# Universidad de las Ciencias Informáticas
# Facultad 3


# Componente para la persistencia de procesos de negocio del componente de flujos de trabajo del marco de trabajo Sauxe.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

**Autor(es):** William Salmerón Pérez

Adrian José Castellanos Puertas

**Tutor(es):** Ing. René Bauta Camejo

Ing. Yoriangel Rivero González

"[insertar mes y año de la defensa]"

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizo al <nombre área> de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Autor: William Salmerón Pérez

_____

Autor: Adrian José Castellanos Puertas

_____

Tutor: René Rodrigo Bauta Camejo

_____

Tutor: Yoriangel Rivero González

_____

DEDICATORIA

*De William:*

*A mis padres por creer en mí y brindarme siempre su apoyo, por darme la fuerza para llegar hasta aquí, por estar siempre presentes y por confiar siempre en mí. A mi herma que es mi vida y la quiero mucho.*

*De Adrian:*

*A mi mamá, porque eres lo mejor de mi vida, a mi hermana que eres la niña de mis ojos, y al tato por tu afecto incondicional.*

AGRADECIMIENTOS

*De William:*
*A mis tutores por habernos ayudado tanto y por estar disponibles siempre que los necesitamos.*

*A el tercer miembro de este trabajo, el Ing. José Francisco Collado por todo su apoyo y ayuda.*

*A Elvis y Ernesto Adrian por haber sido y por seguir siendo como mis hermanos aunque estén lejos.*

*A todas las personas del laboratorio #11 que de una forma u otra han aportado algo para el desarrollo de esta tesis. En especial a Fidel por su ayuda incondicional.*

*A toda la gente del grupo de primer año que fue el mejor grupo de la UCI, en especial a "la cabezona" (Gloria), por aguantarme por tanto tiempo.*

*A mis amigos Ernesto Mato, el venezolano (Pedro), Pedro Luis, Luis, Jean Pablo, Vladimir, por ser las personas con las que pasé más tiempo en mi carrera.*

*A todos los profesores que contribuyeron a mi formación como futuro profesional y que de una forma u otra han hecho de mí una mejor persona.*

AGRADECIMIENTOS

*De Adrian:*
*A mi familia por su apoyo y ayuda incondicional, sin ustedes nada de esto hubiera sido posible y en especial a mi mamá por su optimismo.*

*A Mayte, gracias por estar siempre ahí y ayudarme sin pedir nada a cambio, por brindarme su apoyo y darme fuerzas cuando lo necesité para seguir adelante.*

*A mis amistades, por los buenos momentos compartidos, al Chino, al Gordo, a Yudel, a mis compañeros de la Universidad, los que quedamos y lo que ya no están. A mi gente del laboratorio y a los de mi apartamento.*

*A mis tutores, por su apoyo y por todo el conocimiento que fueron capaces de brindarme para realizar este trabajo y a mi formación como profesional.*

RESUMEN

Debido a la necesidad de contar con sistemas informáticos que administren y gestionen los recursos empresariales, que se adapten a cualquier estructura organizativa y que faciliten e integren la información que se generan en las empresas surgen los sistemas de Planificación de Recursos Empresariales (ERP). Estos sistemas vienen diseñados para realizar procesos de negocio específicos para cada área de una empresa, dichos procesos de negocio vienen definidos en el código fuente del sistema, por lo que no pueden ser cambiados.

Los ERP cuentan con un grupo de parámetros editables para darle un poco de flexibilidad a dichos procesos. Para lograr la gestión de estos procesos de negocio surge la idea de crear sistemas que combinen las funcionalidades de los ERP y los sistemas Workflow, ya que estos permiten el modelado análisis y administración de dichos procesos de negocio.

Actualmente en la Universidad de las Ciencias Informáticas (UCI) se está desarrollando un editor de proceso de negocio que tiene la necesidad de llevar las definiciones de flujo de trabajo al estándar XPDL (del inglés XML Process Definition Language). Por lo que este trabajo tiene como objetivo general el desarrollo de una herramienta que permita importar y exportar modelos de procesos de negocio, logrando así su estandarización y persistencia, además del intercambio de dichos procesos entre diferentes herramientas que implementen el lenguaje XPDL.

**Palabras claves:** proceso de negocio, flujo de trabajo, persistencia.

ÍNDICE

ÍNDICE DE FIGURAS.

ÍNDICE DE TABLAS.

ÍNDICE DE GRÁFICAS.

# INTRODUCCIÓN.

El rápido crecimiento y la evolución vertiginosa del mundo han traído consigo el surgimiento de las tecnologías de la información y las comunicaciones (TIC) las que han sido indispensables para el desarrollo global. Debido al constante movimiento y cambio de los procesos de negocio[i] y los volúmenes de información que se manejan, se requiere de herramientas que permitan el manejo y gestión de estos recursos empresariales y que se acoplen a cualquier estructura organizativa. En esta época en la que la globalización es parte fundamental del desarrollo de la economía mundial, es necesario implementar técnicas que ayuden a optimizar los procesos operativos de un negocio. Además, el ambiente actual crea la necesidad de contar con procesos y actividades más eficientes que proporcionen una mayor competitividad y productividad para mejorar su lugar en el mercado mundial actual.[1]

En una estructura organizativa, se ejecutan procedimientos los cuales son llevados a cabo por sus trabajadores. Cada trabajador tiene asignado un cargo y por este cumple con una actividad específica, de esta manera se pueden identificar cada uno de estos procedimientos como una secuencia de actividades que identifican flujos de trabajo, los cuales organizan y controlan las tareas, recursos y reglas necesarias para completar los procesos de negocio. Los procesos de negocio no son más que un grupo de tareas lógicamente relacionadas y ejecutadas para alcanzar un resultado específico en el negocio.[1]

Para mantener una empresa es necesario realizar continuas optimizaciones de sus costos además de un aumento constante de sus producciones, es necesario contar con una forma de integración y organización de la información que se genera en sus departamentos, además de integrar esta información por áreas vitales para la empresa.

Los primeros sistemas desarrollados para manejar estos recursos eran creados para un departamento específico de cada empresa, así cada uno de estos tenía sus propias aplicaciones. Todo esto traía consigo una gran adaptabilidad de estos sistemas a las áreas para las que eran desarrollados, pero entorpecía el flujo de información y la comunicación entre cada uno de los departamentos. Por lo que las grandes empresas del mundo se deciden por el uso de un sistema informático que les permita la integración de las operaciones. Debido a esta necesidad es que surgen los sistemas de Planificación de Recursos Empresariales (ERP).[2]

Los sistemas ERP son básicamente un software para empresas, que facilita e integra la información, ayudan a las diferentes partes de la organización en la compartición de información y conocimiento, así como a su comunicación, también ayuda en la reducción de costos al simplificar los procesos de negocio.[3]

Los ERP son diseñados para realizar los procesos de negocio de cada área de una forma específica, es decir, vienen diseñados para sectores específicos de la industria de una empresa, que no tienen porque aplicarse a otras empresas. Además de traer la definición de estos procesos de negocio en su código fuente de manera tal que no pueden ser cambiados, por lo que cuentan con un gran número de parámetros variables para lograr que se adapten lo mejor posible. Con el paso del tiempo estos procesos pueden ya no ser los que mejores se adapten a lo que necesita la empresa y realizarle cambios o actualizaciones puede ser tan costoso como el precio del mismo software. [3]

Por lo que en muchos casos a partir de este punto, las empresas tienen que pensar en cómo adaptarse a los sistemas ERP, en lugar de estos adaptarse a las necesidades de cada empresa, de modo que esto no puede ser una solución para una empresa cambiante.

