los procesos y tiende a permitir cambios de los procesos sobre una base en uso (Bonillo 2008).

Los esfuerzos han tendido a la integración temprana de aplicaciones empresariales (IAE) y la tecnología de la Extracción, Transformación y Carga (ETC o ETL por sus siglas en inglés *Extraction-Transformation-Load*) que permite la generación de reportes operacionales y multidimensionales sin afectar el desempeño de las aplicaciones, que han sido fortalecidas con nuevas características que proporcionan la integración de la información de la empresa (EII), el flujo de trabajo, la gerencia de los procesos del negocio (GPN o BPM por sus siglas en inglés), la integración a través de un modelo de objetos único para todos los sistemas (canónico), la utilización de portales y la ubicuidad por medio de los dispositivos móviles.

Sin embargo los BPMS, adolecen de un marco referencial integral, que indique como lograr su implementación, a través de un proceso de desarrollo de software, tomando en cuenta los patrones, de tal forma de gestionar el conocimiento que se produce a fin de medir la calidad del proceso y del producto, en este trabajo se presenta una propuesta metodológica para la gerencia de los procesos de negocio sustentada en el uso de patrones.

En (Bonillo, 2008) se propone una Metodología de Gestión de Procesos de Negocios sustentada en el uso de Patrones, en ella se toman aspectos importantes para ayudar al desarrollador a tomar decisiones que afectan la calidad del software. En su trabajo, se propone la implementación de Patrones a través de una taxonomía que se va recorriendo a medida que el producto se va desarrollando, validando paso a paso que se

vayan tomando las “mejores decisiones” de acuerdo a soluciones exitosas previamente certificadas a través de un patrón.

1.2 Formulación del Problema

1.2.1 El Problema

En base a lo anterior surge las siguientes interrogantes:

Si se consigue capturar y representar patrones de forma no ambigua, ¿Se podrían construir procesos de desarrollo asistidos que traduzcan correctamente los patrones modelados al espacio de la solución?,

¿Cómo verificar, de manera estática y asequible, hasta dónde es posible asegurar con el conocimiento disponible, que al combinar ciertos patrones no se han violado las condiciones de funcionamiento correcto de ninguno de ellos, garantizando su eficacia en el dominio BPM? , y

¿En el dominio de la Gerencia de los Procesos de Negocio (*Business Process Management BPM*), es posible desarrollar una aplicación que automatice la metodología de gestión de procesos sustentada en el uso de patrones y permita garantizar que se siguen cumpliendo con los procedimientos estandarizados?

1.3 Objetivos

Seguidamente se presenta el objetivo general y los objetivos específicos de esta investigación.

1.3.1 Objetivo General

Desarrollar un Sistema para el Control y Seguimiento de la Implementación de la Metodología de Gestión de Procesos de Negocio sustentada en el Uso de Patrones.

1.3.2 Objetivos Específicos

- Estudiar el entorno de la Gestión de Procesos de Negocio para el Desarrollo de Software.
- Analizar la Metodología de Gestión de Proceso de Negocio sustentada en el Uso de Patrones.
- Definir los requerimientos funcionales y no funcionales del proceso de negocio a analizar y modelar.
- Integrar la solución generada, con un Administrador de Patrones diseñado en años anteriores.
- Analizar las herramientas Open Source de Modelado de Procesos de Negocio.
- Desarrollar el prototipo que permita el uso de la METODOLOGÍA en cuanto a la reducción de los esfuerzos en el proceso de desarrollo de software usando una notación gráfica basada en estándares internacionales.
- Configurar el motor de ejecución para ejecutar el proceso Metodología de Gestión de Proceso sustentada en el uso de Patrones.
- Realizar pruebas funcionales para verificar el correcto funcionamiento de la solución.

1.4 Justificación

En la actualidad, grandes, medianas y pequeñas empresas o corporaciones se encuentran en búsqueda de mejorar el rendimiento de su compañía. Para ello deben contar, de acuerdo a investigaciones realizadas por expertos, con un manejo eficiente de la información que pueda afectar positivamente el futuro de la organización y poder tomar las decisiones más adecuadas con respecto al campo o ámbito en donde se desenvuelven.

Estas organizaciones se están enfocando en adoptar el uso de herramientas que les permitan incrementar o establecer un mayor rendimiento, utilizando como base nuevos enfoques gerenciales, a fin de alcanzar el éxito a corto, mediano y largo plazo, con el

propósito de establecerse metas que permitan el alcance de los Planes Estratégicos del Negocio, enfocados al cumplimiento de la Visión, Misión, Valores, entre otros, en donde se compromete cada rol relacionado dentro de la empresa que juega un papel de importancia para el cumplimiento de cada objetivo.

Esta investigación se justifica desde tres puntos de vista. El primero es el práctico, ya que se propone al problema planteado una estrategia de acción que al aplicarse arrojará datos que servirán para medir el desempeño de la Metodología de Gestión de Procesos de Negocios sustentada en el Uso de Patrones, propuesta por Bonillo (2008). Aunado a lo anterior, se garantiza la factibilidad y viabilidad del proyecto en el sentido de que en un futuro cercano será usado por usuarios que deseen estudiar, investigar y desarrollar software usando la metodología planteada.

Desde el punto de vista teórico, esta investigación generará reflexión y discusión tanto del conocimiento existente del área investigada, como dentro del ámbito de las Tecnologías de Información, ya que de alguna manera u otra, se confrontan teorías (en nuestro caso se analizan dos cuerpos teóricos dentro de las Tecnologías de Información, ellos son:

- Gestión de Proceso del Negocio (Sistemas de Información).
- Patrones de Ingeniería de Software (Interacción Humano Computador).

Por todo lo anterior, es necesario hacer epistemología del conocimiento existente.

Desde el punto de vista metodológico, esta investigación ocasionará la aplicación de un nuevo método de investigación para generar conocimiento válido y confiable dentro del área de la Gerencia de Proceso de Negocio, ya que permite visualizar y analizar cada paso de la metodología.

1.5 Delimitaciones

Este estudio está circunscrito al área de Gerencia de Procesos de Negocios y al Área de Sistemas de Información, dentro del dominio tecnológico existente en las ciencias

computacionales para el año 2010: Software Libre, Ambientes de Soporte de Decisiones y Modelización de Procesos de Negocio.

1.6 Limitaciones

Para la realización de este trabajo se contó con un período de tiempo corto, ya que solo se contó con seis (6) meses para el desarrollo del producto, donde la tecnología ya había sido seleccionada. Por ende, las primeras semanas fueron cruciales porque se necesitaba instrucción sobre las herramientas a usar para el desarrollo del Sistema de Control y Seguimiento de la Implementación de la Metodología de Gestión de Procesos de Negocio sustentada en el uso de Patrones.

1.7 Alcance

En este Trabajo Especial de Grado se llegará hasta la automatización del módulo crear proceso de la Metodología de Gestión de Procesos de Negocio sustentada en el uso de Patrones, bajo una arquitectura orientada a servicios, por tanto, no se contempla el uso de la disciplina BAM (*Business Activity Monitoring*, para el desarrollo del módulo administrar patrones .

CAPÍTULO II – MARCO TEÓRICO

2.1 Antecedentes de la Investigación

Para iniciar el marco teórico resulta conveniente citar algunos estudios preexistentes vinculados al tema de investigación:

- Fabio A. Zorzan y Daniel Riesco publicaron un artículo titulado Automatización de Procesos de Desarrollo de Software definidos con Spem. Este trabajo propone "*utilizar motores workflow que son utilizados para automatizar procesos de negocio. Para lograr esta Automatización se deberá definir una transformación del metamodelo SPEM al metamodelo de Business Process Modeling Notation (BPMN) definido por la Object Management Group (OMG) por medio del lenguaje Relations que forma parte de Query/Views/Transformations (QVT). La especificación BPMN resultante podrá ser transformada a un lenguaje estándar para la implementación de procesos workflow, como ser Business Process Execution Language for Web Services (BPEL4WS) o XML Process Definition Languaje (XPDL). Con esto se lograría fundamentalmente la automatización de cualquier proceso de desarrollo de software especificados bajo el SPEM a través de su transformación a proceso workflow estándar".*
- Marcelo Mejía y León Arzarte publicó un artículo titulado Automatización de Procesos de Negocio utilizando un BPMS. Este trabajo "*describe el uso de un BPMS para automatizar la administración de los procesos de negocio de una empresa, teniendo capacidades de rapidez y eficacia en respuesta a cambios. Se presenta la arquitectura de los BPMS, sus principales características, las ventajas que ofrece, y un caso práctico que ejemplifica su uso".*

- Walter Itamar Mourao publicó un artículo titulado Evaluación del Uso de Herramientas para Procesos de Flujo de Trabajo Típico en Ingeniería de Software *(Avaliação do uso de ferramentas de workflow em processos típicos de engenharia de software).* Este trabajo *"describe una selección de una herramienta de workflow y una experiencia de su uso en un proceso de caso de uso hacer consideraciones acerca de las posibles mejoras en el proceso que se pueden aplicar con este tipo de herramienta."*
- Andrea Delgado publicó un artículo titulado Desarrollo de Software con enfoque en el Negocio. Este trabajo *"describe como cambiar las visiones de cómo se relacionan el negocio con la información".*
- Pedro Nolasco Bonillo, en 2008, publicó un trabajo titulado Metodología para la Gerencia de Procesos de negocio sustentada en el Uso de Patrones. Durante el desarrollo de este trabajo se enuncia que "*las empresas actuales requieren de modelos de negocios complejos con una estructura organizacional, procesos y sistemas que deben ser diseñados explícitamente. El trabajo de diseñar estos modelos de negocio es claramente interdisciplinario, ya que requiere conocimientos de desarrollo del negocio, los diferentes procesos que ocurren en la empresa y de la gerencia de los procesos y las aplicaciones tecnológicas. En el ámbito de la ingeniería de software sería conveniente poder contar con un sistema de métodos, herramientas y técnicas que permitan reutilizar las mejores prácticas durante el proceso de desarrollo de software según cada uno de los procesos que se implementen en cada dominio. En base a esto, en este trabajo se realiza una propuesta de marco teórico referencial integral y una metodología que abarca desde el análisis de los requerimientos hasta el monitoreo de los procesos, apoyando las etapas de análisis, diseño, modelaje y configuración, a través del uso de patrones. La propuesta metodológica está conformada por dos macro-procesos: uno relacionado con la creación del proceso en sí mismo y otro que corresponde a la administración, y comprende: el mantenimiento, administración del proceso en producción y el monitoreo a través de indicadores de gestión".*

2.2 Bases Teóricas

A continuación se presenta los fundamentos para esta investigación.

Tecnología de Información (TI).

El manejo de la información, hoy en día, se ha convertido en un recurso indispensable en la toma de decisiones a nivel empresarial. Esto es debido a que su uso ha reducido o reduce notablemente la probabilidad de fracaso al momento de una decisión trascendental en las operaciones y/o estrategias de una determinada empresa. Su concepto nos dice que se basa en el estudio, diseño, desarrollo, implementación, soporte o dirección de los sistemas de información computarizados, en particular, de software de aplicación y hardware de computadoras.

Todo esto nos indica que el manejo de la información es de vital importancia en las relaciones empresariales, por lo que a continuación se estará tratando términos relacionados con la Tecnología de Información.

Procesos de Negocio y Gerencia de Procesos de Negocio

Según la Vigésima Edición del Diccionario de la Real Academia Española el término **proceso** se refiere al "conjunto de las fases sucesivas de un fenómeno natural o de una operación artificial". Esto indica que se trata de un conjunto de actividades que persiguen un determinado fin y que dependiendo de la rama de operación en la que se ejecuta puede recibir cierto tipo de entradas. En la figura 1, se muestra un ejemplo de algunos tipos de procesos, sus características, propósito y la rama en la que desenvuelven.

Tipos de procesos:	Industriales	de Información	de Negocio
Foco	COSAS	DATOS	RELACIONES
Propósito	Transformar y ensamblar materiales y componentes en otros componentes y productos finales, usando recursos	Procesar y transmitir datos estructurados y no estructurados, y conocimiento	Alcanzar las condiciones que satisfacen las necesidades de los participantes, clientes o usuarios
Caracterísi-cas	Tradiciones de la ingeniería industrial	Tradiciones de la ingeniería informática	Basados en estructuras de comunicación y coordinación humanas encontradas en todos los lenguajes y culturas
Acciones	Ensamblar, Transformar, Transportar, Almacenar, Inspeccionar	Enviar, Invocar, Grabar, Recuperar, Consultar, Clasificar,	Solicitar, Prometer, Ofrecer, Rechazar, Proponer, Cancelar, Medir

Figura 1: Algunos Tipos de procesos

Fuente: Barrera 2007

El tipo de proceso que engloba esta investigación es el denominado **Proceso de Negocio,** término un poco controversial que ha tenido unas cuantas definiciones a través de la historia. A continuación se enuncian algunas de esas definiciones:

- Un Proceso de Negocio es una colección de actividades que, tomando una o varias clases de entradas, crean una salida que tiene valor para un cliente. (Hammer y Champy, 1993)
- Los Procesos de Negocio representan el flujo de trabajo y de información a través del negocio. (BOMSIG, 1995)
- Para Jeffrey N. Lowenthal, "*Un proceso de negocio es un conjunto de tareas relacionadas lógicamente llevadas a cabo para lograr un resultado de negocio definido Son dos las características importantes de los procesos: 1. Tienen clientes (internos o externos), 2. Cruzan fronteras organizacionales, es decir, operan entre subunidades organizacionales.…".*

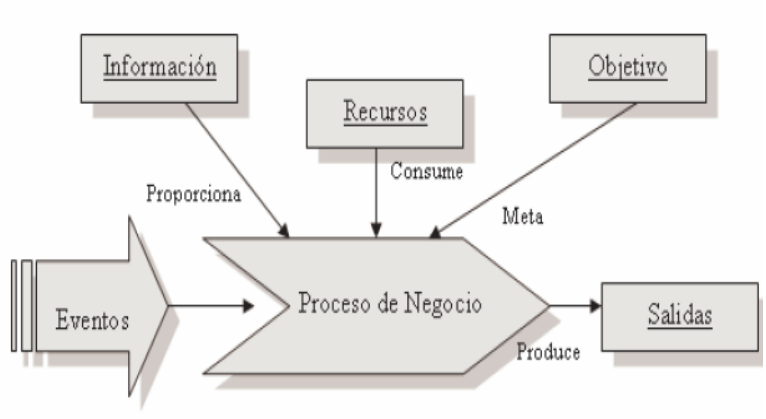


Figura 2: Representación Esquemática de un Proceso de Negocio

Fuente: Barrera 2007

Los procesos de negocios son las formas particulares en que las organizaciones coordinan y organizan las actividades de trabajo, la información y los conocimientos, para producir un bien o servicio valioso. La problemática está en que los procesos de negocio del mundo real se expanden por diferentes organizaciones, departamentos, sistemas y aplicaciones.

Los procesos de negocio tienen las siguientes características:

- Complejos.
- Dinámicos.
- Distribuidos y particularizados.
- Duración prolongada (pueden durar incluso meses o años)
- A veces automatizados, aunque sea parcialmente.
- Dependen de la inteligencia y el juicio de las personas (características que los hacen muy subjetivos).
- Difíciles de visualizar.

La Gerencia de Procesos de Negocio es la traducción del término *Businnes Process Management (BPM)* y se refiere a la disciplina empresarial que tiene como objetivo mejorar la eficiencia a través de la gestión sistemática de los procesos de negocio. Se trata de una estrategia para la gestión y mejora de la ejecución del negocio a través de la continua optimización de los procesos de negocio en un ciclo cerrado de modelado,

ejecución y medición. BPM modela, simula, ejecuta, gestiona, monitoriza y optimiza los procesos de negocio.

Según Giga (Vollmer et al., 2004) "La gerencia de proceso del negocio se refiere a diseñar, ejecutar y optimizar los procesos funcionales del negocio a través de toda la organización incorporando los sistemas, a los procesos y a la gente". Forrester (Harris et al., 2003) definen BPM como "integración caracterizada por flujo de trabajo orquestado, orientado a aplicaciones a través de usos internos múltiples y/o entre los socios externos". Los BPMS consisten en dos elementos: (1) el diseñador de procesos; (2) y el motor de proceso.

BPMS (Business Process Management Systems) son aquellas aplicaciones mediante las cuales se puede implementar una estrategia de Tecnología de Información orientada a la automatización de los procesos claves de las distintas organizaciones que operan en los distintos segmentos del mercado. Es decir, un BPMS es la herramienta en la cual se diseñan, modelan y ejecutan los procesos de negocios.

De esta forma, BPMS es una categoría estratégica dentro del mercado de desarrollo de software, que explota la infraestructura middleware IT, permitiendo a las corporaciones unir sus tecnologías ya implementadas, con las personas y con los procesos que participan en el desarrollo del negocio diario. Esta capacidad de operar automáticamente los procesos de negocios reduce drásticamente los costos relacionados.

Las empresas han convalidado la necesidad de desarrollarse más allá de la optimización de los recursos y del tratamiento transaccional de sistemas, reorientándose a explotar la información y el conocimiento que se refiere a la acción de los recursos internos y externos de la organización.

Un enfoque de automatización de procesos deviene en una reducción de errores, asegurando que los procesos clave se comporten siempre de la misma manera y brindando elementos que permitan visualizar el estado de los mismos.

La administración de los procesos permite asegurar que los mismos estén ejecutándose eficientemente y obtener información que luego puede ser utilizada para mejorarlos. Es a través de la información que se obtiene de la ejecución diaria de los procesos, que se pueden identificar posibles ineficiencias en los mismos y de esta forma

optimizarlos y por consiguiente, mejorar el desempeño corporativo en los distintos departamentos.

Para soportar esta estrategia es necesario contar con un conjunto de herramientas que den el soporte necesario para cumplir con el ciclo de vida de BPM.

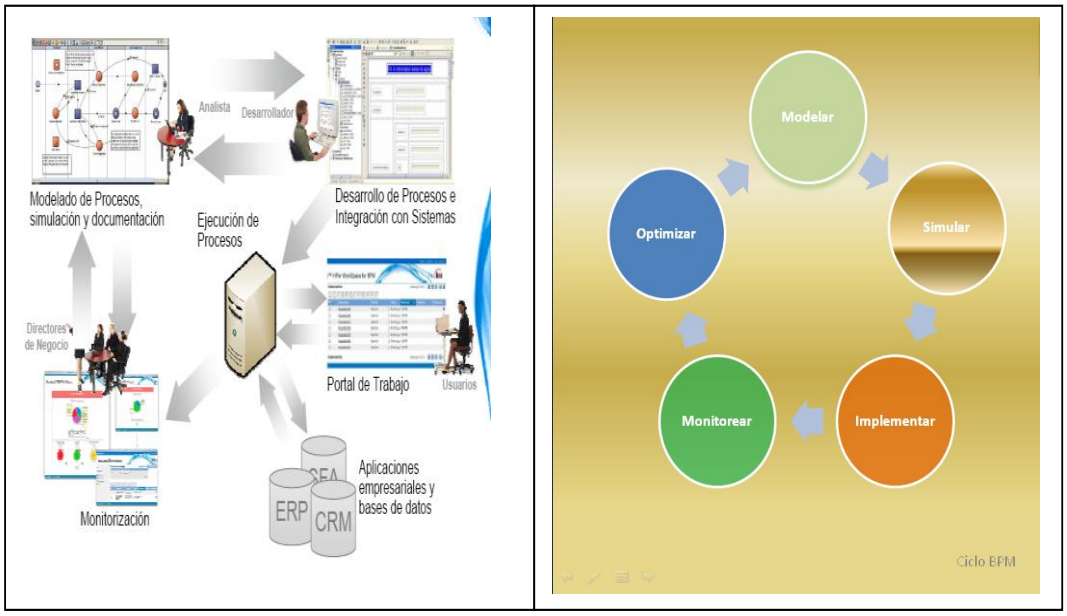


Figura 3: Ciclo de Vida BPM

Fuente: http://www.oracle.com/technology/pub/articles/dev2arch/2006/09/aqualogic-bpm.html

Este conjunto de herramientas son las que corresponden a un BPMS y en base a las mismas se construyen las aplicaciones BPM.

Además de lo ya mencionado, existen diversos motivos que promueven la implementación de una Gerencia de Procesos de Negocio (BPM), dichos motivos son:

- Extensión del programa institucional de calidad.
- Cumplimiento de legislaciones.
- Creación de nuevos y mejores procesos.
- Analizar los procedimientos correctos e incorrectos mediante la comprensión de los procesos.

- Documentación de procesos para outsourcing y definición de SLA (Service Level Agreement)
- Creación y mantenimiento de la cadena de valor.

BPM es la evolución del concepto de gestión y reingeniería de procesos que fue inicialmente desarrollado a principios de los años noventa. Hoy en día BPM representa la convergencia de metodologías gerenciales y prácticas tecnológicas alrededor de la Gerencia por Procesos.

La Gerencia de Procesos de Negocio permite estandarizar el flujo de actividades del negocio, permitiendo la ganancia de puntos en la productividad y la eficiencia. Las soluciones de BPM también sirven para medir, analizar y mejorar la Gerencia de los Procesos de Negocio y análisis financiero de una empresa.

El objetivo de BPM es hacer un seguimiento sistemático de cómo los recursos (físicos, económicos, humanos, tecnológicos, etc.) son asignados a una organización operativa y convertidos en acciones en la búsqueda de los objetivos de la organización, desde la definición de prioridades.

El BPM permite el análisis, definición, ejecución, supervisión y administración de procesos, incluido el apoyo para la interacción entre personas y ordenadores de diferentes aplicaciones. Sobre todo, permite a las reglas de negocio de la organización, disimular en la forma de procesos, y la creación de áreas de gestión por sí mismos, sin injerencia de las áreas técnicas.

Así, por ejemplo, BPM como filosofía y metodología de gestión basada en procesos incluye el desarrollo de mapas de procesos, la identificación de los procesos claves del negocio, el desarrollo de sistemas de medición (*Benchmarking*) y monitoreo basados en procesos, la innovación y mejora continua de los procesos, el uso de técnicas de la calidad total y six-sigma, la aplicación e implantación de tecnologías de información habilitadoras del paradigma de procesos (por ejemplo: ERP, CRM, SCM, BPMs) y el desarrollo de una estructura organizativa basada en roles y equipos capaz de soportar este paradigma de gestión.

Los usos de BPM permiten a las compañías implementar y administrar procesos a través de unidades de negocio tanto internas como externas y sus relaciones.

El desarrollo de servicios Web y los lenguajes de integración basados en XML ha permitido el desarrollo de patrones para los Procesos de Negocio. En consecuencia, la Gerencia de los Procesos del Negocio abarca la Integración de los Procesos de Negocios (BPI) y por lo tanto la Integración de las Aplicaciones Empresariales (EAI). (Ver, Figura 4).

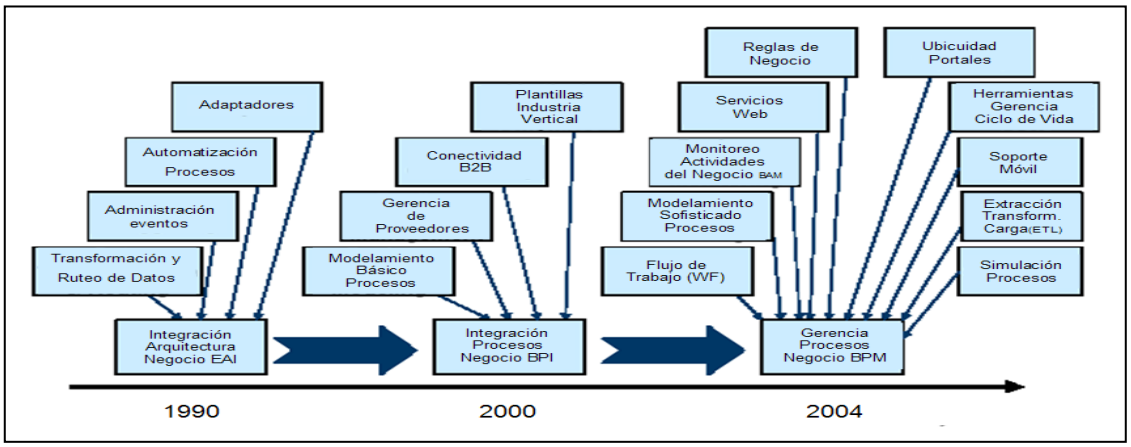


Figura 4: Evolución de las Aplicaciones EAI/BPM

Fuentes: (Vollmer et al., 2004)

Las herramientas de BPM son capaces de apoyar este tipo de actividad debido a su capacidad de coordinar interacciones entre: (1) los sistemas de información; (2) los Procesos del Negocio, y; (3) la gente que los utiliza. Esto da visibilidad de las empresas en el estado de los procesos y tiende a permitir cambios de los procesos sobre una base en uso.

En la Figura 5 se presenta la arquitectura que soporta los BPMS, en la base inferior se encuentra la capa de integración de datos (bases de datos distribuidas, replicación de datos, extracción transformación y carga de datos, integración de la información de la empresa, intercambio electrónico de datos y los elementos de dato como tal), a continuación la capa de integración de aplicaciones (modelo de objetos canónico y distribuido, integración de aplicaciones empresariales y las aplicaciones), seguidamente la capa de integración de la presentación (portales, dispositivos de presentación y adaptación de aplicaciones a dispositivos móviles) y finalmente la capa de integración de los procesos (BPM y flujo de trabajo).

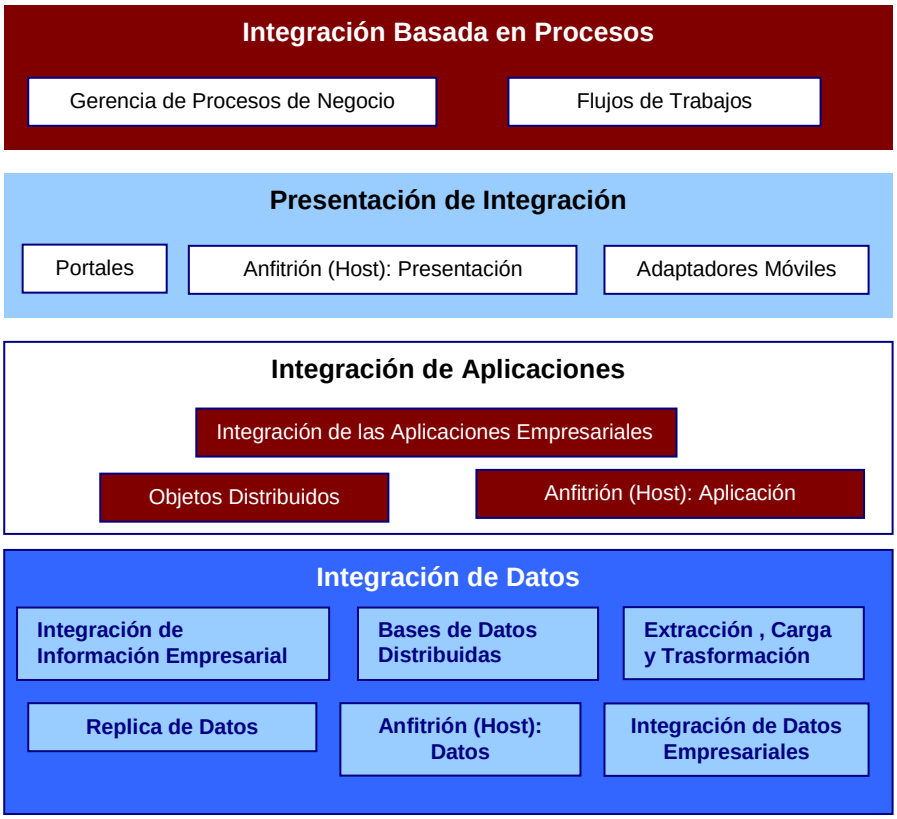


Figura 5: Arquitectura BPM.

Fuente: (Vollmer et al., 2004)

En la Iniciativa para la Gerencia de los Procesos de Negocio (BPMI, hoy unida a la OMG), se definen especificaciones estándares y abiertas, tales como la notación en la que se modelan los Procesos del Negocio (BPMN) y el lenguaje de consulta de procesos (BPQL), que permite la gerencia estándar del análisis del proceso, basada en el negocio y a través de los BPMS. Los estándares en los cuales se enfoca BPMI son los siguientes (ver, Figura 7):

1. BPML: Es el lenguaje en el que se modelan los Procesos de Negocio, se puede definir como un metalenguaje para modelar los procesos. BPML proporciona un modelo abstracto de ejecución para los procesos de colaboración y transaccionales, basado en el concepto de una máquina transaccional de estado. Se ha definido como medio para dar convergencia al uso del proceso dentro de las empresas, tanto de las transacciones distribuidas síncronas como asíncronas.

2. BPMN: Una notación estándar para el modelaje de los procesos de negocio (BPMN), la cual permite entender los procedimientos internos a través de una notación gráfica (*Business Process Diagram-BPD-)* permitiendo la comunicación de estos en una forma estándar. Esta notación facilita además el entendimiento de las colaboraciones, del rendimiento y de las transacciones, permitiendo la reutilización. Establece una relación entre los elementos gráficos y los constructores de los bloques estructurados del lenguaje de ejecución de procesos (BPEL), incluyendo BPML y BPEL4WS (BPEL para servicios Web).

3. BPSM: Es un "*framework*" conceptual que incluye patrones arquitecturales y de flujo de trabajo para BPM.

4. BPXL: Es un estándar del BPMI para extender BPEL4WS a fin de que pueda manipular transacciones, reglas de negocio, administración de tareas e interacción humano-humano asistida por el computador.

5. BPQL: Es una interface para administrar la infraestructura de los procesos de negocio que incluye facilidades de ejecución de procesos ("*Process Server*") y facilidades para el desarrollo de procesos ("*Process Repository*"). Esta interface permite a los analistas de proceso revisar el estado de los mismos y controlar su ejecución, se basa en el protocolo simple de acceso a objetos (SOAP). La utilidad correspondiente al repositorio permite implementar procesos desde el administrador de modelos y está basada en el protocolo de autorización de versionamiento distribuido (*WebDAV*). La administración de las interfaces BPQL puede ser expuesta a través de servicios UDDI (*Universal Description, Discovery and Integration*) a fin de registrar los procesos, adquirir y descubrir los mismos en un catálogo de servicios Web.

6. BPEL: (*Business Process Execution Language*) es un lenguaje basado en XML diseñado para compartir tareas en ambientes distribuidos –incluso a través de múltiples organizaciones- usando una combinación de servicios Web. Escrito por desarrolladores de *BEA Systems*, IBM y Microsoft, BPEL combina y substituye *IBM's WebServices Flow Language* (WSFL) y la especificación *Microsoft's XLANG*. (BPEL es también conocido como BPELWS o BPEL4WS). Usando BPEL, un

programador describe formalmente un Proceso del Negocio que ocurre a través de la Web de tal forma que cualquier entidad pueda realizar unos o más pasos en el proceso de la misma manera.

7. WS-CDL: Es un lenguaje de descripción de coreografía de servicios, este lenguaje está basado en XML y describe las colaboraciones entre las entidades a través de un punto focal con un comportamiento común y complementario, donde el orden del intercambio de mensajes se determina a partir de los objetivos del negocio. Esta especificación de servicios Web ofrece un puente de comunicación entre los ambientes computacionales heterogéneos.

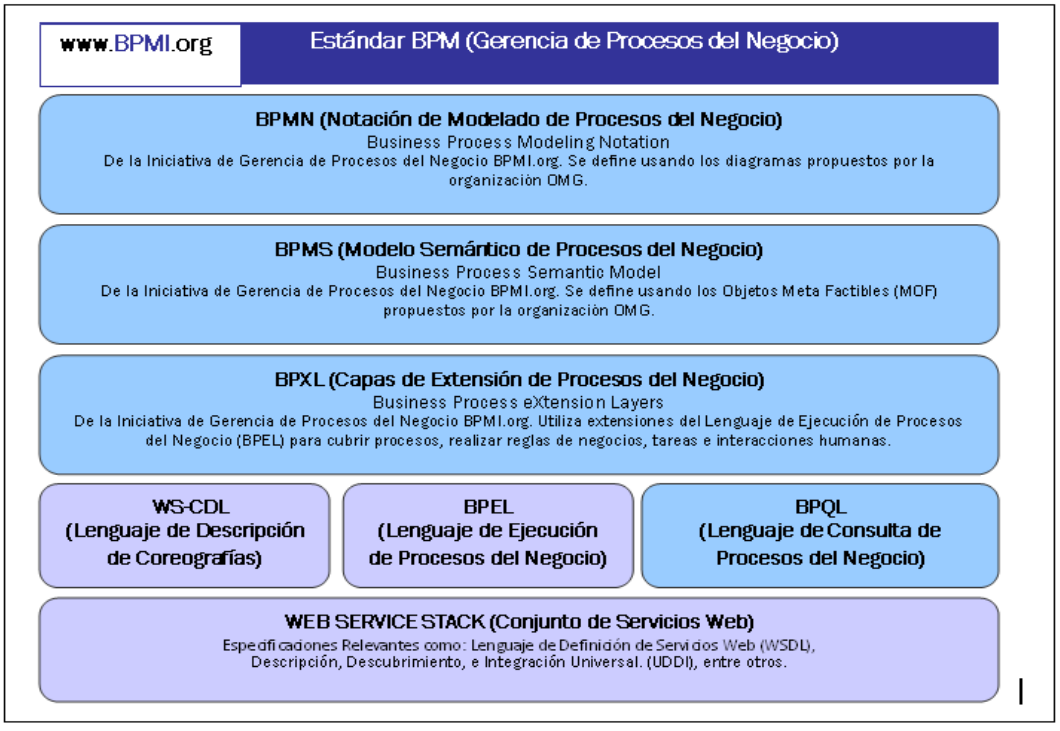


Figura 6: Estándares BPM según BPMI

Fuente: www.BPMI.org

Gartner (2005), propone un conjunto de criterio que permiten evaluar las arquitecturas de BPM, estos son: (1) Herramientas gráficas – diseño, análisis, modelado y definición de procesos.; (2) Motor de la ejecución ("*run-time*"), máquina de estado que

ejecuta el flujo de proceso definido. El motor puede invocar los servicios o las tareas automatizadas que el ser humano tiene que terminar. Los servicios se pueden proporcionar a través de la arquitectura de servicios y por otros proveedores ("*outsourcers*"); (3) Agilidad, implica la disponibilidad de realizar cambios a los procesos en ejecución, la administración de la lista de tareas y las prioridades del trabajo; (4) Herramientas para supervisar y para administrar flujos, incluye el funcionamiento de los procesos, el grado de terminación o las condiciones de finalización. Puede también cubrir los elementos que produjeron la finalización del proceso, compensación, balanceo de carga y transferencia, y; (5) Herramientas para el control de gestión a través de los datos almacenados que permiten las mediciones y los ajustes (control de proceso estadístico BAM) del negocio.

De acuerdo a esto, los requisitos funcionales con los que debe cumplir una aplicación de BPM, son los siguientes:

- *Análisis de procesos de negocio* (BPA): se refiere a cuales procesos deben ejecutarse y cómo estos están estructurados, lo que implica las actividades de administración del modelo de procesos, actualización de formas, modelo organizacional (roles, grupos, usuarios, departamentos, localidades, etc.), especificación de flujos, interfaz gráfica amigable y manejo estructurado de procesos.

- *Business Rule Engine* (BRE): incluye las actividades de configuración de usuarios, configuración de áreas y localidades, asignación de responsables, tipos de casos y tipificaciones, reglas de notificación y escalamientos, detección de solapamientos, configuración de mecanismos de notificación, manejo de soluciones, modelos de acuerdos de servicio y capacidad para el manejo de excepciones.

- *Generación de códigos:* se refieres las actividades de especificación de estándares, integración con tecnologías EAI, integración con herramientas de análisis, arquitectura WEB, soporte de arquitectura de servicios, compilación de procesos, soporte a procesos de larga duración, compensación y reverso de transacciones, manejo de colas, manejo de versiones, generación del flujo de trabajo, optimización de lista de trabajo y soporte e interacción.

- *Integración y Colaboración:* incluye las actividades de actualización y modificación de procesos, control de configuraciones, desactivación y activación de procesos, exportación de datos, exportación de procesos, gestión de históricos, gestión de procesos, importación de datos, importación de procesos, migración de flujos de trabajo, administración de usuarios, facilidades de comunicación, reportes y administración.

- *Monitoreo de las Actividades del Negocio (BAM):* incluye las actividades de monitoreo de procesos en ejecución, consola flexible de presentación gráfica ("Dashboard"), manejo de eventos y capacidad de análisis.

- *Interfaz de Aplicación:* incluye las actividades de administración de patrones de eventos de interacción, administración de patrones de interfaz, mantenibilidad de la interfaz y usabilidad de la interfaz.

- *Optimización:* incluye las actividades de manejo de escenarios, administración y manejo de servicios y administración y manejo de interfaces.

Esto conlleva a la necesidad de un marco referencial que englobe tanto a la Gestión de Procesos de Negocio como lo que es Patrones, relación que se describe a continuación. Pero primero hay que tener en cuenta que es un patrón.

Patrones

Los patrones fueron inicialmente concebidos por el arquitecto Christopher Alexander como una manera de formalizar la solución a problemas comunes a muchos proyectos de arquitectura. Más tarde, vista la eficacia de los patrones en el campo de la arquitectura, otras disciplinas los añadieron a su repertorio de herramientas.

La informática no ha sido una excepción y la adaptación de este concepto a la ingeniería del software ha dado lugar a lo que conocemos como **patrones software**. Estos patrones capturan soluciones a situaciones de especial interés, bien por ser muy comunes, bien por ser especialmente complejas.

Normalmente, un patrón está compuesto por el enunciado del problema y una o varias propuestas de solución. Para formalizar estas soluciones, se necesita una notación expresiva y rica en semántica que permita plasmar eficientemente los atributos de las

mismas. Además esta notación ha de ser fácil de entender e interpretar. La notación más extendida es **UML** (*Unified Modeling Language*) que permite capturar con precisión las soluciones a problemas de todo tipo.

Taxonomía de Patrones

La presente taxonomía o jerarquización de patrones fue propuesta por Bonillo (2008). A continuación, se explican con mayor detalle cada uno de los patrones que integran la taxonomía propuesta: análisis, arquitectural, diseño e interacción. Al final, se presentan los patrones de flujo de trabajo como escenarios de los patrones de análisis, por lo que no se incluyen los mismos dentro de la taxonomía. De igual forma no se incluye el tema particular de los *idioms*, dado que son específicos por tecnología, sin embargo es conveniente mencionar que para la mayoría de los BPMS, el *idiom* es JAVA.

Actualmente, el uso de patrones en el proceso de producción de software es uno de los temas más tratados, generando un gran interés entre investigadores y desarrolladores (Barros, 2003).

Los patrones pueden aplicarse a nivel del: análisis de requisitos, el diseño de la arquitectura, el diseño detallado, la interacción con el usuario, el flujo de trabajo y el código. Con lo que puede establecerse la siguiente clasificación:

Patrones de Análisis: Los patrones de análisis son grupos de conceptos que representan una construcción común en el mundo del modelado conceptual. Pueden ser relevantes a un dominio o ser genéricos a diferentes dominios. La idea central es la construcción de escenarios utilizando patrones. Se pretende tener una visión conceptual y estructural de las situaciones, con el fin de identificar la naturaleza intrínseca de las mismas. Con esa visión, es posible determinar el tipo de escenario correspondiente a cada situación y así, elegir un patrón de un catálogo, re-usando su estructura con el fin de derivar el escenario más fácil y directamente. **Los Patrones de Análisis consisten en un texto guía**, que para cada componente del escenario incluye pautas acerca del contenido que deberá tener dicho componente. Por ejemplo el patrón de análisis para realizar una actividad productiva es aplicado al escenario de diseño de una agenda de reuniones de tal forma de reutilizar las características de una actividad productiva en el dominio específico de una agenda de

reuniones. Los patrones de Flujo de Trabajo (Bonillo, 2005c) son escenarios de los patrones de análisis descritos en el Anexo 1.1, por lo que no se mencionan en la taxonomía.

Patrones de Arquitectura (Arquitecturales): **Son esquemas fundamentales de organización de un sistema**. Especifican una serie de subsistemas y sus responsabilidades respectivas e incluyen las reglas y criterios para organizar las relaciones existentes entre ellos. Algunos ejemplos son: capas, filtros y conexiones; modelos, vistas y controles (MVC).

Estilos Arquitectónicos: representan una **generalización de los patrones de arquitectura,** dado que **expresan los componentes y las relaciones del esqueleto estructural y general** de una aplicación independiente del contexto y de otros estilos, son una categorización de sistemas. Algunos estilos típicos son las arquitecturas basadas en flujos de datos, las de invocación implícita, las jerárquicas, las centradas en datos o en intérprete-maquina virtual.

Patrones de Diseño: Son patrones de un nivel de abstracción menor que los patrones de arquitectura. Están por lo tanto **más próximos a lo que sería el código fuente final**. Su uso no se refleja en la estructura global del sistema. Por ejemplo: "*abstract factory*", "constructor" y "*chain responsability*". En 1995, Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides publicaron el libro "*Design Patterns: Elements of Reusable Object Oriented Sofware"* Ellos recopilaron y documentaron 23 patrones de diseño aplicados usualmente por expertos diseñadores de software orientado a objetos. A partir de este momento comenzó a reconocer a estos autores como el grupo de los cuatro o **GoF**, en su libro clasificaron los patrones de diseño en 3 grandes categorías basadas en su propósito: creacionales, estructurales y de comportamiento.

Patrones de Interacción: también conocidos como **Patrón de Interfaz**, describe una solución exitosa a un problema recurrente concerniente a la interfaz de usuario, en un contexto dado. Un Patrón de Interacción es un **medio de comunicación** que se expresa en una notación sencilla, a fin de ser entendida por las personas del equipo de diseño de la interacción que generalmente es multidisciplinario. Algunos ejemplos son: formatos de datos de fechas y representación visual jerárquica del estado del sistema.

Patrones de Implementación: Se refieren a la **forma de programar o implementar** una solución en un lenguaje específico, se asocia con los términos en inglés "*kit*" e "*Idiom*".

Cada uno de los tipos de patrones se relaciona con el nivel de abstracción en el cual se utilizan en el proceso de desarrollo de software, desde los patrones de análisis en el nivel de abstracción superior, hasta los patrones de implementación en el menor nivel de abstracción.

A su vez cada tipo de patrón genera conocimiento y tiene una calidad; de igual forma en la relación que existe entre un patrón de un nivel de abstracción superior con uno de menor nivel, se crea también conocimiento y calidad, lo que determina la reutilización de la combinación de los patrones en sus diferentes niveles de abstracción. Es decir la taxonomía de patrones propuesta, tiene implícito un modelo de gestión de conocimiento para la reutilización del patrón individual en su nivel de abstracción y de la relación entre los patrones. Al utilizar esta taxonomía dentro de la propuesta metodológica, se establece una diferencia con el resto de las metodologías, que utilizan los patrones de forma individual.

Figura 7: Diferentes tipos de Patrones y su Relación

Fuente: (Bonillo, 2008)

En el Anexo 1, describe con mayor detalle cada uno de los patrones antes mencionados. A partir de esta descripción, es preciso notar que **lo que diferencia a los tipos de patrones entre sí, está relacionado con su nivel de abstracción** (patrones

aislados versus familias – lenguajes o catálogos – de patrones). Esta clasificación, basada en lo planteado por Bushmann et al. (1996).

Según Bonillo, **en general los patrones tienen una limitación: son difíciles de especificar y evaluar en base a un modelo de calidad particular**, por lo que el estudio de los temas de ADL (*Architecture Definition Language*) y ABAS (*Atribute-Based Architecture Styles*), como estructuras que extienden la representación dada por el Grupo de los Cuatro (GoF) con la finalidad de especificar la información sobre los patrones y las características de calidad relativas, se consideran relevantes y fueron especificados en la investigación doctoral del autor antes mencionado.

De acuerdo a todo lo anterior, el autor reflexiona lo siguiente: que las descripciones de GoF pueden ser extendidas utilizando ABAS como un estilo y no una técnica (forma eficiente y eficaz de utilizar una herramienta), de tal forma de expresar las características relacionadas con la calidad. La relación entre los aspectos teóricos desarrollados se puede resumir como se muestra en la Figura 8.

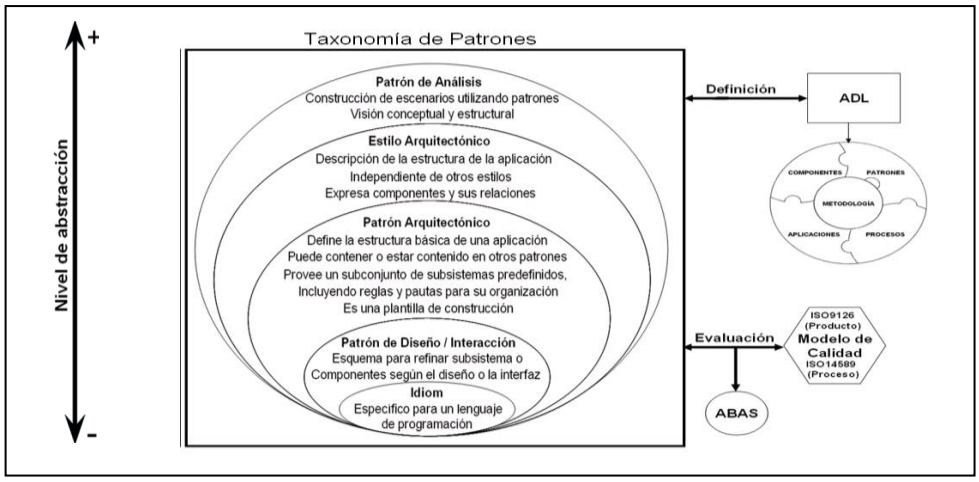


Figura 8: Relación entre las Bases Teóricas

Fuente: (Bonillo, 2008)

Bonillo 2006, tomando como referencia la representación de patrones de interacción propuesto por Acosta y Zambrano (2004), muestra una representación extendida del metapatrón, con la finalidad de incorporar el concepto de ABAS en la taxonomía de patrones expuesta con anterioridad.

Tabla 1: Metapatrón adaptado de Acosta, Zambrano (2004) y Kazman, Klein (2004)

Nombre (título), autor clasificación, dominio (usos conocidos) y rango.	Nombre: Idea Central por la cual se identifica el patrón. Autor: Nombre de la persona que creó el patrón. Clasificación: Se refiere al tipo de patrón de acuerdo a la taxonomía antes mencionada. Dominio: Indica el o los dominios en los cuales el patrón ha sido implementado. Rango: es el grado de confiabilidad de este patrón con respecto a los dominios en los cuales ha sido implementado.
Problema	Describe el problema que será resuelto, desde el punto de vista del usuario.
Solución	Describe, en una forma narrativa y gráfica, la solución al problema. Con base en: Objetivo, Intención, Motivación, Actores o Participantes, Recursos, Estructura, Episodios, Colaboraciones e Implementación. (Ver Anexo 1, para ejemplos)
Atributos de Calidad	Atributos de calidad de interés, el contexto de uso, contrastes y atributos relevantes (requerimientos específicos).
Medidas de atributos de Calidad	Un resumen de lo que es discutido en la descripción del problema, usando términos específicos relacionados con aspectos medibles de los atributos del modelo de calidad. Esto incluye una discusión de los eventos que pudieran hacer que la arquitectura responda o cambie.
Parámetros de atributos de Calidad	Un resumen de lo que será discutido en la sección de solución, pero especificando términos relevantes a los parámetros de los atributos especificados en el modelo de calidad.
Análisis del Modelo de Calidad	Una descripción de cómo los atributos del modelo de calidad serán relacionados formalmente con los elementos del patrón y las conclusiones sobre la conducta arquitectural que se obtiene con el modelo. Esta información se utiliza indicar el resultado de la medición de los atributos de calidad establecidos para el patrón con sus respectivas medidas y parámetros.
Contexto	Presenta las condiciones bajo las cuales el patrón es utilizado.
Fuerza	Conflictos que pudiesen restringir la solución. Incluyendo las excepciones.

Consecuencias	Describe el resultado de aplicar el patrón.
Ejemplos/Contra-ejemplos	Muestra ejemplos y contraejemplos de la solución propuesta.
Patrón Relacionado	Otros patrones que se relacionan con el patrón descrito.

Fuente: Bonillo, 2008

Se cita que la usabilidad del patrón indicada en la representación de Acosta y Zambrano (2004) es un atributo de calidad. Además un patrón no requiere, generalmente, todos los elementos antes mencionados, a fin de **cumplir su objetivo, un patrón debe tener como mínimo** los siguientes elementos: Nombre, Problema, Solución, Contexto y Fuerzas.

En base a lo descrito anteriormente (BPM y Patrones), Bonillo propone la metodología donde integra la Metodología de Proceso de Negocio con el concepto de Patrones

Metodología de Gerencia de Procesos de Negocio Sustentada en el Uso de Patrones

Bonillo (2008), plantea una metodología sustentada en el uso de patrones. Su investigación estuvo definida bajo los parámetros establecidos por la Metodología SESL (*Systemic Evaluation for Stakeholder Learning*) y el propósito fue la instanciación o instanciamiento de la propuesta original realizada por Magnus Ramage en 1979 basada en Sistemas de Trabajo Cooperativo (*Cooperative Work*), el cual nos indica que "la combinación de la tecnología, personas y la organización facilita la comunicación y coordinación necesaria para que un grupo trabaje efectivamente con la finalidad de alcanzar un objetivo común".

La representación del sistema de estudio se realizó proponiendo un enfoque *top-down* utilizando la teoría general de sistemas. La metodología que propone la investigación de Bonillo, está basada en una conjunción de los conceptos relativos a la Gerencia de los Procesos de Negocio y los patrones de software, la organización de soporte interno, los objetivos del evaluador y la teoría general de sistemas. (Bonillo, 2004a).

El trabajo de Bonillo (2008) busca validar y relacionar conceptos de las ciencias de la computación, específicamente en el área de la ingeniería de software a fin de administrar Procesos de Negocio a través de la utilización de patrones, permitiendo la reutilización de mejores prácticas, estableciendo relaciones y auspiciando la calidad del proceso y el producto de software a través de la revisión del uso y la relación de los diversos patrones en el dominio BPM.

Diseño de la Metodología de Gerencia de Proceso de Negocio sustentada en el uso de Patrones

Bonillo (2008), plantea el diseño de la metodología tomando en cuenta que la metodología a diseñar es un Proceso de Negocio (para la gerencia sustentada en el uso de patrones de otros Procesos de Negocio) y utiliza la Notación para el Modelamiento de Procesos de Negocio (BPMN por sus siglas en inglés).