Los beneficios serían muchos si se le diera una flexibilidad a los procesos de negocios y pudieran ser cambiados a conveniencia según las necesidades de una empresa, debido a que ante la necesidad de una industria competitiva, dinámica, que se adapte rápidamente a los cambios, y donde se aprovechen al máximo los recursos disponibles, el modelado, análisis y administración de estos procesos, tomaría gran importancia. Por esta razón, la creación de una combinación entre ERPs y la tecnología Workflow sería la respuesta.

La Workflow Management Coalition (WfMC) define Workflow como:"*automatización, total o parcial, de los procesos de negocio, que involucra el transporte de documentos, información o tareas de un participante a otro, de acuerdo a un conjunto de reglas establecidas para conseguir el objetivo global del negocio*" La WfMC es un consorcio industrial formado por compañías, vendedores, organizaciones de usuarios, y consultores para definir estándares para la interoperabilidad de sistemas de gestión de Workflow.[4]

Las técnicas de modelado de procesos de negocio son muy importantes en un sistema de Workflow, ya que permiten definir formalmente, secuencias de actividades relacionadas lógicamente entre sí para lograr un objetivo, lo que define un proceso, de tal forma que puedan ser interpretados y ejecutados

correctamente. Hoy en día el modelado de procesos ayuda a generar una representación exacta de la ejecución actual de los procesos y de la ejecución que la empresa necesita o desea, estos son importantes para la configuración del ERP porque los modelos de procesos describen las actividades de la empresa, las reglas del negocio, las funciones de organización y otros asuntos que se deben configurar en los sistemas de información. Los sistemas Workflow proporcionan a la empresa ahorro de tiempo, mejora de la productividad y eficiencia de la empresa, debido a la informatización de muchos procesos de negocio, optimización de la circulación de información interna con clientes y proveedores; mayor flexibilidad de acuerdo con las necesidades empresariales y una adecuada integración con sistemas de información actuales como es el caso de los ERP. Por esta razón surge la idea de crear sistemas que integren las funcionalidades que brinda un sistema ERP y las de un sistema Workflow, proporcionando a los ERP mayor adaptabilidad a los procesos de negocio y los cambios que estos necesiten.[5]

Nuestro país propone insertarse de manera gradual en este proceso de informatización en el que están inmersas la mayoría de las empresas a nivel mundial. Actualmente se encuentra en desarrollo, en la Universidad de Ciencias Informáticas (UCI), el ERP Cedrux, el cual tiene como base tecnológica el marco de trabajo Sauxe, elaborado en el departamento de Tecnología del centro CEIGE (Centro de Informatización y Gestión de Entidades).

Como parte de la evolución de Sauxe se le está desarrollando una herramienta para la gestión de flujos de trabajo con el objetivo de lograr la maleabilidad en los procesos de negocio. Esta herramienta tiene la necesidad de llevar las definiciones de flujo de trabajo al estándar XPDL (del inglés XML Process Definition Language) y de este estándar a las definiciones de flujo de trabajo para lograr la estandarización y la persistencia, posibilitando la gestión de estos procesos a conveniencia. El estándar XPDL es un lenguaje para la definición de flujos de trabajo definido por la WfMC. El uso de éste estándar para la definición de los procesos brinda la posibilidad de almacenar y permitir el intercambio de diagramas de procesos por cualquier editor que lo implemente para la representación gráfica del modelo.[6]

Lo antes expuesto genera el siguiente ***problema a resolver***: ¿Cómo lograr la persistencia de una definición de flujos de trabajo a partir de un proceso modelado en el componente de gestión de flujos de trabajo del marco de trabajo Sauxe?

Para darle solución a este problema se definió como ***objetivo general***: Desarrollar un componente que permita importar y exportar modelos de procesos de negocio.

Este objetivo general se desglosa en los siguientes ***objetivos específicos***:

- Estudio del estado del arte sobre las herramientas que implementan XPDL como lenguaje de especificación de flujos de trabajo.
- Análisis y diseño de la solución propuesta.
- Implementación de la solución propuesta a partir de las reglas de transformación identificadas.
- Validación de la solución propuesta.

Teniendo como ***objeto de estudio***: El modelado de procesos de negocio.

Centrando su ***campo de acción*** en: Las herramientas para el modelado de procesos que implementen el estándar XPDL.

La investigación parte de la siguiente ***idea a defender***: Con el desarrollo de un componente que permita importar y exportar modelos de procesos de negocio se logrará la persistencia de definiciones de flujos de trabajo a partir de un proceso modelado en el componente de gestión de flujos de trabajo del marco de trabajo Sauxe

Para el desarrollo de la solución se definen las siguientes ***tareas de la investigación***:

- Realizar un estudio del estado del arte de las principales herramientas relacionadas con el objetivo de la investigación.
- Realizar el análisis y diseño de la solución generando los artefactos definidos por la metodología seleccionada.
- Implementar las reglas de mapeo para el objeto BPMN “eventos de inicio” a sus elementos XPDL correspondientes.
    - Evento de mensaje.
    - Evento de tiempo.
    - Evento de regla.
    - Evento de vínculo.

    - Evento múltiple.
- Implementar las reglas de mapeo para el objeto BPMN “eventos de fin” a sus elementos XPDL correspondientes.
    - Evento de mensaje.
    - Evento de error.
    - Evento de cancelación.
    - Evento de vínculo.
    - Evento de terminación.
    - Evento múltiple.
- Implementar las reglas de mapeo para el objeto BPMN “eventos intermedios” a sus elementos XPDL correspondientes.
    - Evento de mensaje.
    - Evento de tiempo.
    - Evento de error.
    - Evento de cancelación.
    - Evento de compensación.
    - Evento de regla.
    - Evento de vínculo.
    - Evento múltiple.
- Implementar las reglas de mapeo para el objeto BPMN “tareas” a sus elementos XPDL correspondientes.
    - Tarea de servicio.
    - Tarea de recibir.
    - Tarea de envío.
    - Tarea de usuario.
    - Tarea de guión.
    - Tarea manual.
    - Tarea referenciada.

- Implementar las reglas de mapeo para el objeto BPMN “subprocesos” a sus elementos XPDL correspondientes.

    - Subprocesos embebidos.
    - Subprocesos independientes.
    - Subprocesos de referencia.
- Implementar las reglas de mapeo para el objeto BPMN “nodos de decisión” a sus elementos XPDL correspondientes.
    - Decisión exclusiva / fusión (XOR).
    - Decisión inclusiva / fusión (OR).
    - Decisión compleja / fusión.
    - Tenedor paralelo / Ingreso (AND).
- Implementar las reglas de mapeo para el objeto BPMN “transiciones” a sus elementos XPDL correspondientes.
    - Transición de Flujo de secuencia.
    - Transición de Flujo de mensaje.
    - Transición de asociación.
- Implementar las reglas de mapeo para el objeto BPMN “mensajes” a sus elementos XPDL correspondientes.
- Implementar las reglas de mapeo para el objeto BPMN “piscinas” a sus elementos XPDL correspondientes.
- Implementar las reglas de mapeo para el objeto BPMN “carriles” a sus elementos XPDL correspondientes.
- Validar la solución propuesta.