El Modelamiento en BPMN se realiza mediante diagramas simples con un conjunto muy pequeño de elementos gráficos. Las cuatro categorías básicas de elementos son:

Objetos de flujo: Eventos, actividades, rombos de control de flujo.

Objetos de conexión: Flujo de secuencia, flujo de mensaje, asociación.

Carriles: Representan los roles y por cada uno de ellos se coloca un carril.

Artefactos: Objetos de datos, grupo y anotación.

La propuesta metodológica para la Gestión de los Procesos de Negocio sustentada en el Uso de Patrones está conformada por dos sub-procesos (ver, Figura 9): **Crear** Proceso y **Administrar** Proceso.

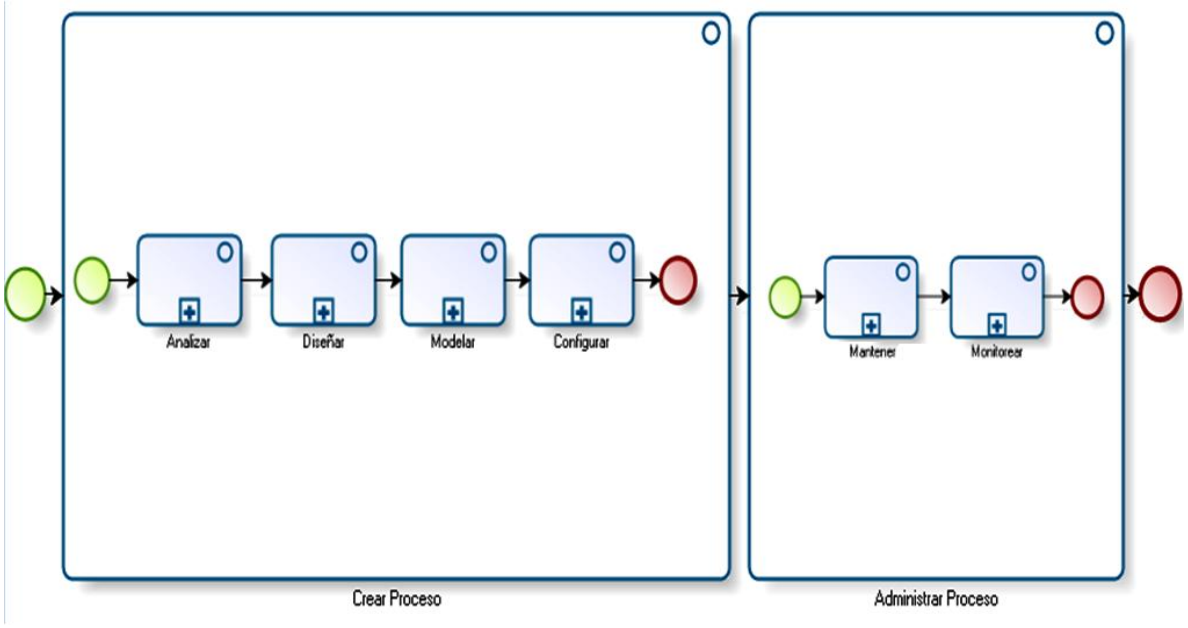


Figura 9: BPD del proceso de la Metodología de Gerencia de Procesos Sustentada en el Uso de Patrones.

Fuente: Bonillo 2008

El proceso de la Metodología de Gerencia de Procesos Sustentada en el Uso de Patrones, inicia con la solicitud de creación de un nuevo proceso o modificación sobre un proceso existente, esta solicitud pasa por los sub-procesos: **Analizar, Diseñar, Modelar y Configurar** del sub-proceso **Crear**, este sub-proceso finaliza al colocar en producción la solicitud de nuevo proceso o mejora.

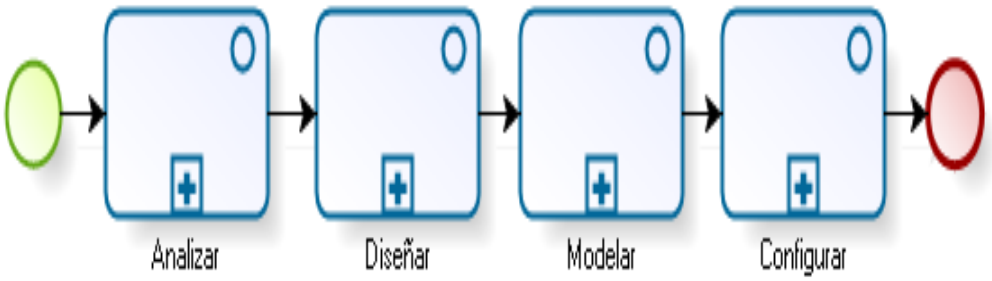


Figura 10: BPD del sub-proceso Crear Proceso.

Fuente: Bonillo 2008

A continuación con los procesos en producción se inicia el sub-proceso **Administrar** (ver Figura 11), a través del cual se Mantienen las Aplicaciones, Configuraciones y la Plataforma (sub-proceso **Mantener**) y se Monitorean las Funciones y la Plataforma (sub-proceso **Monitorear**).

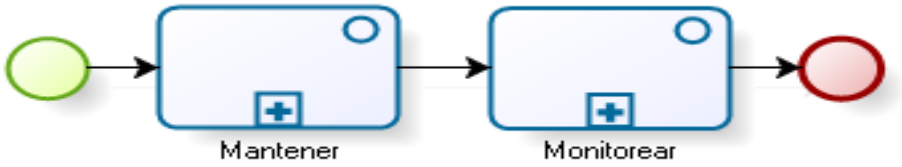


Figura 11: BPD del sub-proceso Administrar.

Fuente: Bonillo 2008

A continuación se describen, los sub-procesos que integran el sub-proceso **Crear**. El primero de los sub-procesos de Crear es el sub-proceso **Analizar**, el cual se divide en dos sub-procesos: **Identificar los Requerimientos y Levantar Información**. (Ver Figura 12).

La función principal del sub-proceso **Analizar** es recibir los requerimiento de nuevos procesos o mejoras que realizan los Usuarios a través del Escritorio de Ayuda; asignarlos a los Analistas de Negocio o al equipo de Mantenimiento de Aplicaciones, a fin de identificar el requerimiento y levantar la información detallada; relacionar el requerimiento con los patrones de análisis disponibles en un catálogo, de tal forma de reutilizar el conocimiento que se genera con la utilización de los patrones y la medición previa de su calidad, o generar nuevo conocimiento sobre los patrones seleccionados, y preparar la documentación necesaria para continuar con los sub-procesos **Diseñar** y **Gestionar cambio**.

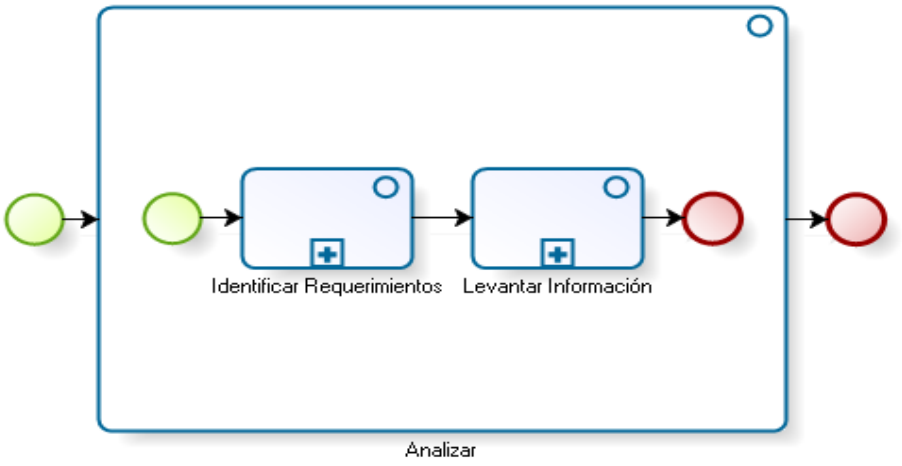


Figura 12: BPD del sub-proceso Analizar.

Fuente: Bonillo 2008

El sub-proceso **Identificar Requerimientos** inicia cuando el Usuario solicita la creación de un nuevo proceso o mejora sobre un proceso existente, al Escritorio de Ayuda.

En caso que el requerimiento sea para crear un nuevo proceso, el Escritorio de Ayuda registra el requerimiento según la planilla que se presenta en el Anexo 2, y asigna el requerimiento al Usuario con la finalidad de que realice seguimiento, y también lo asigna al Analista de Negocio para que estudie y describa con mayor detalle el requerimiento. (Ver Figura 13).

En el caso en que el requerimiento del Usuario sea una mejora sobre un proceso existente, se envía este requerimiento al sub-proceso de **Administrar Proceso**.

Una vez que el Analista de Negocio tiene asignado el requerimiento de nuevo proceso, describe con mayor detalle el mismo, en la misma planilla de solicitud (ver, Anexo 2). A continuación asigna una prioridad al requerimiento, esto en base a la urgencia y al impacto del proceso en la organización.

Luego evalúa sobre una cartelera previa de actividades (ver, Anexo 3), cuáles serán las actividades que provisionalmente se asociarán en el tiempo a este requerimiento, y pasa toda esta información, con el requerimiento, al sub-proceso **Levantar Información**.

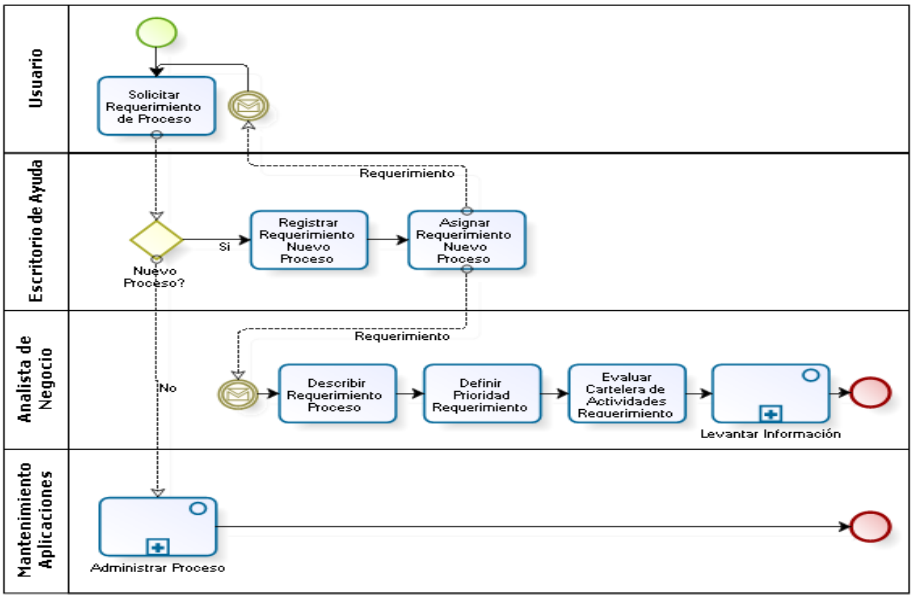


Figura 13: BPD del sub-proceso Identificar Requerimiento

Fuente: Bonillo 2008

A continuación en el sub-proceso **Levantar Información**, el Analista de Negocio, toma el requerimiento asignado previamente por el Escritorio de Ayuda y recupera de un catálogo existente, los Patrones de Análisis. (Ver, figura 14).

Para recuperar el Patrón de Análisis, el Analista de Negocio se conecta a una Herramienta de Administración de Patrones, en la cual se encuentran los catálogos (ver, Anexo 4). En esta Herramienta se recuperan los Patrones de Análisis realizando una búsqueda en base al Nombre del Patrón, el Autor, el Problema, la Solución o el Contexto.

Posteriormente el Analista de Negocio relaciona el Requerimiento con los Patrones de Análisis seleccionados, de tal forma de reutilizar el conocimiento previamente almacenado en el catalogo sobre estos patrones.

Luego el Analista de Negocio, junto con el Usuario levanta la información correspondiente al funcionamiento actual del proceso (utilizando para esto el formato disponible en el Anexo 5, en el cual se indican las características particulares del proceso, sus elementos y atributos, los roles asociados, la estructura organizacional y la localidad en la cual se ejecuta el proceso).

Toda esta información sirve de base a la ejecución de los siguientes sub-procesos: **Diseñar** y **Gestionar Cambio**, con los cuales se finaliza el sub-proceso de **Levantar información**.

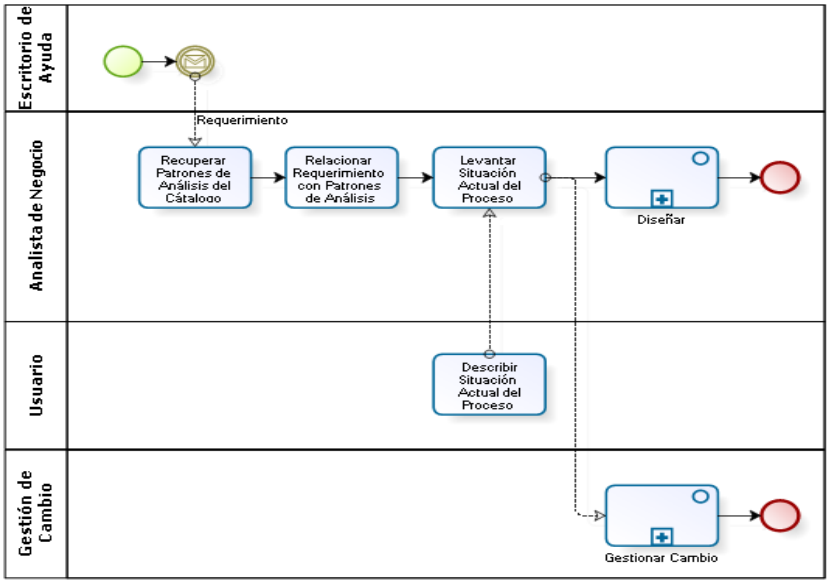


Figura 14: BPD del sub-proceso Levantar Información

Fuente: Bonillo 2008

El siguiente es el sub-proceso de **Diseño**, en este el Analista de Negocio compara el proceso de acuerdo a su funcionamiento actual con los Patrones de Análisis seleccionados, esto lo hace con la finalidad de comparar el "ser" contra el "deber ser". (Ver, Figura 15).

Con la comparación de los patrones y la funcionalidad actual del proceso el Analista de Negocio identifica brechas y riesgos sobre diseñar el proceso según el patrón o según su funcionalidad actual.

A continuación el Analista de Negocio, discute las brechas y los riesgos con el Usuario, de tal forma que el usuario le indique si se modelará el proceso en papel de acuerdo a su funcionamiento actual o como lo indican los patrones de análisis, en cualquiera de los dos casos el Analista de Negocio realiza un modelo en papel del proceso.

Una vez realizado el modelo en papel el Analista de Negocio, le envía este modelo al Arquitecto de Sistemas, el cual a través de la Herramienta de Administración de

Patrones, recupera los Estilos Arquitecturales (ver, Anexo 6), recupera los Patrones Arquitecturales (ver, Anexo 7), relaciona los Patrones Arquitecturales con los Estilos Arquitecturales, los Estilos Arquitecturales con los Patrones de Análisis, y los Patrones en general con el Proceso Modelado en Papel.

Es preciso aclara que todas estas relaciones se realizan con el fin de gestionar el conocimiento que produce cada uno de los Patrones de manera individual y grupal, de tal forma que el Arquitecto pueda luego reutilizar este conocimiento.

Luego el Arquitecto selecciona el proceso de desarrollo de software que utilizará para construir los componentes que se requieran, entre AUP, XP y FDD (la Gestión de Procesos de Negocio es un sub-proceso del Desarrollo de Software, a través del cual se obtiene un producto terminado, Figura 17), y propone una cartelera definitiva de actividades, teniendo en cuenta los patrones seleccionados y relacionados y el proceso de desarrollo de software elegido.

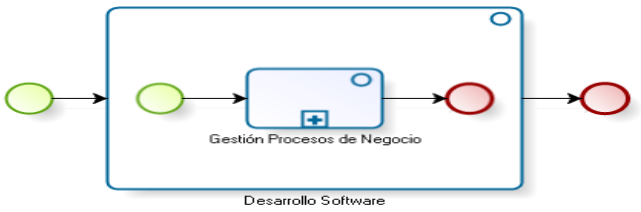


Figura 15: BPD Relación Desarrollo de Software y Gestión Procesos de Negocio

Fuente: Bonillo 2008

Seguidamente, el Analista de Negocio, presenta esta cartelera de actividades definitivas al Usuario para que la acepte o no, en el caso de aceptar se pasan los patrones relacionados y el modelo en papel al sub-proceso **Modelar**, en caso contrario se continúan negociando las actividades con el Usuario.

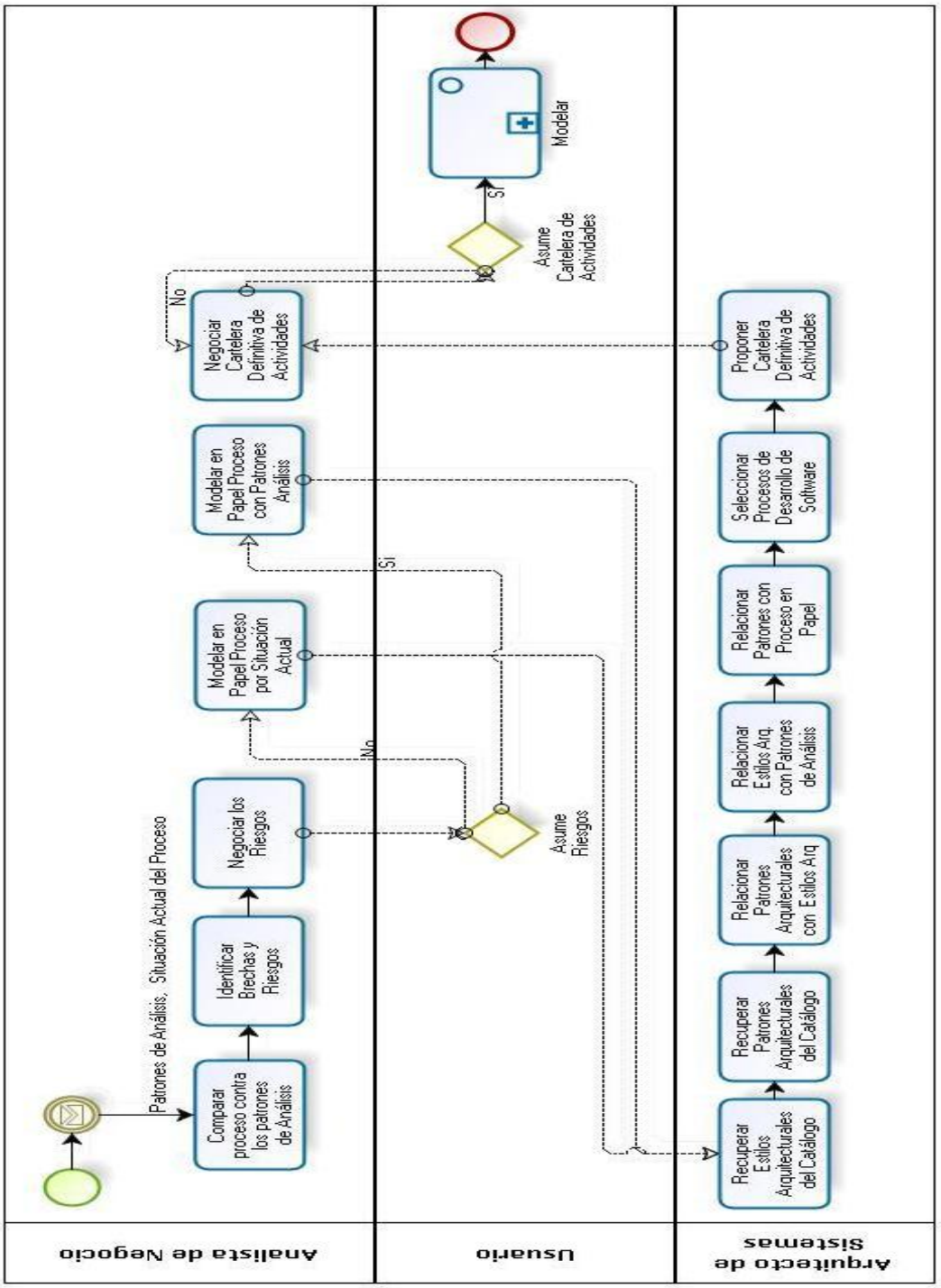


Figura 16: BPD del sub-proceso Diseñar

Fuente: Bonillo 2008

Una vez analizado y diseñado el proceso, el siguiente sub-proceso consiste en **Modelar**. En este sub-proceso el Analista de Negocio, realiza un diagrama, de acuerdo al modelo realizado en la fase anterior, en una herramienta de análisis de proceso BPA.

Es importante citar que durante la diagramación del proceso de negocio se utiliza una herramienta que administra de forma integral los objetos del negocio que serán accedidos desde el BPA (ver, Anexo 8).

Seguidamente el Analista de Negocio realiza una simulación funcional en la herramienta BPA a fin de detectar tendencias (cuellos de botellas, ciclos, etc.) y solicita la aprobación del Usuario. Si el usuario no aprueba el resultado de la simulación funcional, el Analista de Negocio, continua con la diagramación del proceso según las observaciones realizadas por el Usuario.

En el caso en que el Usuario aprueba el resultado de la simulación funcional, el Arquitecto integra el diagrama del proceso actual con el diagrama general de todos los procesos diagramados para la organización.

Seguidamente el Analista de Negocio realiza una simulación integral del proceso dentro de todos los procesos de la organización a fin de verificar las tendencias con respecto a la arquitectura general de procesos.

A continuación el Analista de Negocio solicita la aprobación del usuario sobre esta simulación integral. Si el usuario no aprueba la simulación integral o existe algún tipo de error, entonces el Analista de Negocio vuelve a diagramar el proceso.

En el caso en el que el Usuario aprueba la simulación integral, el Arquitecto exporta a UML y a BPEL el diagrama del proceso. Es importante citar que la mayoría de los BPA, tiene mecanismos de exportar de BPMN a UML y BPEL que son los estándares con los cuales trabajan los desarrolladores de sistemas, en el caso particular del UML el BPA lo que exporta es una estructura general es decir el

desarrollador debe realizar los diagramas por completo, en cuanto al BPEL igualmente el desarrollador deberá realizar una tarea de configuración del mismo.

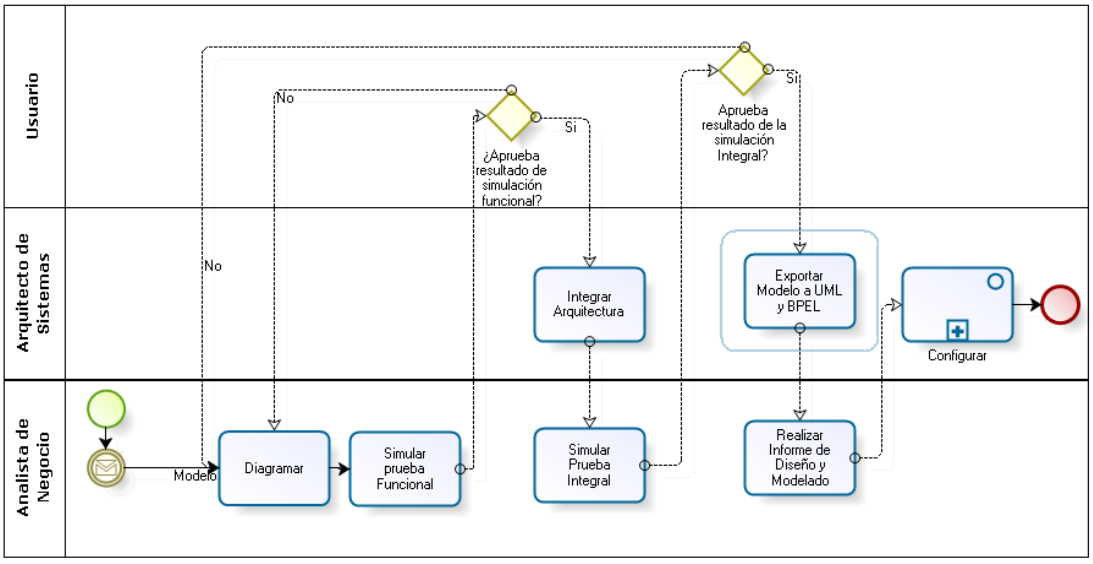


Figura 17: BPD del sub-proceso Modelar

Fuente: Bonillo 2008

Posteriormente, el Analista de Negocio, realiza un informe final del diseño y el modelado del proceso y pasa toda esta información al siguiente sub-proceso **Configurar.**

En el sub-proceso **Configurar** (ver, Figura 19), el Arquitecto de Sistemas, toma el UML el BPEL y los patrones seleccionados hasta el momento y los envía al Desarrollador para que realice dos sub-proceso: **Construir la Lógica de Negocio** y **Construir la Interfaz**.

Es importante citar que se habla de configurar y no programar directamente dado que con el BPEL se tiene una buena parte de la aplicación y sus componentes programados.

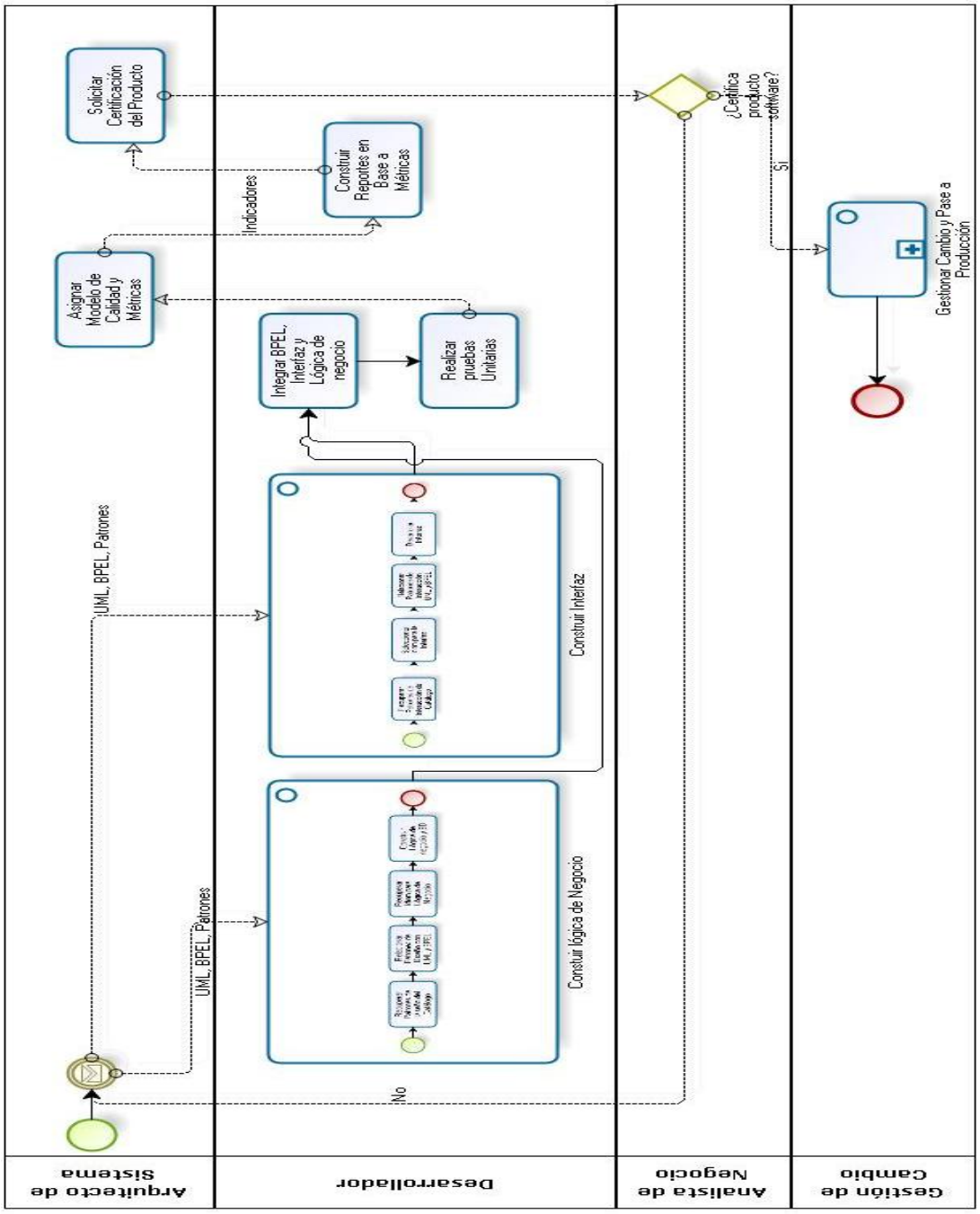


Figura 18:BPD Configurar

Fuente: Bonillo 2008

Construir la Lógica de Negocio, significa crear programas que realizan entradas de datos, consultas a los datos, generación de informes y más específicamente todo el procesamiento que se realiza detrás de la aplicación visible para el usuario (actualmente estos programas se expresan como servicios web). La Lógica de Negocio se compone de: (1) Operaciones Internas; (2) Aplicación de Lógica de Negocio; (3) Adaptador de Aplicaciones; (4) Componentes Lenguaje de Consulta a la Base de Datos (SQL), y (5) Componentes de Programación.

En el sub-proceso **Construir Lógica de Negocio** (ver, figura 10), el Desarrollador recupera los patrones de diseño del catálogo (ver, Anexo 10), relaciona estos Patrones de Diseño con el UML y el BPEL, recupera un *Idiom* o Lenguaje de Programación y construye en base al UML el BPEL y los patrones la Lógica de Negocio y una Base de Datos (la estructura de esta base de datos dependerá de las variables definidas para el proceso, y la tecnología será según el *Idiom* seleccionado) que apoya a esta Lógica de Negocio.

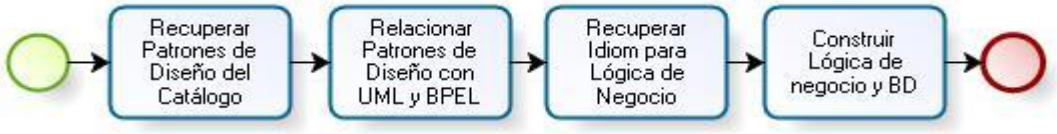


Figura 19: BPD del sub-proceso Construir Lógica de Negocio

Fuente: Bonillo 2008

En el sub-proceso **Construir Interfaz** el Desarrollador recupera los patrones de interacción del catálogo (ver, Anexo 10), selecciona un "*idiom*" para programar la interfaz, relaciona los patrones de interacción con el UML y el BPEL y desarrolla la interfaz.

Figura 20: BPD del sub-proceso Construir Interfaz

Fuente: Bonillo 2008

A continuación el Desarrollador integra el BPEL, la Interfaz y la Lógica de Negocio y realiza Pruebas Unitarias.

Con la finalidad de establecer mediciones sobre el proceso de negocio que se está configurando, seguidamente el Arquitecto asigna un modelo de calidad, este modelo se utilizará para medir el producto de software y el proceso, es por esto que se recomienda la utilización del modelo de calidad ISO-9126 para la medición de la calidad del Producto e ISO-14598 para la medición del proceso de software. A su vez, estos modelos de calidad son utilizados para asociar mediciones a los patrones que se han relacionados con el proceso de negocio.

Una vez que el Arquitecto asigna el Modelo de Calidad, el Desarrollador crea los Reportes asociados, estos reportes están basados en los indicadores de desempeño del proceso y que luego serán presentados en el sub-proceso **Monitorear.** El BPMS contiene normalmente un modulo de BAM el cual almacena estos indicadores en el tiempo de acuerdo a estructuras de inteligencia de negocio.

A continuación el Arquitecto le solicita al Analista de Negocio que certifique el Producto de software final.

Si el Analista de Negocio certifica el producto, entonces se iniciara el sub-proceso de **Gestión de Cambio** y el de **Pase a Producción.** En caso contrario se inicia nuevamente todo el sub-proceso **Configurar** con las observaciones que indique el Analista de Negocio.

En este punto se finaliza todo el sub-proceso **Crear Proceso**. A continuación con los procesos en producción se inicia el sub-proceso **Administrar** (ver, Figura 11).

El primero de los sub-procesos de **Administrar** es el sub-proceso **Mantener**, el cual su vez está confirmado por los sub-procesos **Mantener Aplicación**, **Mantener Configuración** y **Mantener Plataforma**.

En el sub-proceso **Mantener Aplicación**, el Escritorio de Ayuda le envía al área de Mantenimiento de Aplicaciones, una solicitud de requerimiento de modificación o mejora sobre un proceso que ya se encuentra en producción.

El área de Mantenimiento de Aplicaciones analiza el requerimiento, si considera que la solicitud tomará más de 40 horas de trabajo, devuelve el requerimiento al Escritorio de Ayuda con la finalidad de que se inicie un nuevo sub-proceso **Crear**.

Si el área de Mantenimiento de Aplicaciones analiza el requerimiento y considera que la solicitud tomará 40 horas o menos de trabajo, realiza un diseño de la solución y la prueba, para finalmente invocar al sub-proceso **Gestionar Cambio.**

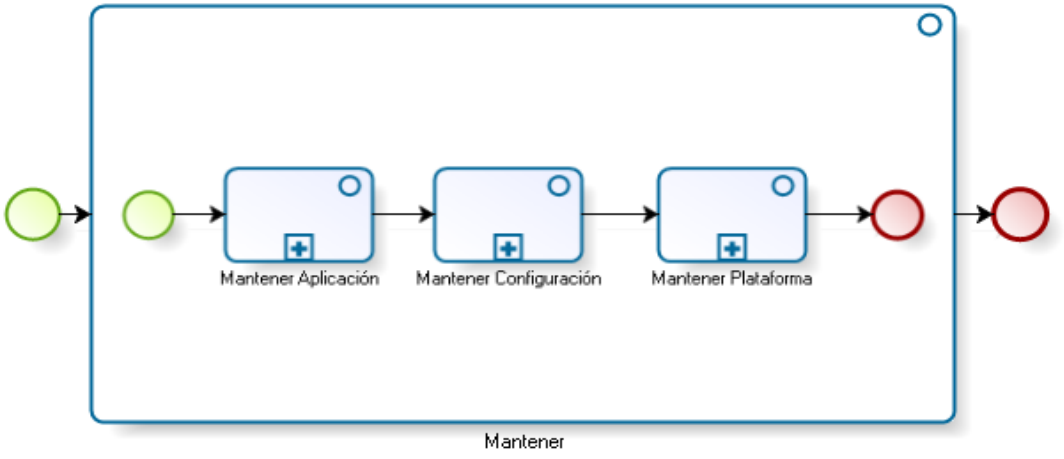


Figura 21: BPD del sub-proceso Mantener Configuración

Fuente: Bonillo 2008

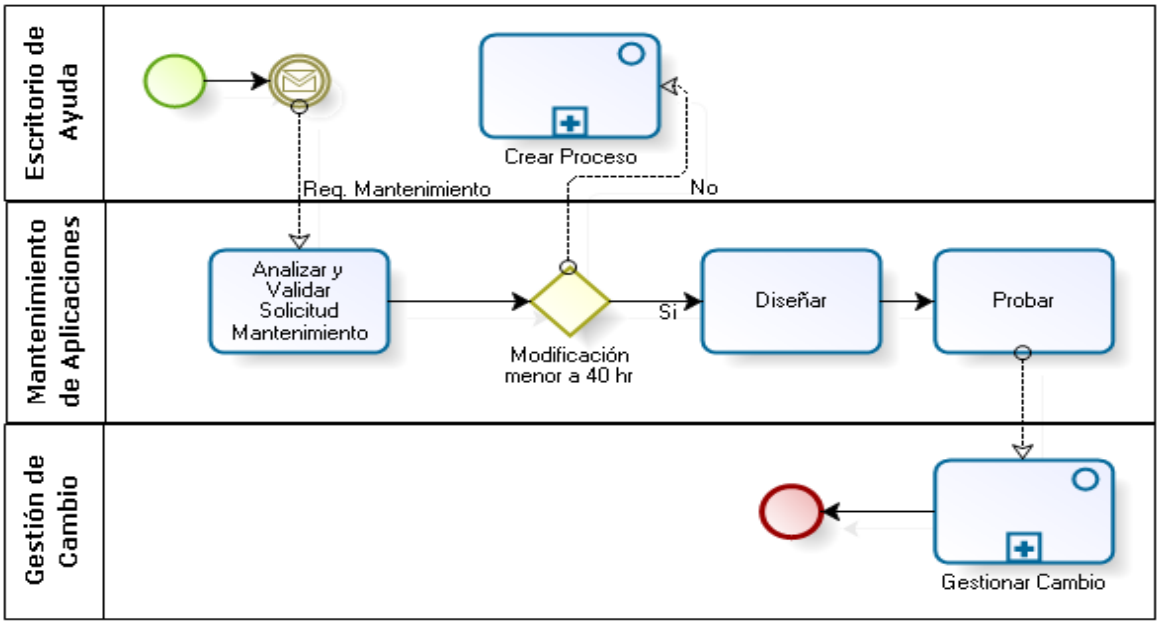


Figura 22: BPD del sub-proceso Mantener Aplicación

Fuente: Bonillo 2008

En el siguiente sub-proceso **Mantener Configuración** (ver, Figura 23), el Escritorio de Ayuda envía un requerimiento al área de Mantenimiento de Aplicaciones para modificar alguna variable del proceso (en los BPMS el flujo de control se realiza mediante variables es por esto que a cada decisión que exista en BPMN o en el BPEL se le asocian una o más variables de control).

El área de Mantenimiento de Aplicaciones decide si acepta el cambio de variables, en caso afirmativo modifica la variable, prueba el cambio e inicia el sub-proceso **Gestionar Cambio**; cuando decide no modificar la variable le indica las razones al Escritorio de Ayuda el cual debe notificarle al usuario.

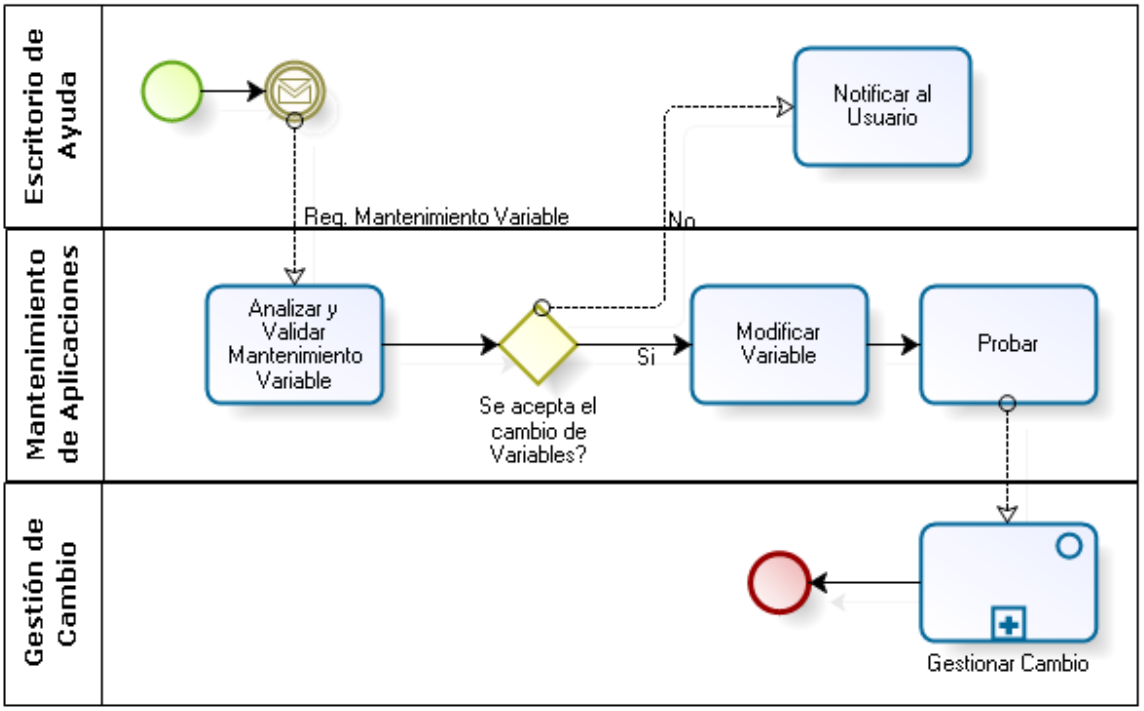


Figura 23: BPD del sub-proceso Mantener Configuración

Fuente: Bonillo 2008

El último de los sub-procesos de mantenimiento es **Mantener Plataforma** (ver, Figura 25), inicia cuando el Escritorio de Ayuda envía una solicitud de cambio en el hardware o el software donde se ejecuta el BPMS, al área de Mantenimiento de Plataforma, la cual analiza y acepta o no el cambio, en el caso de aceptar el cambio en la plataforma, realiza la modificación, prueba e inicia el sub-proceso **Gestionar Cambio**, cuando no se acepta el cambio se envía la razón al Escritorio de Ayuda, el cual notifica al Usuario.

A continuación tenemos los sub-procesos de **Monitorear** (ver, Figura 26): **Monitorear Funciones** y **Monitorear Plataforma.**

En el sub-proceso **Monitorear Funciones** (ver, Figura 27), el Gerente de Proceso realiza un requerimiento de medición en tiempo real al área de Mantenimiento de Aplicaciones, quienes analizan y validan la solicitud, si se decide

aceptar el requerimiento, se configura el reporte, se prueba y se inicia el sub-proceso Gestionar Cambio, en caso contrario el área de Mantenimiento de Aplicaciones notifica al Gerente de Proceso con la finalidad de ajustar el requerimiento.

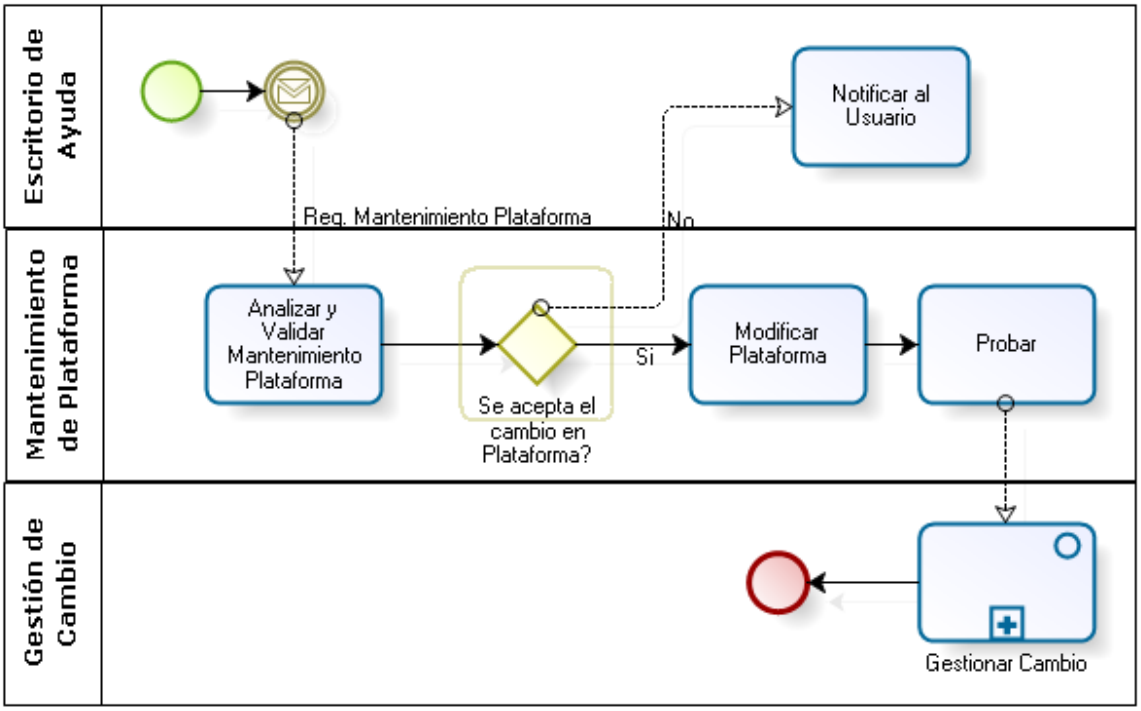


Figura 24: BPD del sub-proceso Mantener Plataforma

Fuente: Bonillo 2008

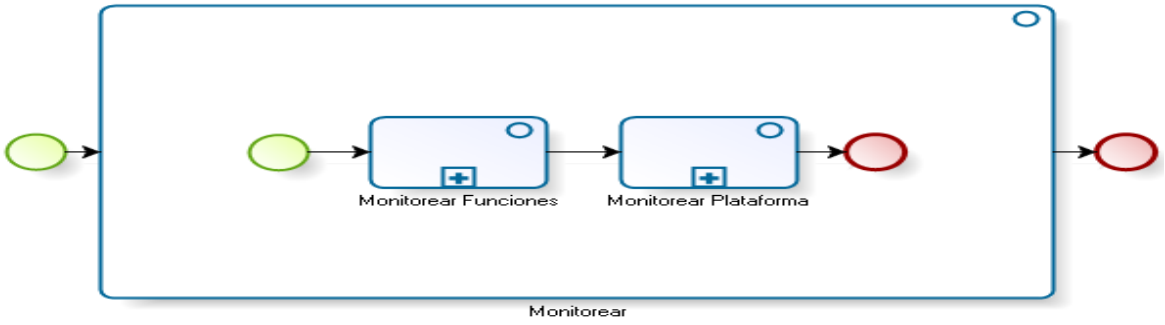


Figura 25: BPD del sub-proceso Monitorear

Fuente: Bonillo 2008

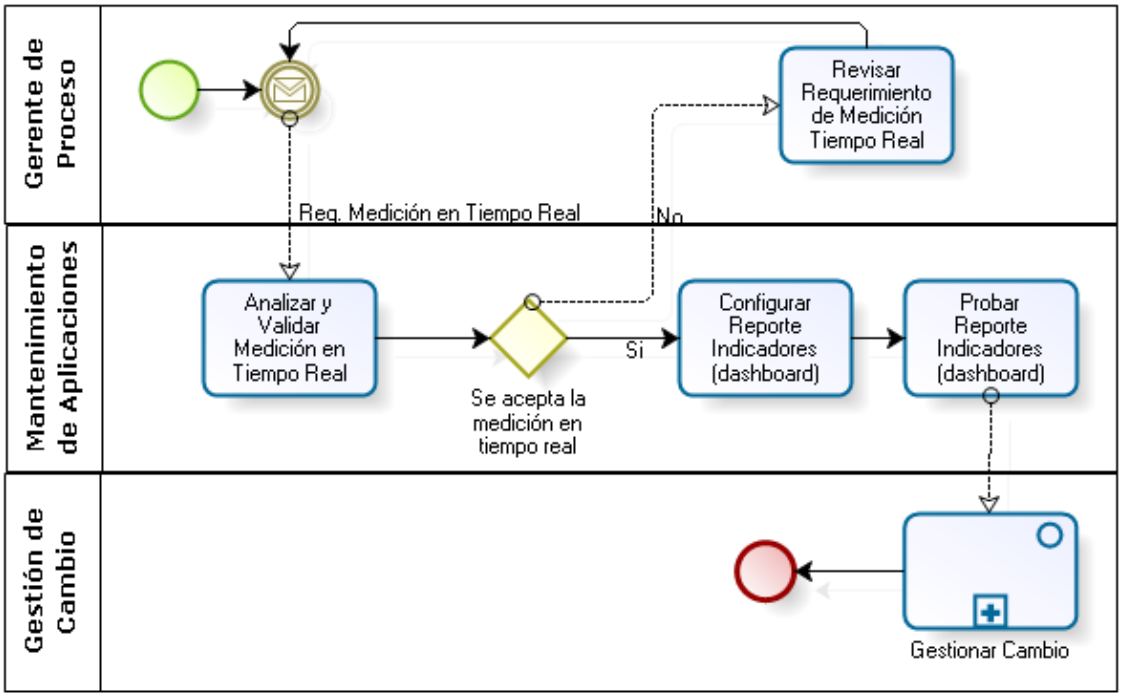


Figura 26: BPD del sub-proceso Monitorear Funciones

Fuente: Bonillo 2008

El último de los sub-proceso es el de **Monitorear Plataforma** (ver, Figura 28), inicia cuando el área de Mantenimiento de Aplicaciones solicita al área de Monitoreo y Control, analizar y validar la plataforma de hardware y software del BPMS (estas validaciones se refieren a espacio en disco, memoria utilizada, procesos, etc.), el área de Monitoreo control acepta la validación de la plataforma, realiza una configuración y prueba e inicia el sub-proceso **Gestionar Cambio**, en el caso de no aceptar la solicitud le notifica al área de Mantenimiento de Aplicaciones para que realice los ajustes en el requerimiento.

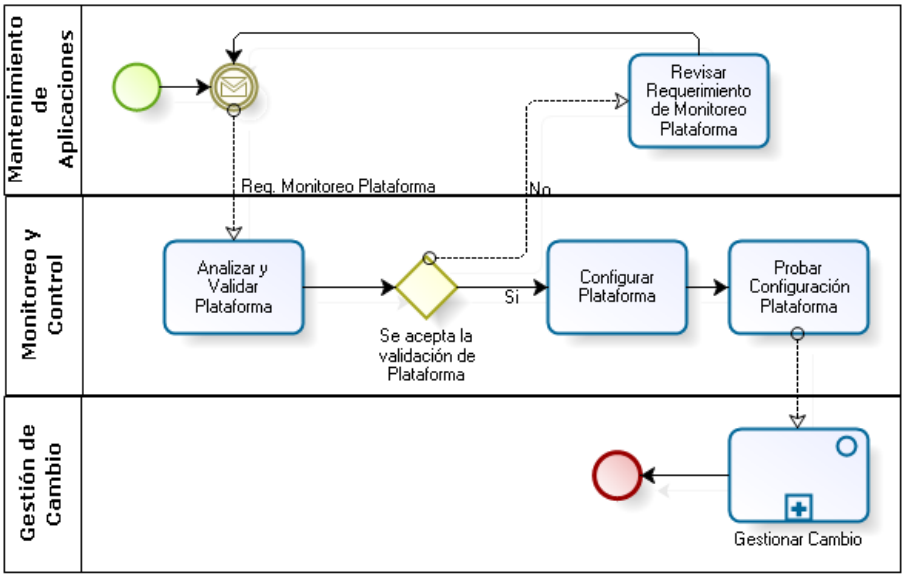


Figura 27: BPD del sub-proceso Monitorear Plataforma

Fuente: Bonillo 2008

Retroalimentación

Esta es la etapa final de la metodología, una vez puestos en producción los procesos de negocio, estos son analizados con el objetivo de obtener mediciones sobre los atributos de calidad seleccionados durante el sub-proceso Configurar, de tal forma de obtener conclusiones relevantes en cuanto al comportamiento de los mismo, y en especial lograr gestionar el conocimiento referente al uso de los patrones y sus relaciones.

Para la implementación de dicha metodología es necesario conocer las características de las tecnologías usadas para el desarrollo del sistema de control y seguimiento de la metodología de gestión de procesos de negocio sustentada en el uso de patrones, que será enunciado a continuación.

CAPÍTULO III - MARCO METOLÓGICO

En este Capítulo se describe la metodología utilizada para realizar la investigación y es por lo que se hace necesario señalar: metodología de desarrollo de software, diseño de la investigación, el tipo de estudio, así como también la selección de la población, la elaboración del instrumento, recolección de datos y análisis de los mismos. “La metodología constituye la médula del plan, se refiere a la descripción de las unidades de análisis, o de investigación, las técnicas de observación y recolección de datos, los instrumentos, los procedimientos y las técnicas de análisis”. (Tamayo, 1997).

3.1 Metodología de Desarrollo de Software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software.

Una metodología puede seguir uno o varios modelos de ciclo de vida, es decir, el ciclo de vida indica qué es lo que hay que obtener a lo largo del desarrollo del proyecto pero no cómo hacerlo. “La metodología indica cómo hay que obtener los distintos productos parciales y finales”.

Al momento de elegir una metodología general para desarrollar software se debe tener en cuenta aquella que permita en todas sus fases la fácil adaptabilidad a cualquier tipo de requerimientos que el desarrollo exija.

El ciclo o proceso de desarrollo de sistemas de información a lo largo de los años ha madurado considerablemente, aprendiendo de los errores del pasado e incorporando cada día mejores prácticas y herramientas en pro de la satisfacción del cliente, que es el objetivo final de cualquier proyecto. Dentro de esta línea de

crecimiento y madurez existe como punta de lanza, las metodologías llamadas Ágiles. Por lo cual, se entiende como Desarrollo ágil de Software a un paradigma de Desarrollo de Software basado en procesos ágiles. Los procesos ágiles de desarrollo de software, conocidos anteriormente como metodologías livianas, intentan mejorar la eficiencia y eliminar la burocracia en los pasos de las metodologías tradicionales enfocándose en la gente y los resultados, para reducir los tiempos y costos de desarrollos, pero manteniendo la calidad de los productos.

En la actualidad se plantean proyectos donde se requieren que el resultado final sea un producto de gran calidad en el menor tiempo posible, por esta razón surgen las metodologías ágiles, las cuales se basan en el desarrollo incremental del software con iteraciones muy cortas, adaptándose de una mejor manera a los constantes cambios que se van presentando durante el desarrollo.

Un proceso de desarrollo de software iterativo es aquel que se representa por una serie de tareas agrupadas en pequeñas etapas repetitivas, conocidas como iteraciones. Para cada una de las etapas que conformen el desarrollo final de la aplicación, se define una serie de pasos para completarla, estos pasos son usados en los modelos de desarrollos comunes y se conocen como Modelo, Implementación, Pruebas, Despliegue, Gestión de Configuración, Gestión y Entorno.

El desarrollo de la aplicación *Sistema para el Control y Seguimiento de la Implementación de la Metodología de Gestión de Procesos de Negocio sustentada en el Uso de Patrones* se basa en la metodología de desarrollo ágil (Proceso Unificado Ágil - AUP), donde se definen una serie de iteraciones, en las cuales se cumplen cada uno de los pasos nombrados anteriormente, para el caso del desarrollo del presente sistema solo se tomarán las disciplinas de Modelo, Implementación y Pruebas, que se detallarán a continuación.

3.1.1 Proceso Unificado Ágil (AUP)

Es una versión simplificada de Proceso Unificado de Negocio (*Unified Process - UP*). El enfoque aplica técnicas de prueba ágil impulsando el desarrollo,

Dirigida por Modelos de Desarrollo Ágil, la gestión del cambio ágil, y la base de datos de refactorización para mejorar su productividad.

AUP se preocupa especialmente de la gestión de riesgos. Propone que aquellos elementos con alto riesgo obtengan prioridad en el proceso de desarrollo y sean abordados en etapas tempranas del mismo. Para ello, se crean y mantienen listas identificando los riesgos desde etapas iníciales del proyecto. Especialmente relevante en este sentido es el desarrollo de prototipos ejecutables durante la base de elaboración del producto, donde se demuestre la validez de la arquitectura para los requisitos clave del producto y que determinan los riesgos técnicos.

3.1.2 Ciclo de Vida de UP Ágil

Fases

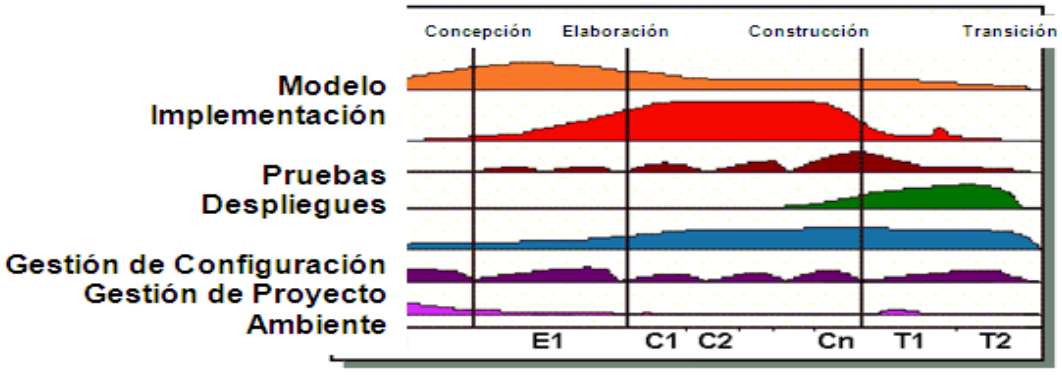


Figura 28: Ciclo de Vida de Ágil UP

Fuente: http://www.ambysoft.com/unifiedprocess/agileUP.html

Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva y que acaban con hitos claros alcanzados:

- **Concepción:** El objetivo de esta fase es obtener una comprensión común cliente-equipo de desarrollo del alcance del nuevo sistema y definir una o varias arquitecturas candidatas para el mismo.

- **Elaboración**: El objetivo es que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema y en validar la arquitectura.
- **Construcción**: Durante la fase de construcción el sistema es desarrollado y probado al completo en el ambiente de desarrollo.
- **Transición**: el sistema se lleva a los entornos de preproducción donde se somete a pruebas de validación y aceptación y finalmente se despliega en los sistemas de producción.

El proceso AUP establece un Modelo más simple que el que aparece en RUP por lo que reúne en una única disciplina, las disciplinas de Modelado de Negocio, Requisitos, Análisis y Diseño. El resto de disciplinas (Implementación, Pruebas, Despliegue, Gestión de Configuración, Gestión y Entorno) coinciden con las restantes de RUP.

Las disciplinas se llevan a cabo de manera sistemática, a la definición de las actividades que realizan los miembros del equipo de desarrollo a fin de desarrollar, validar, y entregar el software de trabajo que satisface las necesidades de sus interlocutores. Las disciplinas son:

- **Modelo:** El objetivo de esta disciplina es entender el negocio de la organización, el problema de dominio que se abordan en el proyecto, y determinar una solución viable para resolver el problema de dominio.

- **Aplicación:** El objetivo de esta disciplina es transformar su modelo (s) en código ejecutable y realizar un nivel básico de las pruebas, en particular, la unidad de pruebas.

- **Prueba:** El objetivo de esta disciplina consiste en realizar una evaluación objetiva para garantizar la calidad. Esto incluye la búsqueda de defectos, validar que el sistema funciona tal como está establecido, y verificando que se cumplan los requisitos.

- **Despliegue:** El objetivo de esta disciplina es el plan para la prestación del sistema y ejecutar el plan para que el sistema a disposición de los usuarios finales.

- **Gestión de la configuración:** El objetivo de esta disciplina es la gestión de acceso a artefactos de su proyecto. Esto incluye no sólo el seguimiento de las versiones artefacto con el tiempo, sino también el control y gestión del cambio para ellos.

- **Gestión de Proyectos**: El objetivo de esta disciplina es dirigir las actividades que lleva a cabo en el proyecto. Esto incluye la gestión de riesgos, la dirección de personas (la asignación de tareas, el seguimiento de los progresos, entre otros), y de coordinación con el personal y los sistemas fuera del alcance del proyecto para asegurarse de que es entregado a tiempo y dentro del presupuesto.
- **Medio Ambiente**: El objetivo de esta disciplina es apoyar el resto de los esfuerzos por garantizar que el proceso adecuado, la orientación (normas y directrices), y herramientas (hardware, software, entre otros) están disponibles para el equipo según sea necesario.

3.2 Tipo de Investigación

De acuerdo Arias (2004), el tipo de investigación se refiere al "grado de profundidad con que se aborda un fenómeno u objeto de estudio". El tipo de ésta investigación es de carácter del Tipo Proyecto Factible.

Los pasos propuestos por esta metodología representaran las cinco fases que rigen un Proyecto Factible:

- Diagnostico de necesidades.
- La Formulación de la propuesta metodológica.

- Información de la Empresa, Usuarios del Sistema, del Proceso y del Medio Ambiente. Recopilando la información por algunos de los siguientes métodos: entrevistas, cuestionarios, observación y muestreo.
- El análisis de la factibilidad del proyecto.
- Luego de la investigación y el análisis se debe diseñar e implementar el desarrollo, verificando, entrenando usuarios y realizando pruebas respectivas por módulos realizados.
- Implantarla y seguir realizando pruebas, para que el usuario final lo acepte.
- Finalmente, seguir controlando y realizando el mantenimiento previo del producto.

De acuerdo al problema planteado, referido a la necesidad de realizar un desarrollo y modelado eficiente del proceso de Metodología de Gestión de Proceso sustentada en el uso de Patrones.

3.2 Diseño de la Investigación

El diseño de la investigación que sustenta este proyecto de tesis de pregrado será el de Campo, ya que se recolectarán datos obtenidos directamente de la realidad, lo que permitirá que la investigación adquiera gran valor debido a la posibilidad de levantar la información necesaria para el diseño de la metodología. Dado que, "consiste en la recolección de datos directamente de la realidad donde ocurren los hechos, sin manipular y controla variable alguna". (Arias, 2004).

El diseño de esta investigación también es denominado documental, dado que se sustentara en investigaciones de otros autores cuya fuente de información siempre procede de documentos escritos, pues esa es la forma uniforme en que se emiten los informes científicos.

3.3 Población y Muestra

Para Tamayo y Tamayo, una población está determinada por sus características definitorias, por tanto, el conjunto de elementos que posea esta características se denominará población, la cual es la totalidad del fenómeno a estudiar, en donde las unidades poseen una característica común, que se estudia y da origen a los datos de la investigación. (Tamayo, 2000).

Dado que la investigación pretende presentar una metodología, la muestra deberá ser definida de tal forma de poder abstraer características que permitan su aplicación y estudio a otras organizaciones ofreciendo conclusiones pertinentes.

La población de esta investigación de pregrado se corresponde con los procesos de negocio de las empresas actuales en diferentes dominios. Dado que esta población es muy amplia es necesario reconocer un dominio específico, el cual se enmarcará a una *población finita*, debido a que este depende de los *stakeholders* (personas que pueden afectar o son afectados por las actividades de una empresa) implicados en el proceso de Metodología de Gestión de Procesos sustentada en el uso de Patrones de Negocio, tales como el Analista de Negocio, Supervisor de Negocio y todas las personas involucradas en el proceso.

Básicamente las muestras se dividen en dos grandes ramas: las muestras no probabilísticas y las muestras probabilísticas. En esta última todos los elementos de la población tienen la misma posibilidad de ser seleccionados. En las muestras no probabilísticas, la elección no depende de la probabilidad, sino de las causas relacionadas con las características del investigador o del que hace la muestra. Aquí el procedimiento no es mecánico, ni en base a fórmulas de probabilidad, sino que depende del proceso de toma de decisiones de una persona o grupo de personas.

En particular, esta investigación de pregrado busca muestras no probabilísticas, debido a que suponen un procedimiento de selección informal y un proceso arbitrario. La ventaja de este tipo de muestra es su utilidad para un

determinado diseño de estudio, que requiere no tanto de una representatividad de los elementos de la población, sino de una cuidadosa y controlada selección de sujetos con ciertas características especificadas previamente en el planteamiento del problema.

La muestra será establecida de forma tal que abarque todo el dominio de las personas involucradas con la Metodología de Gestión de Procesos sustentada en el Uso de Patrones, bien sea directa o indirectamente.

3.4 Técnicas e Instrumentos de Recolección de Datos

La técnica e instrumento de recolección de datos de datos empleada en esta investigación de pregrado será: la revisión bibliográfica, y la infografía, las cuales se describen a continuación:

- Revisión Bibliográfica: Se debe recurrir a la técnica de revisión bibliográfica; tanto de libros, folletos, documentos, revistas, artículos y seminarios, los cuales vienen a brindarle al investigador todo el soporte del marco teórico, lo que significa que se percata de todo lo escrito o que esté relacionado con el tema que escogió como investigación. La Revisión Bibliográfica, se utiliza como base complementaria a la investigación central, con el fin de recopilar y revisar todos aquellos documentos que permitan confrontar el aspecto teórico con la situación real o práctica dentro del modelo de evaluación de las capacidades tecnológicas necesarias en el mercado de la gerencia de procesos de negocio a través del uso de patrones. Es importante señalar que esta revisión se efectúa antes y durante la investigación, con el objetivo de contrastar información, obtener nuevas ideas, indagar la naturaleza de los datos y realizar nuevas conclusiones.
- Fuentes Infográficas: fue el instrumento de mayor uso durante el desarrollo del Trabajo Especial de Grado. Estas fuentes comprendía amplios medios en línea para recopilar información como webinars, foros, documentaciones, etc.

CAPÍTULO IV - MARCO APLICATIVO

Los entregables del Trabajo Especial de Grado corresponden a los que establece el macro-proceso “Crear Proceso” de la Metodología de Gerencia de Procesos sustentada en el Uso de Patrones propuesta por el Dr. Pedro Bonillo para el modelado de los procesos.

4.1 Fase de Modelado

Esta etapa corresponde a entender el negocio, dominio del problema que el proyecto aborda e identificar las soluciones, es decir, la respectiva investigación realizada para llevar a cabo esta investigación. Esto implicó indagar sobre temas como Gestión de Procesos de Negocio, tecnologías web, evaluación de la herramienta BPM Intalio y otros aspectos relacionados a la notación BPMN.

La siguiente figura muestra el nivel 0, el cual explica que un usuario desea interactuar con el sistema.

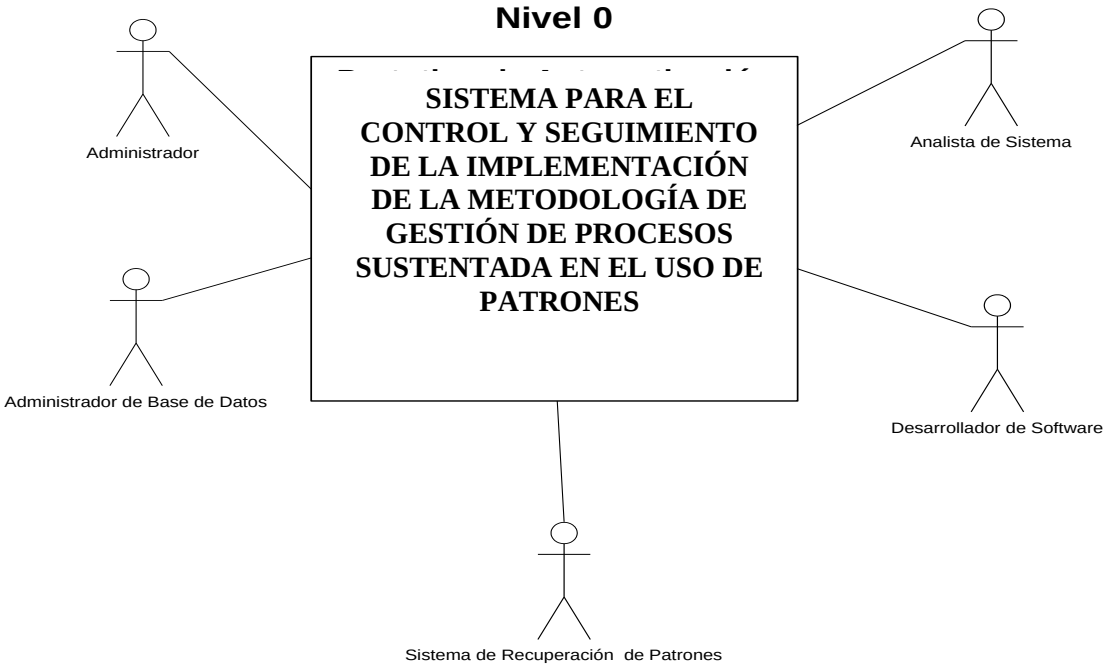


Figura 29: Nivel 0 del Sistema para el Control y Seguimiento de la M.G.P.N.S.U.P.

4.1.1 Iteración 1: Análisis de Requerimientos Funcionales y No Funcionales

Un requerimiento define como debe comportarse los sistemas.

Análisis de Requerimientos Funcionales

Los requerimientos funcionales definen, como el sistema interacciona con su entorno. A continuación se listan los requerimientos funcionales del Sistema para el Control y Seguimiento de la Metodología de Gestión de Proceso de Negocio sustentada en el Uso de Patrones

- Identificar requerimientos del proceso.
- Definir roles.
- Levantar Situación Actual del Proceso
- Conectarse con el Sistema Repattern y Recuperar los Patrones de Análisis.
- Recuperar los Estilos Arquitecturales y Patrones de Arquitecturas.
- Definir Riesgos y Brechas
- Definir Cartelera de Actividades.
- Elaborar la lógica de negocio.
- Modelar y Diseñar los procesos.
- Mantener y Monitorear los procesos.

Los requerimientos no funcionales definen las restricciones del sistema. Los requerimientos no funcionales son:

- Requisito de Look & Feel
  - Desarrollar las interfaces del sistema, se desarrolla siguiendo los lineamientos establecidos por C.A.N.T.V
- Requisitos de Usabilidad
  - Establecer un calendario al momento de colocar la fecha de nacimiento o cualquier otra fecha.
  - Permitir crear usuarios

- Permitir avanzar e ir hacia atrás entre una página y otra.
- Evitar el uso del teclado por parte de los usuarios.

- Portabilidad
  - El sistema puede ser ejecutado desde el Sistema Operativo Windows o desde el Sistema Operativo Linux.
  - Soporte de Navegador Mozilla Firefox e Internet Explorer.

- Requisitos de Seguridad
  - El sistema permite almacenar la clave secreta en un formato encriptado.

4.1.2 Iteración 2: General

En la primera iteración se hace el análisis de requerimientos de lo que debe cumplir la aplicación, mientras que las iteraciones siguientes están basadas en el resultado de las pruebas realizadas en las iteraciones pasadas.

En el nivel 1 de los casos de uso mostrado en la Figura 39, se establece la funcionalidad que tendrá el sistema, especificando la relación entre los usuarios y la aplicación mediante casos de usos.

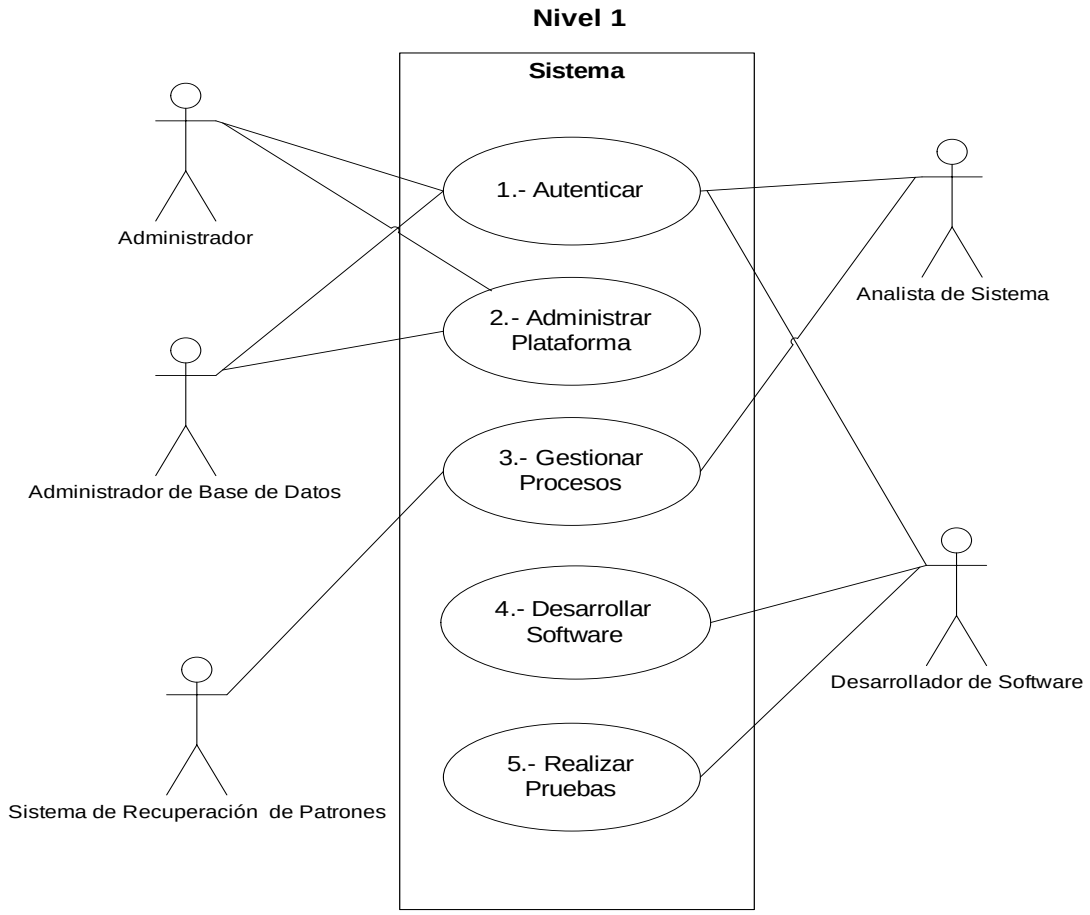


Figura 30: Nivel 1 Interacción de los usuarios con la funcionalidades del sistema

A continuación se describe las especificaciones de los casos de usos en este nivel.

Tabla 2: Especificar Caso de Uso Nivel 1- Autenticar

Nombre	Autenticar
Actor Involucrado	Administrador, Administrador de Base de Datos, Analista de Sistema, Desarrollador de Software
Pre- Condición	Ninguna

Descripción	El usuario introduce un login y una contraseña para intentar acceder al sistema. El sistema valida los datos introducidos, pudiendo ocurrir los siguientes eventos: Los datos introducidos no se encuentran registrados en la base de datos y/o son inválidos, por lo tanto, se niega el acceso al sistema y se muestra el respectivo mensaje de error. Los datos introducidos son correctos y se permite el acceso al sistema, específicamente al menú principal de acuerdo al perfil del usuario.
Post- Condición	De ser exitosa la verificación de datos, el usuario puede interactuar con el sistema.

Tabla 3: Especificar Caso de Uso Nivel 1- Administrar Plataforma

Nombre	Administrar Plataforma
Actor Involucrado	Administrador, Administrador de Base de Datos
Pre- Condición	El usuario se ha autenticado exitosamente.
Descripción	De acuerdo al tipo de administrador se muestra las opciones a las que tiene ingreso, pudiendo ocurrir los siguientes flujos de información: Si es administrador general, podrá configurar la interfaz que se le muestra al usuario, configuración de variables de entorno asociadas al servidor, carpetas donde se almacena la información y el acoplamiento de las distintas herramientas de diseño. Si es administrador de base de datos, podrá establecer los perfiles de ingreso al sistema de acuerdo a la información suministrada por el líder o responsable del proyecto. Además, es posible crear o eliminar las diversas estructuras necesarias durante el manejo de datos en un flujo de procesos.
Post- Condición	Cambios realizados en el sistema

Tabla 4: Especificar Caso de Uso Nivel 1- Gestión de Procesos

Nombre	Gestión de Procesos
Actor Involucrado	Analista de Sistema, Sistema de Recuperación de Patrones
Pre- Condición	El usuario se encuentra autenticado y tiene acceso a esta opción mediante su

	perfil de ingreso.
Descripción	En esta fase se podrá implementar la Metodología de Gerencia de Procesos de Negocio sustentada en el uso de Patrones, pudiendo acceder a las siguientes etapas de la metodología, las cuales son: Crear Proceso Administrar Proceso
Post- Condición	Se han supervisado requerimientos

Tabla 5: Especificar Caso de Uso Nivel 1- Desarrollar Software

Nombre	Desarrollar Software
Actor Involucrado	Desarrollador de Software
Pre- Condición	El usuario se encuentra autenticado
Descripción	En esta fase el actor involucrado se encarga de modelar e implementar los procesos establecidos por la empresa
Post- Condición	El proceso se encuentra implementado

Tabla 6: Especificar Caso de Uso Nivel 1- Realizar Pruebas

Nombre	Realizar Pruebas
Actor Involucrado	Desarrollador de Software, Analista de Sistema
Pre- Condición	El usuario se encuentra autenticado y existen procesos en fase de prueba.
Descripción	Esta fase el actor o actores que se encuentran involucrados realizan pruebas a cada módulo finalizado para verificar el acoplamiento adecuado entre cada componente del proceso implementado. Una vez realizado todas las pruebas exitosamente se da el visto bueno para que el proceso pase a producción y pueda ser usado en su respectivo ámbito.
Post- Condición	El proceso es enviado a revisión o autorizado para el pase a producción.

En las siguientes iteraciones se describe el análisis de cada caso de uso en el nivel 2.

4.1.3 Iteración 3: Diseño de Casos de Usos

Se muestra el nivel 2 de los casos de uso mostrado en la Figura 39. En la figura 40 se establece la funcionalidad de logueo de los usuarios en el sistema.

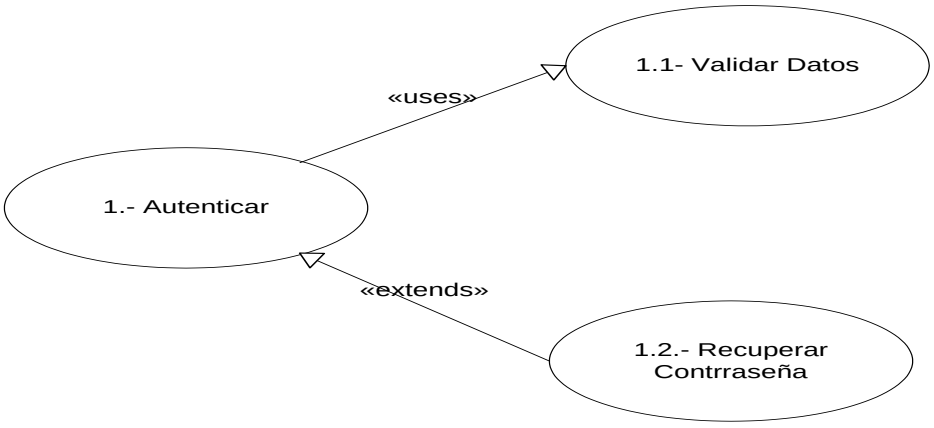


Figura 31 : Nivel 2 Autenticación de los usuarios

En la figura N°41, se administra la plataforma en cuanto a la base de datos y la aplicación para observar si se está ejecutando el proceso de manera correcta.

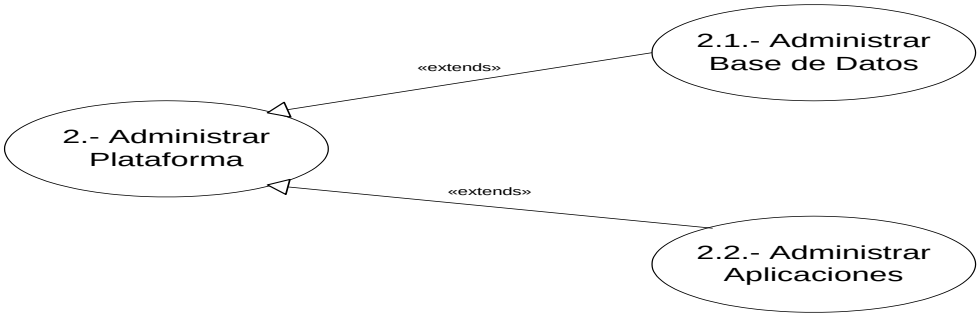


Figura 32: Nivel 2 Administración de la plataforma

La figura 42, se puede observar cómo se despliegan dos procesos el crear y administrar proceso, estos son macros procesos de la metodología de gestión de procesos de negocio sustentada en el uso de patrones.

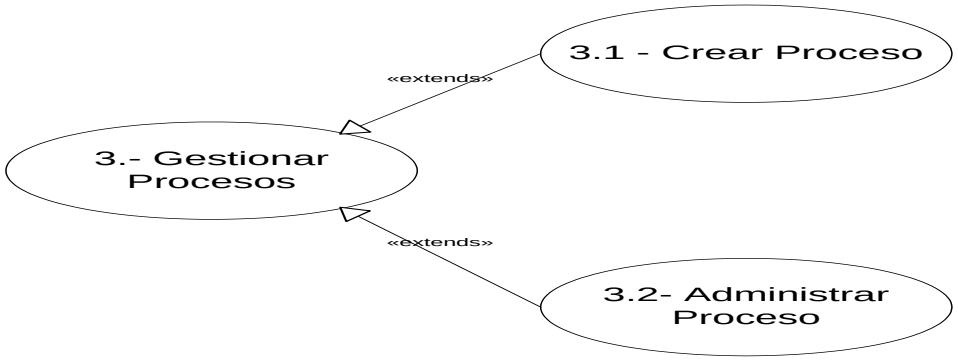


Figura 33: Nivel 2 Gestionar Procesos

A continuación se muestra las especificaciones de los casos de uso correspondiente del nivel 2.

Tabla 8: Especificar Caso de Uso Nivel 2- Autenticar Usuarios

Nombre	Autenticar Usuario
Actor Involucrado	Administrador
Pre- Condición	No tiene
Descripción	Se debe validar que los datos ingresado por los usuarios sean correctos. Y darle la opción de recuperar contraseña si la olvido.
Post- Condición	El proceso es enviado a revisión para solicitar modificación en a base de datos

Tabla 9: Especificar Caso de Uso Nivel 2- Administrar Plataforma

Nombre	Administrar Plataforma
Actor Involucrado	Desarrollador de Software, Analista de Sistema
Pre- Condición	Usuario debe estar logueado y módulo crear proceso estar ejecutado
Descripción	Esta fase el actor o actores que se encuentran involucrados realizan pruebas

	para mantener aplicaciones y base de datos
Post- Condición	Seguimiento de los procesos usando indicadores claves de desempeño.

Tabla 10: Especificar Caso de Uso Nivel 2- Gestionar Proceso

Nombre	Gestionar Procesos
Actor Involucrado	Desarrollador de Software, Analista de Sistema
Pre- Condición	El usuario se encuentra autenticado y existen procesos en fase de prueba.
Descripción	Esta fase el actor o actores que se encuentran involucrados realizan pruebas a cada módulo finalizado para verificar el acoplamiento adecuado entre cada componente del proceso implementado. Una vez realizado todas las pruebas exitosamente se da el visto bueno para que el proceso pase a producción y pueda ser usado en su respectivo ámbito.
Post- Condición	El proceso es enviado a revisión o autorizado para el pase a producción.

Según las funcionalidades de los casos de usos, se procede a diseñar la estructura que mejor se adapte a sus necesidades. Se puede observar mediante los siguientes diagramas:

4.1.4 Iteración 3: Diseño de Casos de Usos Nivel 3

En esta iteración se puede visualizar los casos de usos referentes al nivel 3, donde los macros procesos son desplegados en subprocesos. En la siguiente figura se observa el proceso crear proceso el cual se desplegó en subprocesos analizar, diseñar, modelar y configurar.

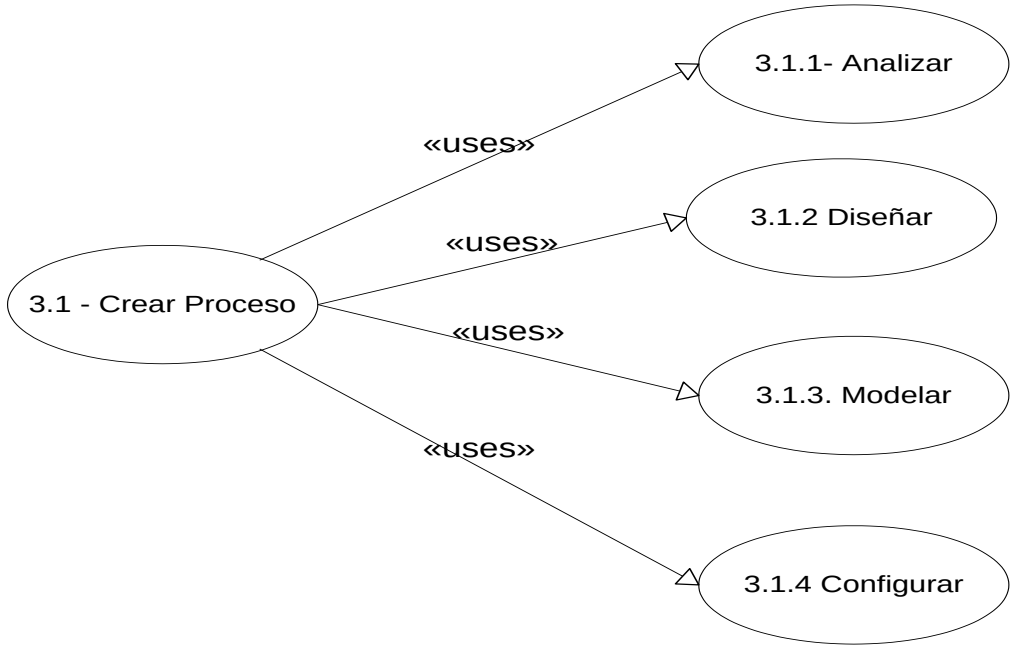


Figura 34: Nivel 3 Crear Procesos

El caso de uso que se muestra en la parte inferior detalla como el administrar proceso es desplegado.

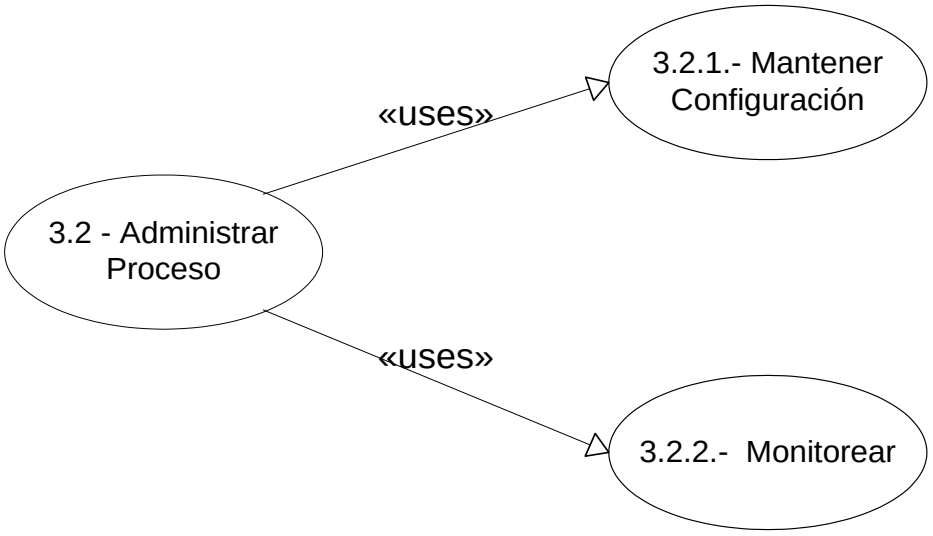


Figura 35: Nivel 3 Administrar Procesos

A continuación se muestra las especificaciones de los casos de uso correspondiente del nivel 3.

Tabla 11: Especificar Caso de Uso Nivel 3- Crear Procesos

Nombre	Crear Proceso
Actor Involucrado	Analista
Pre- Condición	Usuario se autenticó correctamente
Descripción	Se realiza análisis, diseño, modelo y configuración del proceso.
Post- Condición	Módulo proceso Crear Proceso Automatizado.

Tabla 12: Especificar Caso de Uso Nivel 3- Administrar Plataforma

Nombre	Administrar Plataforma
Actor Involucrado	Supervisor
Pre- Condición	Debe estar finalizado módulo crear proceso
Descripción	E realiza configuraciones y se chequean constantemente los procesos.
Post- Condición	Seguimiento de los procesos usando indicadores claves de desempeño. Y metodología automatizada.

4.1.5 Iteración 4: Diseño de Casos de Usos Nivel 4

A continuación se muestra como el proceso analizar es desplegado en dos subprocesos Identificar Requerimientos y Levantar Información.

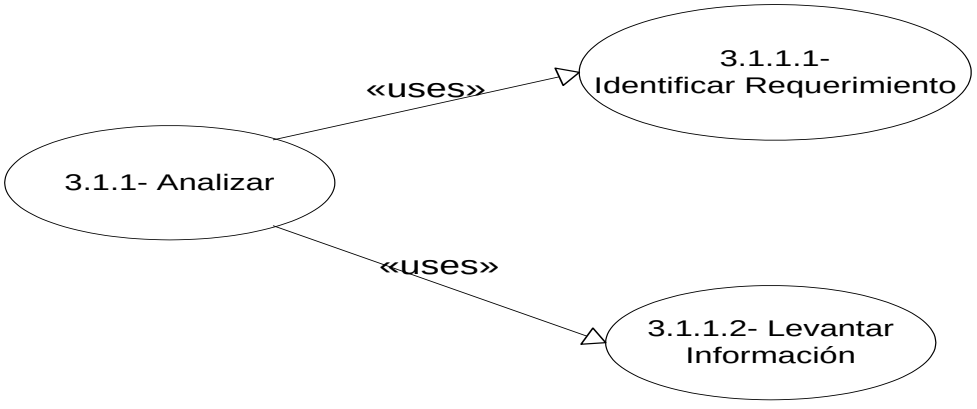


Figura 36: Despliegue de Analizar

El siguiente caso de usos del presente nivel es el proceso diseñar, donde se estandarizan los procesos, se evalúan los posibles riesgos, se propone una propuesta y se realiza el setup del laboratorio.

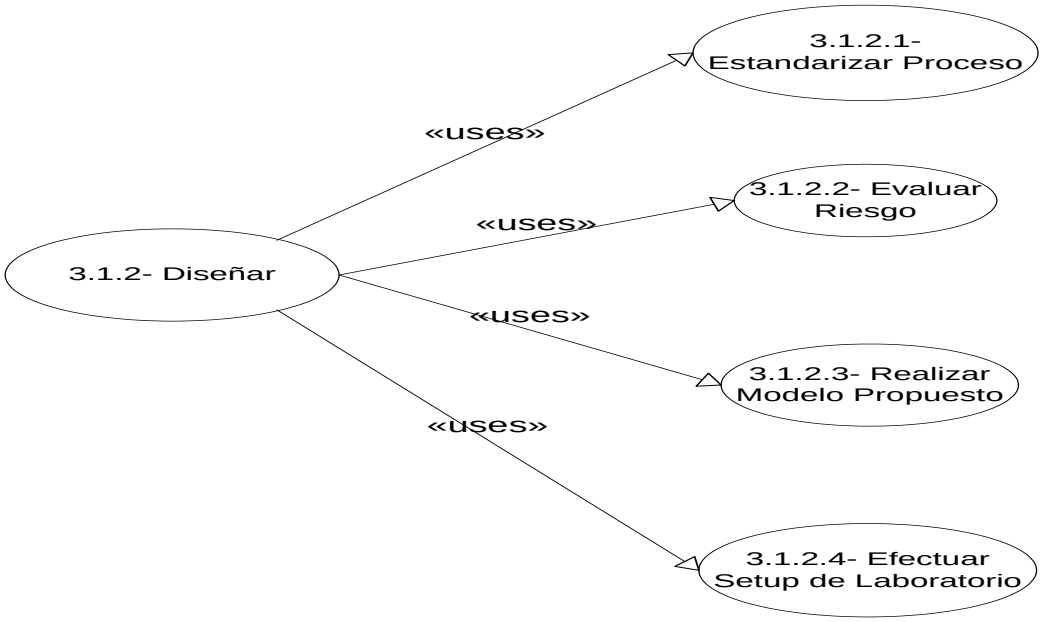


Figura 37: Despliegue de Diseñar

Seguidamente observamos la figura del caso de uso Modelar, donde se crean subprocesos para diagramar, realizar pruebas funcionales, integrar con otras arquitecturas y simular.

La figura a continuación muestra el proceso configurar, donde se definen una serie de funcionalidades de este proceso.

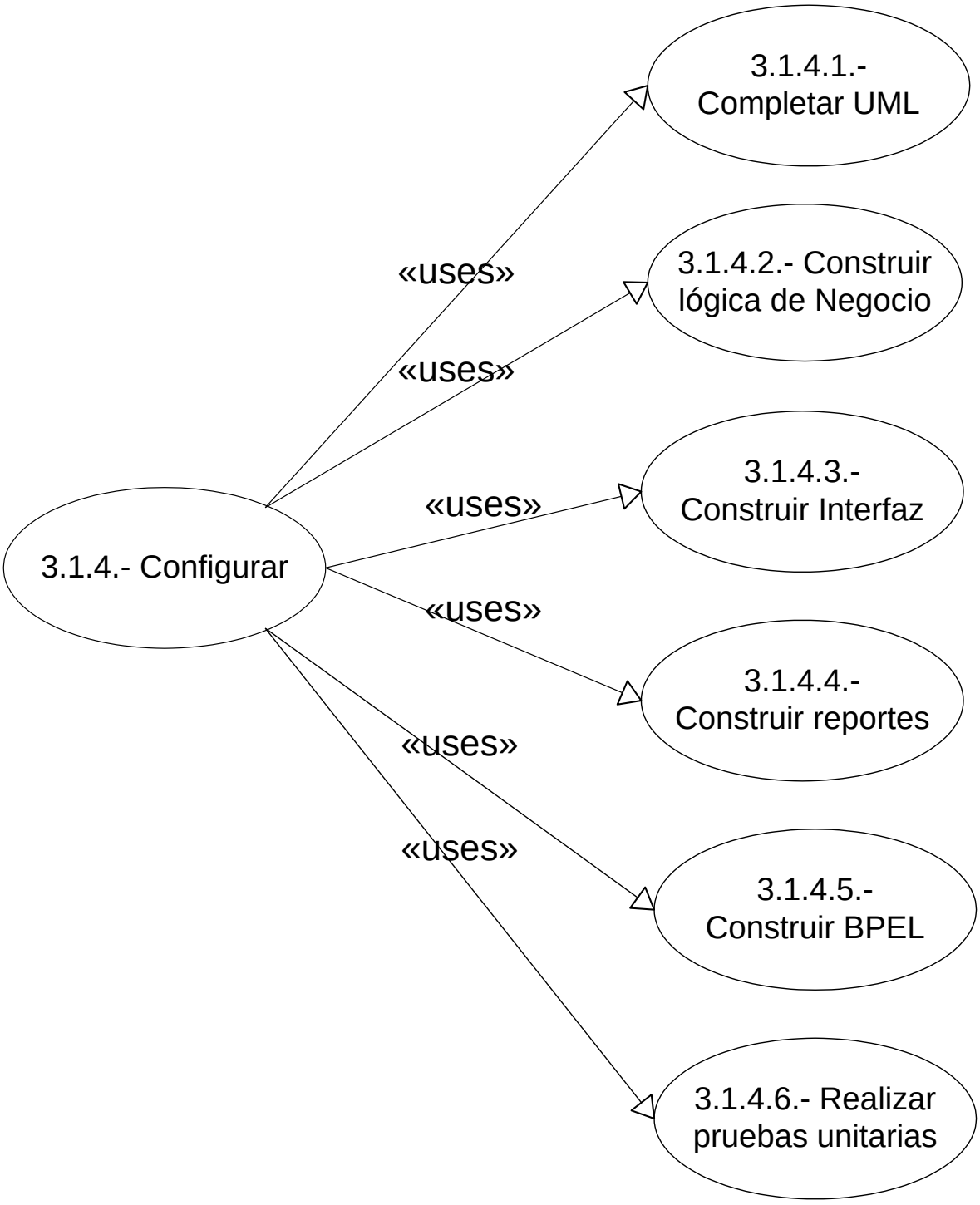


Figura 38: Despliegue Proceso Configurar

Luego de configurar se prosigue a mantener la configuración de las aplicaciones, variables de procesos, y plataforma.

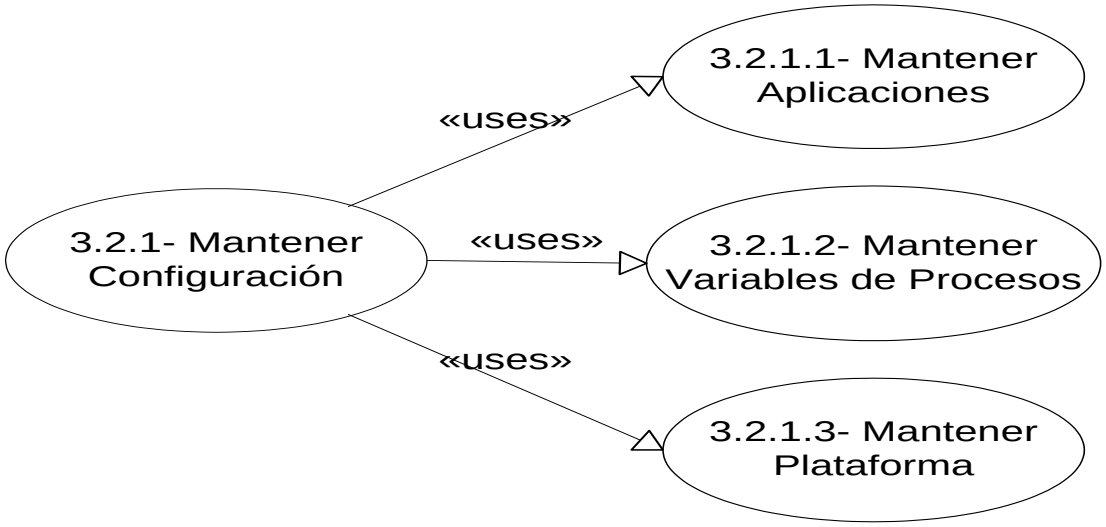


Figura 39: Despliegue Mantener Configuración

Realizado el seguimiento a las configuraciones respectivas se procede a chequer como el proceso se esta comportando, como se puede observar en la siguiente figura.

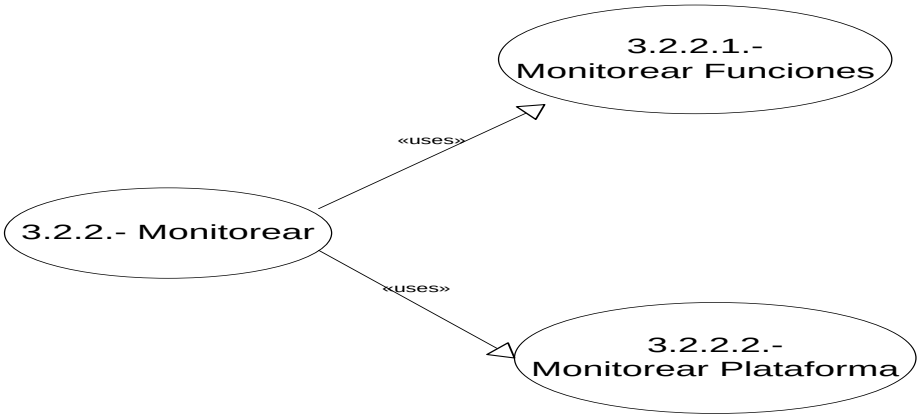


Figura 40: Despliegue de Monitorear

A continuación se muestra las especificaciones de los casos de uso referente al nivel 4. En la siguiente tabla se muestra las especificaciones de Analizar.

Tabla 7: Especificar Caso de Uso Nivel 4- Analizar

Nombre	Analizar
Actor Involucrado	Analista, Usuario
Pre- Condición	Usuario del sistema logueado
Descripción	Se identifica los requerimientos y se hace el levantamiento de la información del proceso.
Post- Condición	Se debe hacer análisis de requisitos.

Tabla 8: Especificar Caso de Uso de Nivel 4- Diseñar

Nombre	Diseñar
Actor Involucrado	Analista
Pre- Condición	Definidos los requerimientos del sistema
Descripción	Se debe estandarizar el proceso, evaluar los posibles riesgos, realizar un modelo y configurar máquinas donde se realizará desarrollo.
Post- Condición	Se debe tener un modelo diseñado en papel

Tabla 9: Especificar Caso de Uso Nivel 4 -Modelar

Nombre	Modelar
Actor Involucrado	Analista
Pre- Condición	Diseño en papel
Descripción	Se realiza diagramado, se van realizando pruebas y se integra con otra arquitectura
Post- Condición	Proceso diagramado en el modelador

Tabla 10: Especificar Caso de Uso Nivel 4 Configurar

Nombre	Configurar
Actor Involucrado	Administrador
Pre- Condición	Diseño en modelador de proceso
Descripción	Se debe completar UML, se construye toda la lógica del negocio interfaz, bpel, reportes y se hacen pruebas unitarias.
Post- Condición	Se debe tener un modelo diseñado en papel

4.1.6 Iteración 5: Diseño de Casos de Usos Nivel 5

En el caso de uso Identificar Requerimiento se procede a llenar una serie de datos de los requerimientos, luego se le da una prioridad y se realiza una cartelera de actividades estableciendo días para el desarrollo del proceso como lo observan en la siguiente figura.

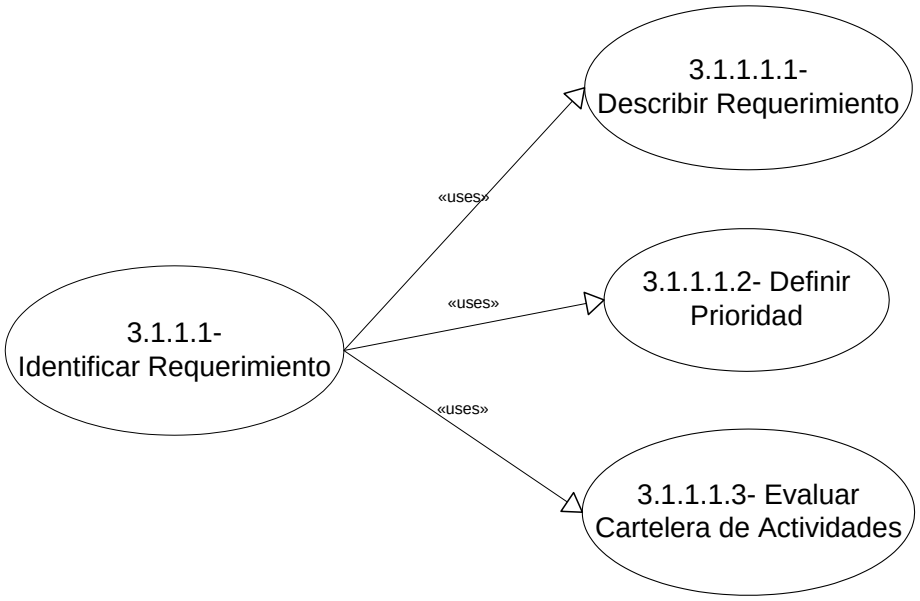


Figura 41: Identificar Requerimientos

Luego de describir el proceso s prosigue con levantar la información actual del proceso y mapearlo con un marco referencial.