El presente trabajo está estructurado en 3 capítulos, a continuación se muestra una breve descripción de cada uno de ellos:

- ***Capítulo 1: “Fundamentación Teórica”***. Este capítulo comprende el estudio del estado del arte acerca de los elementos fundamentales a tener en cuenta para la solución del problema planteado, se analizan y definen las herramientas, tecnologías y modelo de desarrollo a utilizar en la solución.
- ***Capítulo 2: “Análisis y Diseño del Sistema”***. Este capítulo describe la propuesta de solución para el problema planteado y el diseño de la misma. Se describen los artefactos que propone el modelo de desarrollo para esta fase.

- ***Capítulo* 3: *"Implementación y Prueba del Sistema"***. En este capítulo se presentan los diagramas de componentes que se definieron durante la implementación de la aplicación, y las pruebas realizadas para comprobar sus funcionalidades.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

## 1.1 Introducción.

Este capítulo constituye el marco teórico de la investigación a realizar. Se precisan un grupo de conceptos necesarios para entender el objetivo fundamental del trabajo, profundizando en temas como la persistencia de procesos del negocio en flujos de trabajo, haciendo análisis de estas tecnologías a nivel mundial. Además se describen las tecnologías que son utilizadas en la implementación.

## 1.2 Persistencia.

Persistencia es la acción y efecto de persistir, mantenerse constante en algo, durar por largo tiempo. Persistencia en informática de modo genérico, se refiere a la propiedad de los datos para que estos sobrevivan de alguna manera y existen varios tipos de persistencia, uno de ellos es la Persistencia de Objeto. [7]

La persistencia de objetos consiste en la inicialización de objetos con sus atributos por defecto. Esto es posible si sobre un medio (de almacenamiento) fijo se guarda (cuando el objeto fue definido) un conjunto de datos que son recuperados cuando el tipo de objeto en cuestión es creado, dichos datos son transferidos a las propiedades del objeto.[7]

## 1.3 Procesos de negocio.

Un proceso de negocio no es más que una actividad o conjunto de actividades que utilizan recursos, y que se gestionan con el fin de permitir que los elementos de entrada se conviertan en resultados. Los procesos de negocio describen cómo es realizado el trabajo en las empresas, se caracterizan por ser observables, medibles, mejorables y repetitivos, estos pueden incluir o desencadenar otros procesos. [5]

Los procesos poseen las siguientes características:

- Pueden ser medidos y están orientados al rendimiento.
- Tienen resultados específicos.
- Responden a alguna acción o evento específico.

## 1.4 Flujo de Trabajo (workflow).

El concepto de workflow en la historia moderna se remonta a 1912 cuando F. Taylor y a H. Gantt, juntos iniciaron el estudio de la organización racional de trabajo, referido al ámbito de la manufactura. Sin embargo, la acepción que interesa es la referida a sistemas informáticos de modo que se incluye la que utiliza la WfMC, que plantea:

"Workflow es la automatización de un proceso de negocio, sea parcial o totalmente, durante el cual documentos, información o tareas son pasados desde un participante a otro para la ejecución de otra acción, de acuerdo a un conjunto de reglas"

La evolución de la gestión de flujo de trabajo consiste en la automatización de procesos de negocio, durante el cual los documentos, información o tareas pasan de un participante a otro de una manera que se rige por reglas o procedimientos.[8]

## 1.5 Tecnologías usadas mundialmente.

A nivel mundial existe un gran número de herramientas que utilizan el estándar XPDL para intercambiar definiciones de modelado de procesos de flujos de trabajos, las escogidas para realizárseles el estudio son reconocidas por la WfMC y se analizan de acuerdo con los criterios de: utilización de estándar XPDL, tipo de aplicación (web o escritorio), lenguaje de programación, el tipo de licencia y el consumo de servicios:

**Fujitsu Interstage (i-Flow).**

Con las soluciones de FUJITSU, se pude realizar tareas de forma paralela y documentar todo el proceso, lo que ofrece la oportunidad de hacer una reingeniería de los procesos de negocio. La solución está orientada a:[9]

- Mejora de los procesos de negocio.
- Automatizar de tareas.
- Facilitar la monitorización y el control de procesos.
- Simplificar la dinamización de la lógica de los procesos.
- Disminuir los costes de desarrollo.

La solución integral de BPM, propuesta por FUJITSU se basa en:

- Consultoría de reingeniería de procesos.
- Servicios profesionales de implantación, desarrollo y personalización.
- Tecnología de workflow: Interstage® BPM (i-Flow) tecnología que posibilita el modelado de procesos de negocio para su automatización y monitorización on-line, reduciendo tiempos de desarrollo y aumentando el control. [9]

Tecnológicamente i-Flow es un producto 100% basado en Java.

La herramienta posee mecanismos de integración con sistemas externos, como pueden ser las herramientas de EAI (Enterprise Application Integration), a través de Web Services, Java y XML. También, se conecta con bases de datos externas a través de adaptadores.

Adicionalmente, admite programación a través de Java y Java Script con la posibilidad de crear formularios de cliente de forma automática (HTML). [9]

Resumiendo las principales funcionalidades del producto i-Flow:

- Gestión de tiempos con distintos calendarios.
- Control de notificaciones basadas en correos electrónicos.
- Soporte para Servicios Web.
- Posibilidad de personalización del cliente (css).
- Soporte a Java 1.4.
- Módulos de reportes, analíticos y de auditoría.
- Modificación de procesos en tiempo de ejecución.
- Generación automática de formularios HTML.
- Sistema de gestión documental.
- Balanceo de carga.
- Plataforma de alta disponibilidad.
- Firma digital.

**Interstage Business Process Manager Studio.**

Es una suite que ayuda a las empresas y los profesionales para la refinación, automatización y optimización de procesos de negocio, reducir los errores y los costos, mejorar la eficiencia mediante la automatización de las tareas de seguimiento y las métricas de negocio, y optimizar los procesos en tiempo real. Es un modelador de procesos basado en XPDL 2.0. Es una plataforma de escritorio, capaz de trabajar a través de web.[10]

**Características principales:**

- 100% basado en el navegador de la consola.
- Proceso de modelado en línea.
- Las capacidades avanzadas de simulación del proceso.
- Potente y fácil de utilizar con tablas de decisión.
- Proceso de seguimiento de variables.
- Búsqueda avanzada, ordenación, filtro de tareas y procesos.
- Calendarios de negocios de los temporizadores.
- Plazos y notificaciones.
- Modificación en tiempo real.
- Análisis de procesos avanzados.
- Utiliza agentes de servicios para la integración con los sistemas existentes.
- Disparadores para controlar los eventos externos.
- Compatible con BPMN, XPDL, BPEL y Wf-XML 2.0.
- API extensa.
- Plantillas predefinidas de procesos como kits de iniciación.
- Compatible con los principales servidores de aplicaciones J2EE; también está disponible como una oferta independiente.

**TIBCO Business Studio.**

TIBCO Business Studio es un entorno unificado basado en Eclipse que permite, de una forma muy sencilla, modelar, simular y gestionar procesos de negocio. Se ocupa de los retos de orquestar personas y sistemas en los procesos de negocio, dentro y fuera de la organización. Es un software de plataforma de escritorio que permite el modelado y la gestión de procesos de negocio y el paso a la ejecución se realiza

de un modo más rápido y preciso. El producto incluye herramientas y visualizaciones específicamente dirigidas al nivel de experiencia de cada tipo de usuario para que tanto usuarios como técnicos puedan contribuir de forma eficiente y efectiva para crear un proceso de negocio ejecutable. Basado en los estándares de modelado de procesos con BPMN, y para la persistencia usa XPDL.[9]

A continuación se enumeran las principales características:

- **Vista de Negocio.**

  Visualización del proceso de negocio orientada al análisis. En esta vista no se representan detalles técnicos de la implementación del proceso de negocio.