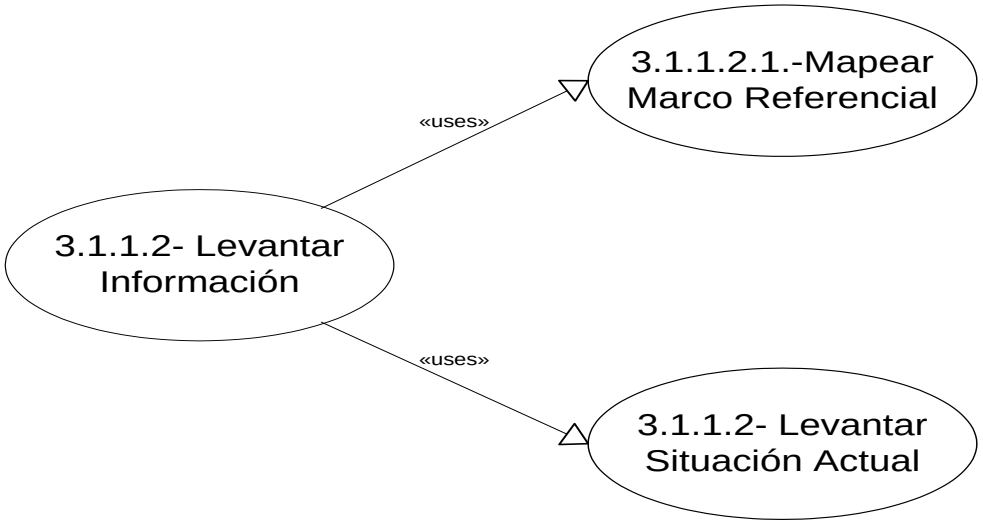


Figura 42: Levantar Información

En la figura presentada en la parte inferior se observa que para mantener la aplicación es necesario validar los tipos de requerimientos, diseñarlos y realizar pruebas para verificar que se esté ejecutando correctamente.

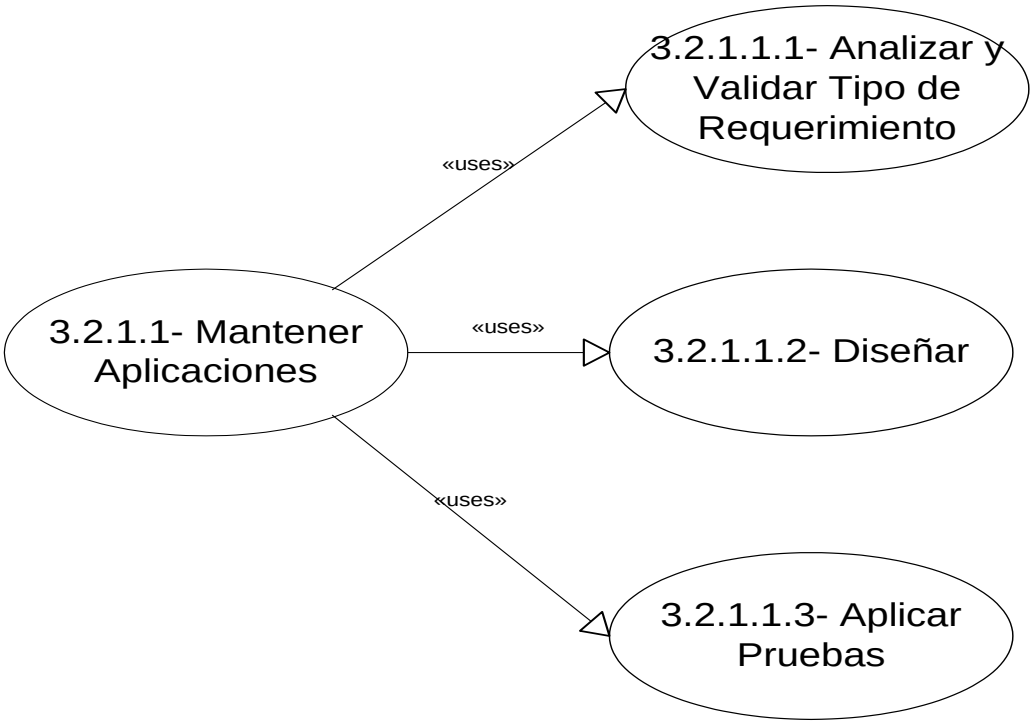


Figura 43: Mantener Aplicaciones

Se prosigue con la especificación de los casos de usos referentes al nivel 5. El primero es Identificar Requerimiento.

Tabla 11: Especificar caso de uso Identificar Requerimiento

Nombre	Identificar Requerimiento
Actor Involucrado	Usuario, Analista
Pre- Condición	Usuario logueado
Descripción	Se debe describir los requerimientos, definir prioridad y evaluar cartelera de actividades
Post- Condición	Usuario registrado

Tabla 12: Especifica Levantamiento de Información

Nombre	Levantar Información
Actor Involucrado	Analista, Usuario
Pre- Condición	Requerimientos definidos
Descripción	Luego de obtenido la información de los requerimientos se debe mapear

	con un marco referencial y se debe levantar la situación actual del proceso
Post- Condición	Se debe tener los requerimientos definidos

Tabla 13: Especificar caso de usos Mantener Aplicaciones

Nombre	Mantener Aplicaciones
Actor Involucrado	Administrador, Supervisor
Pre- Condición	Creado el módulo aplicaciones
Descripción	Se debe completar UML, se construye toda la lógica del negocio interfaz, bpel, reportes y se hacen pruebas unitarias.
Post- Condición	Se debe tener automatizado el módulo crear proceso

4.1.7 Iteración 5: Diagramas de UML

Luego de finalizar el diagramado de los casos de usos se procede a diagramar las clases.

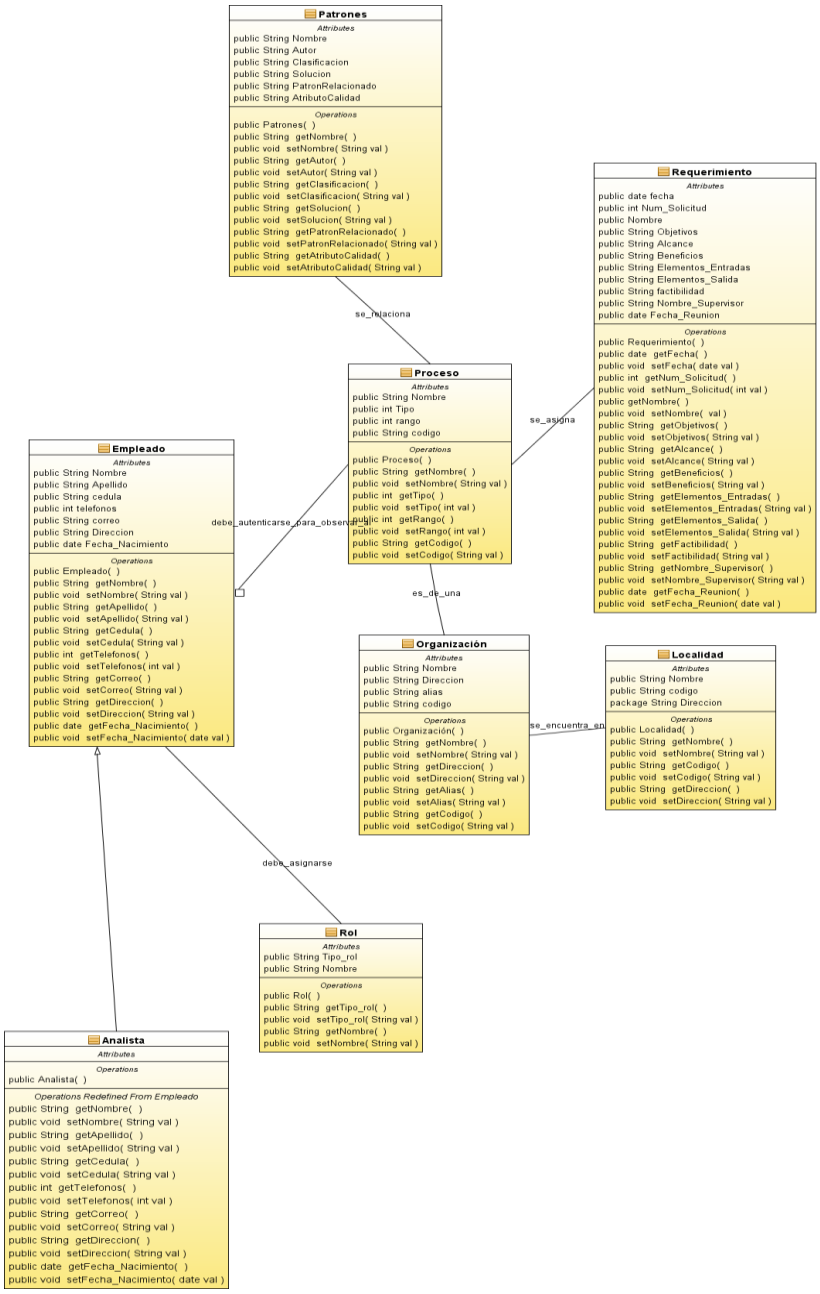


Figura 44: Diagrama de Clases

A continuación el modelo de datos del sistema para el manejo de la lógica dentro de la base de datos.

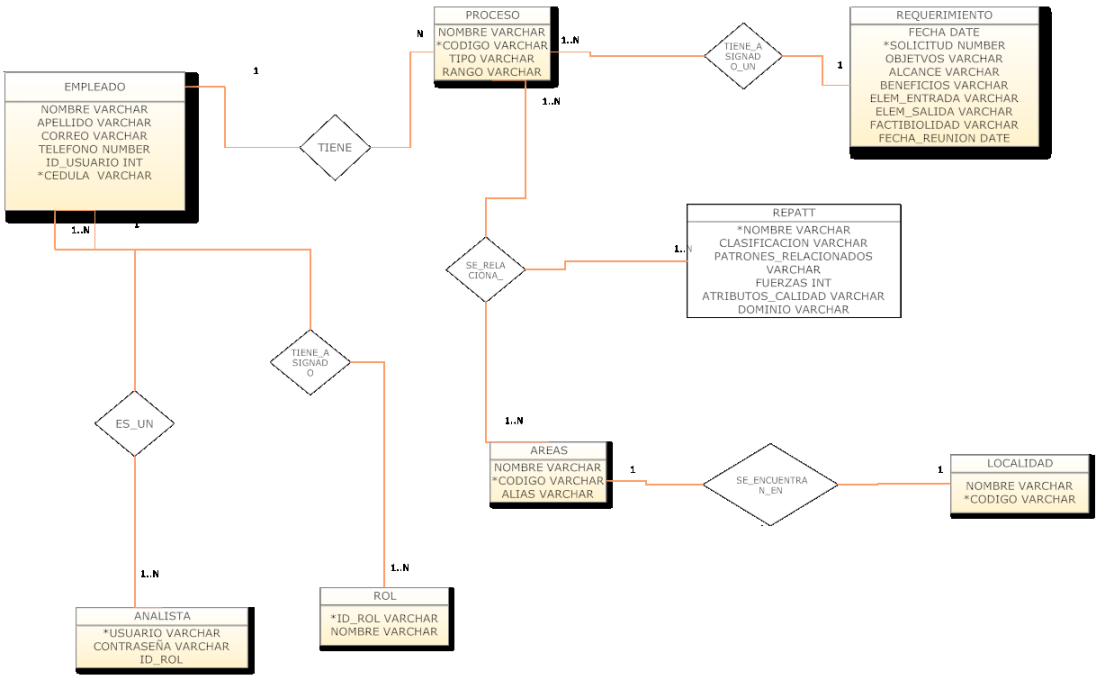


Figura 45: Entidad - Relación

A continuación se muestra el diagrama de secuencias usado para el analizar la interacción entre objetos del sistema a desarrollar.

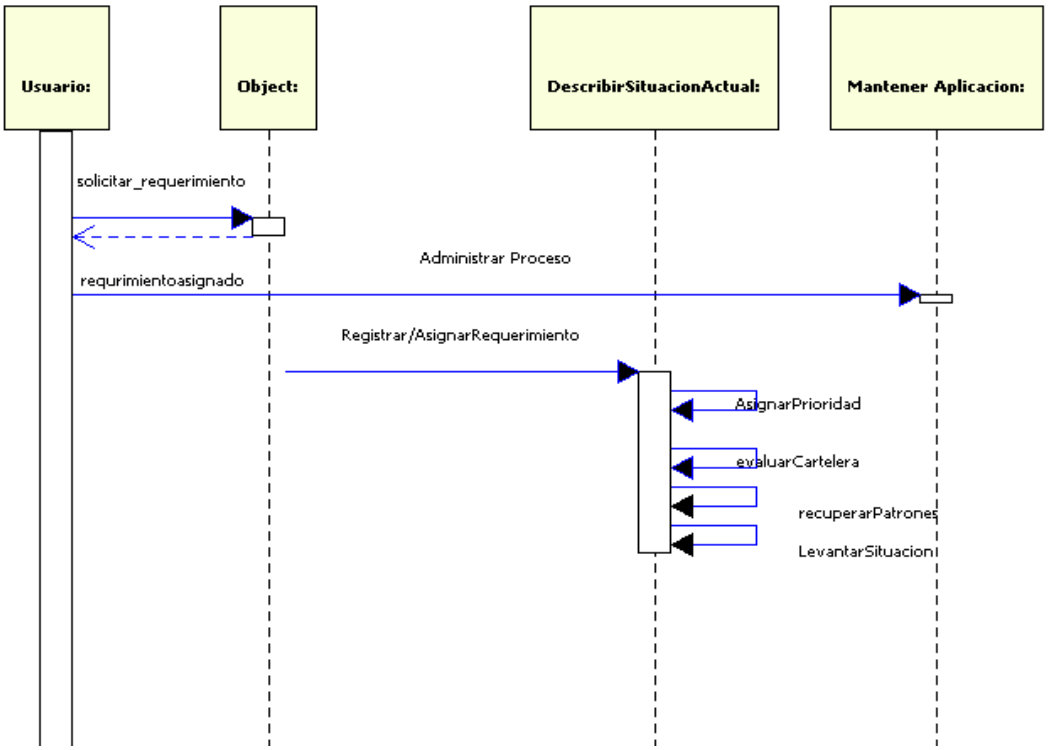


Figura 46: Diagramas de Secuencias para Analizar

Luego de realizar el diagrama de secuencia de analizar es necesario el diagrama de colaboración, el cual permite observar las interacciones entre los objetos.

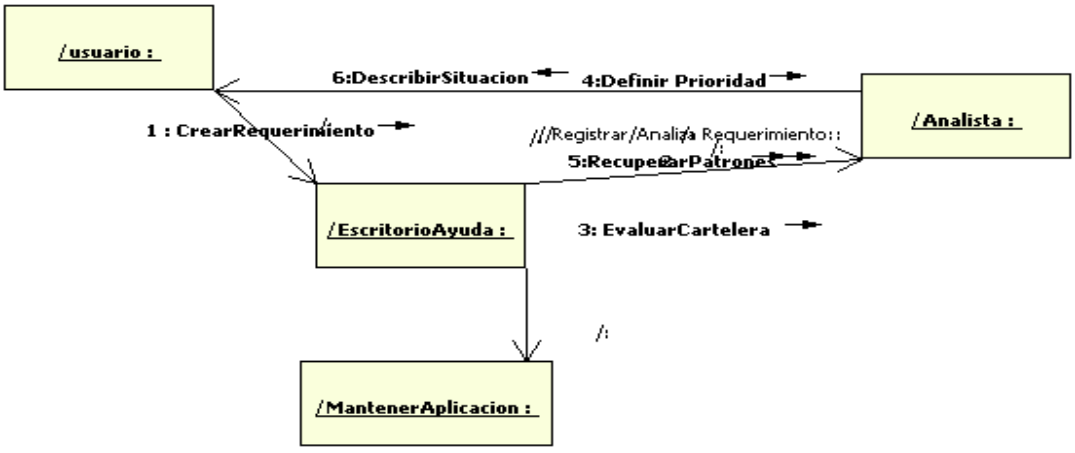


Figura 47: Diagrama de Colaboración de Analizar

En la figura que se encuentra en la parte de abajo, se puede visualizar las rutas que puede seguir el flujo en donde se encuentra el proceso.

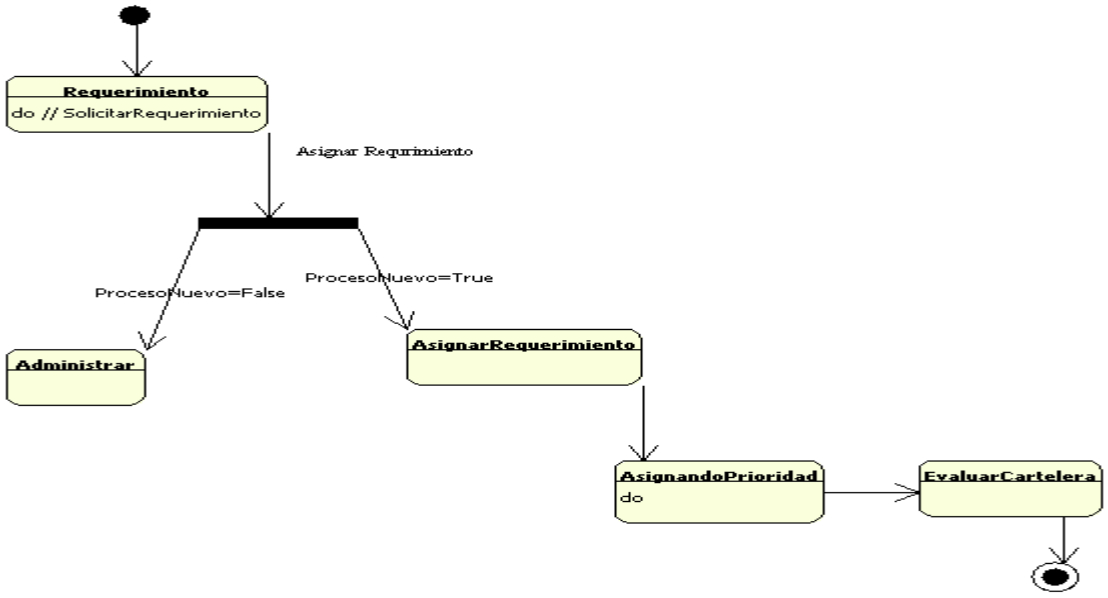


Figura 48: Diagrama de Estado Analizar

Seguidamente, se muestran una serie de diagramas de actividades que permiten visualizar el flujo de trabajo, paso a paso de los procesos que se están desarrollando.

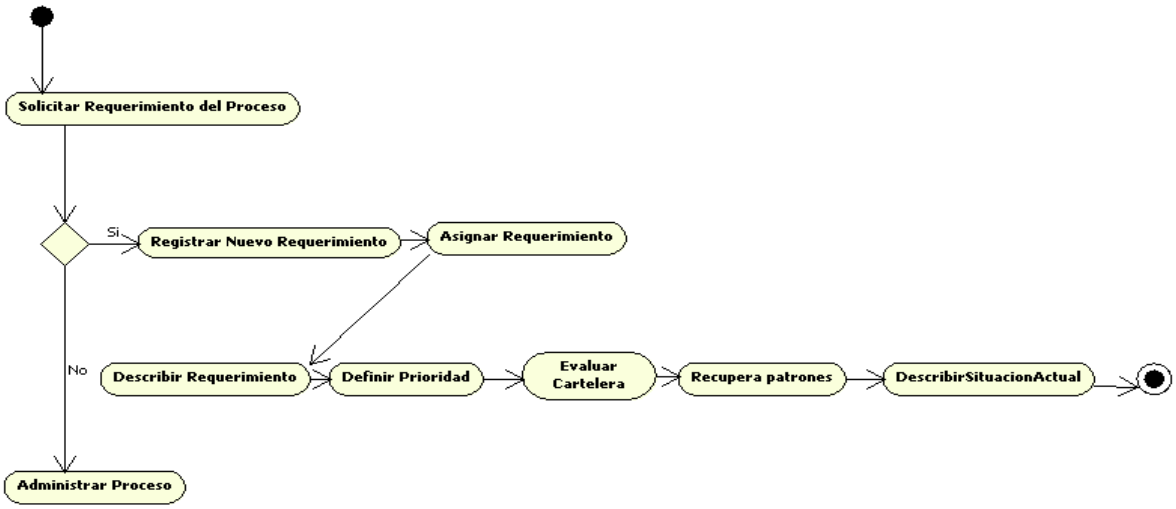


Figura 49: Diagrama de Actividad Analizar

La figura abajo mostrada arroja información del flujo que genera diseñar.

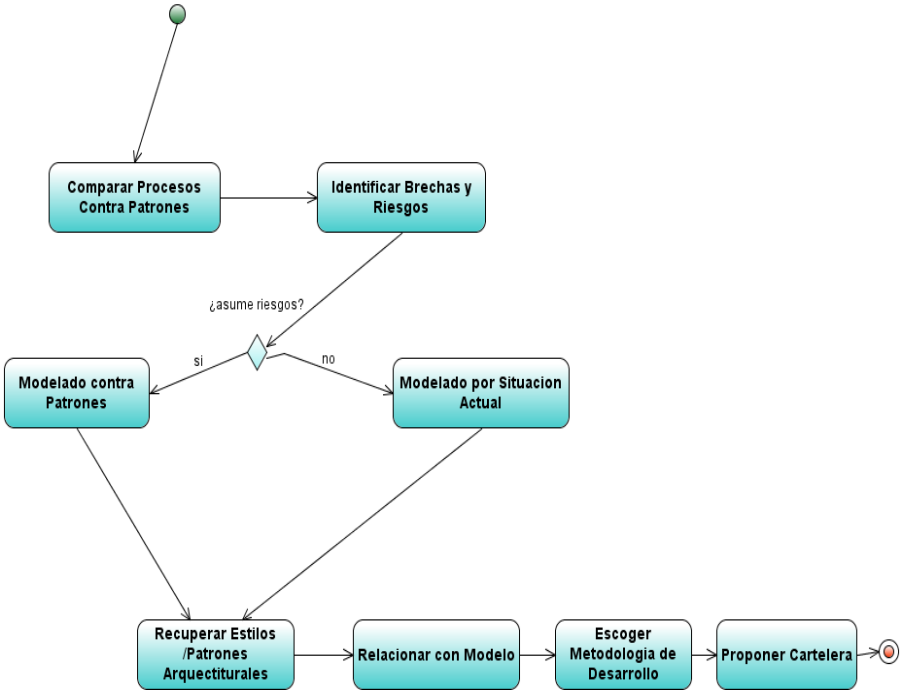


Figura 50: Diagramas de Actividad Diseñar

Luego de diseñar se muestra el flujo de información del proceso modelar.

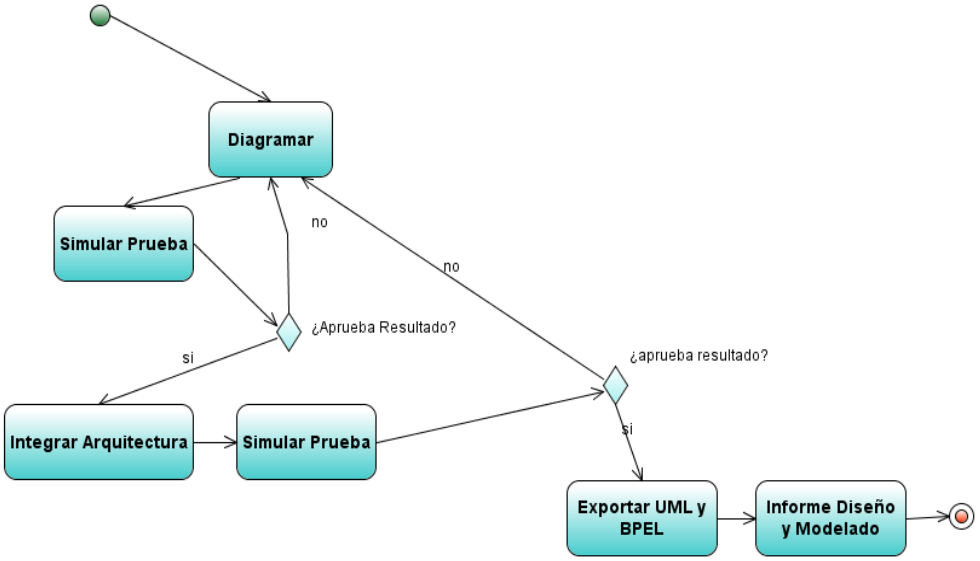


Figura 51: Diagrama de Actividad Modelar

Luego de modelar se procede a diagrama las actividades del proceso configurar, como se puede observar en la figura abajo mostrada.

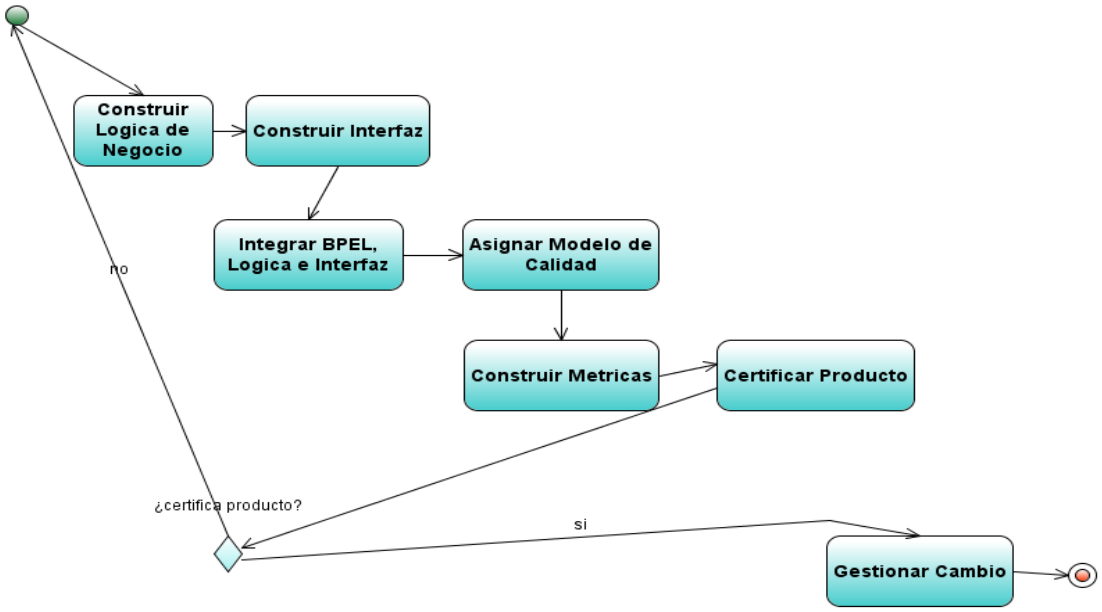


Figura 52: Diagrama de Actividad Configurar

4.1.8 **Iteración 6:** Descripción de la Plataforma

Realizado el diseño del sistema, es un buen momento para describir la plataforma y las herramientas de software que nos permitirán realizar la implementación.

Unos de los objetivos de este trabajo es "Desarrollar el prototipo que permita el uso de la METODOLOGÍA en cuanto a la reducción de los esfuerzos en el proceso de desarrollo de software usando una notación gráfica basada en estándares internacionales" creemos que las tecnologías seleccionadas permiten cumplir con dicho objetivo.

A continuación se muestra lo componentes de la aplicación necesarias para la implementación.

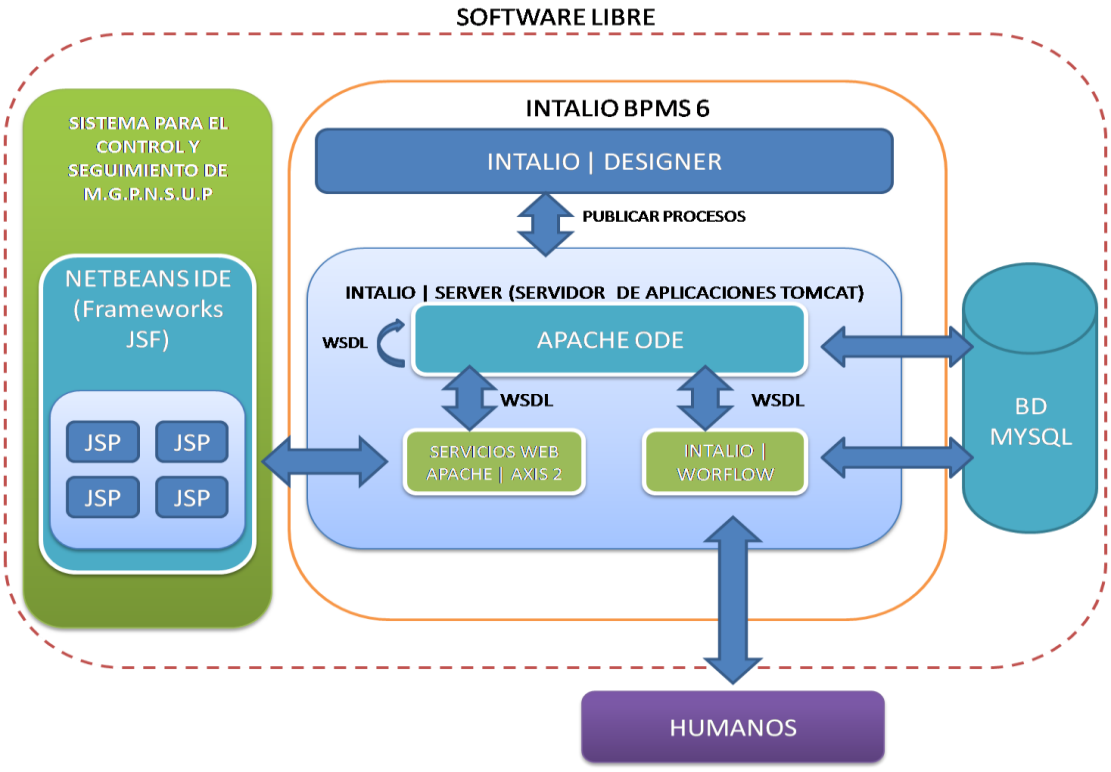


Figura 53: Adaptación de Arquitectura Intalio BPMS para la solución planteada – Fuente: Elaboración propia

Todas las herramientas seleccionadas son de software libre, para mayor información de las herramientas, véase el Anexo 12

Tabla 14: Herramientas Utilizadas

Características	Herramientas
SMBD	MYSQL 5.1
Modelador de Diseño	Diseñador de Intalio 6.0.1
Motor de Workflow	Servidor de Intalio (TOMCAT)
Entorno de Desarrollo	NetbeansIDE 6.7.1
Framework (MVC)	Java Server Faces (JSF)
Páginas	Java Server Pages (JSP)
Servicios Web	Axis2

4.2 Fase de Implementación

Luego de finalizar la fase de concepción se procede a la implementación, como se mencionó anteriormente el modelado de los procesos se realizó con la herramienta Intalio BPMS (notación BPMN) la edición comunitaria. Y para el diseño *look & feel* Netbeans IDE 6.7.1, con el frameworks Java Server Faces.

4.2.1 Configuración del motor de ejecución del BPMS

Este paso corresponde a la configuración del motor de ejecución de la herramienta de software libre Intalio. Se realizó la creación de los distintos diagramas, creación de los esquemas XML, consultas a la base de datos MySQL y las invocaciones de los servicios web, para comunicar el modelado con el cliente desarrollado mediante el frameworks Java Server Server (JSF). Los detalles de la creación del modelado del proceso de Metodología de Gestión de Procesos sustentada en el Uso de Patrones lo podemos ver los diagramas a partir de la siguiente página.

4.2.2 Ejecución del proceso de Metodología de Gestión de Procesos sustentada en el Uso de Patrones

Al momento de generar la solución para el proceso de Metodología de Gestión de Procesos sustentada en el Uso de Patrones se definieron los siguientes roles:

- **Administrador:** es el encargado de dar los respectivos permisos, que tiene cada usuario al ingresar al sistema, además, tiene la facultad de registrar nuevos usuarios.
- **Supervisor:** se encarga de velar por el buen desempeño del procesamiento de las peticiones y, en caso de ser posible, valida el trabajo realizado por los analistas.
- **Analista:** es el encargado de procesar las peticiones de nuevos requerimientos.
- **Operador:** es un usuario que solicita los requerimientos de los procesos a tratar.

4.2.3 **Iteración 1:** Modelado de Procesos en Intalio

En el siguiente diagrama se modela la validación de solitud de datos para entrar al sistema, donde se solicita un usuario y una contraseña, si ocurre un error la misma es captada mediante una excepción.

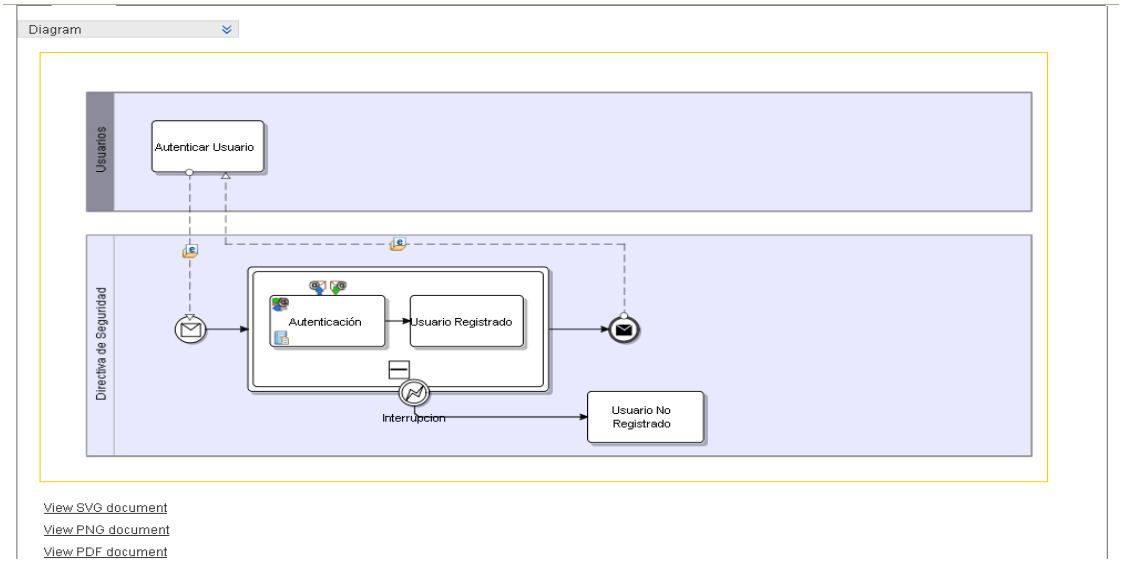


Figura 54: Autenticar Usuarios

La siguiente figura realiza las solicitudes de los datos de los requerimientos, se tiene que conectar a la base de datos donde pide datos básicos del usuario

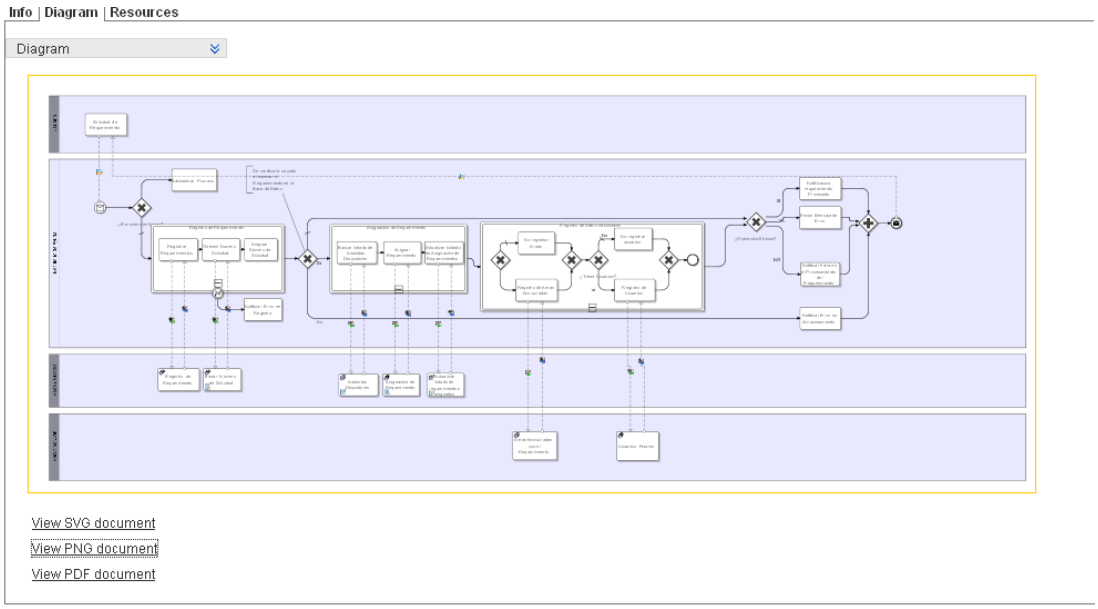


Figura 55: Solicitar Requerimientos

En esta imagen se toman los datos que han sido solicitados por el analista de sistema para luego almacenarlo en la base de datos.

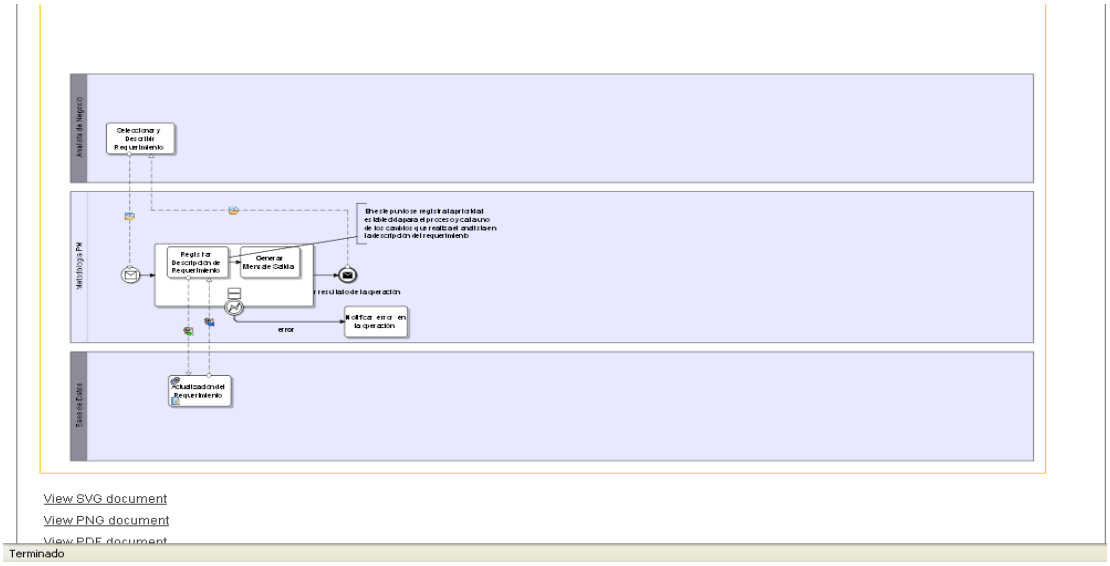


Figura 56: Registrar Descripción de Requerimientos

Esta figura se buscar un número de requerimiento luego busca las áreas de trabajo.

Figura 57: Buscar Requerimiento

Este diagrama lo que hace es buscar lo requerimientos asignado y luego asignarlos a un analista.

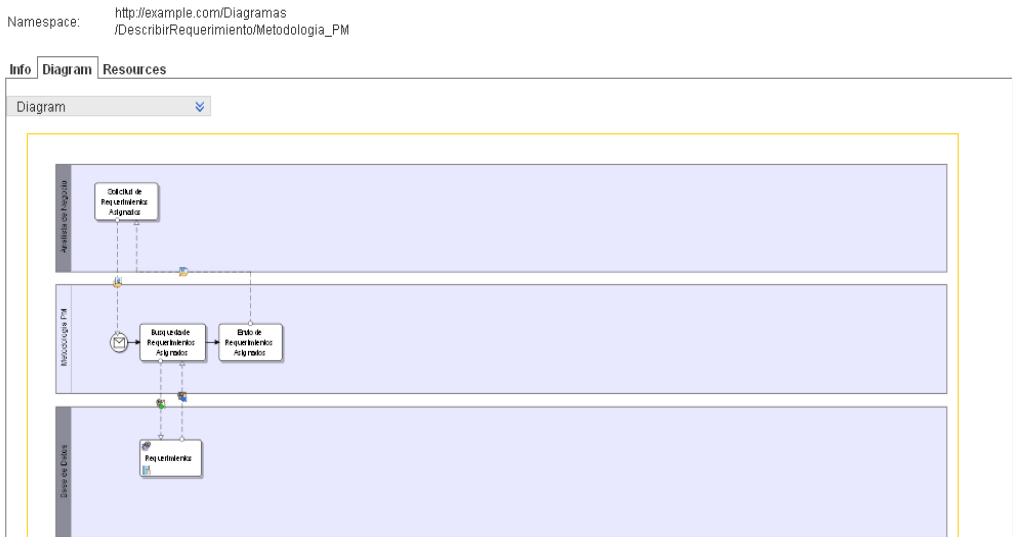


Figura 58: Describir Requerimiento

La función del presente diagrama es registrar nuevos usuarios.

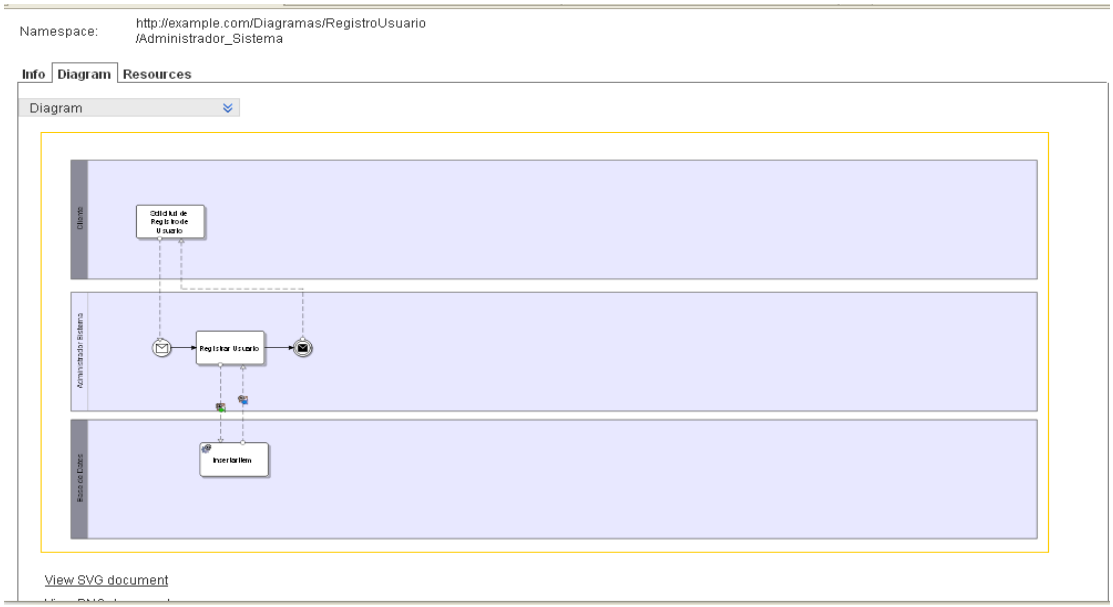


Figura 59: Registrar Usuarios

Inserta cada usuario que fueron registrados a través de la interfaz.

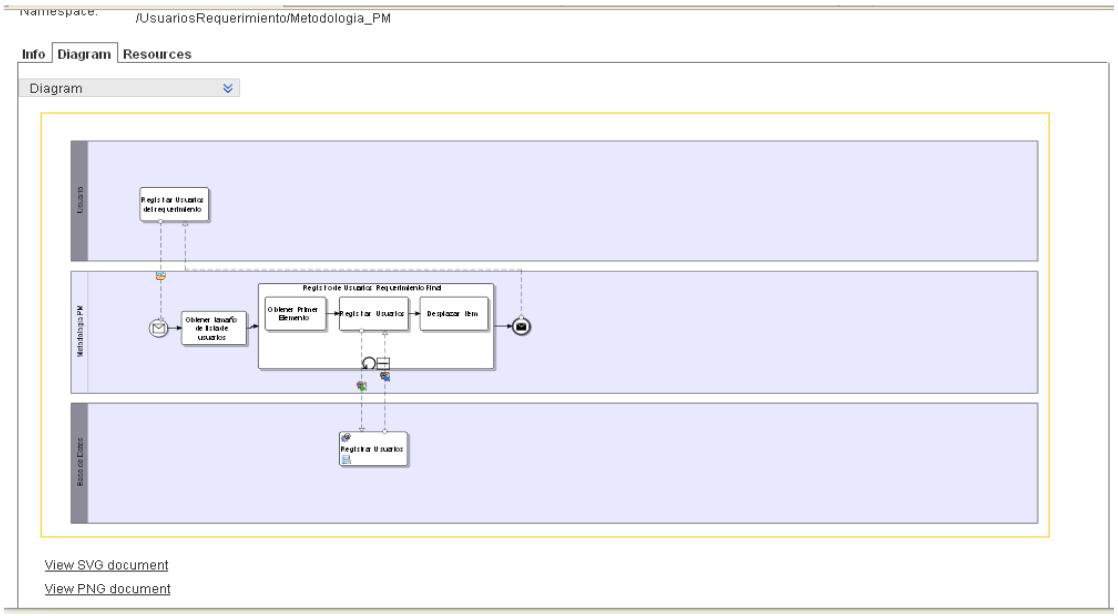


Figura 60: Ingresar Nuevo Usuario

Busca todas las carteleras que han sido aprobadas.

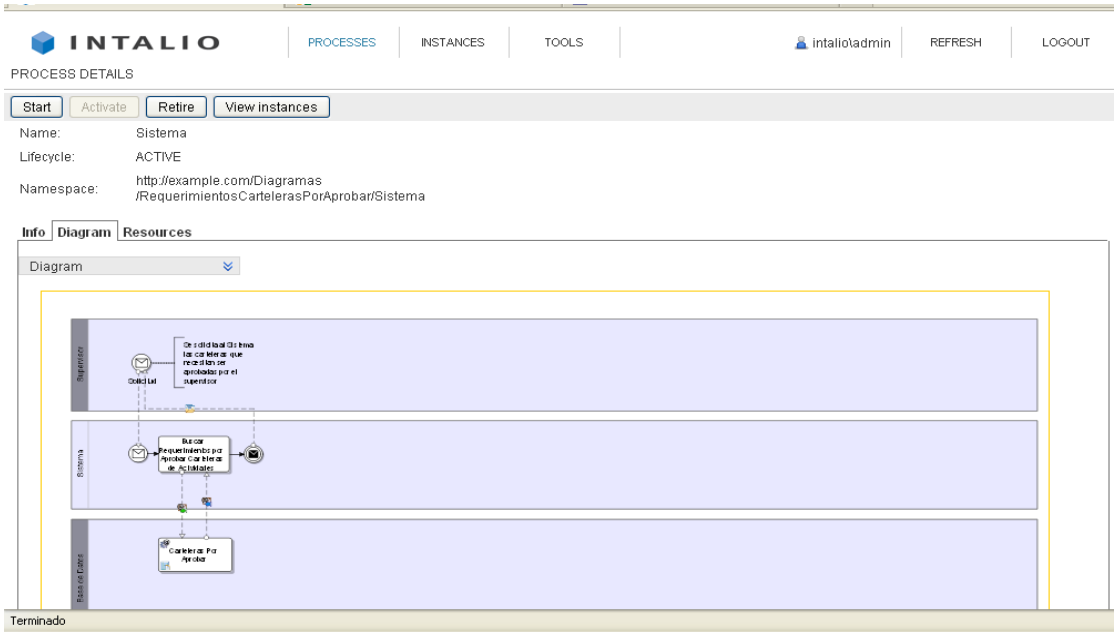


Figura 61: Requerimiento Cartelera Aprobado

Busca todas las carteleras no aprobadas para que la aprueben y cambiar estado.

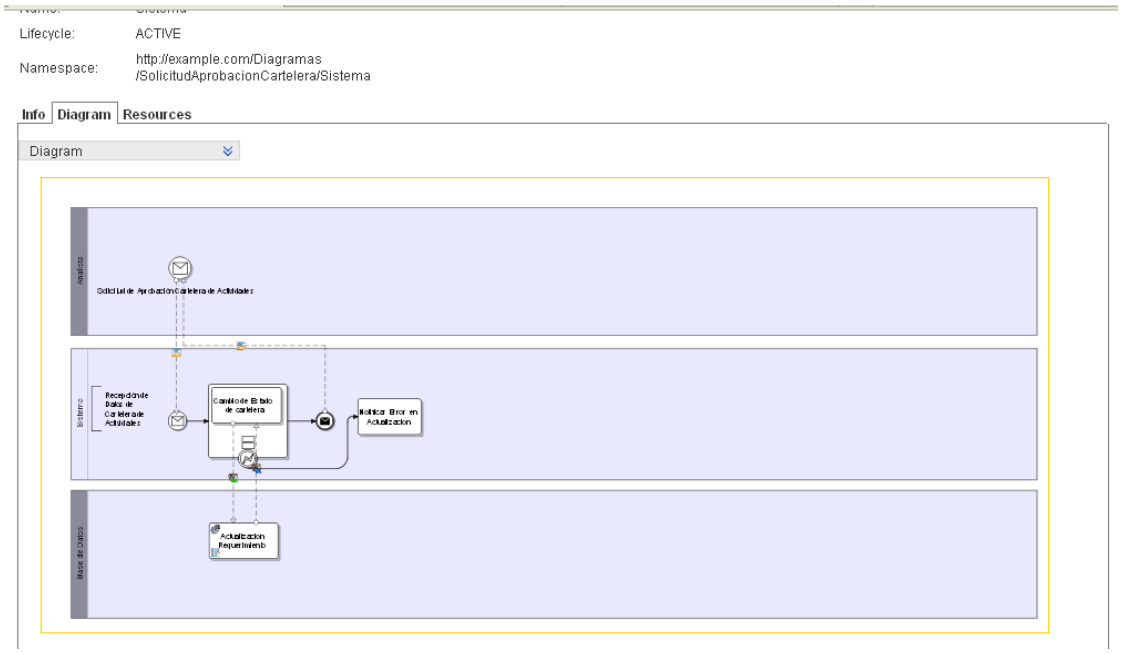


Figura 62: Solicitud de Aprobación de Cartelera

Realiza una consulta sobre las carteleras que no han sido aprobadas.

Figura 63: Carteleras por Aprobar

Busca una cartelera para que el supervisor la apruebe y aprobada cambia estado en la base de datos.

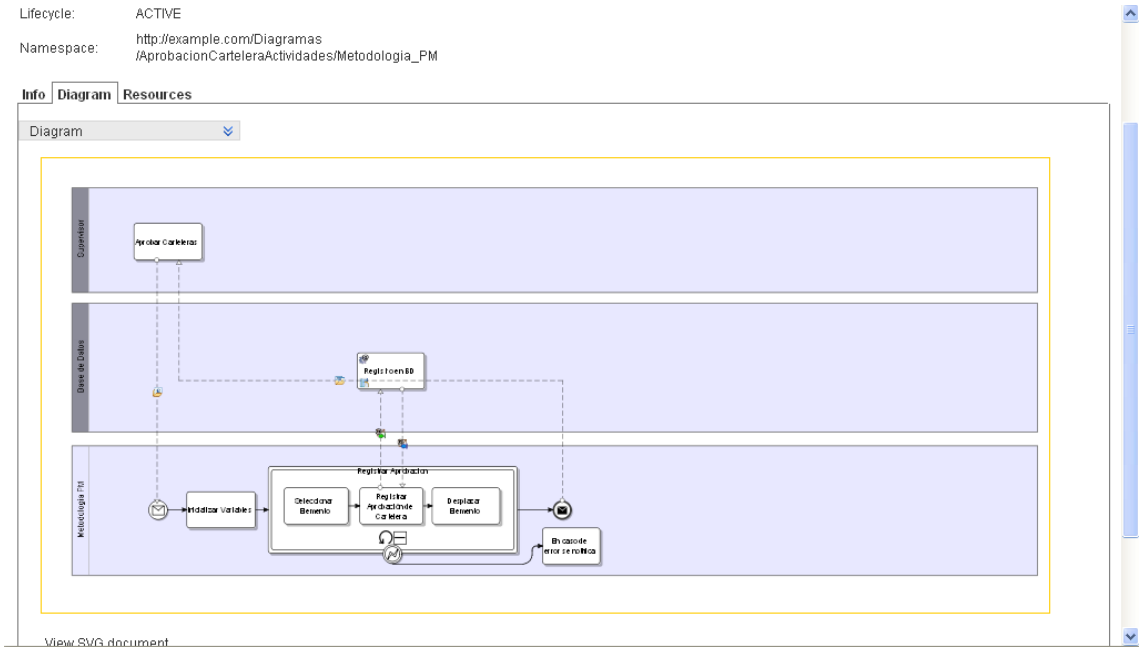


Figura 64: Aprobación Cartelera de Actividades

En la siguiente figura se buscar el número de solicitud del requerimiento.

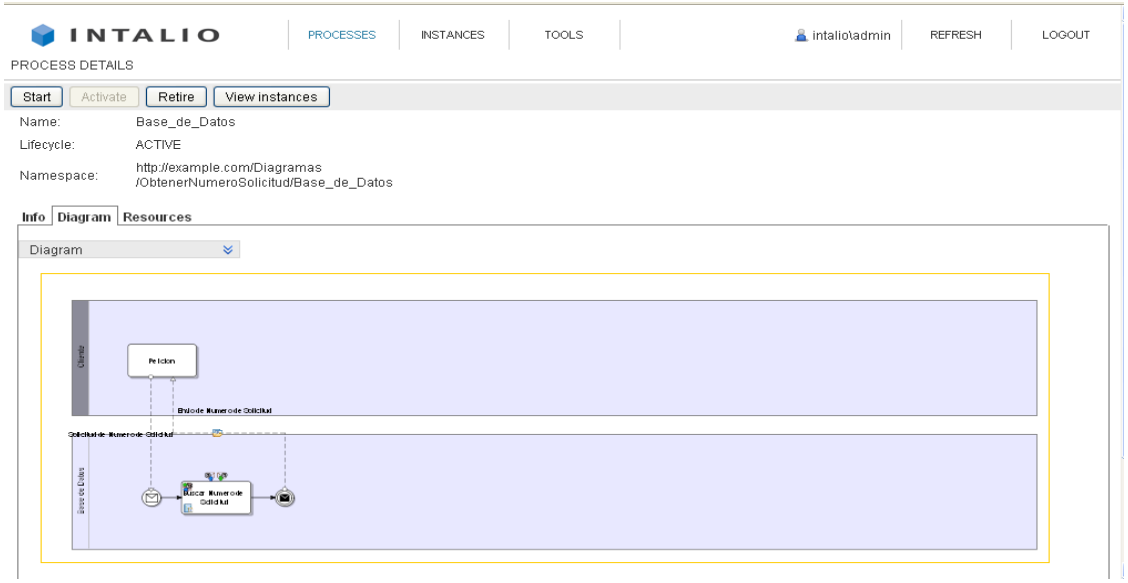


Figura 65: Obtener número de solicitud

Luego de seleccionar las áreas involucradas se procede a guardar la información en la base de datos.

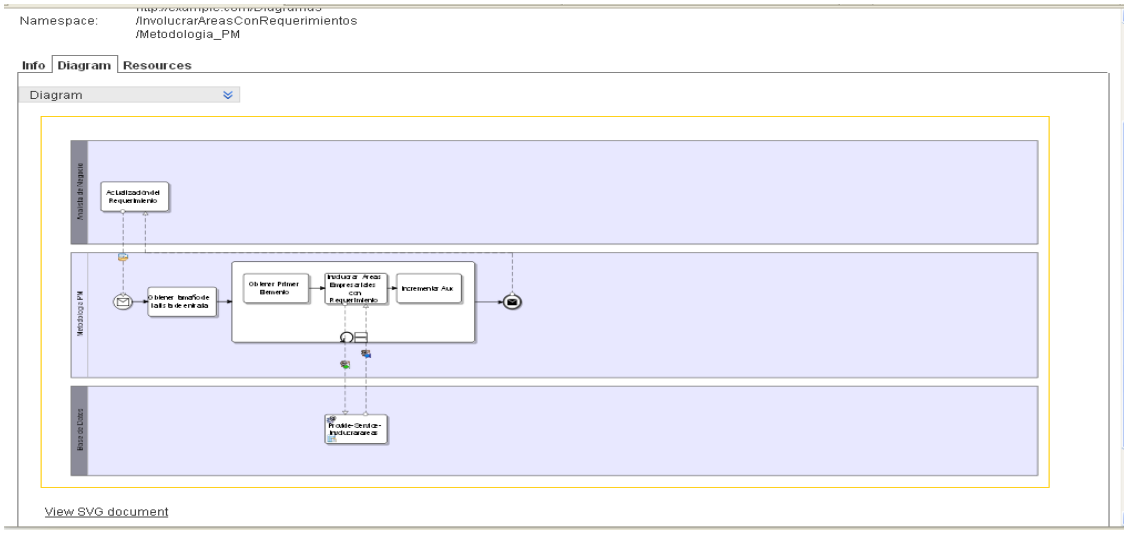


Figura 66: Involucrar Áreas con Requerimientos

Se buscan todos los patrones para que el usuario lo seleccione luego.

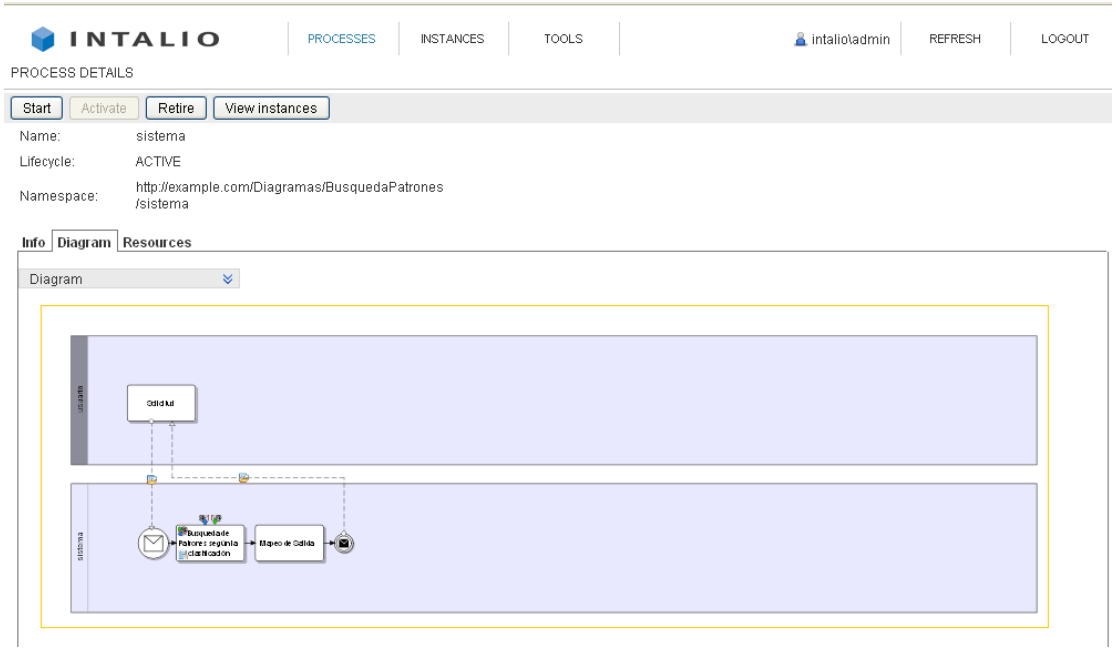


Figura 67: Búsqueda de Patrones

Se hace una consulta sobre la base de datos para que arroje todas las áreas involucradas.

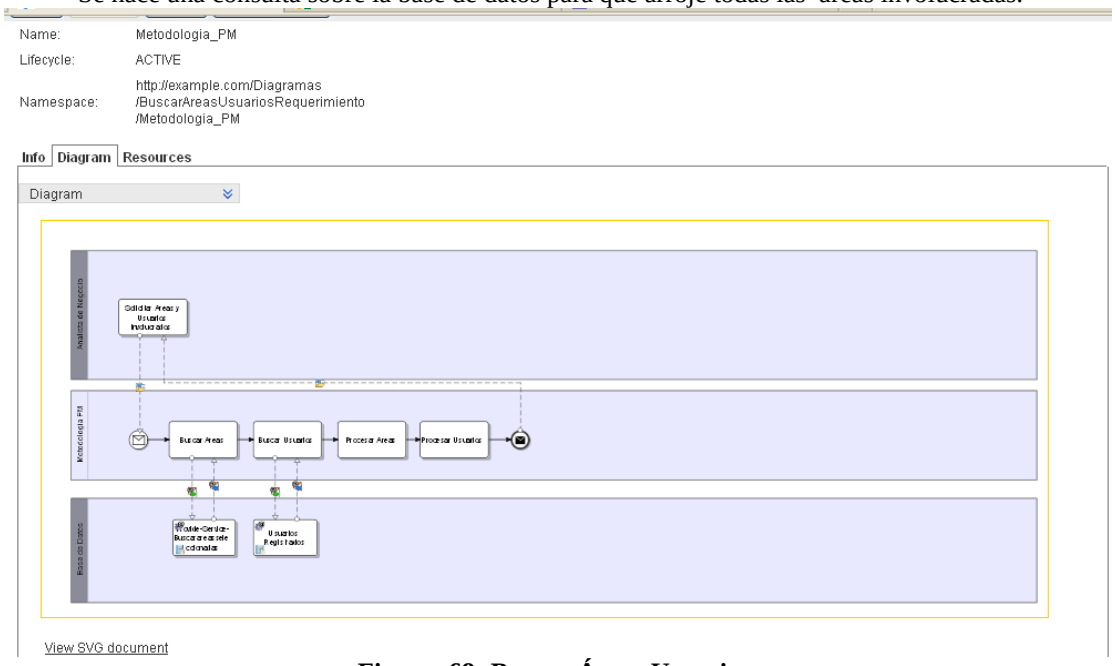


Figura 68: Buscar Áreas Usuarios

Esta vista es la generada cuando se crean elementos o atributos que luego se usan para mapear valores.

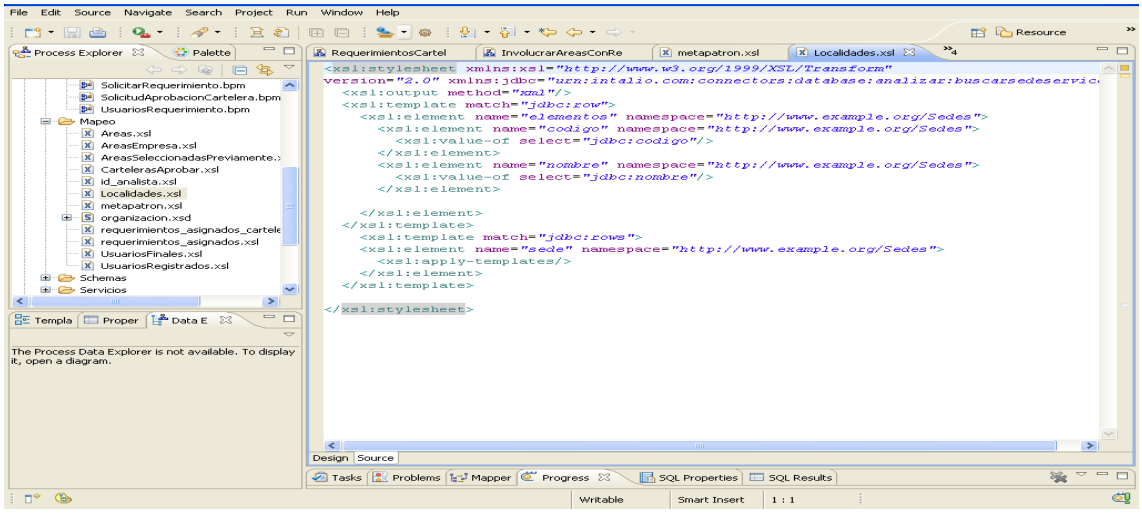


Figura 69: Mapeo de Información

Luego de generar los .xcd necesarios para mapear los valores. Se necesita de la interfaz que permite realizar dicho trabajo. La siguiente imagen muestra como la variable A se mapea con la variable B

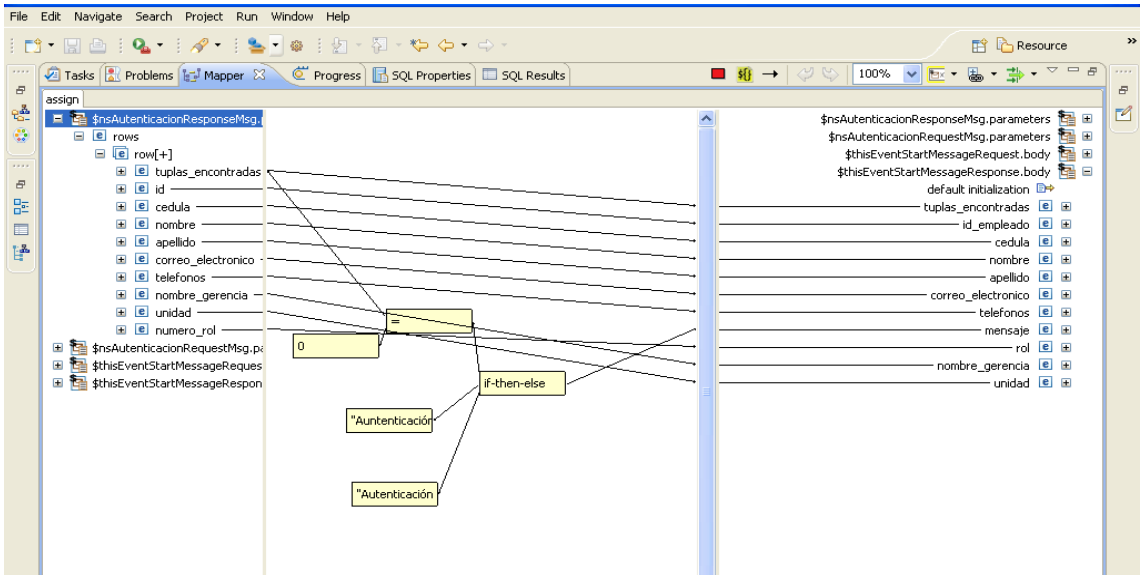
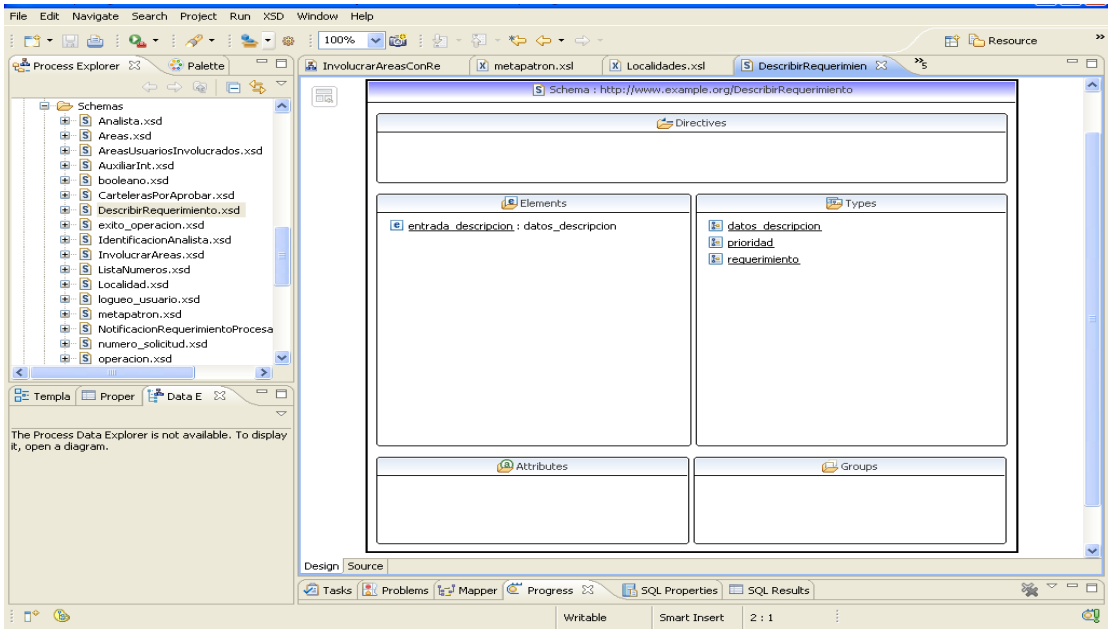


Figura 70: Mapeo de Datos



Es interfaz permite crear tipos de datos ya sean atributos o elementos.

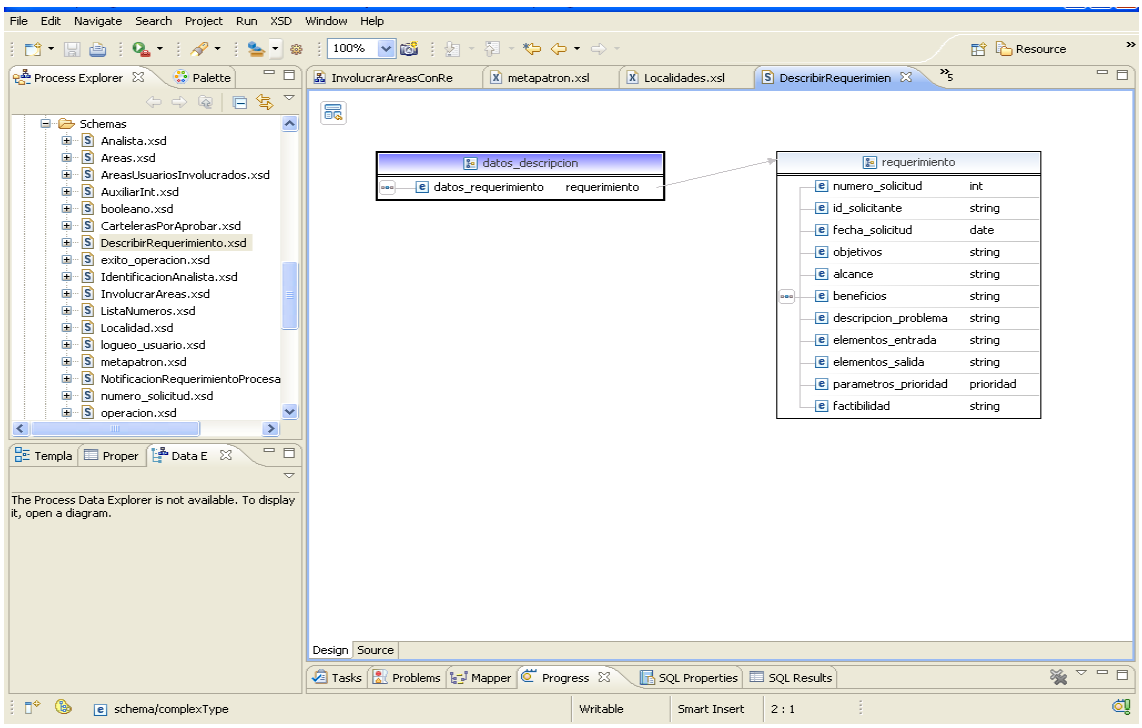


Figura 71: Creación de Schemas

4.2.3 **Iteración 7:** Diseño del Sistema Vistas principales

Para poder ingresar al sistema los usuarios deben autenticarse con algunos de los roles mencionados en la iteración pasada, en la siguiente figura que se muestra a continuación.

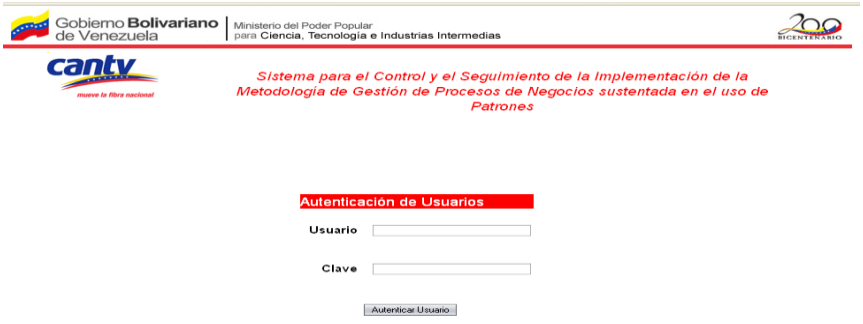


Figura 72: Autenticar Usuarios

Una vez el usuario se ha autenticado, según su rol podrá visualizar una serie de opciones, pueden observarse en las imágenes siguientes.

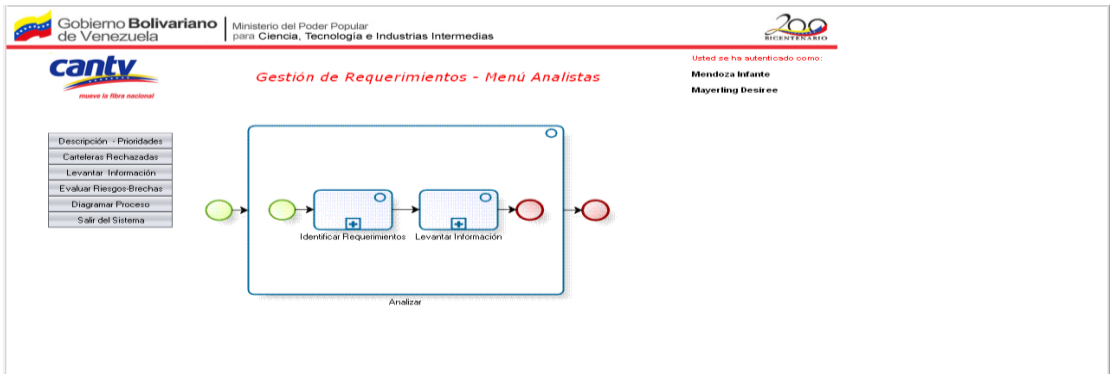


Figura 73: Menú Analista

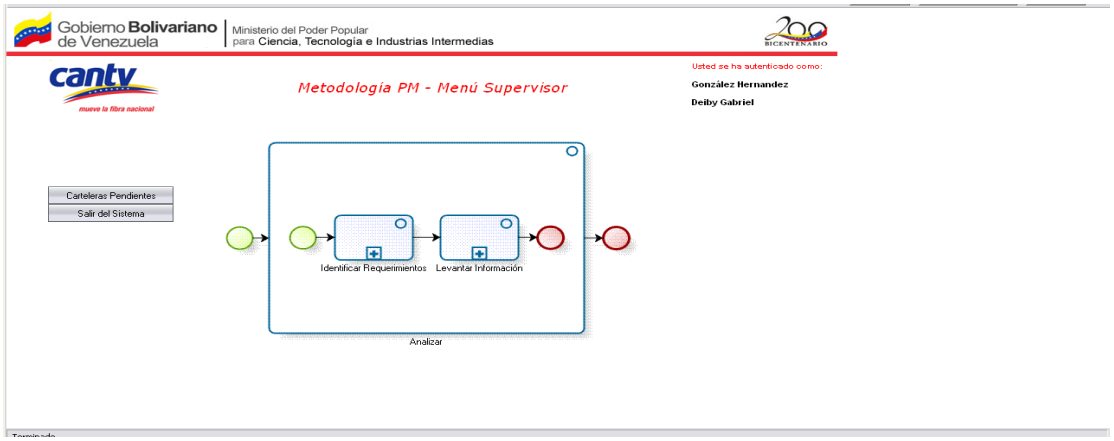


Figura 74: Menú Supervisor

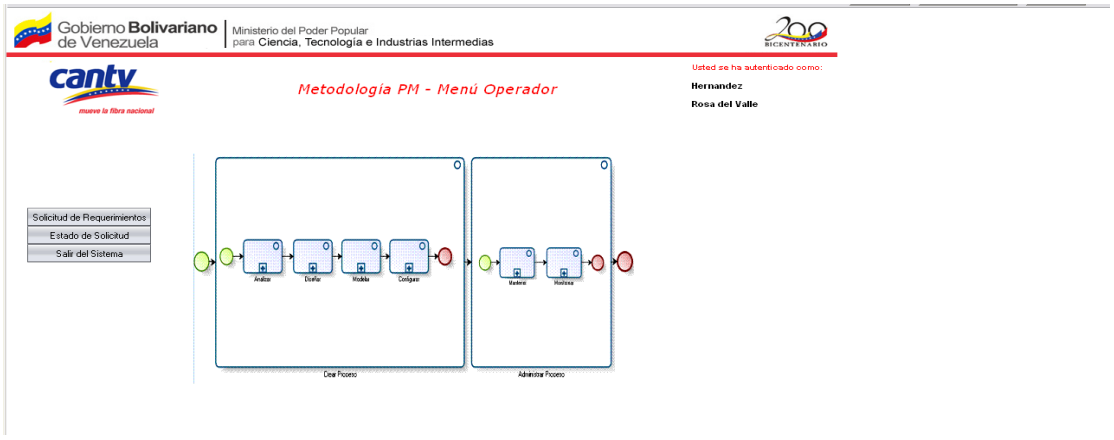


Figura 75: Menú Operador

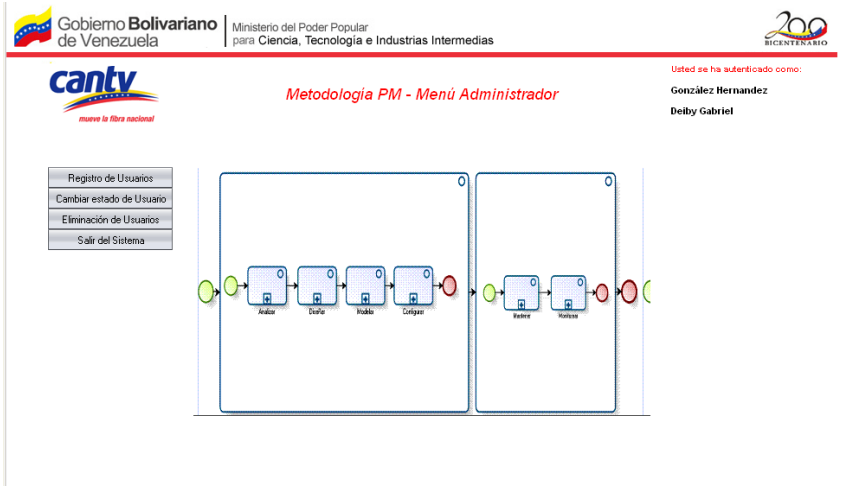


Figura 76: Menú Administrador

4.2.4 **Iteración 8:** Vistas que dependen del Usuario admin (Administrador del Sistema)

Este usuario es el encargado de registrador nuevos usuarios y asignarles su rol, cambiarles su estado dentro del sistema y eliminarlos.

A continuación se muestra la interfaz para dicho usuario.

Figura 77: Registrar Usuarios

4.2.5 **Iteración 9:** Vistas que dependen del Usuario Supervisor

En esta vista se muestran todas las carteleras que tienen que ser aprobadas o rechazadas por el supervisor del sistema.

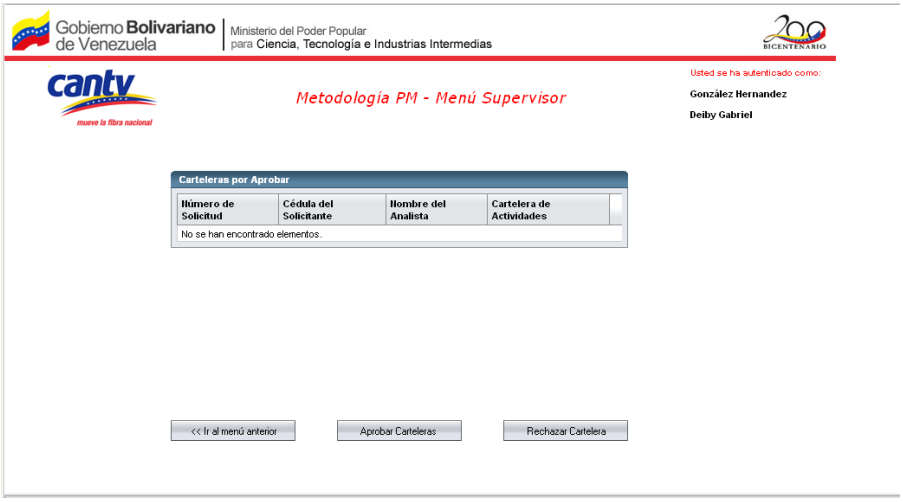


Figura 78: Carteleras por Aprobar

4.2.6 **Iteración 10:** Vistas que dependen del Usuario Operador

El operador es quien hace la solicitud del requerimiento, luego de realizar dicha operación puede consultar el estado en el cual se encuentra.

Figura 79: Solicitar Requerimiento

Figura 80: Solicitar Requerimientos Información

4.2.7 **Iteración 11:** Vistas que dependen del Usuario Analista

Un analista hace la descripción del requerimiento, asigna prioridad, observa las carteleras que tiene rechazadas, hace el levantamiento de la información del

proceso, evalúa brechas y riesgos, para luego realizar un informa de lo que ha sucedido con el requerimiento, puede realizar simulaciones de los procesos.

Figura 81: Requerimientos asignados a un Analista

Según su urgencia e impacto se le asigna la prioridad.

Figura 82: Asignar Prioridad

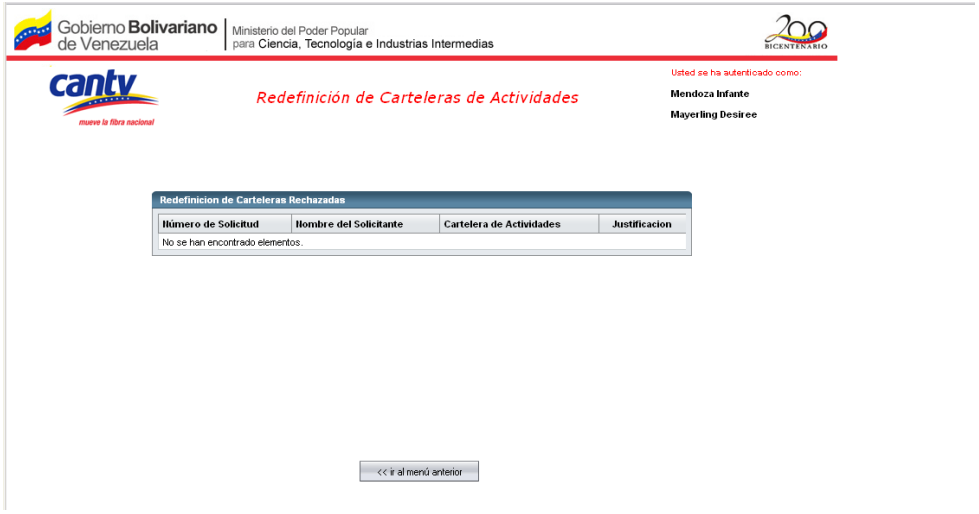


Figura 83: Carteleras Rechazadas

Se verifica los datos de requerimiento para luego asociarlo con un patrón de análisis.

Figura 84: Levantamiento de Información

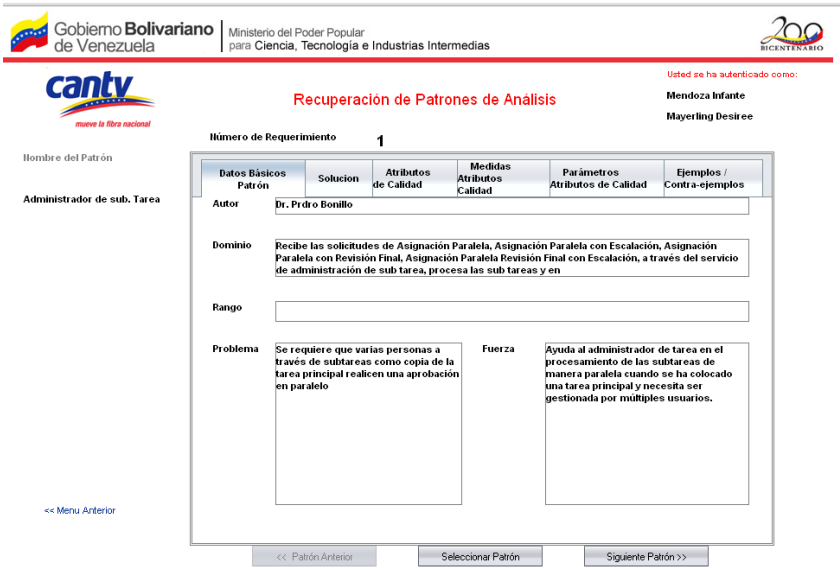


Figura 85: Recuperar Patrones de Análisis

Luego de recuperar el Patrón se prosigue a realizar el levantamiento de la situación actual.

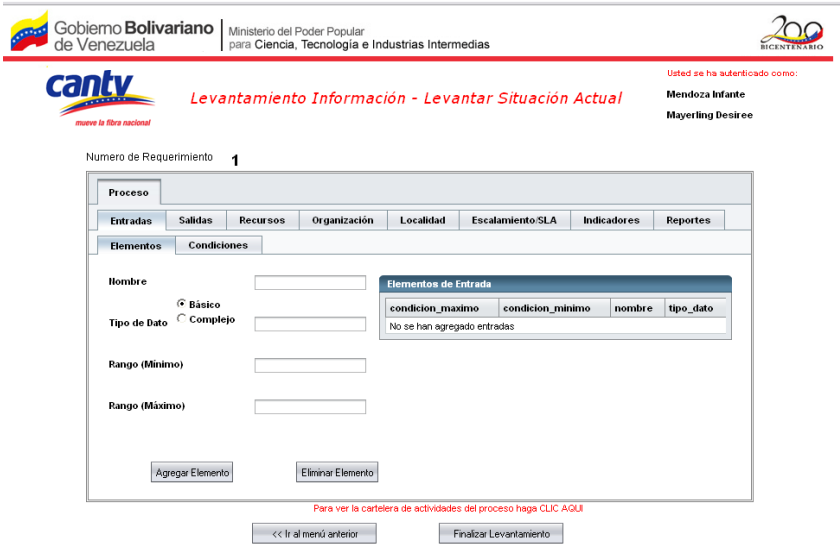
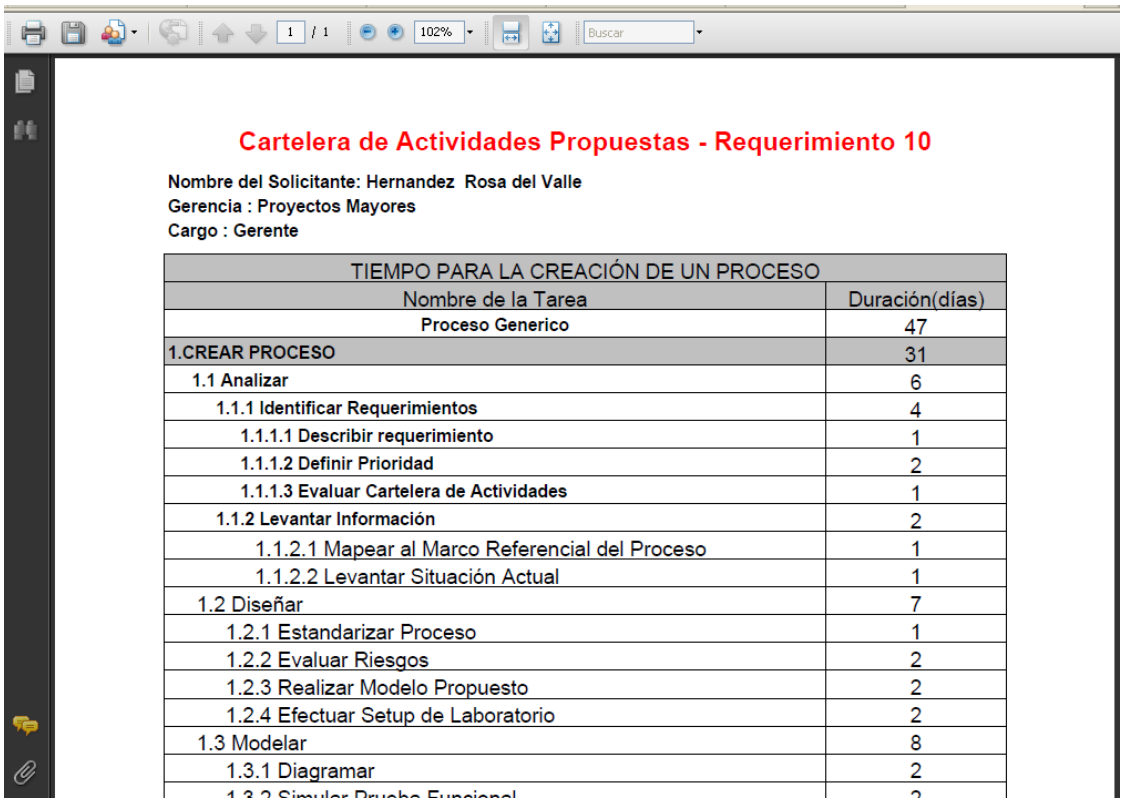


Figura 86: Levantar Situación Actual



Cartelera de Actividades Propuestas - Requerimiento 10

Nombre del Solicitante: Hernandez Rosa del Valle
Gerencia : Proyectos Mayores
Cargo : Gerente

TIEMPO PARA LA CREACIÓN DE UN PROCESO	
Nombre de la Tarea	Duración(días)
Proceso Generico	47
1.CREAR PROCESO	31
1.1 Analizar	6
1.1.1 Identificar Requerimientos	4
1.1.1.1 Describir requerimiento	1
1.1.1.2 Definir Prioridad	2
1.1.1.3 Evaluar Cartelera de Actividades	1
1.1.2 Levantar Información	2
1.1.2.1 Mapear al Marco Referencial del Proceso	1
1.1.2.2 Levantar Situación Actual	1
1.2 Diseñar	7
1.2.1 Estandarizar Proceso	1
1.2.2 Evaluar Riesgos	2
1.2.3 Realizar Modelo Propuesto	2
1.2.4 Efectuar Setup de Laboratorio	2
1.3 Modelar	8
1.3.1 Diagramar	2

Figura 87: Cartelera de Actividades en PDF

Figura 88: Diagramar Proceso

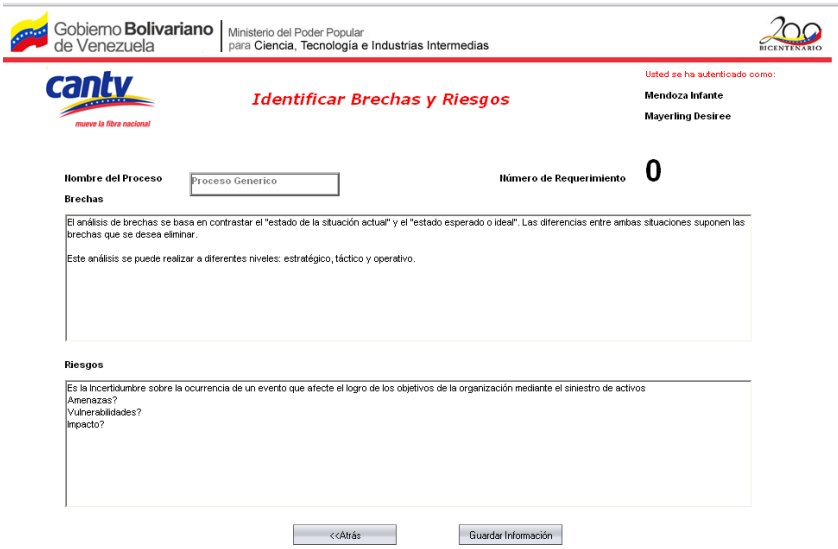


Figura 89: Evaluar Riesgos y Brechas

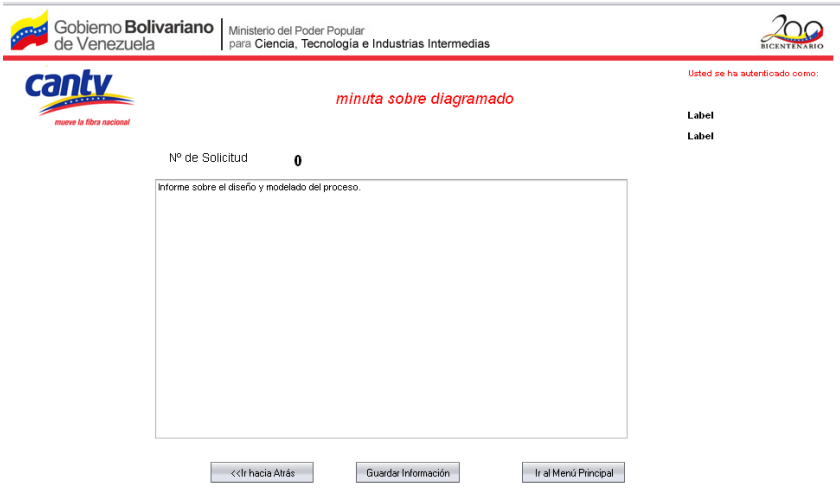


Figura 90: Minuta sobre el Proceso

4.3 Fase de Prueba

Al finalizar cada iteración se debe aplicar pruebas funcionales, para observar si se obtiene los resultados esperados.

4.3.1 Iteración 0: General

Esta prueba consiste en verificar si se está autenticando los usuarios correctamente y si muestra mensajes de errores al momento de autenticarse.

El usuario se ha logueado correctamente.

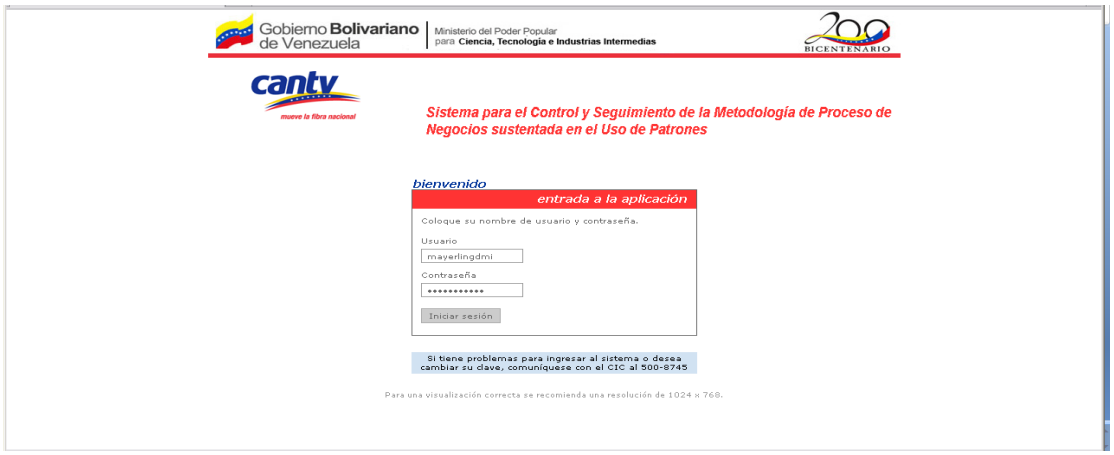


Figura 91: Validar autenticación

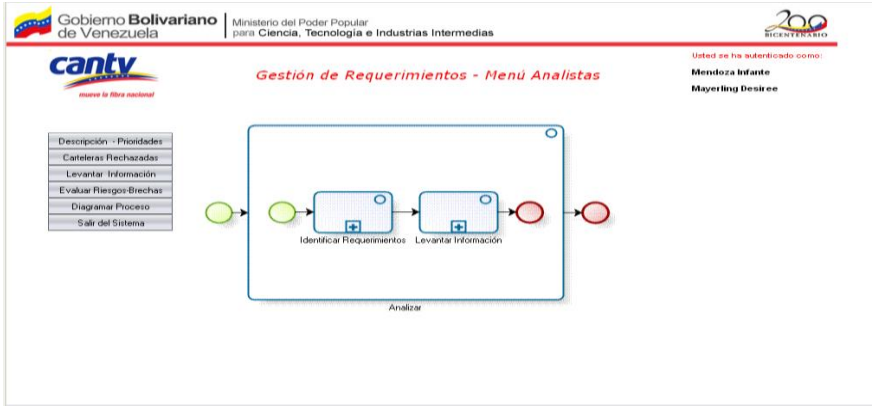


Figura 92: Usuario ingresó correctamente

Figura 93: Usuario y/o Contraseña incorrecta

4.3.2 Iteración 1: Permitir Registros de Nuevos Usuarios

En esta prueba se verifica que permita crear nuevos usuarios y si el nombre del usuario da un alerta que no puede usarse dicho usuario.

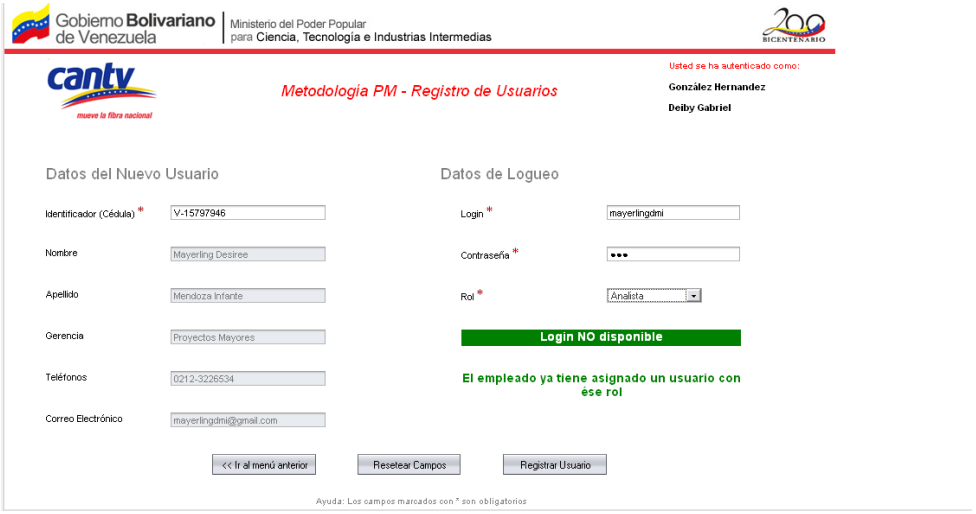


Figura 94: Registro Nuevos Usuarios

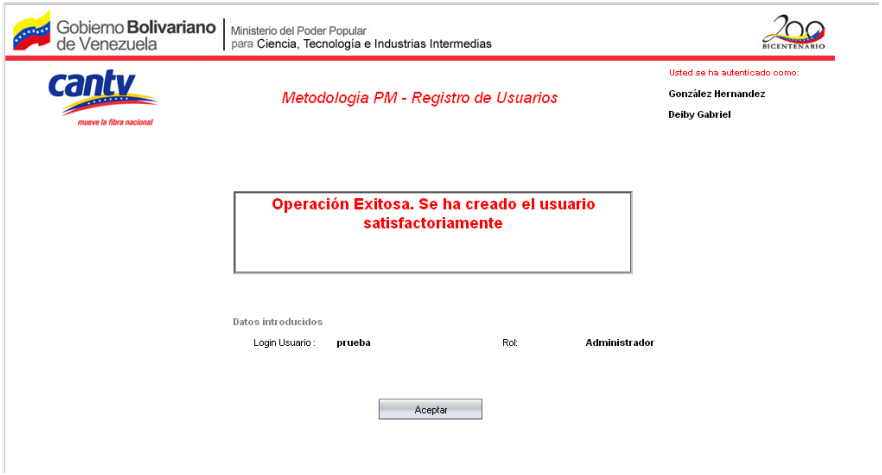


Figura 95: Registro de Usuario Exitoso

4.3.3 Iteración 2: Soporte Navegadores

Esta funcionalidad a verificar es que se pueda visualizar el sistema tanto en mozilla firefox como internet explorer.