- **Vista Técnica.**

  Visualización del proceso orientada a los analistas técnicos. Se muestran detalles técnicos de la implementación del proceso de negocio. Permite definir la implementación de:

  - Tareas manuales de Usuario. Se especifican los parámetros de entrada y salida.
  - Llamada a servicios definidos con WSDL.
  - E-mail. Definir el envío de un correo electrónico.
  - Llamada a base de datos. Es posible invocar procedimientos almacenados.
  - Java Script. Ejecución de código de Java Script en tiempo de ejecución.
  - Disparadores. Configurar el lanzamiento de eventos a una determinada fecha y hora.
  - Ejecución de código Java.

- **Simulación.**

  Es posible simular los procesos definidos con TIBCO Business Studio para detectar cuellos de botella, áreas de alto coste o zonas donde es posible reducir el nivel de servicio. Para la simulación se dispone de varios parámetros que pueden modificarse:

  - Coste de la tarea.
  - Número de agentes.
  - Distribución de la carga de trabajo.

**Together Workflow Editor.**

Consiste en una herramienta gráfica de código abierto que permite el diseño de procesos y genera el código XPDL, esta herramienta no se utiliza para el desarrollo de componentes en sí, sino para la creación de aplicaciones de demostración.[11]

Estas son algunas de las características claves de "Together Workflow Editor":

- Acciones gráficas para seleccionar todas las transiciones entre las actividades seleccionadas.
- Referencias de paquete XPDL transitorias.
- Admite copiar y pegar patrones de diseño de flujo de trabajo configurables.
- Iconos de actividad personalizables, utilizando los atributos de archivo XPDL nativos.
- Compatibilidad con configuraciones definidas por el usuario.
- Cambio de configuración dinámico en el tiempo de ejecución.
- Carga/guarda el diseño gráfico de procesos en atributos extendidos de XPDL.
- Compatibilidad con deshacer/rehacer para cada acción del editor.
- Ve elementos de referencia para un elemento XPDL.
- Historial de navegación con sistema de navegación hacia atrás y hacia delante.
- Botón de reversión en cada cuadro de diálogo de edición.
- Propagación de selecciones de elemento para todos los paneles.
- Navegación directa a elementos referenciados.
- Compatibilidad con la edición de expresión incluyendo la selección de variables de proceso.
- Rotación de gráfico (horizontal o verticalmente).

Luego de un estudio de estas herramientas reconocidas por la WfMC, se concluyó que estas no se pueden integrar con el marco de trabajo Sauxe debido a que todas brindan soporte para servicios de tipo web y una de las peculiaridades que tiene el marco de trabajo es que los servicios que consume son de integración por medio del mecanismo de inversión de control (IOC).

Por otro lado muchas de estas herramientas son de plataforma escritorio, y otra de las necesidades es que sea de plataforma web. Además que todas estas herramientas están desarrolladas en Java y el marco de trabajo Sauxe está realizado en php.

Se debe tener en cuenta que la mayoría de estas herramientas son propietarias y desarrolladas para el sistema operativo Windows y no se acoplan con la política que está teniendo nuestro país de emigrar hacia el software libre.

## 1.6 Modelo de Desarrollo Propuesto.

Para el desarrollo de cualquier producto de software se realizan una serie de tareas entre la idea inicial y el producto final, un modelo de desarrollo establece el orden en el que se harán las cosas en el proyecto y provee de requisitos de entrada y de salida para cada una de las actividades.[7]

La propuesta de modelo de desarrollo fue concebida por el equipo de producción del centro CEIGE teniendo en cuenta los riesgos con los que se cuentan en los proyectos. Este modelo está orientado a la reutilización de componentes parametrizables, donde se simplifican las actividades y se eleva el nivel de especialización de los implicados. Dicho modelo está basado en los principios de buenas prácticas y algunos elementos de las metodologías RUP, fue elaborado teniendo en cuenta las características especiales que presenta la UCI, por ejemplo: casi todos los proyectos están compuestos en su mayoría por estudiantes.[12]

**Características:**

- **Centrado en la arquitectura.**

  La arquitectura determina la línea base, los elementos de software estructurales a partir de los elementos de la arquitectura de negocio. Interviene en la gestión de cambios y diseña la evolución e integración del producto. La arquitectura orienta las prioridades del desarrollo y resuelve las necesidades tecnológicas y de soporte para el desarrollo. [12]

- **Orientado a componentes.**

  Las iteraciones son orientadas por el nivel de significado arquitectónico de los componentes, los mismos son abstracciones arquitectónicas de los procesos de negocio y requisitos asociados que modelan; el componente es la unidad de medición y ordenamiento de las iteraciones. [12]

- **Iterativo e incremental.**

  Las iteraciones son planificadas y coordinadas con el equipo de arquitectura, los clientes y la alta gerencia. Cada iteración constituye el desarrollo de componentes, los cuales son integrados al término de la iteración, permitiendo de esta manera la evolución incremental del producto. [12]

- **Ágil y adaptable al cambio.**

  El desarrollo de las partes formaliza solamente las características principales de la solución, priorizando los talleres y las comunicaciones entre las personas. Los clientes y funcionales están involucrados en el proyecto y poseen parte de la responsabilidad del éxito del mismo. Los cambios son conciliados semanalmente, discutidos y aprobados.[12]

## 1.7 Tecnologías, lenguajes y herramientas de desarrollo.

### 1.7.1 Lenguajes de programación.

Un lenguaje de programación es un idioma artificial diseñado para expresar procesos que pueden ser llevadas a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana.

**PHP.**

PHP, acrónimo de "Preprocesador de Hipertexto" por sus siglas en inglés, es un lenguaje interpretado de alto nivel embebido en páginas HTML. La mayoría de su sintaxis es similar a C, Java y Perl, con solamente un par de características PHP específicas. La meta de este lenguaje es permitir escribir a los creadores de páginas web, páginas dinámicas de una manera rápida y fácil. Una de sus características más potentes es su soporte para gran cantidad de bases de datos: como InterBase, mSQL, MySQL, Oracle, Informix, PostgreSQL, entre otras. PHP también ofrece la integración con varias bibliotecas externas. Como producto de código abierto, goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y se reparan rápidamente. El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.[13]

Entre sus desventajas se encuentran:

- La ofuscación de código es la única forma de ocultar las fuentes.
- El manejo de errores no es tan sofisticado.
- Promueve creación de código desordenado y con un mantenimiento complejo.