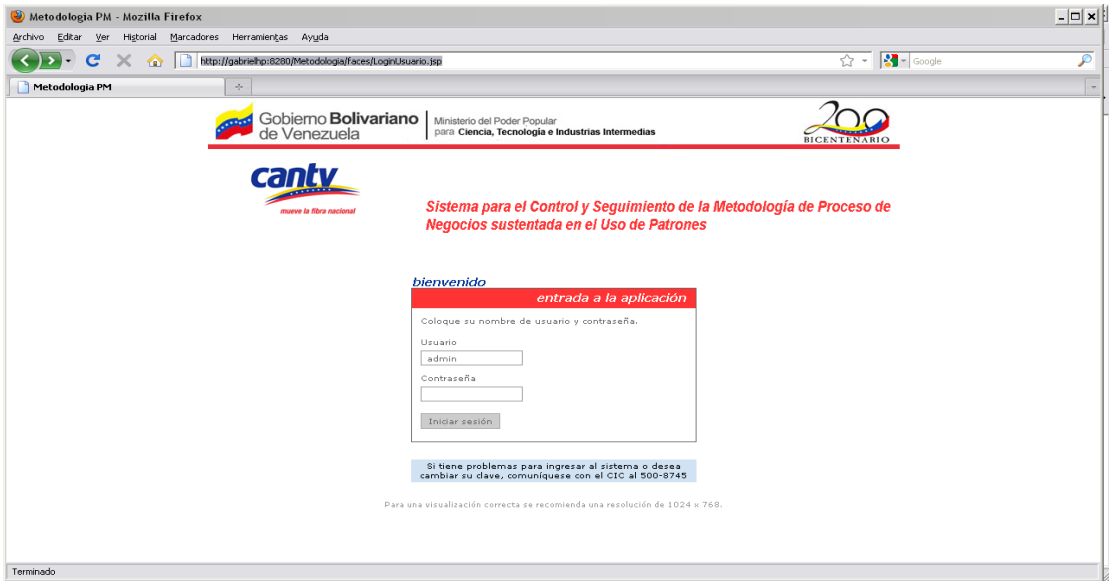


Figura 96: Soporta Mozilla Firefox

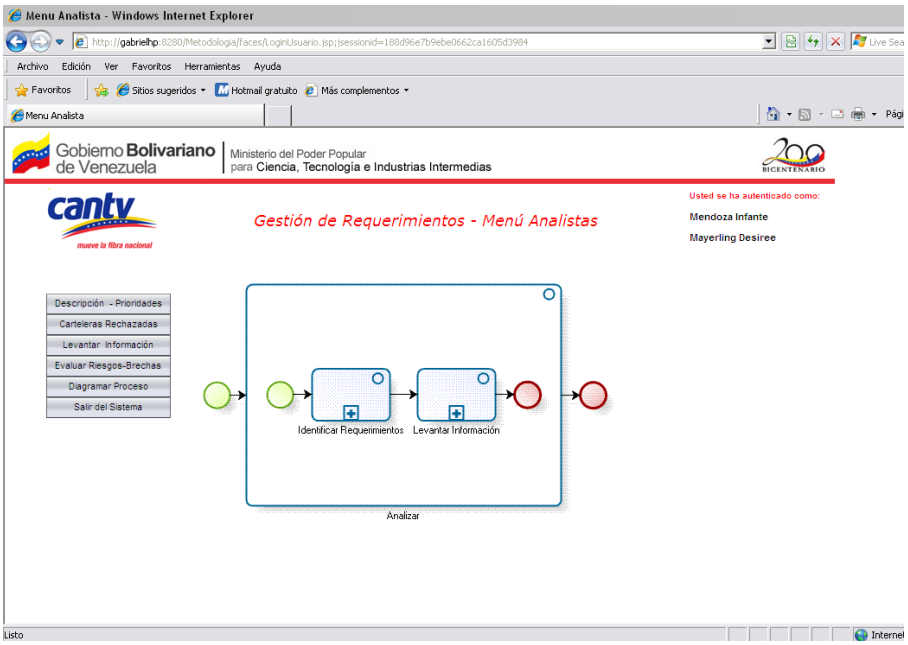


Figura 97: Soporte de Internet Explorer

4.3.4 Iteración 3: Encriptación Clave Secreta

Verificar que cuando se introduzca la clave la misma no sea visualizada por el navegador ni por el text input.

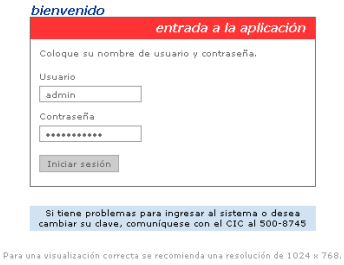


Figura 98: Encriptar claves

4.3.5 Iteración 4: Mensajes cuando está incompleta la operación

Cuando está incompleta la información muestra mensajes de errores, donde dice que la operación no ha sido registrada.

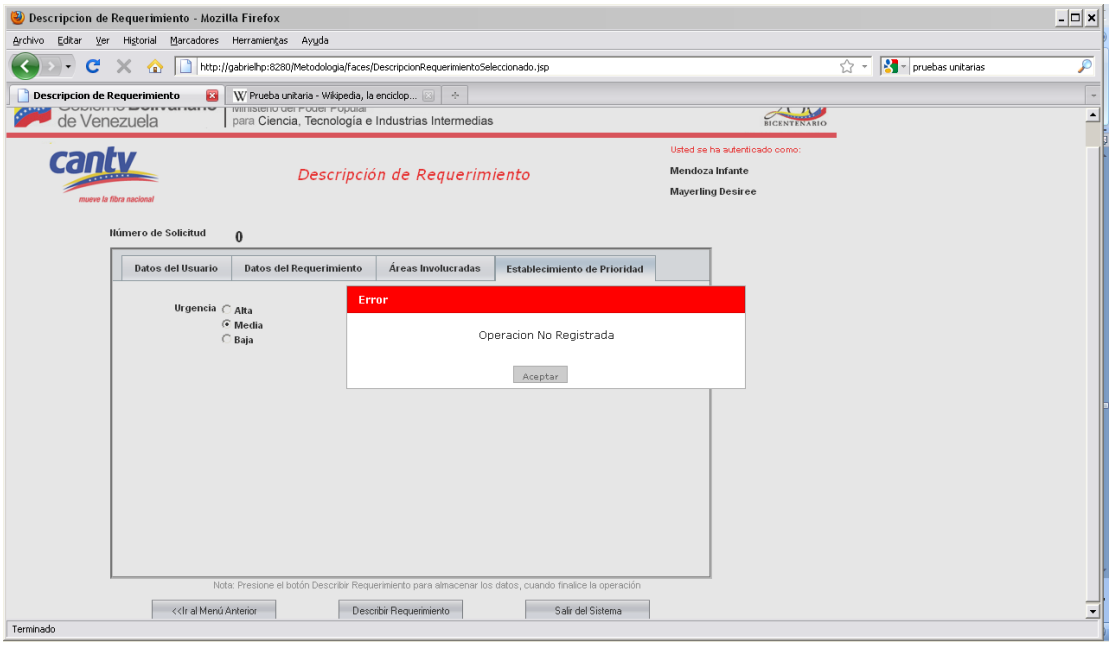


Figura 99: Mensaje de error

4.3.6 Iteración 5: Ancho de Diagramación

Es verificar que las interfaces estén usando el estándar para diagramación de las vistas. Se debe usar 1024 por 768.

Se puede observar que dicha configuración en una resolución de ese tamaño se ve la vista perfectamente.

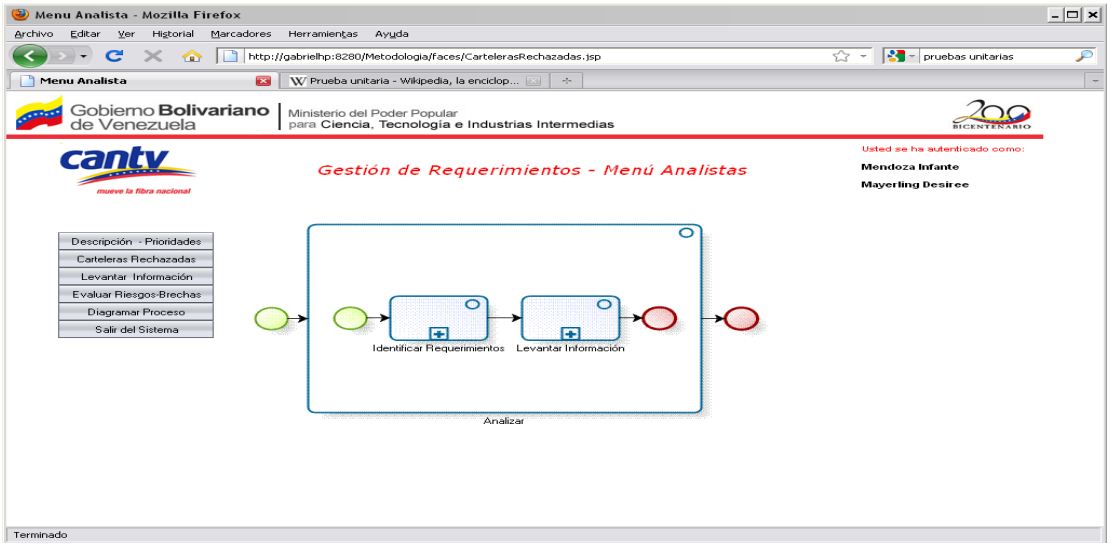


Figura 100: Observando desde Navegador

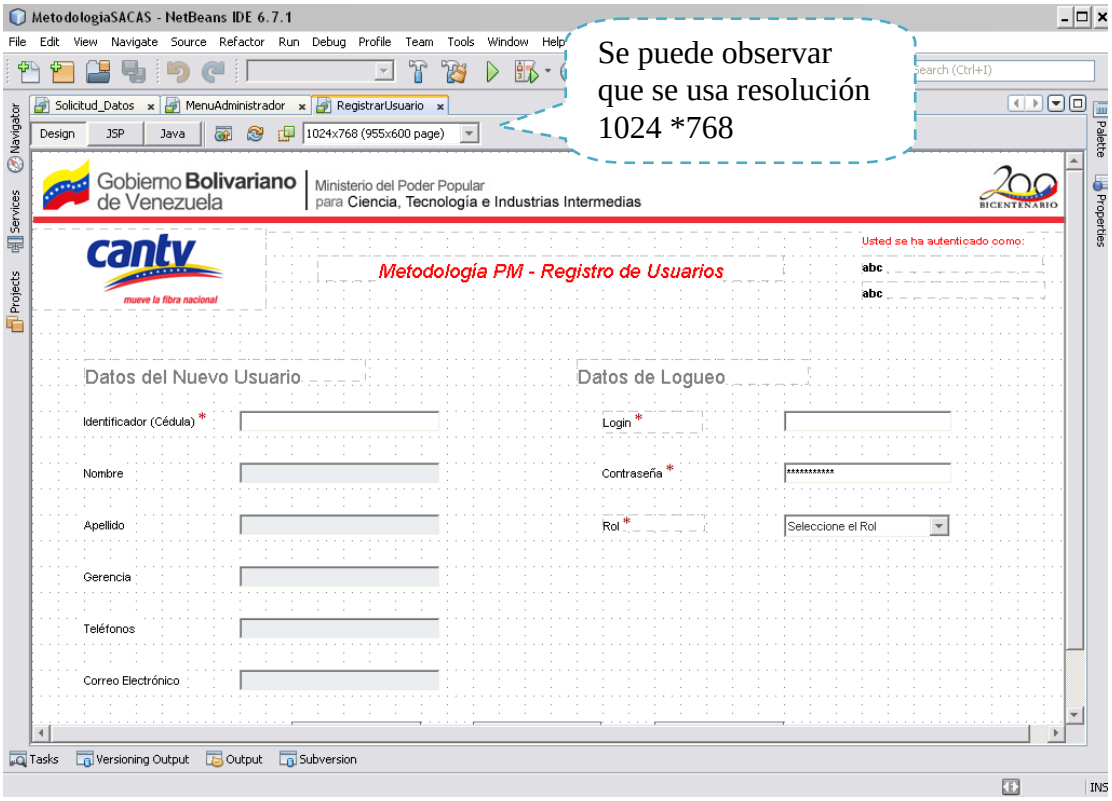


Figura 101: Observando desde NetbeansIDE

CONCLUSIONES

Luego de terminar este TEG podemos decir que se lograron alcanzar los objetivos planteados, en los cuales llegamos a concluir que:

La Tecnología de Información ha evolucionado de manera exponencial en los últimos años. Ha sido capaz de producir muy buenos resultados para la optimización de procesos empresariales de acuerdo a datos arrojados por análisis de desempeño.

Con respecto a primer objetivo específico, los procesos de negocio se han convertido en la médula espinal del futuro de una organización empresarial. Es por ello que han surgido diferentes técnicas que abordan dicha situación o, si se quiere, el mencionado análisis de procesos.

Soluciones como la Gestión de Procesos de Negocio han sido estrategias bastante acertadas para lograr una optimización de procesos de negocio pero no se ha asociado esta actividad con el desarrollo de software que permita la implementación de los ideales presentes en el flujo de trabajo. Además, al momento de crear un software se evalúa sólo el producto final y se deja a un lado el proceso de desarrollo del mismo, sin establecer métricas u experiencias exitosas que originen la calidad adecuada tanto del producto final como del proceso de desarrollo.

En relación al segundo objetivo; es necesario un repositorio de patrones, a fin de lograr la reutilización de los componentes en otros proyectos, a través, de su adaptación y la asociación de un modelo de calidad, permitiendo la trazabilidad y optimización de cada proceso de negocio.

Se definieron los requerimientos funcionales y no funcionales para el desarrollo de todo sistema, estudiar y analizar paso a paso que se necesita para ser implantado y así corregir o avalar la hipótesis de la metodología de Gestión de Proceso de Negocio sustentada en el uso de Patrones.

Con respecto al tercer objetivo, se logró extraer información proveniente del Sistema de Administración y Recuperación de Patrones. Sin embargo, se hace

necesario indagar sobre las métricas de calidad que posee cada patrón ya que no están definidas de manera clara en la mencionada aplicación.

Con respecto al cuarto objetivo, se estudió y analizaron correctamente la herramienta (Intalio BPMS), que ayudaron a implantar dicho sistema. Destacando su facilidad para generar lógica de negocio a través de la notación BPMN y la generación automática de Servicios Web con cada una de sus funcionalidad.

En cuanto al quinto objetivo, se logró desarrollar parte del sistema, se diseñó la lógica de negocio y logró observar su funcionalidad, para el caso de estudio que fue la Compañía Anónima de Teléfonos de Venezuela. Permitió mirar que sucede cuando se realiza una incidencia y cambio dentro la empresa.

Se logró configurar y desplegar el motor de ejecución para observar los procesos que se generaron para la implantación del sistema (séptimo objetivo).

Se realizaron pruebas y las mismas fueron satisfactorias ya que cumplieron con los requerimientos y actualmente se encuentra en fase de prueba.

La metodología de desarrollo escogidas fue correcta, ya que permitió agregar o adaptar según el requerimiento que iba surgiendo en cada fase, y así, un efectivo diseño de la solución.

La combinación de los beneficios que ofrece la herramienta Intalio BPMS junto al concepto de BPM, permitió lograr los objetivos propuestos, debido a que maneja flujo de trabajos usando la notación BPMN y la ejecución de tareas.

Como la aplicación desarrollada genera información que puede ser tomada como métricas, por el estado que se encuentra al momento del diseño del proceso de desarrollo, la misma, puede ser integrada con alguna aplicación de inteligencia de negocio, se recomienda pentaho por su particularidad de permitir el desarrollo de dashboard, que permitan visualizar el estado en que se encuentra cada objeto consultado.

RECOMENDACIONES

Como recomendaciones tenemos:

- Integrar la solución con el proceso de Gestión de Cambio de la Organización.
- Se propone la creación de los módulos de Administrar Proceso para complementar el sistema.
- Se propone integrarlo con el manejo de Inteligencia de Negocios, usando la herramienta Open Source Pentaho, para la extracción, transformación y carga de los datos (ETL).
- Se recomienda integrar la solución a un Directorio Activo de la Empresa, a fin de establecer políticas de seguridad en cuanto a los recursos, sobre todo en cuanto al acceso a los Servicios Web generados por el BPMS Intalio.

REFERENCIAS BIBLIOGRÁFICAS

Acosta, E. y Zambrano, N. (2001). *Patrones de Interfaces: un concepto integrador en el desarrollo de aplicaciones*. XXVII Conferencia Latinoamericana de Informática CLEI´2001. Mérida.

Acosta, E. y Zambrano N. (2004). *Patterns and Objects for User Interface Construction*, in Journal of Object Technology, vol. 3 no. 3. 75-90.

Alexander, C. (1979). *The Timeless Way of Building*. Oxford University Press.

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fidsdahl-King, I., Angel, Sh. (1977). *A Pattern Language*. Oxford University Press, New York.

Aníbarro C. (2001). *Manual de HTML Versión 1.3*

Barros, O. (2002). Arquitectura de Aplicaciones en e-Business. Documento de Trabajo Nº 26, CEGES, Dpto. de Ingeniería Industrial, U. de Chile.

Barros, O. (2003). *Rediseño de Procesos de Negocios usando Patrones*, (2da ed.), 13-31, Editorial DOLMEN, Chile 2003.

Bonillo P. (2008) *Metodología de Gestión de Procesos de Negocio sustentada en el Uso de Patrones*. Tesis de Doctorado. Facultad de Ciencias. Universidad Central de Venezuela.

Bosch, J. (2000). *Design & Use of Software Architectures*. Addison Wesley.

Delgado A. *Desarrollo de Software con enfoque en el Negocio.* Universidad de la República Montevideo – Uruguay, Facultad de Ingeniería, Instituto de Computación.

Centro Nacional de Tecnología de Información. (2007). *Filosofía del Software Libre.* Venezuela 2007.

Cobo, C. (2006). *SOA (Clave para una Apreciación Objetiva)*. Sopra Profit.

Gamma, E., Helm, R., Johnson, R., y Vlissides, J.(1995). *Design Patterns*. Addison Wesley.

ISO/IEC 9126.(2001). *Information Technology-Software Product Evaluation-Quality Characteristics and Guideline for Use.*

ISO/IEC 9126-1. (2001). *Software Engineering-Product Quality. Part 1: Quality Model*.

ISO/IEC: FCD 9126-1.2.(1998). *Information Technology - Software Product Quality. Part 1: Quality Model*.

ISO/IEC14598-3.(1999). *Information Technology - Software Product Evaluation - Part 3: Process for Developers. Software Engineering*.

Itamar W. *Avaliação do uso de ferramentas de workflow em processos típicos de engenharia de software*. Arcadian Tecnologia S/A.

ITIL. *Best Practice for Application Management.* (2da. ed.) Londres, Inglaterra: TSO (The Stationery Office) 2002.

Kellet A.(2007). *Oracle BPM*. Boutler Group.

Kenneth C. Laudon y Jane P. Laudon, (1996). *Sistemas de Información Gerencial*. Pearson Prentice Hall.

Metastorm INC. (2007). *Metastorm Bpm Versión 7.5*. Metastorm Inc.

Ridao, M. (2001). *Uso de Patrones en el Proceso de Construcción de Escenarios*. Tesis Maestría en Ingeniería de Software.

Runa Wfe. (2008). *RUNA WFE* – an Open Source Enterprise Business Process Management (BPM) System Versión 2.1. Runa Wfe

Mejía M. y Alzarte L. *Automatización de Procesos de Negocio utilizando un BPMS.* Instituto Tecnológico Autónomo de México.

Saroka Raúl Horacio, (2002). *Sistemas de Información en la era digital*. Fundación OSDE.

Schildt H. (2001). *Java 2 Manual de Referencia*. Editorial Osborne MacGraw-Hill.

Shenone M. (2004). *Diseño de una Metodología Ágil de Desarrollo de Software*. Universidad de Buenos Aires, Facultad de Ingeniería.

Tamayo, M. (2000). El Proceso de Investigación Científica. Limusa Noriega Editores. México.

Tapscott, D. (1998*). La Economía Digital*, McGraw-Hill S.A., Bogotá.

Vollmer, K. (2004). Market Overview: *Business Process Management*, Giga Research, Planning Assumption, RPA-022004-00001.

Zorzan F. y Riesco D. *Automatización de Procesos de Desarrollo de Software definidos con SPEM.*

BPM. [En Línea] [Citado el 15 de Diciembre de 2008] http://www.oracle.com/global/es/products/technologies/bpm/index.html

BPM Jbpm. [En Línea] [Citado el 18 de Febrero de 2009] http://www.jboss.com/products/jbpm/

Intalio . [En Línea] [Citado el 15 de Marzo de 2009] http://www.intalio.com/products/

Características de UP Ágil. [En Línea] [Citado el 28 de Diciembre de 2008] http://www.ambysoft.com/unifiedprocess/agileUP.html

JavaServer Faces [En Línea][Citado el 14 de Octubre de 2010] http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html

ANEXOS

Anexo 1 Descripción Taxonomía de Patrones

A continuación, en este anexo se explican con mayor detalle cada uno de los patrones que integran la taxonomía propuesta: análisis, arquitectural, diseño e interacción. Al final, se presentan los patrones de flujo de trabajo como escenarios de los patrones de análisis, por lo que no se incluyen los mismos dentro de la taxonomía. De igual forma no se incluye el tema particular de los lenguajes, dado que es específico por tecnología, sin embargo es conveniente mencionar que para la mayoría de los BPMS, el *lenguaje* es JAVA.

|*A1.1 Patrón de Análisis.*

Según Alexander, "Cada patrón describe un problema que ocurre una y otra vez, y describe la solución a ese problema, de tal manera que dicha solución pueda ser usada un millón de veces más, sin hacerlo necesariamente dos veces del mismo modo" (Alexander et al., 1977).

La idea central de la construcción de escenarios utilizando patrones consiste en estudiar las situaciones desde un punto de vista diferente al utilizado por otras técnicas de elaboración de escenarios. Se pretende tener una visión más conceptual y estructural de las situaciones, con el fin de identificar la naturaleza intrínseca de las mismas. Con esa visión, será posible determinar el tipo de escenario correspondiente a cada situación y así, elegir un patrón de un catálogo, reutilizar su estructura con el fin de derivar el escenario más fácil y directamente.

Los patrones son presentados como escenarios descritos a los que se ha agregado un reducido número de reglas de conformación como meta componentes del escenario. Cada componente del escenario, según la estructura definida en (Leite et al., 2000), ha sido completado con un texto nominal que se espera sea reemplazado en el escenario real generado al usar el patrón pero que a su vez guíe en la redacción del componente. (Ridao, 2001). En la figura A1.1 se presenta el patrón de Producción y en la figura A1.2 el escenario DISEÑAR LA AGENDA DE REUNIONES correspondiente al caso de estudio Sistema de Agenda de Reuniones.

PATRÓN: **PRODUCCIÓN**

TÍTULO:	**Realización de una actividad productiva**
OBJETIVO:	Producir un efecto sobre el macrosistema
CONTEXTO:	Ubicación geográfica: generalmente el lugar de trabajo del actor principal Precondiciones: puede tener precondiciones Ubicación temporal: generalmente determinado por el actor principal y posiblemente prolongado
ACTORES:	Varios, al menos uno
RECURSOS:	Al menos uno, generalmente muchos
EPISODIOS:	**POR LO MENOS DOS COMO EL SIGUIENTE** Un actor realiza alguna actividad que produce algún efecto sobre el macrosistema. **PUEDEN ESTAR EN SECUENCIA O CONSTITUIR GRUPOS NO SECUENCIALES**
EXCEPCIÓN:	Circunstancia que obstaculiza el cumplimiento del objetivo

Figura A1.1: Descripción del Patrón Producción. (Ridao, 2001)

TÍTULO:	**Diseñar la agenda de reuniones**
OBJETIVO:	Determinar los requerimientos, oportunidad y lugar de la reunión
CONTEXTO:	Debe presentarse previamente la necesidad de una reunión
ACTORES:	convocante secretaria
RECURSOS:	agenda de reuniones listado para convocatoria horarios disponibles temario
EPISODIOS:	1. [El convocante obtiene los datos de la reunión a diseñar del esquema de base] 2. **#SI** los horarios disponibles de los convocados no están registrados **ENTONCES** SOLICITAR HORARIOS DISPONIBLES 3. El convocante consulta en su agenda de reuniones sus horarios disponibles# 4. ESTABLECER LA FECHA DE REUNIÓN 5. [El convocante determina el material para repartir] 6. #El convocante o la secretaria registran en la agenda el objetivo, la fecha, la hora, el lugar, los temas a discutir, los convocados, el material a presentar y el material para repartir 7. [El convocante confecciona el temario] 8. GENERAR EL LISTADO PARA CONVOCATORIA 9. El convocante o la secretaria registran la reunión en el cronograma de reuniones 10. El convocante o la secretaria reservan el espacio físico 11. [El convocante o la secretaria reservan el material físico] 12. [El convocante o la secretaria registran el material físico en el cronograma de reuniones]#
EXCEPCIONES:	Conflictos en los horarios disponibles de los convocados Conflictos en la disponibilidad de espacio Conflictos en la disponiblidad de material físico

Figura A1.2: Ejemplo de un Escenario de Tipo Producción. (Ridao, 2001)

En cada uno de los ejemplos se puede observar que todos los componentes, además de los episodios, se ajustan a la descripción de los mismos en el patrón correspondiente. Analizando los episodios del escenario de la Figura A1.2 puede verse que se trata de producciones. Los episodios 2, 4 y 8 corresponden a subescenarios, que fueron analizados en forma independiente y resultaron ser también casos de producción. Por ello es posible determinar que se trata de un ejemplo puro de Producción.

A continuación un listado de los patrones de Análisis:

1. Producción
2. Servicio
3. Colaboración
4. Negociación Inconclusa
5. Negociación Inconclusa con Disparador de Escenarios
6. Fin de Negociación
7. Etapa de Negociación
8. Etapa de Negociación con Disparador de Escenarios
9. Negociación Terminada
10. Producción + Servicio + Colaboración
11. Negociación inconclusa con producción o servicio o colaboración
12. Fin de Negociación con producción o servicio o colaboración
13. Etapa de Negociación con producción o servicio o colaboración
14. Negociación terminada con producción o servicio o colaboración
15. Negociación inconclusa con Disparador de Escenarios y producción o servicio o colaboración
16. Etapa de Negociación con Disparador de Escenarios y producción o servicio o colaboración
17. Un tablero de anuncios sencillos
18. Un tablero de anuncios estructurados
19. Biblioteca Virtual
20. Biblioteca Virtual con Agentes Activos
21. Administrador de sub. Tarea
22. Asignación Automática Secuencial con Escalación
23. Asignación Automática Secuencial
24. Asignación con Escalación

25. Asignación Paralela Revisión Final con Escalación
26. Asignación Paralela
27. Asignación sin Notificación
28. Asignación
29. Patrones Básico de Control
30. Secuencia
31. División Paralela
32. Sincronización
33. Elección Exclusiva
34. Fusión Sencilla
35. Ramificación avanzada y Patrones de Sincronización
36. Elección Múltiple
37. Fusión Sincronizada
38. Fusión Múltiple
39. Discriminador
40. Enlace de N a M
41. Patrones Estructurales
42. Ciclos Arbitrarios
43. Terminación Implícita
44. Patrones de Cancelación
45. Cancelar Actividad
46. Cancelar Caso
47. Patrones que implican Múltiples Instancias
48. MI con conocimiento previo al momento de diseño
49. MÍ con conocimiento previo en la ejecución
50. MÍ sin conocimiento previo en la ejecución
51. MÍ sin sincronización
52. Patones basados en Estados
53. Elección diferida Choice
54. Enrutamiento paralelo Interpolado

A1.2 Patrón Arquitectural

Es frecuente que se confundan los términos: estilo y patrón arquitectural; a continuación se profundizara en el término patrón arquitectural, a fin de establecer las diferencias (Tabla A1.1).

Patrón arquitectural. Buschmann et al. (1996) lo define como una regla que consta de tres partes, la cual expresa una relación entre un contexto, un problema y una solución:

- Contexto. Es una situación de diseño en la que aparece un problema de diseño.
- Problema. Es un conjunto de fuerzas que aparecen repetidamente en el contexto.
- Solución. Es una configuración que equilibra estas fuerzas.

Estilo Arquitectónico	Patrón Arquitectónico
Sólo describe el esqueleto estructural y general *para aplicaciones*	Existen en varios rangos de escala, comenzando con patrones que definen la estructura básica de *una* aplicación
Son independientes del contexto al que puedan ser aplicados	Partiendo de la definición de *patrón*, requieren de la especificación de un contexto del problema
Cada estilo es independiente de los otros	Depende de patrones más pequeños que contiene, patrones con los que interactúa, o de patrones que lo contengan
Expresan técnicas de diseño desde una perspectiva que es independiente de la situación actual de diseño	Expresa un problema recurrente de diseño muy específico, y presenta una solución para él, desde el punto de vista del contexto en el que se presenta
Son una categorización de sistemas	Son soluciones generales a problemas comunes

Tabla A1.1 Diferencias entre estilo y patrón arquitectónico

Para Buschmann et al. (1996) son plantillas para arquitecturas de software concretas, que especifican las propiedades estructurales de una aplicación y tienen un impacto en la arquitectura de subsistemas.

La selección de un patrón arquitectónico es, por tanto, una decisión fundamental de diseño en el desarrollo de un sistema de software. Algunos patrones arquitecturales se muestran en la Tabla A1.2.

Patrón Arquitectónico	Descripción
Capas	Consiste en estructurar aplicaciones, las cuales pueden ser descompuestas en grupos o sub-tareas y se clasifican de acuerdo con un nivel particular de abstracción.
Tuberías y Filtros	Provee una estructura para los sistemas que procesan un flujo de datos. Cada paso de procesamiento está encapsulado en un componente filtro. El dato pasa a través de conexiones (*tuberías*), entre filtros adyacentes.
Pizarra	Aplica para problemas cuya solución utiliza estrategias no determinísticas. Varios subsistemas ensamblan su conocimiento para construir una posible solución parcial ó aproximada.
Agente	Puede ser usado para estructurar sistemas de software distribuido con componentes desacoplados que interactúan por invocaciones a servicios remotos. Un componente Agente es responsable de coordinar la comunicación, el reenvío de solicitudes y la transmisión de resultados y excepciones.
Modelo Vista-Controlador	Divide una aplicación interactiva en tres componentes. El modelo contiene la información central y los datos. Las vistas despliegan información al usuario. Los controladores capturan la entrada del usuario. Las vistas y los controladores constituyen la interfaz del usuario.
Control de Abstracción-Presentación	Define una estructura para sistemas de software interactivos de agentes de cooperación, organizados de forma jerárquica. Cada agente es responsable de un aspecto específico de la funcionalidad de la aplicación y consiste en tres componentes: presentación, abstracción y control.
Micronúcleo	Aplica para sistemas de software que deben estar en capacidad de adaptar los requerimientos de cambio del sistema. Separa un núcleo funcional mínimo del resto de la funcionalidad y de partes específicas pertenecientes al cliente.
Reflexión	Provee un mecanismo para sistemas cuya estructura y comportamiento cambia dinámicamente. Soporta la modificación de aspectos fundamentales como estructuras, tipos y mecanismos de llamadas a funciones.

Tabla A1.2 Patrones Arquitecturales

A continuación un listado de los estilos arquitecturales:

1. Estilos de Flujo de Datos
2. Tubería y filtros
3. Estilos Centrados en Datos
4. Arquitecturas de Pizarra o Repositorio
5. Estilos de Llamada y Retorno
6. Modelo Vista-Controlador (MVC)
7. Arquitecturas en Capas
8. Arquitecturas Orientadas a Objetos
9. Arquitecturas Basadas en Componentes
10. Estilos de Código Móvil
11. Arquitectura de Máquinas Virtuales
12. Estilos heterogéneos
13. Sistemas de control de procesos
14. Arquitecturas Basadas en Atributos
15. Estilos Par-a-Par
16. Arquitecturas Basadas en Eventos
17. Arquitecturas Orientadas a Servicios
18. Arquitecturas Basadas en Recursos

Seguidamente un listado de los patrones arquitecturales:

1. Capas
2. Tubería-filtros
3. Pizarra
4. Agente
5. Modelo Vista-Controlador
6. Control de Abstracción-Presentación
7. Reflexión (metanivel que hace al software consciente de sí mismo)
8. Micronúcleo (núcleo de funcionalidad mínima)

A1.3 Patrones de Diseño

Los patrones de diseño, se ubican en el nivel de diseño detallado, estos describen un problema a ser resuelto, una solución y el contexto en el cual la solución trabaja. Nominan una técnica y describen su costo y su beneficio.

Según Gamma et al. (1995) un patrón de diseño tiene cuatro elementos esenciales:

- El nombre: a cada patrón, se le ha asignado una denominación que permite que los entendidos en el tema, puedan conversar usando un diccionario común. Permite un mayor grado de abstracción y permite la comunicación entre colegas, construyendo un lenguaje compartido. Según GoF, uno de los problemas que tuvieron, fue encontrar un nombre apropiado para cada patrón que catalogaron.
- El problema: se explica el problema original, y su contexto. Puede describir desde detalles específicos, como algoritmos, o clases y estructuras que se han encontrado inflexibles a la hora de implementarse.

  Todo patrón nace de un problema a solucionar.
- La solución: cada patrón puede tener una solución a un problema. Un mismo problema real, puede tener dos soluciones parecidas, correspondientes a dos patrones (muchas veces pasa esto con el Abstract Factory ó Construcción abstracta, versus el Factory Method ó Método de Construcción, por ejemplo). Pero la elección seguramente recaerá en el patrón que mejor se adapte al contexto particular del problema que se tenga. La solución que un patrón describe, no necesariamente es detallada al nivel de implementación, sino que provee una descripción abstracta, una enumeración de elementos y sus relaciones, para solucionar el problema planteado.
- Las consecuencias: son los resultados de aplicar el patrón, las compensaciones, compromisos, que se tienen que aceptar al adoptar el mismo.

Las Propiedades de los patrones de diseño son: (Buschmann et al., 1996)

- Tratan los problemas del diseño que se presentan en situaciones específicas y se repite de manera recurrente, mostrando una solución.
- Documentan la experiencia de soluciones probadas a problemas existentes.
- Identifican y especifican las abstracciones sobre el nivel, separándolos en

- objetos, casos o componentes.
- Proveen un vocabulario común y la comprensión de los principios del diseño.
- Son los medios para documentar arquitecturas de software.
- Apoya la creación de software con propiedades definidas.
- Ayudan en la construcción de arquitecturas complejas y heterogéneas.
- Ayudan en el manejo de software complejo.

Gamma y otros (Gamma et al., 1995), consideran que para describir patrones se debe especificar:

- Nombre del patrón y clasificación. Cada patrón tiene un nombre, y una categoría a la que pertenece. El GoF los clasifica en creacionales, estructurales y de conducta. Más adelante se profundizará sobre esta clasificación.
- Intención. Depende del contexto. Cada patrón tiene una intención, una razón, una justificación.
- Otros nombres. Los patrones habían surgido en distintos proyectos y tecnologías. Los primeros autores les dieron un nombre que no siempre coincidía. Estos otros nombres pueden ser enumerados en la descripción del patrón.
- Motivación. Más que la intención, esta parte describe un escenario, un problema en particular que aparece, y que ayuda a entender la descripción algo más abstracta del patrón genérico.
- Aplicación. En qué situaciones puede ser aplicado un patrón. Pueden darse ejemplos de malos diseños, que pueden beneficiarse de la aplicación de este patrón en particular.
- Estructura. La parte más conocida, luego del nombre. Se adopta un diagrama casi siempre basado en UML.
- Participantes. La lista de las clases y objetos que participan del patrón de diseño, y las responsabilidades que tienen.
- Colaboraciones. Descripción de cómo los participantes colaboran para llevar a cabo sus responsabilidades en el patrón
- Consecuencias. Explica cómo el patrón cumple con sus objetivos, y que compromisos se asumen.
- Implementación. Esta es la parte que más puede variar, porque para cada patrón puede haber varias posibles implementaciones, incluso, diferentes implementaciones según la tecnología adoptada.
- Código de ejemplo. Es recomendable dar código de ejemplo de cada patrón, por ejemplo, en Smalltalk o en C++.

- Usos conocidos. Un patrón no es tal, si no ha sido ya empleado en algún caso real. Se debe incluir por lo menos dos ejemplos conocidos de diferentes ámbitos.
- Patrones relacionados. Puede que haya problemas parecidos, que tengan más de una solución. De ahí que muchos patrones estén relacionados entre sí: como el caso del Proxy y del Adaptador. Se trata de explicar cuáles son las similitudes y (no menos importante) las diferencias, entre patrones relacionados.

Gamma et al. (1995), clasifican los patrones de diseño en tres categorías: patrones creacionales, estructurales y de conducta:

- Patrones Creacionales (Creational Patterns). Abstraen el proceso de instanciación. En general, tratan de ocultar las clases y métodos concretos de creación, de tal forma que al variar su implementación, no se vea afectado el resto del sistema. Es común encontrar "competencia" entre estos patrones: hay más de un patrón a adoptar ante una situación: Construcción abstracta, Constructor, Método de Construcción, Prototipo, Singleton.
- Estructurales (Structural Patterns). Se ocupan de cómo se agrupan las clases y los objetos para formar estructuras más grandes. Aquí se puede nombrar al patrón Composite, que permite agrupar varios objetos como si fueran uno solo, y tratar al objeto compuesto de una forma similar al simple: Adaptador, Puente, Compositor, Decorador, Fachada, Proxy.
- De Conducta (Behavioral Patterns). Centrados en la comunicación entre objetos y clases Frecuentemente, describen las colaboraciones entre distintos elementos, para conseguir un objetivo: Cadena de Responsabilidad, Comando, Interpretador, Iterador, Mediador, Observador, Estado, Estrategia, Método de Plantilla, Visitador.

Por otro lado, Buschmann et al. (1996), clasifican los patrones de diseño como: descomposición de estructura, organización de trabajo, control de acceso, gerencia y comunicación:

- Descomposición de la estructura (Structure Decomposition). Descomposición de los subsistemas y componentes complejos en piezas interrelacionadas: Partes Completas.
- Organización de trabajo (Organization of Work). Los componentes colaboran para solucionar problemas complejos: Maestro-Esclavo.
- Control de acceso (Access Control). Guarda y controla el acceso a los servicios y a los componentes: Proxy.
- Gerencia (Management). Maneja la totalidad de las colecciones homogéneas de objetos, servicios y componentes: Procesador de Comando, Manejador de Vistas.

- Comunicación (Communication). Organiza la comunicación entre componentes: Adelantar-Recibir, Cliente-Despachador-Servidor, Publicador-Subscriptor

La Tabla A1.4, muestra algunos patrones de diseño acompañados de sus autores y su propósito. A manera de conclusión se tiene que existe una estrecha relación entre los estilos arquitectónicos y los patrones arquitecturales y de diseño.

N	Patrón de diseño	Autores	Propósito
1	Construcción Abstracta	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Proporcionar interfaces para crear objetos de alguna familia, sin especificar una clase en particular.
2	Adaptador de Clase	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Convertir interfaces de una clase, en otra, que es la esperada por algún cliente.
3	Puente	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Desacoplar una abstracción de su implementación en concreto. Luego, podemos cambiar la implementación, o la abstracción, sin cambiar la otra.
4	Constructor	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Separa la construcción de un objeto complejo, de su representación. De esa manera, el mismo proceso de construcción puede crear diferentes resultados.
5	Cadena de Responsabilidad	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Desacoplar el mensaje enviado de su receptor, permitiendo que haya varios objetos que tengan la oportunidad de manejar el requerimiento. Eso se consigue pasando el requerimiento por la

			cadena de objetos hasta llegar al encargado de atenderlo.
6	Cliente Despachador-Servidor	Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal	Un *despachador* actúa como capa intermedia entre los *clientes* y el *servidor*.
7	Comando	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Encapsular el requerimiento a un objeto, permitiendo incluso el deshacer de la operación
8	Procesador de Comando	Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal	Extiende el patrón comando de Gamma, y añade un procesador del comando explícito
9	Compositor	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Componer objetos en una estructura de árbol, donde los objetos compuestos se tratan de forma similar a los objetos simples.
10	Decorador	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Agregar responsabilidad a un objeto dinámicamente, dándonos una alternativa a la extensión de una clase, en lugar de usar subclases.
11	Fachada	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Proveer una interface unificada a un conjunto de funciones de un subsistema. Es una interface de alto nivel, para facilitar el uso del subsistema.
12	Método de Construcción	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Definir interfaces para crear objetos, como en la Construcción abstracta, pero se delega a las subclases implementar la creación

			en concreto.
13	Flyweight	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Compartir objetos eficientemente, sin repetirlos en el sistema.
14	Adelantar-Recibir	Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal	Contiene toda la funcionalidad de comunicación especificada en el sistema, separada en pares de componentes que pueden comunicarse sin afectar la portabilidad.
15	Interpretador	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Construir una representación de la gramática de un lenguaje, junto con su intérprete.
16	Iterador	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Construir una representación de la gramática de un lenguaje, junto con su intérprete.
17	Maestro-esclavo	Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal	El *master* divide una tarea entre los *slave* idénticos (pero independientes), la combinación de los resultados parciales de cada slave, resulta ser la solución
18	Mediador	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Permitir la interacción de varios objetos, sin generar acoples fuertes en esas relaciones.
19	Memento	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Permite capturar el estado de un objeto sin entrar en su estructura interna, para, por ejemplo, restaurarlo más adelante.
20	Observador	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Definir una relación uno a muchos, entre un objetos y otros que están interesados en sus cambios, sin realizar el acople entre los mismos.
21	Prototipo	Erich Gamma, Richard Helm,	Conseguir otras instancias del

		Ralph Johnson, and John Vlissides	objeto, mediante una instancia prototípica.
22	Proxy	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Proveer un subrogado a otro objeto, para controlar el acceso al mismo.
23	Publicador-Suscriptor	Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal	Es similar al patrón *Observador* de Gamma.
24	Singleton	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Consigue dar un solo objeto de la clase, en cualquier momento de la aplicación
25	Estado	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Permite a un objeto cambiar su conducta cuando cambia su estado interno, simulando que cambia de clase.
26	Estrategia	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Definir una familia de algoritmos, y los hace intercambiables.
27	Método de plantilla	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Definir la estructura de una operación, cuyas operaciones más básicas, quedan delegadas en subclases.
28	Manejador de Vistas	Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal	Separar el manejo de opiniones de los códigos requeridos para presentar o para controlar vistas específicas. Es similar a los patrones *Construcción Abstracta* y *Mediador* de Gamma et al
29	Visitador	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides	Recorrer una estructura (un árbol, por ejemplo), aplicando una operación
30	Parte Completa	Frank Buschmann, Regine Meunier, Hans Rohnert, Peter	Definir un componente que encapsule objetos pequeños. Evita

		Sommerlad, and Michael Stal	que los clientes tengan acceso directo al contenedor de objetos, pero proporciona una interfaz para la agregación.

Tabla A1.4 Algunos patrones de diseño

La Tabla A1.5 muestra que los patrones de diseño pueden aplicarse al nivel de marco de trabajo, dentro de las clases de diseño al nivel de arquitectura y también dentro del diseño detallado [Bra2003].

Arquitectura (Garlan y Shaw)

Categoría	Sub-categoría	Patrones de diseño más aplicados
Flujo de datos	Secuencial por lote	Puede aplicarse decorador
Componentes Independientes	Tuberías y Filtros Procesos de comunicación Paralela Sistema Cliente-Servidor Sistema de eventos	Observador Fachada Estado Observador
Máquinas virtuales	Intérpretes Sistemas basados en reglas	Interpretador
Arquitecturas de Almacenamiento	Bases de datos Sistemas de hipertextos Pizarrones	Observador, iterador Decorador Pizarra
Arquitecturas por capas		Muchos patrones de diseño consisten en capas abstractas y no abstractas.

Tabla A1.5 Arquitecturas alternativas (Braude, 2003)

A continuación un listado de los patrones de Diseño:

1- Construcción Abstracta	28- Control de Presentación-Abstracción
2- Clase Adaptador	29- Prototipo
3- Objeto Adaptador	30- Proxy
4- Pizarra	31- Publicador-Subscriptor
5- Puente	32- Reflexión
6- Agente	33- Singleton
7- Constructor	34- Estado
8- Cadena de Responsabilidad	35- Estrategia
9- Cliente Despachador-Servidor	36- Método de Plantilla
10- Comando	37- Manejador de Vistas
11- Procesador de Comando	38- Visitador
12- Compositor	39- Parte Completa
13- Decorador	40- Administración de Caché
14- Fachada	41- Caja de Objetos (Object Pool)
15- Método de Construcción	42- Delegación (Delegation)
16- Flyweight	43- Ejecución de un único hilo
17- Adelantador-Recibidor	44- El patrón Controlador
18- Interpretador	45- El patrón Creador
19- Iterador	46- El patrón Experto
20- Capas	47- Enlace Dinámico (Dynamic Linkage)
21- Maestro-Esclavo	48- Fotografía (Snapshot)
22- Mediador	49- Inicialización de capas (Layered Initialization)
23- Memento	50- Inmutable (Immutable)
24- Micronúcleo	51- Interfaz (Interface)
25- Modelo Vista-Controlador	52- Marcador de Interfaz
26- Observador	53- Pequeño Lenguaje (Little Language)
27- Filtros y tuberías	54- Proxy Virtual (Virtual Proxy)
	55- Término en dos fases (Two-Phase Termination)

A1.4 Patrón de Interacción

Un patrón de diseño de interfaces, o de interacción, modela un aspecto referente a la interfaz de una aplicación. Estos patrones sirven para describir un problema, su contexto y la solución; para generalizar una solución; para facilitar la comunicación entre disciplinas; para registrar el conocimiento y la experiencia y facilitar las metodologías de diseño de interfaces.

Los patrones de interacción deben ser legibles y entendibles por todos los integrantes del equipo de desarrollo de un proyecto, incluyendo a los expertos en el dominio de la aplicación y los usuarios del sistema. Para lograr alcanzar esas características, estos patrones deben estar expresados haciendo uso de una notación clara, sencilla y bien definida; adicionalmente, los patrones deben estar organizados de tal forma que se facilite su utilización. Entonces, la estructura de los patrones debe facilitar la comunicación. A fin de cumplir su objetivo, existe aceptación en que un patrón tenga, como mínimo, cinco componentes principales: (Acosta et al., 2001).

- Nombre de patrón, el cual debe ser mnemotécnico con respecto al problema de diseño que resuelve. Un nombre de patrón nuevo incrementa el vocabulario de diseño. Este vocabulario permite diseñar a un alto nivel de abstracción y facilita la comunicación entre colegas y dentro de la documentación del diseño.
- Problema, describe una situación en la cual se debe aplicar el patrón; éste explica el problema en sí y su contexto. Algunas veces, el problema puede incluir condiciones que deben ser satisfechas antes de que tenga sentido aplicar el patrón.
- Solución, describe los elementos que conforman el diseño. Provee una descripción abstracta de un ordenamiento general de los elementos que resuelven un problema particular en un contexto dado.
- Contexto, condición(es) en la(s) cual(es) se puede usar el patrón.
- Fuerzas, complementan la definición del problema, debido a que son aspectos del contexto de diseño que deben ser optimizados; en general, estos aspectos son contradictorios y es necesario establecer un equilibrio entre estos.

Uno de los resultados del workshop ChiliPLoP'99 fue el establecimiento de una taxonomía, propuesta inicialmente por Borchers, que agrupa a los patrones con base en tres dimensiones: nivel de abstracción, funcionalidad y dimensión física. Dentro de cada dimensión, se clasifican los patrones de la siguiente manera:

- Según el nivel de abstracción, se puede tener patrones de muy alto nivel que describen una tarea completa que realiza el usuario; luego, en un segundo nivel de abstracción se tienen patrones más concretos que describen el estilo de cierta parte de la interacción, y, por último, patrones que describen objetos de la interfaz de usuarios.
- En lo que respecta a la funcionalidad de los patrones, estos se agrupan en las siguientes categorías: aquellos orientados a describir la percepción (auditiva, visual, etc.) de los usuarios; los patrones que expresan la manipulación de los datos de entrada o salida y aquellos dirigidos a mostrar la navegación a través de la aplicación.
- Por último, se tiene la dimensión física de la interfaz, esto es, en primer lugar se encuentran los patrones orientados a la distribución de los elementos en el espacio o plano físico, posteriormente, los patrones que representan secuencias o series discretas de eventos, de diálogos, etc., y aquellos patrones que muestran continuidad en el tiempo, tales como el diseño de buenas técnicas de animación en la interfaz de usuarios.

Mahemoff y Johnston (1998) proponen un enfoque de diseño de Interacción Humano Computador, basado en patrones orientados a la usabilidad. Estos autores definen cuatro clases de patrones IHC, a saber: Patrones del Sistema, Patrones de Tareas, Patrones de Elementos de la Interfaz, Patrones de Usuarios. Los patrones del sistema capturan aspectos concernientes al desarrollo del mismo, por ejemplo los atributos de usabilidad que se deben garantizar, el propósito del sistema, etc. Los patrones de tareas describen las funcionalidades de la aplicación y cómo se deben llevar a cabo. Los patrones de elementos de interfaz describen elementos tales como los conocidos "widgets" o composiciones de éstos. Los patrones de usuarios describen los perfiles de los usuarios potenciales de la aplicación.

A1.5 Patrón de Flujo de Trabajo

Los patrones de flujo de trabajo o "workflow", se corresponden con un conjunto de actividades básicas que permiten el control de un proceso de negocio, y son escenarios de los patrones de análisis descritos en el Anexo 1.1 (por lo que no se mencionan en la taxonomía), estos patrones de "workflow" son: aprobación simple, aprobación simple con escalación automática, aprobación simple con renovación automática, flujo de trabajo secuencial, flujo de trabajo secuencial con escalación automática, flujo de trabajo paralelo, flujo de trabajo paralelo con revisión final, asignación manual y notificación.

Todos estos patrones de flujo de trabajo, permiten separar la lógica de negocio y la de presentación, garantizando la meta configuración de las aplicaciones asociadas a los procesos de negocio, a continuación se presentan cada uno de estos patrones de flujo de trabajo.

Tabla A1.6 Patrón Asignación

Nombre	Asignación	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	Debe ser ejecutada una tarea por un usuario o grupo de acuerdo a ciertas reglas de asignación y manejando una lista de tareas (Work List).	
Solución	a. Asignación de valores a los atributos de la tarea desde procesos externos. b. Asignación de atributos relacionados a acuerdos de servicio mediante la invocación de una Administrador de Acuerdos de Servicio (Administrador Acuerdos Servicios) c. Asignación de atributos de sistema mediante la invocación de un Servicio Administrador de Sistema (Administrador sistema), d. Invocación de un servicio administrador de tareas (Servicio Administración Tarea), el cual utilizara los atributos de la tarea, para determinar la agregación de dicha tarea a la lista de trabajo de un usuario o grupo de usuarios, además de realizar otras actividades como verificar la permisología, estado de transición, documentación, acuerdos de servicios, calcular tiempo, asignar auditoria, verificar modificaciones, encuestas y soluciones sobre la tarea. e. Posteriormente, los resultados o eventos son manejados por un Servicio Administrador de Eventos de Tarea (ServicioAdministracionEventosTarea) a fin de verificar la expiración, la renovación y los errores asociados.	
Contexto	Cualquier proceso donde se requiera asignar una tarea a un usuario o grupo.	
Fuerzas	Todos los workflow que incluyen interacción humano computador requieren administración de las tareas a ser asignadas a los usuarios, además de manejar una lista de tareas	
Usabilidad	Permite administrar la asignación de tareas de acuerdo a ciertos atributos asignados a las mismas.	
Consecuencias	Asignación de tareas a usuarios o grupos de usuarios tomando en cuenta las características de la tarea para la asignación y manejando una lista de trabajos(work list)	
Ejemplo	Un empleado, a través de un portal de empleados suministra una solicitud de vacaciones. El portal inicia un proceso de negocios que incluye una tarea de usuario modelada usando un simple workflow, la tarea es asignada al supervisor del empleado. Cuando el supervisor aprueba o rechaza la solicitud de vacaciones, el empleado es notificado con la decisión del	

	supervisor por e-mail.
Patrones Relacionados	f. Administrador de Acuerdos de Servicios g. Administrador de Sistema h. Administrador de Tarea i. Administrador de Eventos sobre la Tarea

Tabla A1.7 Patrón Asignación con Escalación

Nombre	Asignación con Escalación	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	Debe escalarse una tarea asignada de un usuario a otro, tomando en cuenta acuerdos de servicios preestablecidos (Service Level Agreement)	
Solución	Utilización del Patrón de Asignación y adicionalmente invocación de una operación denominada "verificarEscalacion", que determina de acuerdo a las condiciones del SLA, si la tarea asignada debe ser escalada, cuando debe ser escalada una tarea a otro usuario, quien es el usuario a ser notificado de la escalación y escala la tarea de ser necesario.	
Contexto	Cualquier proceso donde se requiera escalar una tarea en caso de no ser cumplidas las políticas de SLA.	
Fuerzas	Es necesario manejar escalación de tareas en caso de no ser cumplidos acuerdos de SLA como por ejemplo la expiración de la tarea ya que esto evita retrasos en el proceso y asignación de las tareas a usuarios o grupos que si puedan manejar dichos acuerdos	
Usabilidad	Todos los procesos donde sea necesario escalar tareas en base a un conjunto de acuerdos de SLA.	
Consecuencias	Asignación de tareas a usuarios o grupos de usuarios tomando en cuenta las características de la tarea para la asignación y manejando una lista de trabajos(work list), además de manejar información sobre acuerdos de servicios para realizar escalación de tareas a otros usuarios o grupos	
Ejemplo	El proceso de HelpDesk da a usuarios la capacidad de archivar tickets de petición del servicio. Si la persona que recibe el boleto no actúa dentro de un período especificado, el boleto se escala automáticamente a su encargado. El boleto se escala automáticamente tres veces si nadie ha actuado en él dentro de un tiempo predefinido, hasta que consigue al gerente de la compañía. Si el gerente tampoco actúa, el ticket expira.	
Patrones Relacionados	- Asignación - AdministradorEventosTarea	

Tabla A1.8 Patrón Asignación con Renovación

Nombre	Asignación con Renovación	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	Se debe renovar la asignación de una tarea a un usuario una cantidad definida de veces, cada vez que esta expire. Este tiempo de expiración viene especificado en los acuerdos de SLA(Service Level Agreement)	
Solución	Utilización del patrón de asignación y adicionalmente invocación de una operación denominada "verificarRenovacion", que además de realizar las mismas funciones de la operación "verificarEscalacion" (determinar según los acuerdos de SLA si una tarea debe ser escalada, cuando, a quien y escalar la tarea de ser necesario) también está en la capacidad de renovar la tarea – asignarla al mismo usuario o grupo – una cantidad definida de veces.	
Contexto	Cualquier proceso donde se requiera renovar la asignación de tareas a un usuario o grupo conforme a los acuerdos de SLA.	
Fuerzas	Puede darse el caso en que se flexibilicen los acuerdos de servicios, para dar oportunidad a un usuario o grupo de usuarios de completar una tarea	
Usabilidad	Todos los procesos donde sea necesario escalar tareas en base a un conjunto de acuerdos de SLA.	
Consecuencias	Asignación de tareas a usuarios o grupos de usuarios permitiendo la renovación de la misma al mismo usuario, un número definido de veces antes de ser escalada.	
Ejemplo	El proceso de HelpDesk da a usuarios la capacidad de archivar tickets de petición del servicio. Si la persona que recibe el boleto no actúa dentro de un período especificado, el boleto se escala automáticamente a su encargado. El boleto se escala automáticamente tres veces si nadie ha actuado en él dentro de un tiempo predefinido, hasta que consigue a la última persona con capacidad de atender la petición de servicio, si esta persona tampoco actúa, el ticket expira. En este caso puede ser renovada la asignación a la misma persona cierta cantidad de veces, con el fin de no truncar la posibilidad de prestar el servicio.	
Patrones Relacionados	- Asignación - Administrador de Eventos de Tarea	

Tabla A1.9 Patrón Asignación Automática Secuencial

Nombre	Asignación Automática Secuencial	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	Una tarea debe ser realizada o procesada por un usuario o grupo de usuarios en forma secuencial.	
Solución	Creación de una secuencia de usuarios que procesaran la tarea mediante un servicio Administrador de Secuencia, La tarea será asignada a cada usuario en la secuencia utilizando el patrón de asignación hasta llegar al último usuario.	
Contexto	Un proceso que contiene una lista de usuarios asociados a una tarea a realizar. La tarea tiene un conjunto de acuerdos de SLA asociados.	
Fuerzas	Es frecuente el caso en que existe un orden definido en el cual más de un usuario o grupo de usuario requieren procesar una misma tarea.	
Usabilidad	Cualquier proceso donde se haga necesario el procesamiento de una tarea por varios o grupo de usuarios en forma secuencial.	
Consecuencias	El procesamiento de una tarea por múltiples usuarios en forma secuencial	
Ejemplo	En un sistema aprobador de órdenes de compras, un usuario perteneciente a un grupo supervisor evalúa inicialmente la orden de compra. Después que el usuario inicial aprueba la orden, el gerente del departamento debe aprobarla también.	
Patrones Relacionados	- Asignación - Administrador de Secuencia	

Tabla A1.10 Patrón Asignación Automática Secuencial con Escalación

Nombre	Asignación Automática Secuencial con Escalación	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	Una tarea debe ser realizada o procesada por un usuario o grupo de usuarios en forma secuencial, si alguno de los usuarios no cumple los acuerdos de SLA, la tarea es escalada a otro usuario.	
Solución	Creación de una secuencia de usuarios que procesaran la tarea mediante un servicio AdministradorDeSecuencia, La tarea será asignada a cada usuario en la secuencia utilizando el patrón de asignaciónConEscalacion hasta llegar al último usuario. En este caso, la tarea puede	

	ser escalada en caso de no ser cumplidos los acuerdos de SLA por alguno de los usuarios en la lista.
Contexto	Un proceso que contiene una lista de usuarios asociados a una tarea a realizar y los cuales deben cumplir con un conjunto de acuerdos de SLA.
Fuerzas	Es frecuente el caso en que existe un orden definido en el cual más de un usuario o grupo de usuario requieren procesar una misma tarea, a esto se la agrega la necesidad de escalar la dicha tarea si uno más de dichos usuarios no cumplen con los acuerdos de SLA.
Usabilidad	Cualquier proceso donde se haga necesario el procesamiento de una tatarea por varios o grupo de usuarios en forma secuencial y los cuales deban cumplir con acuerdos de SLA.
Consecuencias	El procesamiento de una tarea por múltiples usuarios en forma secuencial y escalación de dichas tareas en caso de no haber sido ejecutadas según los acuerdos de SLA.
Ejemplo	Un empleado solicita una nueva computadora portátil urgentemente. El workflow secuencial primero encamina la tarea a su jefe para la aprobación y después a una persona en el departamento encargado de adquirir la computadora portátil. Si el SLA indica que la tarea debe escalarse después de dos días, entonces la petición primero va al jefe para la aprobación y si no se aprueba en dos días, la tarea se puede encaminar automáticamente a la persona siguiente en la jerarquía de gerencia. Un escalamiento similar se puede también hacer para el departamento encargado de adquirir la computadora portátil. Esto permite al workflow terminar sin largos retrasos en el proceso de aprobación.
Patrones Relacionados	- Asignación con Escalación - Administrador de Secuencia

Tabla A1.11 Patrón Asignación Manual

Nombre	Asignación Manual	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	Se presenta cuando a un usuario o un grupo de usuarios no han aprobado la tarea, y no se ha remitido la tarea al usuario original que creó la tarea, para su aprobación final.	
Solución	Se puede solucionar, cambiando los atributos iníciales y validando con el administrador de acuerdos de servicios, para determinar cuáles son los grupos de usuarios que van a recibir la tarea para su revisión y validación, una vez creado el grupo al cual se le va a enrutar la tarea se invoca al administrador de Tarea, que en este caso va hacer el encargado de enrutar la tarea cuando cada usuario haya revisado la tarea y la envié a su	

	revisor original.
Contexto	Cualquier proceso donde se requiera tener más de una revisión no automática, para su aprobación final.
Fuerzas	Cuando se requiere que exista una aprobación sin un criterio específico de asignación automática este patrón permite la asignación manual.
Usabilidad	Permite enrutar tareas a un usuario o grupo de usuarios de forma manual, para su posterior aprobación
Consecuencias	La tarea es asignada a una persona o grupo que fue seleccionada por un criterio manual
Ejemplo	Un representante de una Compañía "HR", escribe un nuevo documento, y lo tiene repasado por su encargo. El encargado puede decidir que otra persona debe también repasarlo antes de que se acepte. Esta persona puede alternadamente remitir la tarea a otras antes de aprobarla. Por lo tanto la tarea se puede enrutar a múltiples usuarios, antes de volverse al revisor original, que entonces la aprueba basado en comentarios de otros.
Patrones Relacionados	Proceso Externo, Asignación

Tabla A1.12 Patrón Asignación Paralela

Nombre	Asignación Paralela	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	El problema que resuelve este patrón, es que se lleva un control de la tarea desde su inicio hasta su estado de completitud, ya que el dueño de la tarea le puede hacer un seguimiento al proceso permitiendo la aprobación en paralelo de varios usuarios a través de sub tareas	
Solución	Cuando se completan las subtareas el administrador de eventos tarea, prepara un mensaje de completitud de subtarea, que se le es enviado al administrador de tarea para indicarle al dueño de proceso que ya la tarea padre puede ser retirada, esto ocurre cuando se ha logrado el 75% de completitud de subtarea.	
Contexto	Se utiliza Asignación Paralela cuando una tarea se debe aprobar por múltiples usuarios o simultáneamente por grupos de usuarios, cuando se crea una tarea cada usuario tiene una copia de la tarea, los usuarios podrán agregar información, agregar comentarios independiente de otros usuarios, como premisa los usuarios que repasan las tareas en paralelo no pueden ver las tareas de los otros usuarios (incluyendo comentarios, resultados, etc.). El dueño de la tarea puede ver todas las subtareas en ejecución. Cuando el creador de	

	la tarea retira a la tarea "Padre", todas las subtareas se retiran.
Fuerzas	El administrador de Tareas, se puede saturar a la hora de atender cada petición de subtarea, ya que una tarea se puede dividir en muchas partes. En ese momento el administrador de eventos de tarea prepara un mensaje de completitud de subtarea, que es enviado al administrador de tarea, para sí retirar la tarea padre y finalizar el proceso.
Usabilidad	La usabilidad de este patrón viene dada, con la rapidez con que se desea agilizar una tarea principal, para que sea divida en subtareas, para que varios usuarios puedan resolver un problema de manera rápida.
Consecuencias	
Ejemplo	Un ejemplo concreto es el de emplear a un nuevo aspirante para optar a un cargo dentro de una organización. Cada entrevistador da su voto para emplear o no al candidato, si en el resultado de la votación el candidato obtiene un promedio de 75%, obviamente será aceptado, caso contrario será rechazado, se hace regencia en este ejemplo porque cada entrevistador puede votar independientemente de los otros entrevistadores.
Patrones Relacionados	Proceso Externo, Administrador Sub Tarea, Administrador Tarea, Asignación

Tabla A1.13 Patrón Asignación Paralela con Escalación

Nombre	Asignación Paralela con Escalación	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	El problema que resuelve este patrón, es que se lleva un control de la tarea desde su inicio hasta su estado de completitud, ya que el dueño de la tarea le puede hacer un seguimiento al proceso. Permite también realizar escalaciones cuando no se ha completado una tarea o subtarea	
Solución	Cuando se completan las subtareas el administrador de eventos tarea, prepara un mensaje de completitud de subtarea, que se le es enviado al administrador de tarea para indicarle al dueño de proceso que ya la tarea padre puede ser retirada, esto ocurre cuando se ha logrado el 75% de completitud de subtarea.	
Contexto	Se utiliza Asignación Paralela con Escalación cuando una tarea se debe aprobar por múltiples usuarios o simultáneamente por grupos de usuarios, Si la Tarea no se ha completado se realiza la escalación automática. Cuando se crea una tarea cada usuario tiene una copia de la tarea, los usuarios podrán agregar información, agregar comentarios	

	independientes de otros usuarios, como premisa los usuarios que repasan las tareas en paralelo no pueden ver las tareas de los otros usuarios (incluyendo comentarios, resultados, etc.). El dueño de la tarea puede ver todas las subtareas en ejecución. Cuando el creador de la tarea retira a la tarea "Padre", todas las subtareas se retiran.
Fuerzas	El administrador de Tareas, se puede saturar a la hora de atender cada petición de subtarea, ya que una tarea se puede dividir en muchas partes. En ese momento el administrador de eventos de tarea prepara un mensaje de completitud de subtarea, que es enviado al administrador de tarea, para sí retirar la tarea padre y finalizar el proceso.
Usabilidad	La usabilidad de este patrón viene dada, con la rapidez con que se desea agilizar una tarea principal, para que sea divida en subtareas, para que varios usuarios puedan resolver un problema de manera rápida. Permite además llevar un control de notificaciones a los supervisores por medio de escalaciones, cuando la tarea ha expirado o no ha sido completada.
Consecuencias	
Ejemplo	Una Persona Dueña de un proceso, crea una tarea, y la imparte a los usuarios o grupos de usuarios para que revisen y le agreguen información adicional a las subtareas de manera que sean procesadas en forma independiente de cada usuario, que está procesando la subtarea, cada usuario aprobara la subtarea en porcentajes de completación. Si las subtareas no son completadas en un lapso de tiempo la subtarea escala un nivel hacia los supervisores de los usuarios que son responsables de procesar las subtareas.
Patrones Relacionados	Asignación Paralela, Asignación con Escalación, Administrador Sub Tareas, Administrador Tarea, Proceso Externo

Tabla A1.14 Patrón Asignación Paralela con Revisión Final

Nombre	Asignación Paralela con Revisión Final	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	El problema que resuelve este patrón, es que se lleva un control de la tarea desde su inicio hasta su estado de completitud, ya que el dueño de la tarea le puede hacer un seguimiento al proceso y a su vez es apoyado por un usuario que revisa la tarea para su aprobación final, es decir la tarea pasa por una doble validación.	
Solución	Cuando se completan las subtareas el administrador de eventos de tarea, prepara un mensaje de completitud de subtarea, que se le es enviado al administrador de tarea para indicarle al revisor final que ya la tarea padre puede ser retirada, esto ocurre cuando se ha logrado el	

	75% de completitud de subtarea.
Contexto	Asignación Paralela con un Revisión Final es una extensión del patrón Asignación Paralela, en el cual después de la revisión de la tarea en paralelo por parte de los usuarios, la tarea se asigna a un revisor final. El revisor final puede ser un usuario o un grupo. Cuando hay más de un revisor final o si la tarea se asigna a un grupo, uno de los usuarios tiene que adquirir la tarea antes de repasarla, el revisor final puede ver todas las subtareas.
Fuerzas	El administrador de Tareas, se puede saturar a la hora de atender cada petición de subtarea, así como también si se selecciona a un grupo para que actué como revisor final, se puede perder tiempo en escoger al usuario que va a tomar la tarea para el cambio de estatus de las subtareas para que sea retirada por el dueño del proceso.
Usabilidad	La usabilidad de este patrón viene dada, con la rapidez con que se desea agilizar la tarea principal, la cual se les envía a los múltiples usuarios en paralelo y en seguida se le envía a un revisor final para que el dueño del proceso pueda tomar una decisión.
Consecuencias	
Ejemplo	En un proceso de revisión de documentos, el creador del documento inicia el proceso especificando el documento y a los revisores del documento. Cada revisor puede repasar el documento simultáneamente e independientemente de los otros revisores, después de que todos los revisores hayan hecho el repaso del documento, el creador del documento consigue repasar los comentarios de los revisores.
Patrones Relacionados	Asignación Paralela, Asignación, Proceso Externo, Administrador Tarea, Administrador Sub Tarea.

Tabla A1.15 Patrón Asignación Paralela Revisión Final con Escalación

Nombre	Asignación Paralela Revisión Final con Escalación	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	El problema que resuelve este patrón, es que se lleva un control de la tarea desde su inicio hasta su estado de completitud, ya que el dueño de la tarea le puede hacer un seguimiento al proceso y a su vez es apoyado por un usuario que revisa la tarea para su aprobación final, es decir la tarea pasa por una doble validación. Si la tarea no es completada se escala a otra instancia, para la toma de decisiones.	
Solución	Cuando se completan las subtareas el administrador de eventos de tarea, prepara un mensaje de completitud de subtarea, que se le es enviado al administrador de tarea para indicarle al	

	revisor final que ya la tarea padre puede ser retirada, esto ocurre cuando se ha logrado el 75% de completitud de subtarea.
Contexto	Asignación Paralela con un Revisión Final es una extensión del patrón Asignación Paralela, en el cual después de la revisión de la tarea en paralelo por parte de los usuarios, la tarea se asigna a un revisor final. El revisor final puede ser un usuario o un grupo. Cuando hay más de un revisor final o si la tarea se asigna a un grupo, uno de los usuarios tiene que adquirir la tarea antes de repasarla, el revisor final puede ver todas las subtareas.
Fuerzas	El administrador de Tareas, se puede saturar a la hora de atender cada petición de subtarea, así como también si se selecciona a un grupo para que actué como revisor final, se puede perder tiempo en escoger al usuario que va a tomar la tarea para el cambio de estatus de las subtareas para que sea retirada por el dueño del proceso.
Usabilidad	La usabilidad de este patrón viene dada, con la rapidez con que se desea agilizar la tarea principal, la cual se les envía a los múltiples usuarios en paralelo y en seguida se le envía a un revisor final para que el dueño del proceso pueda tomar una decisión.
Consecuencias	
Ejemplo	En un proceso de revisión de documentos, el creador del documento inicia el proceso especificando el documento y a los revisores del documento. Cada revisor puede repasar el documento simultáneamente e independientemente de los otros revisores, después de que todos los revisores hayan hecho el repaso del documento, el creador del documento consigue repasar los comentarios de los revisores, en caso contrario si el creador de la tarea no logra cerrar el proceso, se realiza una notificación de que una tarea no ha sido completada.
Patrones Relacionados	Asignación Paralela, Asignación Paralela con Escalación, Asignación con Escalación Asignación Paralela con Revisión Final, Administrador Sub Tarea, Administrador Tarea, Proceso Externo.

Tabla A1.16 Patrón Administrador de Sub Tarea

Nombre	Administrador de Sub Tarea	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	Se requiere que varias personas a través de subtareas como copia de la tarea principal realicen una aprobación en paralelo	
Solución	Creación de subtareas copia de la tarea de la tarea original permitiendo a través del administrador de subtareas la aprobación en paralelo	

Contexto	Recibe las solicitudes de Asignación Paralela, Asignación Paralela con Escalación, Asignación Paralela con Revisión Final, Asignación Paralela Revisión Final con Escalación, a través del servicio de administración de sub tarea, procesa las sub tareas y envía respuesta.
Fuerzas	Ayuda al administrador de tarea en el procesamiento de las subtareas de manera paralela cuando se ha colocado una tarea principal y necesita ser gestionada por múltiples usuarios.
Usabilidad	Se utiliza como una funcionalidad que apoya el procesamiento y gestión de las subtareas.
Consecuencias	Cuando se crea la tarea principal la Asignación Paralela en cualquiera de sus modalidades, realiza una invocación al Administrador de Subtarea, el cual es el encargado de recibir procesar (crear, modificar, consultar, eliminar) las subtareas. Permitiendo llevar un control de la gestión de la subtarea, permite monitorear el porcentaje de completitud de subtarea, realiza validaciones de grupos y usuarios a los cuales se les distribuye las subtareas para su revisión, trabaja en paralelo con el administrador de tareas.
Ejemplo	El Administrador es una funcionalidad que permite procesar las tareas en subtareas, en donde todo comienza cuando una tarea se está procesando en paralelo, esta tarea se ejecuta de manera independiente de cada usuario, cada uno de los usuarios desconoce la información que le agregan al momento de la completación de la tarea, el administrador de tarea recibe los atributos necesarios para poder distribuir la tarea al usuario o al grupo de usuarios para su gestión.
Patrones Relacionados	Asignación Paralela, Asignación, Administrador Tarea

Tabla A1.17 Patrón Asignación sin Notificación

Nombre	Asignación sin Notificación	
Clasificación	Workflow	
Autor	MSc. Pedro Bonillo	
Confiabilidad	%	
Problema	Cuando se crea una tarea, no se espera una respuesta del usuario, ya que no interviene un administrador de acuerdos de servicios	
Solución	Para tener una respuesta satisfactoria, se utiliza el administrador de acuerdos de servicio, el cual permite notificar al supervisor, cuando un usuario ha recibido una tarea, cuando fue aprobada y ejecutada por el usuario final.	
Contexto	El FYI es una Tarea que no se bloquea y se enumera como tareas asignadas a un usuario, esta tarea se aloja en el worklist para su aprobación, El proceso que crea la tarea no espera una respuesta del usuario.	
Fuerzas	Si el administrador de acuerdo de servicio no puede determinar la ubicación del usuario	

	o grupos de usuario, no se podrá realizar la notificación de que la tarea ha sido entregada, ni aprobada, esto generaría un error, el cual es lanzado por el administrador de tareas.
Usabilidad	Viene dada, para llevar un control del flujo de la tarea, hasta el momento de su aprobación por el usuario.
Consecuencias	
Ejemplo	Un ejemplo práctico es cuando se utiliza un proceso de sistema de órdenes de compra. Cuando el sistema procesa aprobaciones sobre cierta cantidad como por ejemplo: $, un supervisor tiene que ser notificado, pero sin parar el proceso, esta información es vital para el supervisor ya que le permite monitorear el sistema.
Patrones Relacionados	Asignación Simple

Anexo 2 Formato de Solicitud de Requerimiento

En este anexo se presenta el formato a través del cual se solicita un requerimiento de creación o mejora de un proceso de negocio.

Número de Solicitud	

Fecha:	
Nombre Solicitante:	
Teléfonos Solicitante:	
E-Mail del Solicitante:	
Gerencia:	
Unidad:	

Objetivos: (Indique cada uno de los objetivos que deben ser cubiertos)

Descripción del Problema: (Situación Actual, Diagramas de Procesos, etc.)

Alcance: (Indique cuál será el alcance del desarrollo, qué aspectos deben contemplarse y cuáles no)

Beneficios: (Indique los beneficios que traerá el desarrollo e implementación del proyecto en base a procesos, uso de recursos, etc.)

Elementos de Entrada: (Indique cuáles son los datos o elementos de entrada que deberá recibir el sistema. Ej. Datos, archivos logs, etc.)

Elementos de Salida: (Indique cuáles son los datos o elementos que debe mostrar el sistema como resultado de su procesamiento. Ej. Automatización de configuración, reportes, etc.)

Áreas Involucradas (Indique las áreas y/o personas que estarán relacionadas con el desarrollo solicitado)

Área	Alias Área	Supervisor	Persona Contacto	Teléfono Persona Contacto.

Usuarios: (Indique quiénes serán los usuarios finales, nombres, teléfonos, e-mail, ubicación administrativa y ubicación física. En caso de ser un grupo colocar los datos del grupo del supervisor)

Nombre/ Grupo	Teléfonos	E-mail (personal y del alias de la unidad a la que pertenece)	Ubicación Administrativa	Ubicación Física

Anexos. (Anexe a este formato toda aquella información que considere sea necesaria, así como información que pueda ayudar al desarrollo de este proyecto)
Solo Para ser Llenado por la Unidad de Recepción de la Solicitud:

Responsable:	
Factibilidad:	
Fecha Reunión Inicial:	

Anexo 3 Cartelera de Actividades Propuestas

A continuación se presentan las actividades que conforman la metodología y una duración en días promedio para un proceso medianamente complejo (entre 20 y 60 pasos):

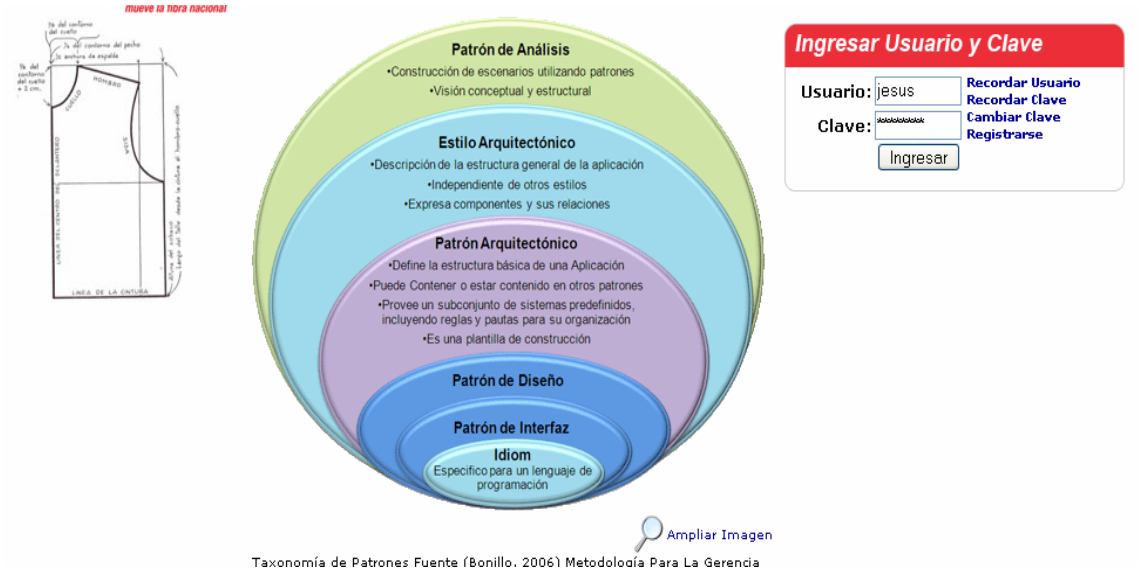
Tiempo para la creación de un proceso	
Nombre de Tarea	**Duración (días)**
Process Mangement (Gerencia de Procesos sustentada en Patrones)	186
1. Crear Procesos	130
1.1 Analizar	28
1.1.1 Identificar Requerimientos	8
1.1.1.1 Describir Requerimiento	5
1.1.1.2 Definir Prioridad	1
1.1.1.3 Evaluar Cartelera de Actividades	2
1.1.2 Levantar Información	20
1.1.2.1 Mapear al Marco Referencial del Proceso	5
1.1.2.2 Levantar Situación Actual	15
1.2 Diseñar	24
1.2.1 Estandarizar Proceso	2
1.2.2 Evaluar Riesgos	2
1.2.3 Realizar Modelo Propuesto	15
1.2.4 Efectuar el Setup del Laboratorio	5
1.3 Modelar	8
1.3.1 Diagramar	5
1.3.2 Simular Prueba Funcional	1
1.3.3 Integrar Arquitectura	1
1.3.4 Simular Prueba Integral	1
1.4 Configurar	50

1.4.1 Completar UML	5
1.4.2 Construir Lógica de Negocios	10
1.4.3 Construir Interfaz	10
1.4.4 Construir Reportes	10
1.4.5 Configurar el BPEL	10
1.4.6 Realizar Pruebas Unitarias	5
2. Administrar Procesos	56
2.1 Mantener Configuraciones	16
2.1.1 Mantenimiento de Aplicaciones	6
2.1.1.1 Analizar y validar el tipo de requerimiento	2
2.1.1.2 Diseñar	3
2.1.1.3 Aplicar Pruebas	1
2.1.2 Mantener Variables del Proceso	5
2.1.3 Mantener Plataforma	5
2.2 Monitorear	40
2.2.1 Monitorear Funciones	20
2.2.2 Monitorear Plataforma	20

Anexo 4 Recuperar Patrón de Análisis

En este anexo se presentan las pantallas de la aplicación de Administración de patrones, que se utilizan para recuperar un Patrón de Análisis.

Inicialmente el usuario se conecta a la aplicación y coloca un usuario y clave.

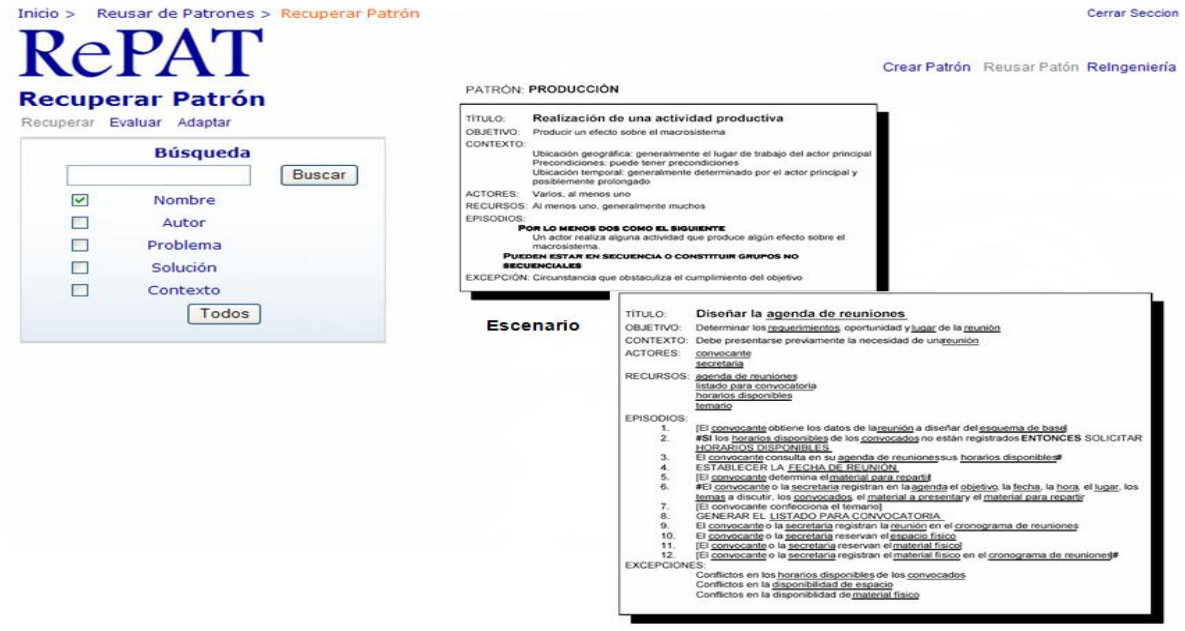


Taxonomía de Patrones Fuente (Bonillo, 2006) Metodología Para La Gerencia

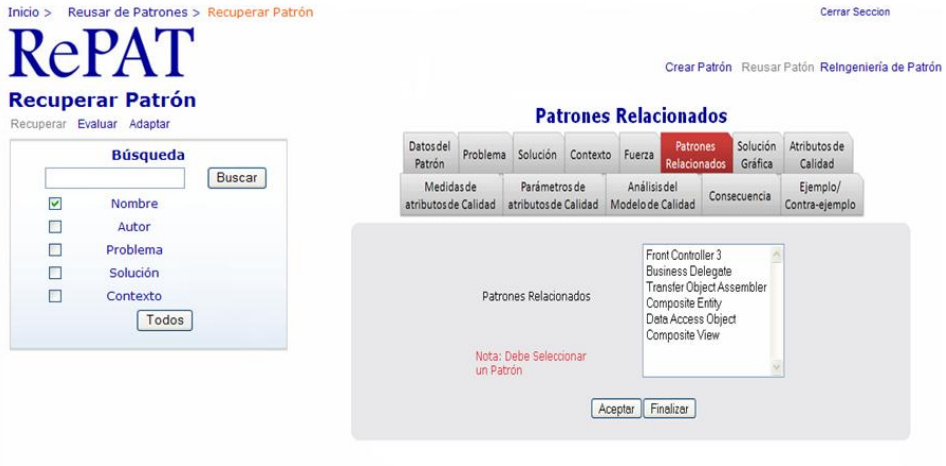
A continuación, indica que desea trabajar con el catálogo de Patrones de Análisis.

Una vez en el catálogo de Patrones de Análisis el usuario indica la heurística a través de la cual desea recuperar un patrón en específico, esta heurística consiste en una búsqueda por combinación de los campos Nombre del Patrón, Autor, Problema que soluciona el Patrón, Solución y Contexto asociado al Patrón.

Al seleccionar un Patrón de Análisis se muestran todos los datos asociados al mismo y sus escenarios, en base al meta-patrón descrito en el marco teórico.



A continuación se puede editar la información del Patrón y ver los otros Patrones con los cuales está relacionado tanto en su mismo nivel de abstracción como en el inferior, es decir en este caso se pudiese ver con que otros Patrones de Análisis está relacionado y con qué Estilos Arquitecturales.


Inicio > Reusar de Patrones > Recuperar Patrón
Cerrar Seccion
RePAT
Crear Patrón Reusar Patón Reingeniería de Patrón
Recuperar Patrón
Recuperar Evaluar Adaptar
Búsqueda
Buscar
Nombre
Autor
Problema
Solución
Contexto
Todos
Patrones Relacionados
Datos del Patrón
Problema
Solución
Contexto
Fuerza
Patrones Relacionados
Solución Gráfica
Atributos de Calidad
Medidas de atributos de Calidad
Parámetros de atributos de Calidad
Análisis del Modelo de Calidad
Consecuencia
Ejemplo/ Contra-ejemplo
Patrones Relacionados
Front Controller 3
Business Delegate
Transfer Object Assembler
Composite Entity
Data Access Object
Composite View
Nota: Debe Seleccionar un Patrón
Aceptar
Finalizar

Anexo 5 Formato Levantamiento Información del Proceso Actual

A continuación se presentan los formatos que se utilizan para levantar la información del proceso de negocio.

PROCESO

Entradas

Código	Nombre	Tipo		Rango		Orden de entrada	Condición (Código)
		Básico	Complejo	Min	Max		

Condiciones

Código	Nombre	Expresión

Salidas

Código	Nombre	Tipo		Rango		Orden de entrada	Condición (Código)
		Básico	Complejo	Min	Max		

Condiciones

Código	Nombre	Expresión

Recursos

Código	Nombre	Tipo	Tiempo de Utilización	Cantidad	Unidad de Medida

Organización

Código	Nombre

Localidad

Código	Nombre

Escalamiento/SLA

Código	Nombre	Expresión

Indicadores

Código	Nombre	Variables	Formula	Periodicidad	Clientes

Reportes

Código	Nombre	Variables	Formula	Periodicidad	Clientes

ELEMENTO

Código	Nombre	Tipo	Regla (Cód)

Atributos

Código	Nombre	Tipo		Rango		Solo Lectura	Estático	Ordenado	Exclusivo	Valor por Omisión	Regla (Cód)
		Básico	Complejo	Min	Max						

Reglas

Código	Nombre	Expresión

RECURSO

Código	Nombre	Tipo	Costo		Cant. Disp	Disponibilidad
			Tipo	Valor	Uni. Medida	
		Transporte				

Cualificaciones

Código	Rol asignado	Dimensión o requerimiento del rol asignado	Valoración alcanzada por el recurso especifico

Atributos

Código	Nombre	Tipo		Rango		Valor Inicial
		Básico	Complejo	Min	Max	

ORGANIZACIÓN

Código	Nombre

Atributos						
Código	Nombre	Tipo		Rango		Valor Inicial
		Básico	Complejo	Min	Max	

LOCALIDAD

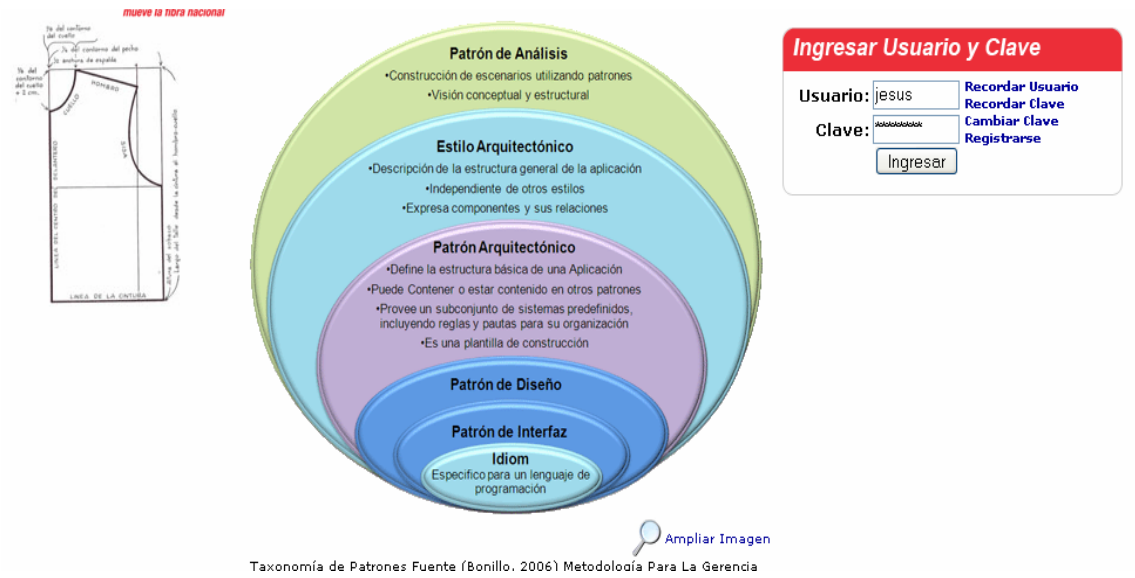
Código	Nombre

Atributos						
Código	Nombre	Tipo		Rango		Valor Inicial
		Básico	Complejo	Min	Max	

Anexo 6 Recuperar Estilo Arquitectural

En este anexo se presentan las pantallas de la aplicación de Administración de patrones, que se utilizan para recuperar un Estilo Arquitectural

Inicialmente el usuario se conecta a la aplicación y coloca un usuario y clave.

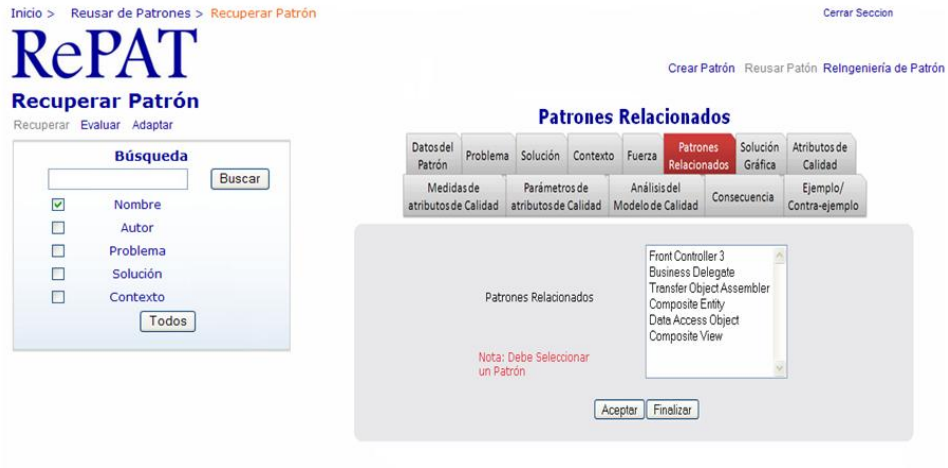


Taxonomía de Patrones Fuente (Bonillo, 2006) Metodología Para La Gerencia

A continuación, indica que desea trabajar con el catálogo de Estilos Arquitecturales

Una vez en el catálogo de Estilos Arquitecturales el usuario indica la heurística a través de la cual desea recuperar un patrón en específico, esta heurística consiste en una búsqueda por combinación de los campos Nombre del Patrón, Autor, Problema que soluciona el Patrón, Solución y Contexto asociado al Patrón.

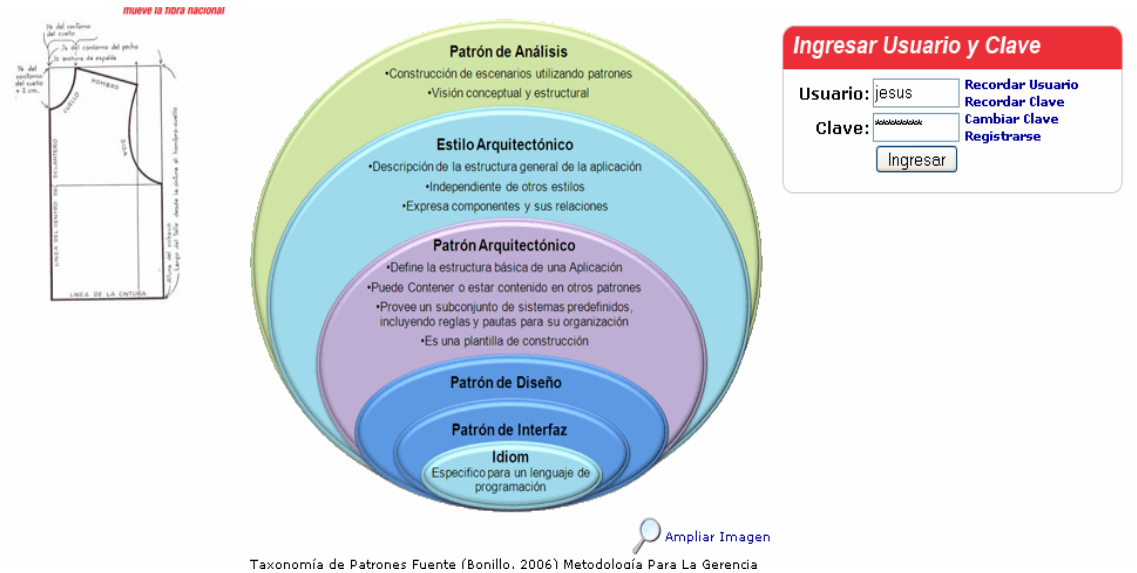
Al seleccionar un Estilo Arquitectural se muestran todos los datos asociados al mismo y sus escenarios, en base al meta-patrón descrito en el marco teórico. A continuación se puede editar la información del Patrón y ver los otros Patrones con los cuales está relacionado tanto en su mismo nivel de abstracción como en el inferior, es decir en este caso se pudiese ver con que otros Estilos Arquitecturales está relacionado y con qué Patrones Arquitecturales.



Anexo 7 Recuperar Patrón Arquitectural

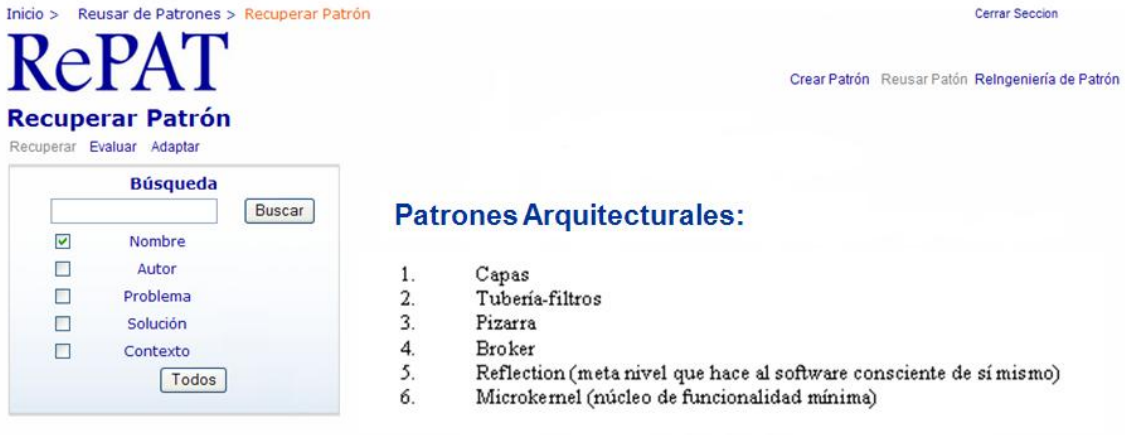
En este anexo se presentan las pantallas de la aplicación de Administración de patrones, que se utilizan para recuperar un Patrón Arquitectural

Inicialmente el usuario se conecta a la aplicación y coloca un usuario y clave.



Taxonomía de Patrones Fuente (Bonillo, 2006) Metodología Para La Gerencia

A continuación, indica que desea trabajar con el catálogo de Patrones Arquitecturales



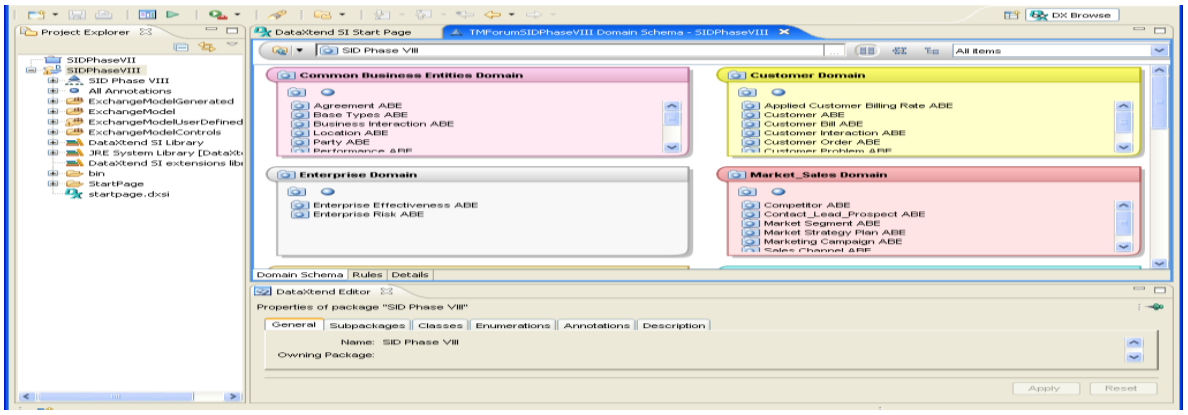
Una vez en el catálogo de Patrones Arquitecturales el usuario indica la heurística a través de la cual desea recuperar un patrón en específico, esta heurística consiste en una búsqueda por combinación de los campos Nombre del Patrón, Autor, Problema que soluciona el Patrón, Solución y Contexto asociado al Patrón.

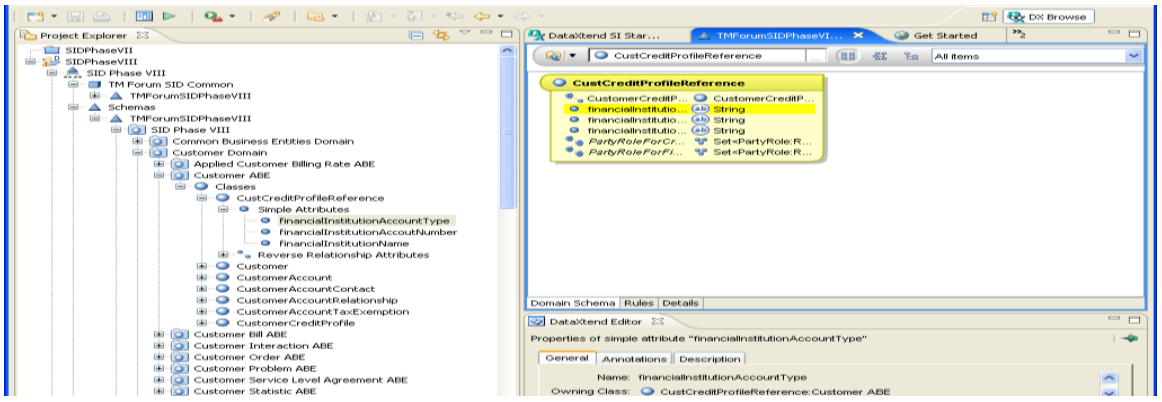
Al seleccionar un Patrón Arquitectural se muestran todos los datos asociados al mismo y sus escenarios, en base al meta-patrón descrito en el marco teórico. A continuación se puede editar la información del Patrón y ver los otros Patrones con los cuales está relacionado tanto en su mismo nivel

de abstracción como en el inferior, es decir en este caso se pudiese ver con que otros Patrones Arquitecturales está relacionado y con qué Patrones de Diseño.

Anexo 8 Administración Objetos del Negocio

En este anexo se presenta la herramienta que permite administrar los objetos del Negocio, está herramienta permite de forma visual diferentes esquemas de dominio del negocio, objetos reglas y detalles, como se muestra a través de las siguientes figuras:

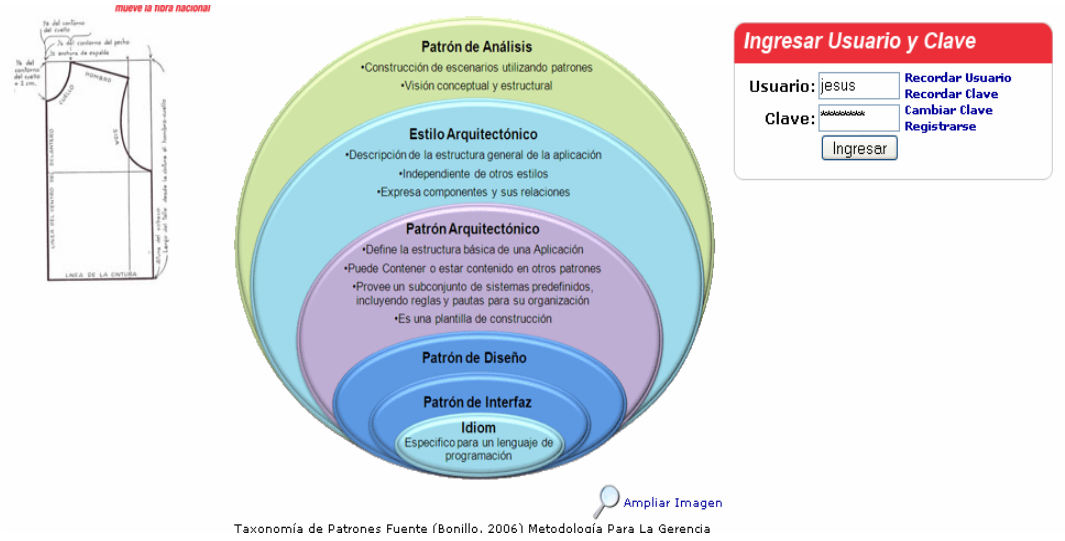




Anexo 9 Recuperar Patrón de Diseño

En este anexo se presentan las pantallas de la aplicación de Administración de patrones, que se utilizan para recuperar un Patrón de Diseño.