### 1.7.2 Marcos de trabajo.

**Marco de Trabajo Sauxe.**

Sauxe es un framework para aplicaciones web de gestión, construido para el ahorro de recursos humanos y materiales, así como para garantizar un entorno de desarrollo sustentado por soluciones estables y de alta calidad debido a la estandarización que este introduce al desarrollo. Además posibilita a los desarrolladores de aplicaciones web de gestión una rápida adquisición de conocimientos sobre la tecnología tipo que se propone, ahorrando tiempo, recursos y mejorando la calidad de los productos finales.[10]

Este marco de trabajo sigue el principio de la independencia tecnológica y garantiza además una alta productividad, extensibilidad, reusabilidad, integración con conocimientos de los lenguajes que soporta y de los escenarios más complejos como son multi-entidad, multi-moneda, multi-idioma, entre otros.[10]

El marco de trabajo Sauxe propone solución a un sin número de escenarios o aspectos arquitectónicos como: gestión y configuración dinámica de caché, integración de componentes de forma distribuida o no distribuida, implementación de mecanismos de mensajería y control de excepciones, gestión y configuración dinámica de precondiciones, pos condiciones y validaciones de variables, gestión y configuración de flujos de trabajo, visualización de las funcionalidades de un sistema, entre otros escenarios de alta complejidad, garantizando la calidad de los sistemas que se desarrollen con el mismo.[10]

**Zend Framework 1.9.5.**

Zend Framework no es más que un framework MVC (Modelo-Vista-Controlador) open-source para el desarrollo de aplicaciones web con PHP. Brinda soluciones para construir sitios web modernos, robustos y seguros. Posee un bajo acoplamiento entre sus componentes, lo que posibilita la utilización de los mismos a conveniencia.

Brinda una alta abstracción de bases de datos, haciendo extremadamente simple la interacción con estas, sin necesidad de escribir ninguna consulta SQL. Cuenta con módulos para el manejo de ficheros PDF, canales RSS, entre otros. Cuenta con clientes para el acceso a servicios web (WS) y robustas clases para la autenticación y el filtrado de entrada; completa documentación y test de alta calidad.[14]

### 1.7.3 Lenguajes de Notación.

**XML 2.0.**

XML, por sus siglas en inglés eXtensible Markup Language ('lenguaje de marcas extensible'), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML (Standard Generalized Markup Language) y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML y Android.[15]

XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable.[15]

XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil. [15]

**XPDL 2.0.**

El lenguaje de Definición de Proceso de XML (XPDL) es un formato estandarizado por la Workflow Management Coalition (WfMC) para intercambiar definiciones de modelado de procesos. XPDL define un esquema XML para especificar la parte declarativa de proceso. [16]

XPDL especifica un formato de diseño de los procesos. Permite una representación gráfica de los procesos incluyendo coordenadas X e Y para cada nodo implementado. Además, los nodos pueden especificar atributos tales como roles y descripción de actividades. Suele ser el más extendido cuando se trata de implementar procesos o workflow con interacciones humanas. Además, gracias a esa representación gráfica, el incorporar cambios a nuestros procedimientos ya modelados es tarea bastante sencilla y rápida. [16]

El objetivo de XPDL es almacenar y permitir el intercambio de diagramas de procesos. Intenta ofrecer una manera estándar para representar procesos de tal manera que puedan ser importados/exportados por cualquier editor que implemente el estándar.[16]

### 1.7.4 Herramientas de desarrollo.

**NetBeans 7.1.**

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios y una comunidad en constante crecimiento. Sun Microsystems fundó el proyecto de código abierto NetBeans en junio del año 2000 y actualmente es su patrocinador principal de los proyectos. [17]

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Es una base modular y extensible usada como estructura de integración para crear aplicaciones de escritorios grandes. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden utilizarse para desarrollar sus propias herramientas y soluciones. El entorno de desarrollo integrado (IDE) NetBeans es una herramienta para programadores pensada para 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 IDE NetBeans. Es un producto libre y gratuito sin restricciones de uso.[17]

**Visual Paradigm 4.3.**

Es una herramienta CASE de modelado que utiliza UML como lenguaje de modelado profesional y que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite realizar ingeniería tanto directa como inversa. La herramienta es colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto; genera la documentación del proyecto automáticamente en varios formatos como HTML, PDF y permite control de versiones. Dentro de sus características principales se puede destacar su robustez, usabilidad y portabilidad. Para el desarrollo de este trabajo se escogió como herramienta CASE de modelado al Visual Paradigm porque tiene un coste favorable y realiza de una forma íntegra los planes de construcción del software. Permite ingeniería inversa, del modelo físico se puede llegar al modelo lógico. Genera de forma automática códigos desde diagramas, además brinda la posibilidad de generar documentación.[18]

**Apache 2.2.16.**

Apache es una tecnología gratuita de código abierto, se usa en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal. Es un servidor altamente configurable de diseño modular. El servidor Apache implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Además de ser: [19]

- Modular.
- Código abierto.
- Multi-plataforma.
- Extensible.
- Popular (fácil conseguir ayuda/soporte).

**PostgreSQL 8.3.**

Es un sistema de gestión de bases de datos libre basado en el proyecto Postgres, perteneciente a la Universidad de Berkeley. Es un sistema objeto-relacional, que incluye características como la herencia, valores no atómicos (atributos basados en vectores y conjuntos), funciones, disparadores, entre otras. Es altamente extensible, permitiendo el uso de operadores, funciones y tipos de datos definidos por el usuario. Soporta la integridad referencial garantizando la integridad de los datos en la base de datos. PostgreSQL permite realizar múltiples conexiones desde procesos clientes, existiendo un proceso maestro en el servidor que siempre se ejecuta y que está a la espera de nuevas conexiones clientes, de forma tal que cuando alguien se conecta, se inicia un nuevo proceso, asegurando que el cliente y la nueva conexión no necesitan del proceso postgres original. Una de sus principales características es la alta concurrencia. Esto permite que mientras se realizan cambios en una tabla, otros procesos accedan a la misma sin la necesidad de bloqueos, además de que cada usuario tiene visión de la última modificación. Presenta el inconveniente de que para bases de datos pequeñas su velocidad de respuesta no es muy eficiente en comparación con otras relativamente grandes.[20]

**Geany 0.20.**

Geany es un editor de programación compatible con multitud de lenguajes, como C, Java, PHP, HTML, Python, Perl o Pascal, entre otros. Entre sus muchas funciones, Geany incluye resaltado de sintaxis, completado de código, autocompletado de construcciones habituales y de etiquetas XML y HTML y lista de símbolos. Geany permite compilar y ejecutar los programas creados o editados, ya sea desde el

terminal o desde su menú. Y además dispone de plugins, destacando autoguardado, buscador de archivos y exportador.[21]

**Características:**

- Compatible con la mayoría de lenguajes
- Varios paneles para acceder mejor a los datos
- Herramientas para compilar
- Buscador integrado

**Navegador Mozilla Firefox 3.0 – 8.0.**

Mozilla Firefox es un navegador de código abierto, multiplataforma, basado en el código de base de Mozilla que nos proporciona una navegación rápida, segura, y eficiente.[22]

Entre sus principales características se encuentran:

- **Velocidad**. Las páginas se abren en menor tiempo y se navega con comodidad en ellas.
- **Seguridad**. Este software de código abierto permite que las fallas de seguridad se corrijan al instante, al tener un grupo de colaboradores detrás dispuesto a ello. Al mismo tiempo Mozilla Firefox usa su propio motor de dibujado de páginas, Gecko, lo que lo hace inmune a las fallas de seguridad del Internet Explorer, que también tienen todos los navegadores basados en él.
- **Sin publicidad**, desde sus inicios Mozilla Firefox procuró evitarnos esas molestas ventanas emergentes de publicidad, de un modo sencillo y eficaz.
- **Navegación por pestañas,** esto nos permite tener varias páginas abiertas, en una misma ventana, permitiendo una navegación en ella mucho más cómoda.
- **Varios Perfiles**, se pueden usar varios perfiles de usuario con configuraciones diferentes para cada uno.
- **Extensiones y Temas**, esto te permite configurarlo a tu gusto con las numerosas extensiones y temas (apariencia), que se nos ofrece. Puedes hacer un Mozilla Firefox a tu medida.