Inicialmente el usuario se conecta a la aplicación y coloca un usuario y clave.



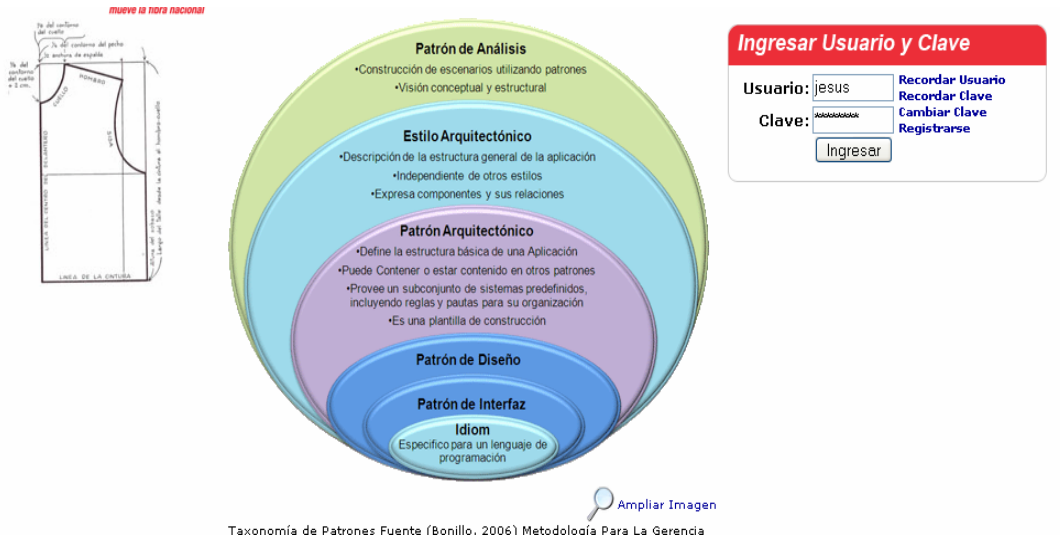
Taxonomía de Patrones Fuente (Bonillo, 2006) Metodología Para La Gerencia

A continuación, indica que desea trabajar con el catálogo de Patrones de Diseño.

Una vez en el catálogo de Patrones de Diseño el usuario indica la heurística a través de la cual desea recuperar un patrón en específico, esta heurística consiste en una búsqueda por combinación de los campos Nombre del Patrón, Autor, Problema que soluciona el Patrón, Solución y Contexto asociado al Patrón.

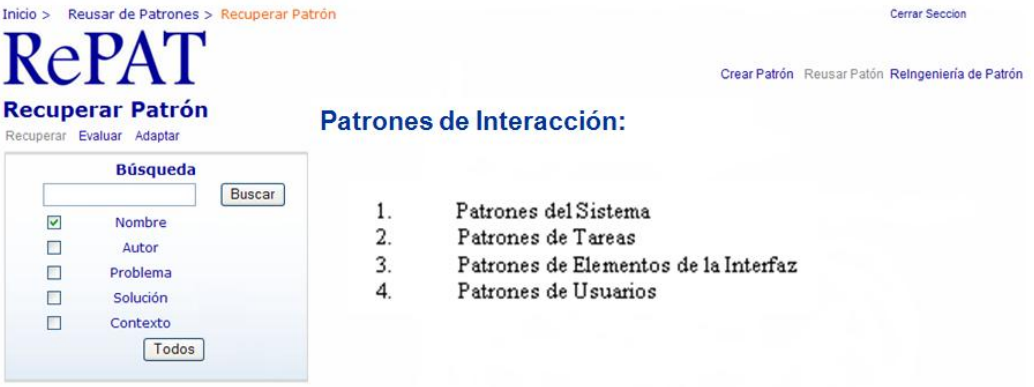
Anexo 10 Recuperar Patrón de Interacción

En este anexo se presentan las pantallas de la aplicación de Administración de patrones, que se utilizan para recuperar un Patrón de Interacción.
Inicialmente el usuario se conecta a la aplicación y coloca un usuario y clave.



Taxonomía de Patrones Fuente (Bonillo, 2006) Metodología Para La Gerencia

A continuación, indica que desea trabajar con el catálogo de Patrones de Interacción.

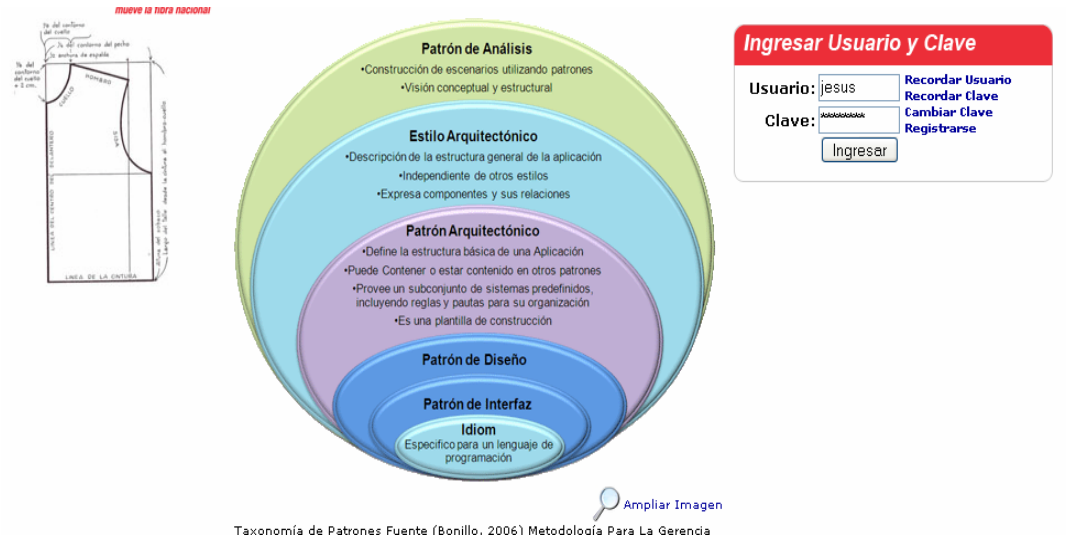


Una vez en el catálogo de Patrones de Interacción el usuario indica la heurística a través de la cual desea recuperar un patrón en específico, esta heurística consiste en una búsqueda por combinación de los campos Nombre del Patrón, Autor, Problema que soluciona el Patrón, Solución y Contexto asociado al Patrón.

Anexo 11 Asignar Modelo de Calidad

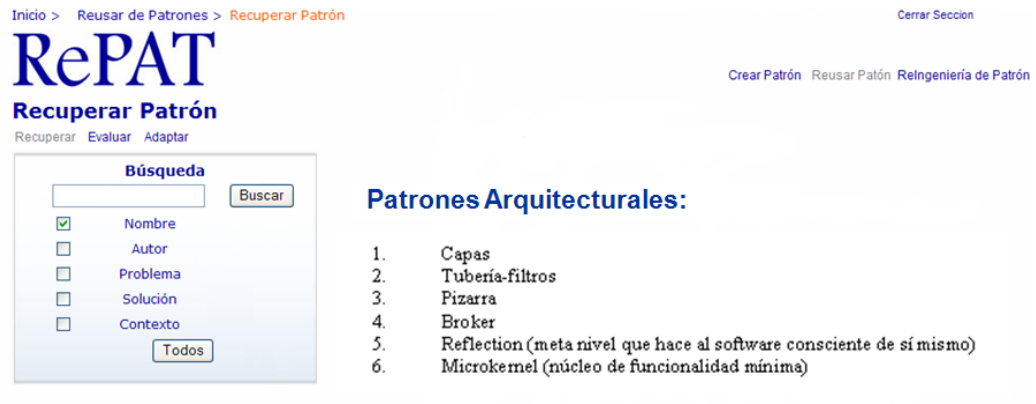
En este anexo se presentan las pantallas de la aplicación de Administración de patrones, que se utilizan para asignar a los patrones un modelo de calidad

Inicialmente el usuario se conecta a la aplicación y coloca un usuario y clave.



Taxonomía de Patrones Fuente (Bonillo, 2006) Metodología Para La Gerencia

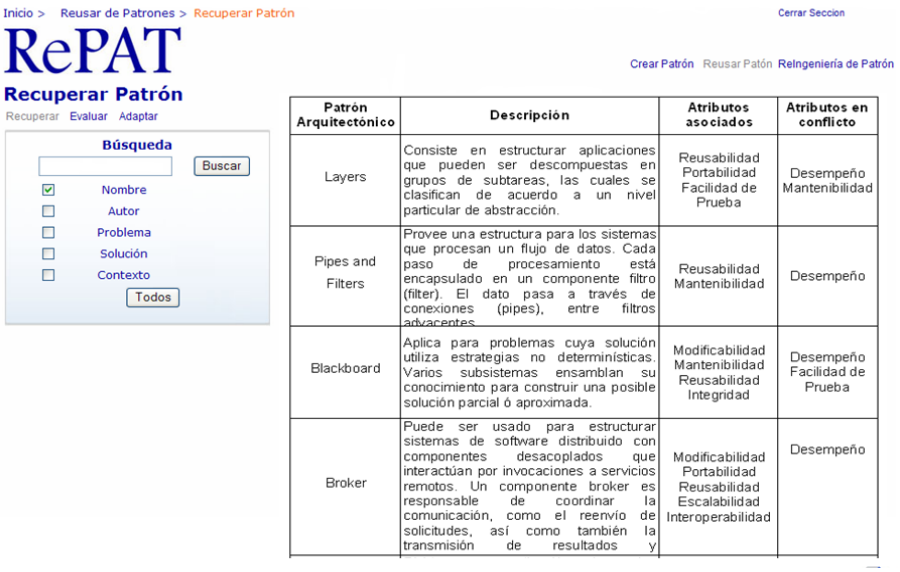
A continuación, por cada uno de los patrones seleccionados los recupera seleccionando el catálogo respectivo, en este caso se describirá el uso de la aplicación a partir de un Patrón Arquitectural.



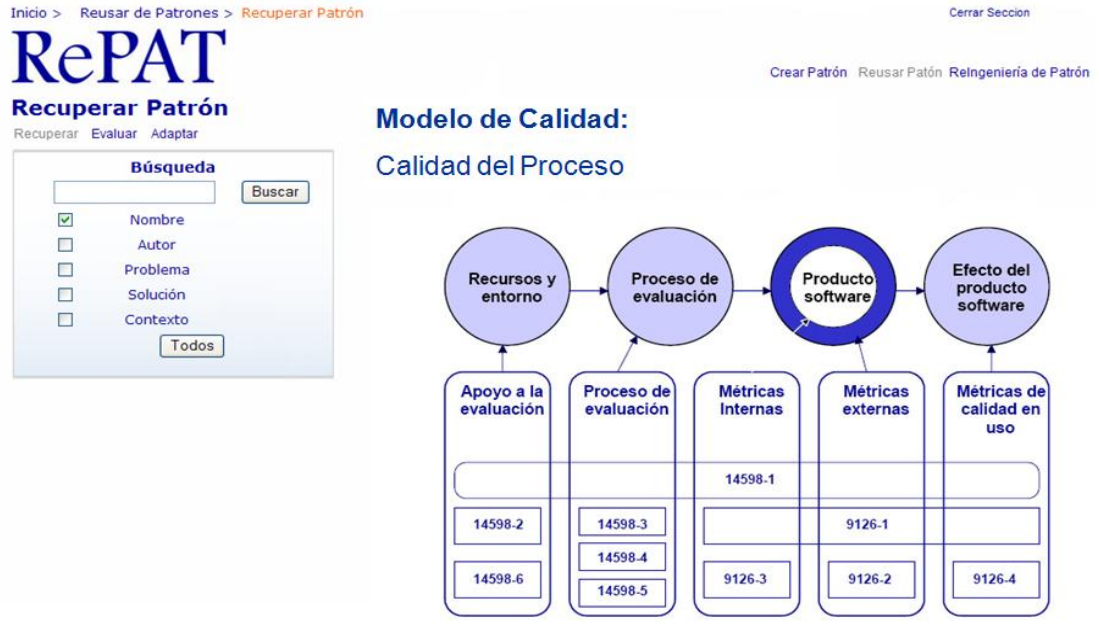
Una vez en el catálogo de Patrones Arquitecturales (en este caso se utiliza como ejemplo los Patrones Arquitecturales pero esto aplica para todos los patrones seleccionados) el usuario indica la heurística a través de la cual desea recuperar un patrón en específico, esta heurística consiste en una búsqueda por combinación de los campos Nombre del Patrón, Autor, Problema que soluciona el Patrón, Solución y Contexto asociado al Patrón.

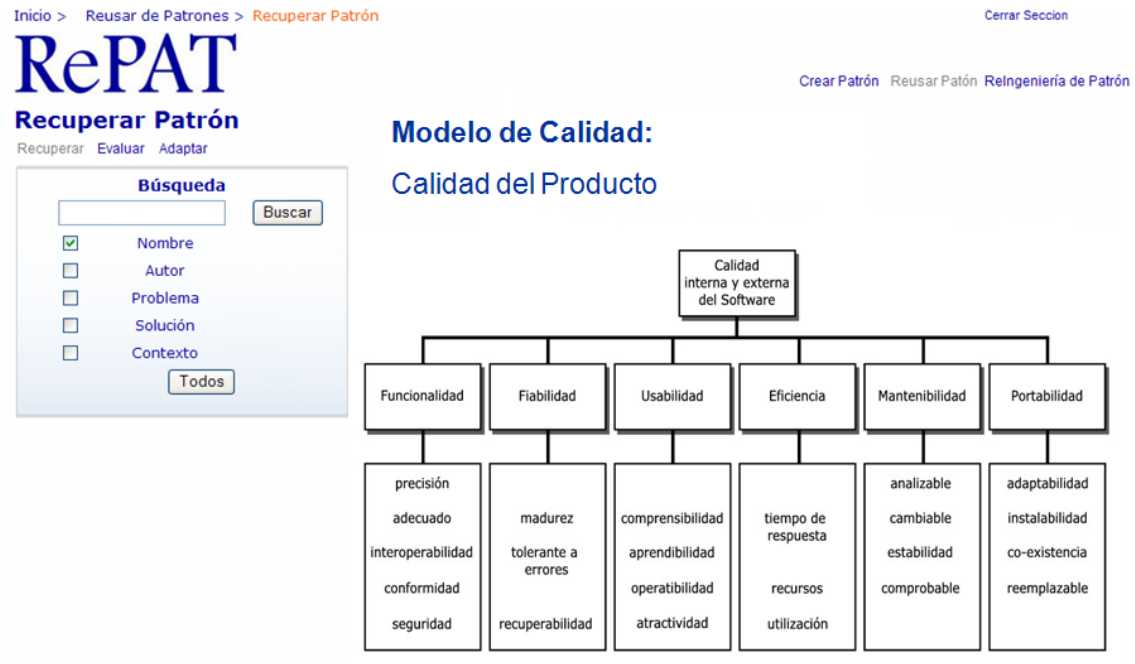
Estando en el patrón especifico se selecciona la pestaña Atributos de Calidad y a continuación se seleccionan los atributos de calidad que se desean asociar al patrón.

A continuación se pueden consultar para diferentes patrones los atributos de calidad asociados y los atributos en conflicto.



También se puede partir del modelo de calidad del proceso, del producto para luego asignar los atributos de calidad correspondientes.


Inicio > Reusar de Patrones > Recuperar Patrón
Cerrar Seccion
RePAT
Crear Patrón Reusar Patón Reingeniería de Patrón
Recuperar Patrón
Recuperar Evaluar Adaptar
Búsqueda
Buscar
Nombre
Autor
Problema
Solución
Contexto
Todos
Modelo de Calidad:
Calidad del Proceso
Recursos y entorno
Proceso de evaluación
Producto software
Efecto del producto software
Apoyo a la evaluación
Proceso de evaluación
Métricas Internas
Métricas externas
Métricas de calidad en uso
14598-1
14598-2
14598-3
14598-4
14598-5
14598-6
9126-1
9126-2
9126-3
9126-4


Inicio > Reusar de Patrones > Recuperar Patrón
Cerrar Seccion
RePAT
Crear Patrón Reusar Patón Reingeniería de Patrón
Recuperar Patrón
Recuperar Evaluar Adaptar
Búsqueda
Buscar
Nombre
Autor
Problema
Solución
Contexto
Todos
Modelo de Calidad:
Calidad del Producto
Calidad interna y externa del Software
Funcionalidad
Fiabilidad
Usabilidad
Eficiencia
Mantenibilidad
Portabilidad
precisión
adecuado
interoperabilidad
conformidad
seguridad
madurez
tolerante a errores
recuperabilidad
comprensibilidad
aprendibilidad
operatibilidad
atractividad
tiempo de respuesta
recursos
utilización
analizable
cambiable
estabilidad
comprobable
adaptabilidad
instalabilidad
co-existencia
reemplazable

Anexo 12 Herramientas Usadas

Intalio BPMS

Intalio es una plataforma *Open Source* (Código Abierto) para modelado de procesos de negocio. Esta construida bajo el estándar de Eclipse Europa, modelador BPMN y el motor que usa es Apache ODE BPEL. Proporciona todos los componentes necesarios para el diseño, despliegue y gestión de los más complejos procesos de negocio, incluida BAM, BRE, ECM, ESB y el Portal.

La Plataforma de Procesos de Negocios Intalio, está disponible como software de servicio que ofrece, diseñado para usuarios (*Business Edition Intalio*), y un ligero proceso de ejecución de servicios públicos dirigidos a los desarrolladores (*Intalio Developer Edition*). En la siguiente figura se muestra las ediciones.

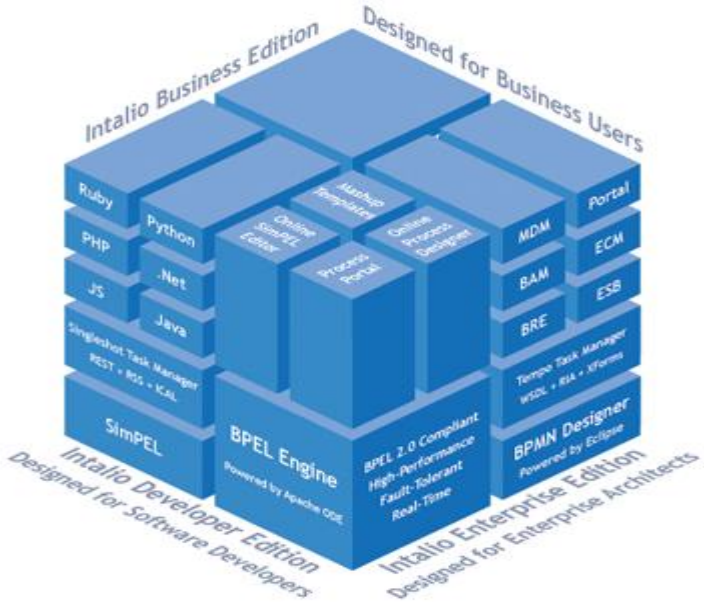


Figura 102: Ediciones de Intalio

Fuente: http://www.intalio.com/products/

Diseñador de Intalio

El diseñador permite modelar cualquier proceso mediante las especificaciones de BPMN 2.0 ó 1.1 para luego generar el código BPEL 2.0.

El modelador BPMN apoya toda la especificación BPMN 2.0 extendido con flujo de trabajo humano con BPEL4People. También es usado cuando no se necesita desplegar en el servidor. Es soportado por Windows, Linux, Mac Os X.

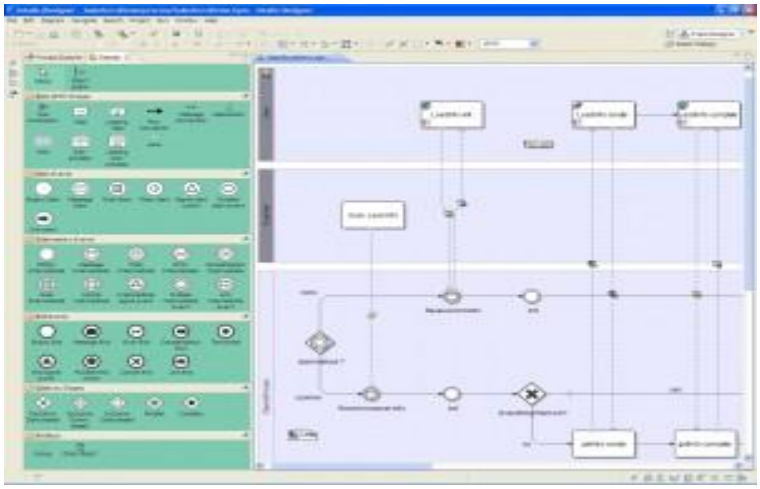


Figura 103: Soporte para Paletas Personalizadas

Fuente: http://www.intalio.com/products/designer/#bpmn

El modelador BPMN está basado en el modelador BPMN de Eclipse, que originalmente fue aportado por Intalio.

Características

Tabla 15: Características del Diseñador de Intalio

Características	Descripción
Generación/Importación de Código	✓ Validación de Procesos transparentes ✓ Importación y generación de código en WS-BPEL 2.0 ✓ Generación de WSDL ✓ Generación automática de código de procesos ✓ Importación de código en BPEL4WS 1.0/1.1 ✓ Importación ARIS EPC.
Entorno	✓ Versión Standalone ✓ Soporte de BPMN 1.1 ✓ Entorno de desarrollo integrado ✓ Basado en Eclipse
Gestión de Ciclo de Vida	✓ Versionamiento colaborativo ✓ Registro/comprobación ✓ Gestor de dependencia grafica ✓ Búsqueda avanzada ✓ Versionamiento local
Editor de Mapeo	✓ Soporte de esquema XML ✓ Mapeo de transformación gráfica ✓ Mapeo reglas graficas ✓ Validación de esquemas ✓ Soporte para esquemas complejos ✓ Editor de Mapeo de esquema a esquema ✓ Soporte a XPath y XQuery ✓ Interfaz de usuario punto y click
Despliegue de Procesos	✓ Comprobación de consistencia ✓ Despliegue de procesos con un click
Modelador de Procesos	✓ Constructor de procesos adaptable

	✓ Manejo de flujos de trabajo ✓ Editor de propiedades visual para los procesos ✓ Validación transparente de semántica ✓ Conjunto avanzado de procesos prefabricados ✓ Manejo de excepción de flujos
Introspección de Sistema	✓ Sistema de interfaz on-line ✓ Generación automática de sistema de interfaz en WSDL

Fuente: http://www.intalio.com/products/designer/#features (Traducción)

Servidor de Intalio

Es nativo de BPEL 2.0, servidor basado en la arquitectura J2EE y certificados para una amplia gama de plataformas de hardware, sistemas operativos, servidores de aplicaciones, y bases de datos. Es el motor de Intalio, y está basado en Apache ODE, el cual implementa el estándar WS-BPEL 2,0, así como XPath, además incluye una nueva función BPEL: doXSLTansform para ejecutar transformaciones XSLT durante la ejecución de procesos.

Características

Tabla 16: Características del Servidor Intalio

Características	Descripción
Soporte de Hardware	✓ HP-PA RISC 32 y 64 bit ✓ IBM Power 32 y 64 bit ✓ Intel x86 y Intel Itanium 32 y 64 bit ✓ Opteron EM64 ✓ Sun Sparc 32 y 64 bit.

Soporte Operativo	✓ HP-UX 11.31 para PA-Risc 64 bit y HP-UX 11.23 para Itanium 64 bit ✓ IBM AIX 5.2, 5.3 — 32 y 64 bit ✓ Sun Solaris 8, 9, 10 — 32 y 64 bit ✓ SUSE Linux 9, 10 ✓ Red Hat Linux 4, 5 ✓ Windows 2003 Server, 2000 Server.
Soporte a Servidor de Aplicaciones	✓ Apache Geronimo 2.0.1 ✓ Apache Tomcat 5.5.26 ✓ Servidor de aplicación JBoss ✓ Servidor de aplicación IBM WebSphere 6.1.0.3
Soporte de Base de Datos	✓ Derby 10.2 ✓ IBM DB2 8.1, 8.2, 9.1, 9.5 ✓ MS SQL 2000, 2005 ✓ MySQL 5.0, 5.1 ✓ Oracle 8i, 9i, 10g ✓ PostgreSQL 8.1.10, 8.2.5 ✓ Sybase ASE 12.5.1
Gestión y Despliegue	✓ Balanceo de carga ✓ Clustering ✓ Instrumentación JMX ✓ Depuración de API's
Consola de Gestión	✓ Gestión de consola basada en navegadores ✓ Control acceso basado en roles ✓ Servidor de tablero de instrumentos indicador de estado ✓ Lista de proceso desplegados ✓ Gestión de proceso (Activos, Retirados) ✓ Lista de instancias de procesos ejecutándose ✓ Búsqueda y filtro de instancias de procesos

Ejecución de Procesos	✓ Compilador de procesos Just-in-Time ✓ Soporta WS-BPEL 2.0 ✓ Soporta BPEL4WS
SOA	✓ Interfaces REST-style ✓ Interfaces Plain Old XML (POX) ✓ WS-Direccionamiento ✓ WS-Transacciones atómicas ✓ WS-Coordinación ✓ WS-Seguridad
Conectores de Tecnología	✓ AS400 Data ✓ Email (SMTP, POP3, IMAP4) ✓ Sistema de archivos ✓ FTP ✓ HTTP, HTTP/S ✓ GigaSpaces ✓ Glue ✓ IIOP ✓ AJAX ✓ Java ✓ JavaSpaces ✓ JBI ✓ JCA ✓ JDBC ✓ JMS ✓ JNDI ✓ JOTM ✓ JTA ✓ MQ Series ✓ Oracle AQ ✓ Plexus

	✓ Remote EJB ✓ REST ✓ RMI ✓ SOAP ✓ Sistema Entrada/Salida ✓ TCP ✓ TLS ✓ VFS ✓ UDP ✓ XFire ✓ XMPP
Conectores de Aplicación	✓ Aplicaciones Google ✓ Aplicaciones Oracle ✓ PeopleSoft ✓ Salesforce.com ✓ Siebel
Soporte Dinámico para Formularios	✓ Soporta XForms 1.1 ✓ Diseñador XForms ✓ Engine XForms ✓ Intalio|AJAX
Gestor de Flujos de Tareas	✓ Planificación de tareas ✓ Notificación de tareas ✓ Manipulación de tareas ✓ Eliminación de tareas ✓ Asignación de tareas ✓ Gestor extensible de tareas ✓ Soporta BPEL4People

Fuente: http://www.intalio.com/products/server/#features (Traducción)

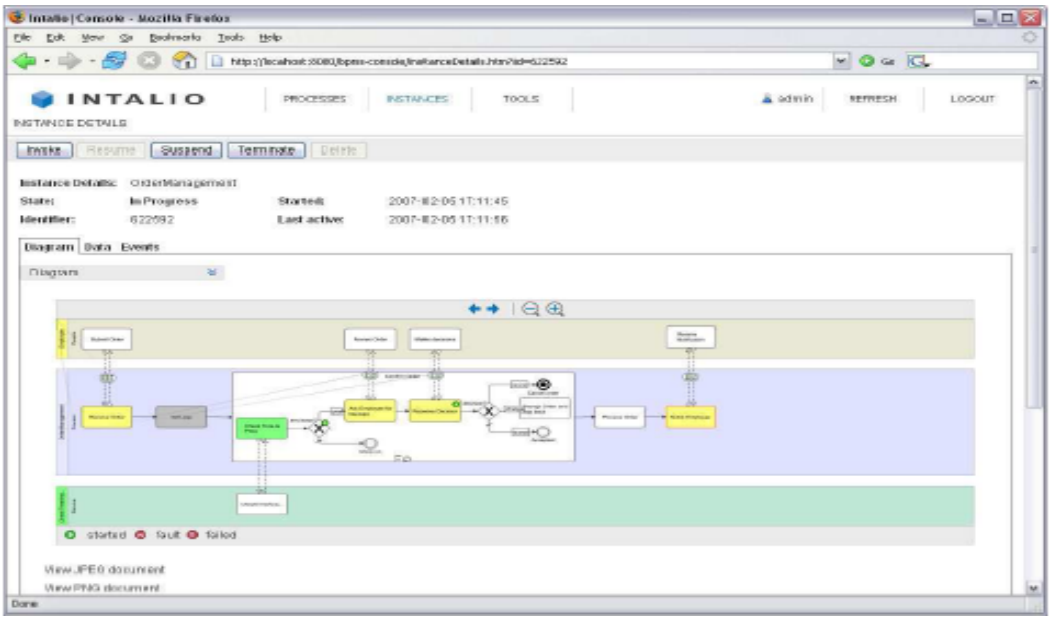


Figura 104: Servidor de Intalio

Fuente: Seminario Realizado por Ing. Cejas J. Junio 2008

Netbeans

Es un proyecto de código abierto con gran base de usuarios. Fundado por SunMicroSystems en junio del año 2000, en la actualidad sigue siendo su mayor patrocinador. Actualmente existen dos productos: el Netbeans IDE y Netbeans Platform. Este trabajo Especial de Grado se contempla es el NetbeansIDE, a continuación una breve definición.

Definición de NetbeansIDE

Según Netbeans.org, es un entorno de desarrollo - una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación.

Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Entre sus características se encuentra un sistema de proyectos basado en Ant[1], control de versiones y refactoring[2]. Todas las funciones del IDE son provistas por módulos. Cada módulo provee una función bien definida, tales como el soporte de Java, edición, o soporte para el sistema de control de versiones. NetBeans contiene todos los módulos necesarios para el desarrollo de aplicaciones Java en una sola descarga, permitiéndole al usuario comenzar a trabajar inmediatamente.

El Netbeans 6.7.1 usado en el trabajo de grado, fue creado el 27 de julio de 2009 esta versión soporta el paquete visual web, el cual, permite construir aplicaciones web estándar, incluido soporte para AJAX y componentes JSF, de manera rápida y visualmente.

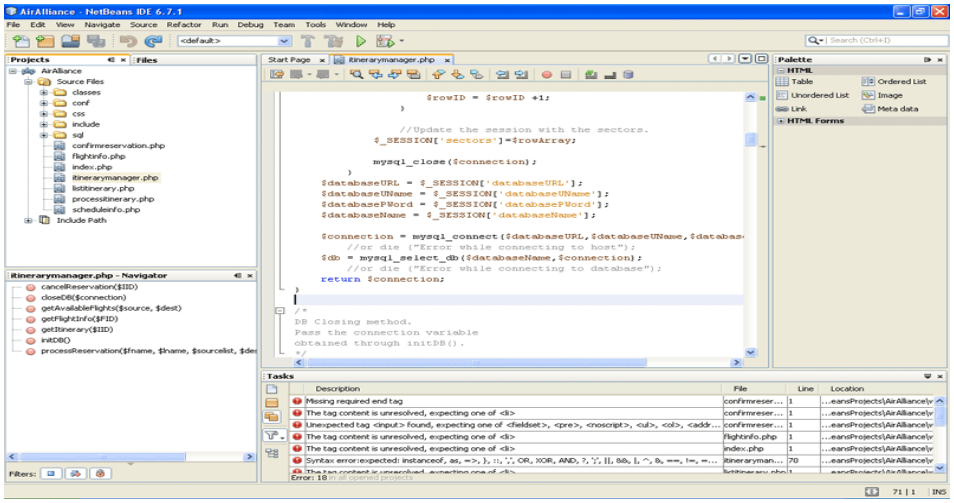


Figura 105: Vista de Netbeans 6.7.1

[1] Ant: es una herramienta del proyecto jakarta de Apache que viene a ser un equivalente de make (mucho más moderno) para los desarrolladores Java.
[2] *Refactoring*: es una técnica de la ingeniería de software para reestructurar un código fuente, alterando su estructura interna sin cambiar su comportamiento externo.

Como se necesita comunicar los procesos desarrollados ya sea a través de Intalio BPMS y/o la Aplicación Web se debe definir Servicios Web, enunciados a continuación:

Servicios Web (Web Services)

Los Servicios Web proponen una nueva forma de intercomunicar la lógica de negocio de las empresas, basándose en protocolos abiertos como XML y SOAP, facilitando así la colaboración entre las mismas.

Servicios Web es "un recurso programable y direccionable mediante una URL". Desde el punto de vista del programador, es "una clase, que se auto-define usando XML, y que es accesible mediante HTTP". La clase es auto-definida en XML para ofrecer al exterior un interfaz estándar con el que utilizar los métodos que proporciona y es accesible mediante HTTP por ser este el protocolo de aplicación utilizado por los navegadores web.

Si se propone la creación de unos servicios, que serán utilizados de forma pública por otras empresas externas a la que proporciona el mismo, se plantea la necesidad de un método de localización y búsqueda de los mismos.

Las capacidades de los Servicios Web permiten a las herramientas de BPM apoyar la ejecución del proceso de forma automatizada. Vollmer et al. (2004) discuten que la puesta en práctica de las soluciones de BPM ha emergido como estrategia dominante, que las organizaciones están adoptando para mejorar la eficacia de sus procesos del negocio.

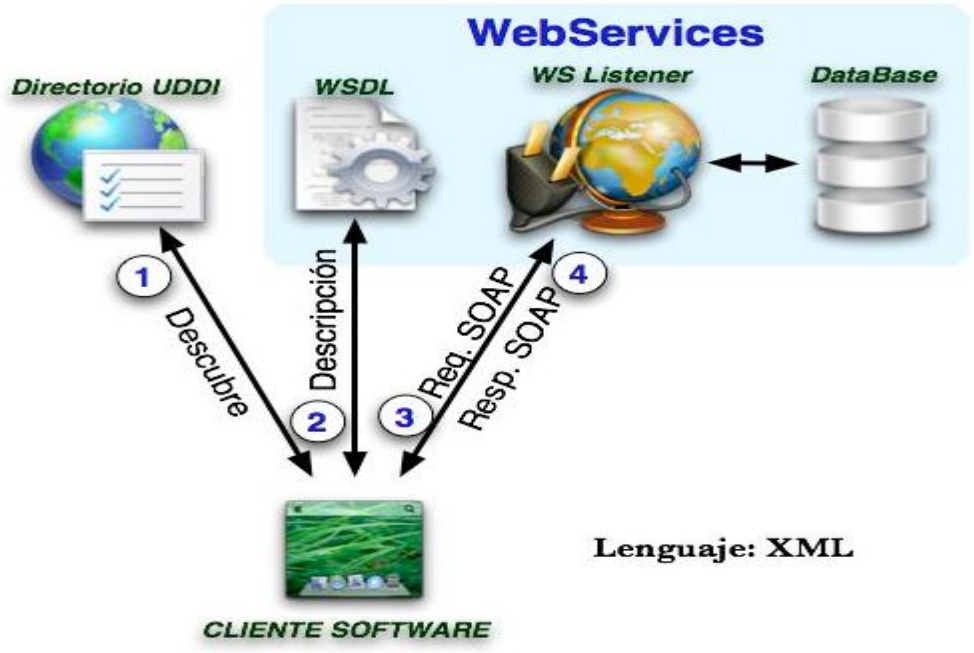


Figura 106: Estructura de un servicio Web

Fuente: GETSI III - http://www.eddysan.com/wp-content/uploads/2009/02/web-services-conceptos.pdf

Los Servicios Web han impulsado la Arquitectura Orientadas a Servicios (SOA), a continuación se mostrará a detalle.

Arquitecturas Orientadas a Servicios (Service-oriented architecture – SOA)

Según Microsoft Corporation, SOA establece un marco de diseño para la integración de aplicaciones independientes de manera que desde la red pueda accederse a sus funcionalidades, las cuales se ofrecen como servicios. La forma más habitual de implementarla es mediante Servicios Web, una tecnología basada en estándares e independiente de la plataforma, con la que SOA puede descomponer aplicaciones monolíticas en un conjunto de servicios e implementar esta funcionalidad en forma modular.

Otro concepto, propuesto por Gartner "SOA es una arquitectura de software que comienza con una definición de interface y construye toda la topología de la aplicación como una topología de interfaces, implementaciones y llamados a interfaces. Sería mejor llamarla "arquitectura orientada a interfaces". SOA es una relación de servicios y consumidores de servicios, ambos suficientemente amplios para representar una función de negocios completa".

SOA tiene las siguientes propiedades:

- **Vista lógica:** el servicio es una abstracción (vista lógica) de los programas, bases de datos, procesos de negocio, etc., definido en términos de lo que hace (llevando a cabo una operación de negocio).
- **Orientación a mensajes: e**l servicio se define formalmente en términos de los mensajes intercambiados entre agentes proveedores y solicitantes, y no está basado en las propiedades de los agentes. La estructura interna del agente (lenguaje de programación, BD, proceso, etc.) se abstrae en SOA. Esto permite incorporar cualquier componente o aplicación a esta arquitectura "decorando" estos componentes con software de gestión y conversión.
- **Orientación a la Descripción:** un servicio se describe con metadatos procesables. La descripción da soporte a la naturaleza pública de SOA: sólo se incluyen en la descripción aquellos detalles que se exponen públicamente y son importantes para el uso del servicio. La semántica de un servicio debe documentarse, directa o indirectamente, por su descripción.
- **Granularidad:** los servicios tienden a usar un pequeño número de operaciones con mensajes relativamente complejos.
- **Orientación a la Red:** los servicios tienden a usarse a través de la red, aunque este no es un requisito absoluto.

- **Neutral a la Plataforma:** los mensajes se envían en un formato estándar y neutral a la plataforma, distribuido a través de las interfaces (XML).

En general, SOA y Servicios Web son apropiados para aplicaciones:

- Que deben operar a través de Internet, donde la fiabilidad y la velocidad no se puede garantizar;
- Donde no existe habilidad de gestionar la instalación de forma que todos los solicitantes (clientes) y proveedores se actualicen a la vez;
- Donde los componentes de un sistema distribuido se ejecuten en distintas plataformas y distintos productos;
- Donde una aplicación existente necesite exponerse para ser usada a través de la red y pueda "decorarse" como un servicio Web. (Acevedo, 2007)

En términos de modelos de interacción abstractos y tal como se aprecia en la Figura Nº 36, SOA requiere de las siguientes interacciones

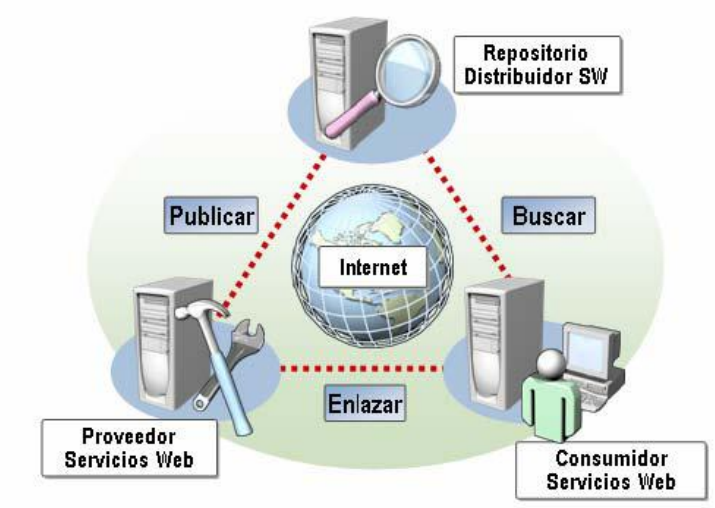


Figura 107: Arquitectura Orientada a Servicios.

Fuente: Acevedo. 2008

Todo Servicio Web necesita de un protocolo de comunicación, el cual se define a continuación.

SOAP (Simple Object Access Protocol – Protocolo Simple de Acceso a Objeto)

La funcionalidad que aporta SOAP es la de proporcionar un mecanismo simple y ligero de intercambio de información entre dos puntos usando el lenguaje XML. SOAP no es más que un mecanismo sencillo de expresar la información mediante un modelo de empaquetado de datos modular y una serie de mecanismos de codificación de datos.

Esto permite que SOAP sea utilizado en un amplio rango de servidores de aplicaciones que trabajen mediante el modelo de comunicación RPC (*Remote Procedure Call*). SOAP consta de tres partes:

• El SOAP *envelope* que define el marco de trabajo que determina qué se puede introducir en un mensaje, quién debería hacerlo y si esa operación es opcional u obligatoria.

• Las reglas de codificación SOAP que definen el mecanismo de serialización que será usado para encapsular en los mensajes los distintos tipos de datos.

• La representación SOAP RPC que define un modo de funcionamiento a la hora de realizar llamadas a procedimientos remotos y la obtención de sus resultados.

Objetivos de SOAP

• Establecer un protocolo estándar de invocación a servicios remotos que esté basado en protocolos estándares de uso frecuente en Internet, como son HTTP (*Hiper Text Transport Protocol*) para la transmisión y XML (*eXtensible Markup Language*) para la codificación de los datos.

• Independencia de plataforma hardware, lenguaje de programación e implementación del servicio Web. El logro de estos objetivos ha hecho de SOAP un

protocolo extremadamente útil, ya que el protocolo de comunicación HTTP es el empleado para la conexión sobre Internet, por lo que se garantiza que cualquier cliente con un navegador estándar pueda conectarse con un servidor remoto. Además, los datos en la transmisión se empaquetan o serializan con el lenguaje XML, que se ha convertido en algo imprescindible en el intercambio de datos, porque es capaz de salvar las incompatibilidades que existían en el resto de protocolos de representación de datos de la red.

Por otra parte, los Servicios Web pueden procesar las peticiones de usuario empleando tecnologías tales como: Servlets, páginas ASP (*Active Server Pages*), páginas JSP (*Java Server Pages*) o sencillamente un servidor de aplicaciones.

Un ejemplo típico de diseño de un Servicio Web, utilizando las ventajas de SOAP en la figura N° 37.

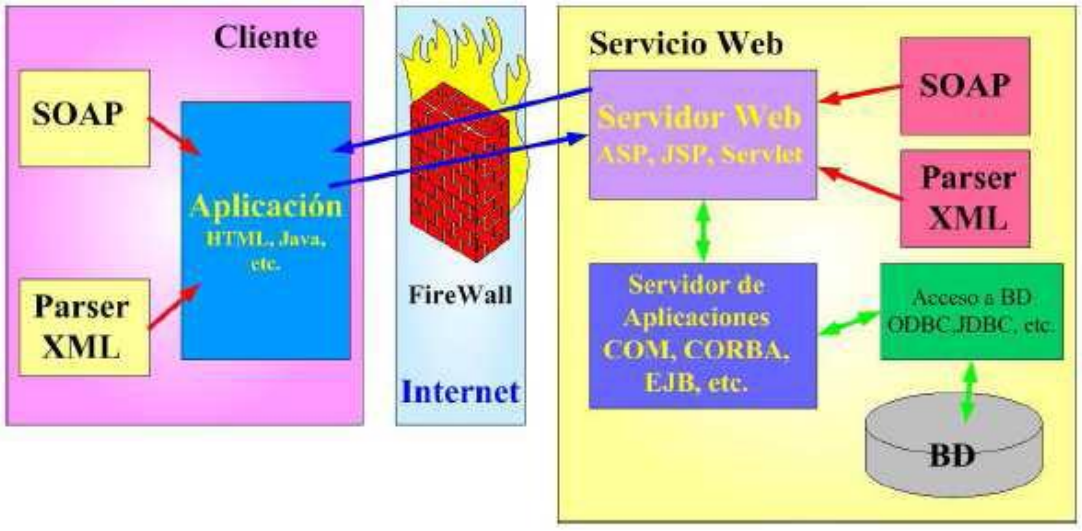


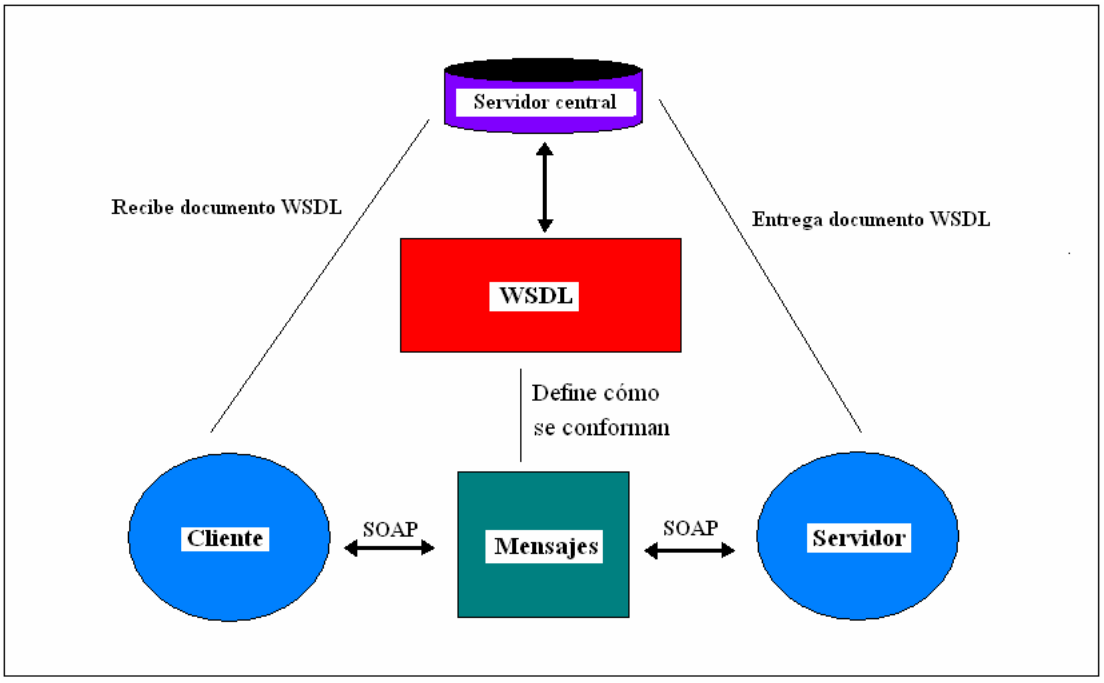
Figura 108: Ejemplo de uso de SOAP

Fuente: Biblioteca de Ingeniería – Universidad de Sevilla – SOAP y WSDL

Todo Servicio Web necesita describir la información que contiene para ello usa el Lenguaje de Descripción de Servicios, enunciado a continuación.

WSDL (Lenguaje de Descripción de Servicios)

El lenguaje de descripción de Servicios Web (WSDL, *Web Service Description Language*) es un dialecto basado en XML sobre el esquema que describe un servicio Web. Un documento WSDL proporciona la información necesaria al cliente para interaccionar con el servicio Web. WSDL es extensible y se pude utilizar para describir, prácticamente, cualquier servicio de red, incluyendo SOAP sobre HTTP e incluso protocolos que no se basan en XML



.

Figura 38: Esquema WSDL

Fuente: Biblioteca de Ingeniería – Universidad de Sevilla – SOAP y WSDL

Dado que los protocolos de comunicaciones y los formatos de mensajes están estandarizados en la comunidad del Web, cada día aumenta la posibilidad e importancia de describir las comunicaciones de forma estructurada. WSDL afronta esta necesidad definiendo una gramática XML que describe los servicios de red como

colecciones de puntos finales de comunicación capaces de intercambiar mensajes. Las definiciones de servicio de WSDL proporcionan documentación para sistemas distribuidos y sirven como fórmula para automatizar los detalles que toman parte en la comunicación entre aplicaciones. Las definiciones de servicio de WSDL proporcionan documentación para sistemas distribuidos y sirven como fórmula para automatizar los detalles que toman parte en la comunicación entre aplicaciones.

Java Server Faces (JSF)

Es un framework de desarrollo ágil, basado en el patrón Modelo-Vista-Controlador (MVC), licencia *Open Source.*

El objetivo de JSF es la separación entre la lógica de la aplicación y la presentación, facilitan la conexión de la capa de presentación con el código de aplicación. Este diseño permite a cada miembro de un equipo de desarrollo de aplicaciones web, centrarse en su proceso de desarrollo, y también proporciona un modelo de programación sencillo para vincular las partes.

La tecnología JSF incluye:

- Un conjunto de APIs para representar componentes de interfaz de usuario y de manejo, el manejo de eventos y validación de entrada, definición de navegación de página, y soporte a la accesibilidad.
- Java Server Pages (JSP), posee biblioteca de etiquetas personalizadas para la expresión de una interfaz Java Server Faces dentro de una página JSP.

Java Server Pages (JSP)

Es una tecnología Web que permite a los desarrolladores y diseñadores desarrollar de manera rápida páginas web dinámicas. La misma es un desarrollo de la compañía Sun Microsystem.

Como parte de la familia de la tecnología Java, la tecnología JSP permite el rápido desarrollo de aplicaciones basadas en Web que son independiente de la plataforma. JSP separa la tecnología de la interfaz de usuario de la generación de contenidos, permitiendo a los diseñadores cambiar el diseño de página sin alterar la dinámica subyacente contenido.

Las JSP's permiten la utilización de código Java mediante scripts. Además, es posible utilizar algunas acciones JSP predefinidas mediante etiquetas. Estas etiquetas pueden ser enriquecidas mediante la utilización de Bibliotecas de Etiquetas (TagLibs o Tag Libraries) externas e incluso personalizadas.

Como todo desarrollo debe insertar, actualizar y almacenar información, es necesario contar con un manejador de Base de Datos. A continuación se detalla brevemente el Manejador de Base de Datos MySQL.

Manejador de Base de Datos MySQL

MySQL es un sistema gestor de bases de datos. Pero la virtud fundamental y la clave de su éxito es que se trata de un sistema de libre distribución y de código abierto.

Lo primero significa que se puede descargar libremente de Internet, por ejemplo de la dirección (http://www.mysql.com); lo segundo (código abierto) significa que cualquier programador puede remodelar el código de la aplicación para mejorarlo.

Esa es también la base del funcionamiento del sistema Linux, por eso MySQL se distribuye fundamentalmente para Linux, aunque también hay versiones para Windows.

Existen cuatro versiones de MySQL:

- Estándar. Incluye el motor estándar y la posibilidad de usar bases de datos InnoDB. Todo el potencial de MySQL, pero sin soporte completo para utilizar transacciones.

- Max. Para usuarios que quieran MySQL con herramientas de prueba para realizar opciones avanzadas de base de datos
- Pro. Versión comercial del MySQL estándar
- Classic. Igual que la estándar pero no dispone de soporte para InnoDB.

El uso de MySQL (excepto en la versión Pro) está sujeto a licencia GNU *public license* (llamada GPL). Está licencia admite el uso de MySQL para crear cualquier tipo de aplicación. Se pueden distribuir copias de los archivos de MySQL, salvo esas copias se cobren a un tercer usuario. Se prohíbe cobrar por incluir MySQL. Se puede modificar el código fuente de MySQL, pero si se distribuye la aplicación con el código modificado, habrá que obtener una copia comercial y consultar sobre el cobro de la licencia. Al distribuir copias, se tiene que poder obtener información sobre las licencias GNU.