**RapidSVN 0.12.0.**

RapidSVN es una plataforma visual para el sistema Subversion escrito en C++. Se utiliza un nuevo marco wxWidgets y también se incluye un cliente de Subversion C++ API en el proyecto.

El objetivo de este proyecto era conseguir un producto de fácil manejo para los usuarios principiantes pero lo suficientemente potente y con herramientas interesantes para los usuarios avanzados.

El programa funciona en cualquier plataforma y se puede ejecutar en Linux, Windows, Mac OS / X, Solaris, etc. Para facilitar más el trabajo los desarrolladores han incluido en su web un manual en línea.[23]

## 1.8 Conclusiones Parciales.

Se estudiaron los principales conceptos asociados con el tema da la investigación para asentar las bases del marco teórico, también se realizó un estudio de herramientas reconocidas por la WfMC con el objetivo de encontrar posibles reutilizaciones y se concluyó que estas no se pueden integrar con el marco de trabajo Sauxe debido a las características peculiares de este.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DEL SISTEMA

## 1.1 Introducción.

En el siguiente capítulo se concibe el análisis y diseño del sistema basado en el modelo de desarrollo propuesto. Se describen los requisitos funcionales para lograr un mejor entendimiento del comportamiento del software y seguidamente una descripción del diseño propuesto para alcanzar el objetivo trazado.

## 1.2 Modelo conceptual.

El modelo conceptual es una representación gráfica de los principales conceptos en un dominio del problema. Este modelo muestra las relaciones entre estos conceptos y los atributos de conceptos de la solución.

A continuación se muestra el diagrama del Modelo Conceptual efectuado para el componente WFTranslatorXPDL, el cual contiene un Editor de Procesos de Negocio (EPN) que genera un Paquete de Objetos que contiene todas las especificaciones de los objetos BPMN modelados en el EPN y que luego serán validados y transformado al estándar XPDL.

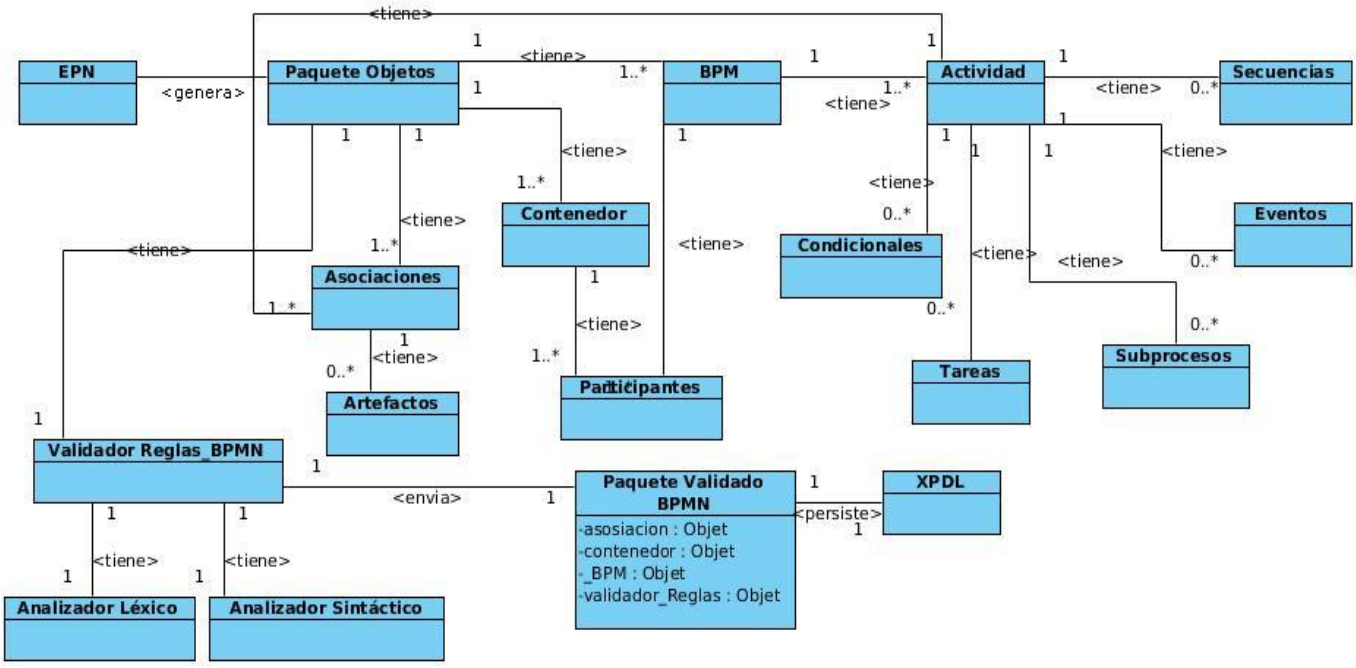



**Figura 1: Modelo Conceptual.**

**Tabla 1: Editor de procesos de negocio.**

| Descripción | Es una herramienta para el modelado de procesos de negocio para el Marco de Trabajo Sauxe. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Paquete de objetos. | Conjunto de clases que dan persistencias a los objetos BPMN. | Object | no | si | | |

**Tabla 2: Paquete de objetos.**

| Descripción | Conjunto de clases que dan persistencias a los elementos BPMN. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Asociación | Transición para asociar información adicional sobre el proceso. | Object | si | si | | |
| Contenedor | Representa un proceso. | Object | no | si | | |
| BPM | Conjunto de actividades asociadas entre sí para cumplir un objetivo. | Object | no | no | | |
| Validador de reglas BPMN | Componente para validar los procesos de negocios modelados con BPMN. | Object | no | si | | |

**Tabla 3: Asociación.**

| Descripción | Transición para asociar información adicional sobre el proceso. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Id | Identifica el objeto entre los otros objetos. | Object | no | si | | |
| Name | Nombre para identificar el objeto. | Object | si | no | | |
| Source | Especifica la salida del flujo de secuencia para cumplir un objetivo. | Object | si | si | | |
| Target | Especifica el objetivo del flujo de secuencia | Object | si | si | | |

**Tabla 4: Contenedor.**

| Descripción | Representa un proceso en el diagrama de proceso de negocio. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Participantes | Participantes en el proceso. | Object | si | si | | |
| Id | Identifica el objeto entre los otros objetos. | Object | no | si | | |

**Tabla 5: BPM.**

| Descripción | Conjunto de actividades asociadas entre sí para cumplir un objetivo. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Id | Unico para cada procesos | Object | no | si | | |
| Actividad | Representa el trabajo realizado dentro de una organización. | Object | si | si | | |

**Tabla 6: Participantes.**

| Descripción | Representa a los participantes de un proceso. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| BPM | Conjunto de actividades asociadas entre sí para cumplir un objetivo. | Object | no | no | | |
| Id | Identifica el objeto entre los otros objetos. | Object | no | si | | |

**Tabla 7: Actividad.**

| Descripción | Representa el trabajo realizado dentro de una organización. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Id | Identifica el objeto entre los otros objetos. | Object | no | si | | |
| Name | Nombre para identificar el objeto. | String | si | no | | |
| ActivityType | Especifica el tipo de actividad (Tarea, Evento, Subproceso, decisión, flujo de secuencia) | String | no | no | | |

**Tabla 8: Atributos.**

| Descripción | Son actividades simples o atómicas que se realizan en un proceso. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Id | Identifica el objeto entre los otros objetos. | Object | no | si | | |
| Name | Nombre para identificar el objeto. | String | si | no | | |
| TaskType | Especifica el tipo de tarea (Service, Receive, Send, User Script, Manual, Reference, None) | String | no | no | | |

**Tabla 9: Eventos.**

| Descripción | Representa algo que ocurre o puede ocurrir durante el curso de un proceso. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Id | Identifica el objeto entre los otros objetos. | Object | no | si | | |
| Name | Nombre para identificar el objeto. | String | si | no | | |
| EventType | Especifica el tipo de Evento que será (Start, intermédiate, End) | String | no | no | | |

**Tabla 10: Decisiones.**

| Descripción | Elementos utilizados para controlar la divergencia y convergencia del flujo. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Id | Identifica el objeto entre los otros objetos. | Object | no | si | | |
| Name | Nombre para identificar el objeto. | String | si | no | | |
| GatewayType | Especifica el tipo de decisión (XOR, OR, Complex, AND) | String | no | no | | |

**Tabla 11: Subproceso.**

| Descripción | Es una actividad compuesta que incluye un conjunto interno lógico de actividades (proceso). | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| id | Identifica el objeto entre los otros objetos. | Object | no | si | | |
| Name | Nombre para identificar el objeto. | String | si | no | | |
| SubProcessType | Especifica el tipo de subproceso (Embedded, Independent, Reference) | String | no | no | | |

**Tabla 12: Flujo de Secuencia.**

| Descripción | Representan el control de flujo y la secuencia de las actividades. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Id | Identifica el objeto entre los otros objetos. | Object | no | si | | |
| Name | Nombre para identificar el objeto. | String | si | no | | |
| Source | Especifica la salida del flujo de secuencia | Object | si | si | | |
| Target | Especifica el objetivo del flujo de secuencia | Object | si | si | | |

**Tabla 13: Paquete de objetos validados.**

| Descripción | Representan el control de flujo y la secuencia de las actividades. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |
| Asociación | Transición para asociar información adicional sobre el proceso. | Object | si | si | | |
| Contenedor | Representa un proceso. | Object | no | si | | |
| BPM | Conjunto de actividades asociadas entre sí para cumplir un objetivo. | Object | no | no | | |
| Validador de reglas BPMN | Componente para validar los procesos de negocios modelados con BPMN. | Object | no | si | | |

**Tabla 14: XPDL.**

| Descripción | Representa el trabajo realizado dentro de una organización. | | | | | |
|---|---|---|---|---|---|---|
| Atributos | | | | | | |
| Nombre | Descripción | Tipo | ¿Puede ser nulo? | ¿Es único? | Restricciones | |
| | | | | | Clases válidas | Clases no válidas |

## 1.2 Requisitos de software.

Los requisitos son un punto clave en el desarrollo de las aplicaciones informáticas, permiten una comunicación efectiva entre los usuarios y el equipo de desarrollo, con el objetivo de llegar a un entendimiento de lo que hay que realizar, y describen el comportamiento esperado en el software una vez desarrollado. Gran parte del éxito de un proyecto de software radicará en la identificación de las necesidades del negocio.

Los requisitos de software se clasifican en funcionales y no funcionales. Los requisitos funcionales describen qué es lo que el sistema debe hacer para dar soporte a las funciones y objetivos del usuario. Los requisitos no funcionales imponen restricciones de cómo los requisitos funcionales deben ser implementados.[24]

## 1.3 Técnicas de captura y validación de requisitos.

La captura de requisitos es la actividad mediante la cual el equipo de desarrollo de un sistema de software extrae, de cualquier fuente de información disponible, las necesidades que debe cubrir dicho sistema. El proceso de captura de requisitos puede resultar complejo, principalmente si el entorno de trabajo es desconocido para el equipo de analistas, y depende mucho de las personas que participen en él. Por la complejidad que todo esto puede implicar, la ingeniería de requisitos ha trabajado desde hace años en desarrollar técnicas que permitan hacer este proceso de una forma más eficiente y precisa.[25]

Los requisitos una vez definidos necesitan ser validados. La validación de requisitos tiene como misión demostrar que la definición de los requisitos precisa realmente el sistema que el usuario necesita o el cliente desea. Es necesario asegurar que el análisis realizado y los resultados obtenidos de la etapa de definición de requisitos son correctos. Pocas son las propuestas existentes que ofrecen técnicas para la realización de la validación y muchas de ellas consisten en revisar los modelos obtenidos en la definición de requisitos con el usuario para detectar errores o inconsistencias.[25]

## 1.4 Requisitos funcionales.

Los requisitos funcionales son los que definen las funciones que el sistema será capaz de realizar. Los mismos se centran en qué y no cómo se deben hacer esas funciones.[26] Utilizando las técnica tormenta de ideas, se definen los siguientes requisitos funcionales.

**RF.1** Persistir procesos de negocio.
**RF.2** Trasformar objetos php a etiquetas XPDL.
**RF.3** Transformar etiquetas XPDL a objetos php.
**RF.4** Representar en el formato XPDL un servicio según las especificaciones del marco de trabajo Sauxe.

**Tabla 15: Especificación del requisito: Persistir procesos de negocio.**

| | |
|---|---|
| **Precondiciones** | Deben existir por cada objeto php, una clase que represente la etiqueta equivalente en XPDL. |
| **Flujo de eventos** | |
| **Flujo básico Persistir procesos de negocio.** | |
| 1. | Crear archivo XPDL vacío. |
| 2. | Escribir las etiquetas en el archivo tipo XPDL. |
| 3. | Salvar el archivo XPDL. |
| **Pos-condiciones** | |

| | | |
|---|---|---|
| 1. | Se ha salvado el proceso de negocio. | |
| **Flujos alternativos N/A** | | |
| **Pos-condiciones** | | |
| 1 | N/A | |
| **Validaciones** | | |
| 1 | Se validan los datos según lo establecido en el Modelo conceptual Modelo conceptual Componente para la persistencia de procesos de negocios del componente de flujos de trabajo del marco de trabajo Sauxe. | |
| **Conceptos** | **Persistir** | Usados internamente:<br>myPackage<br>doc |
| **Requisitos especiales** | N/A | |
| **Asuntos pendientes** | N/A | |

**Tabla 16: Especificación del requisito: Trasformar objetos php a etiquetas XPDL.**

| | | |
|---|---|---|
| **Precondiciones** | Debe existir el paquete de objetos | |
| **Flujo de eventos** | | |
| **Flujo básico Persistir procesos de negocio.** | | |
| 1. | Obtener el paquete de objetos php previamente validado. | |
| 2. | Desensamblar el paquete de objetos php. | |
| 3. | Crear por cada objeto php, una clase que represente la etiqueta equivalente en XPDL. | |
| **Pos-condiciones** | | |
| 1. | Se han transformado las etiquetas XPDL a objetos php. | |
| 2. | | |
| **Flujos alternativos** | | |
| **Flujo alternativo 1.a El archivo no existe en la dirección definida.** | | |
| **Pos-condiciones** | | |
| 1 | N/A | |
| **Validaciones** | | |
| 1 | Se validan los datos según lo establecido en el Modelo conceptual Modelo conceptual Componente para la persistencia de procesos de negocios del componente de flujos de trabajo del marco de trabajo Sauxe. | |
| **Conceptos** | **Cargar** | Usados internamente:<br>myPackage<br>doc |
| **Requisitos especiales** | N/A | |
| **Asuntos pendientes** | N/A | |

**Tabla 17: Especificación del requisito: Transformar etiquetas XPDL a objetos php.**

| | |
|---|---|
| **Precondiciones** | Debe existir un archivo XPDL |
| **Flujo de eventos** | |

| | | |
|---|---|---|
| **Flujo básico Persistir procesos de negocio.** | | |
| 1. | Cargar archivo XPDL. | |
| 2. | Desensamblar las etiquetas. | |
| 3. | Crear por cada etiqueta XPDL un objeto php. | |
| **Pos-condiciones** | | |
| 1. | Se han transformado las etiquetas XPDL a objetos php. | |
| 2. | | |
| **Flujos alternativos** | | |
| **Flujo alternativo 1.a** | | |
| **Pos-condiciones** | | |
| 1 | N/A | |
| **Validaciones** | | |
| 1 | Se validan los datos según lo establecido en el Modelo conceptual Modelo conceptual Componente para la persistencia de procesos de negocios del componente de flujos de trabajo del marco de trabajo Sauxe. | |
| **Conceptos** | **Cargar** | Usados internamente:<br>myXPDL<br>doc |
| **Requisitos especiales** | N/A | |
| **Asuntos pendientes** | N/A | |

**Tabla 18: Especificación del requisito: Representar en el formato XPDL un servicio según las especificaciones del marco de trabajo Sauxe.**

| | |
|---|---|
| **Precondiciones** | Debe existir el objeto Variables.php |
| **Flujo de eventos** | |
| **Flujo básico Persistir procesos de negocio.** | |
| 1. | Cargar Objetos php. |
| 2. | Desensamblar los objetos php |
| 3. | Obtengo el objeto Variables. |
| 4. | Crear la etiqueta Variables especificándole el módulo y el servicio al que se accederá. |
| 5. | Representar el objeto Variables.php en el formato XPDL |
| **Pos-condiciones** | |
| 1. | Se ha realizado la especificación del servicio de forma correcta. |
| 2. | |
| **Flujos alternativos** | |
| **Flujo alternativo 3.a** No existe el objeto Variables.php | |
| 1. Crear la etiqueta Variables vacía. | |
| **Pos-condiciones** | |
| 1 | N/A |
| **Validaciones** | |
| 1 | Se validan los datos según lo establecido en el Modelo conceptual Modelo conceptual Componente para la persistencia de procesos de negocios del componente de flujos de trabajo del marco de trabajo Sauxe. |

| | | |
|---|---|---|
| **Conceptos** | **Cargar** | Usados internamente:<br>myPackage<br>doc |
| **Requisitos especiales** | N/A | |
| **Asuntos pendientes** | N/A | |

## 2.4 Requisitos no funcionales.

Como se ha mencionado con anterioridad, el presente trabajo forma parte de un proceso productivo iniciado por el departamento Tecnología del centro CEIGE y los resultados que se obtengan formarán parte del marco de trabajo desarrollado en el mismo. De esta manera los requisitos no funcionales a los que se debe acoger la aplicación a desarrollar son los que fueron establecidos por el centro al inicio del proceso de desarrollo, a continuación son descritos algunos de los más importantes.

**Rendimiento.**
Los tiempos de respuesta y velocidad de procesamiento de la información serán rápidos, no mayores de 5 segundos para las actualizaciones y 20 para las recuperaciones.

**Seguridad.**
Autenticación, Autorización (Contraseña de acceso) y Auditoría.

**Software.**

**Para el cliente:**
- Navegador Mozilla Firefox 3.0 o superior.
- Sistema operativo Windows 98 o superior o Linux en cualquiera de sus distribuciones.

**Para el servidor:**
- Sistema operativo Linux en cualquiera de sus distribuciones.
- Un servidor Apache 2.0 o superior con módulo PHP 5.0 disponible.
- Un servidor de base de datos PostgreSQL 8.3 o superior.

**Hardware.**

**Para el servidor:**
- Requerimientos mínimos: Procesador Pentium IV a 2GHz de velocidad de procesamiento y 1Gb de memoria RAM.
- Al menos 40Gb de espacio libre en disco duro.

- Tarjeta de red.

**Para el cliente:**

- Requerimientos mínimos: Procesador Pentium III a 1GHz con 256Mb de memoria RAM.
- Tarjeta de red.

## 1.5 Modelo de diseño.

El Modelo de diseño es una abstracción del Modelo de Implementación y su código fuente, el cual fundamentalmente se emplea para representar y documentar su diseño.

Dentro de este epígrafe se verán los elementos que se tuvieron en cuenta durante el diseño de la solución, dando paso así a la exposición de los resultados de este flujo de trabajo, que refleja cómo será implementado el sistema en términos de clases del diseño.

## 1.6 Diagrama de clases del diseño.

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación, contiene información como clases, asociaciones, atributos, métodos y dependencias.

A continuación se muestra el diagrama de clases del diseño que se realizó durante el proceso de desarrollo:

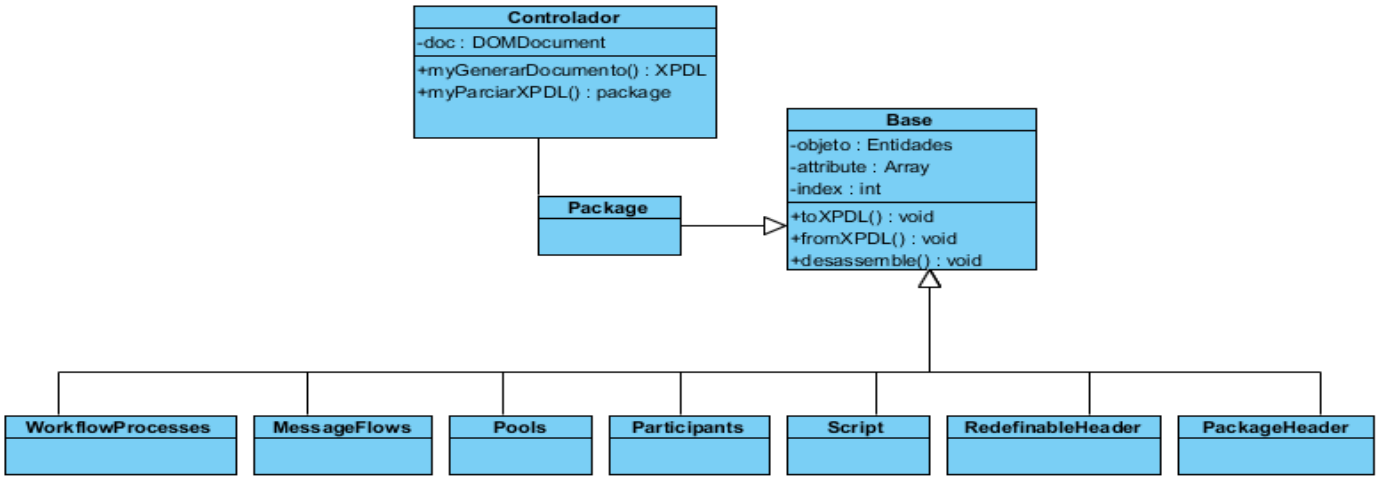



**Figura 2: Diagrama de clases del diseño general.**

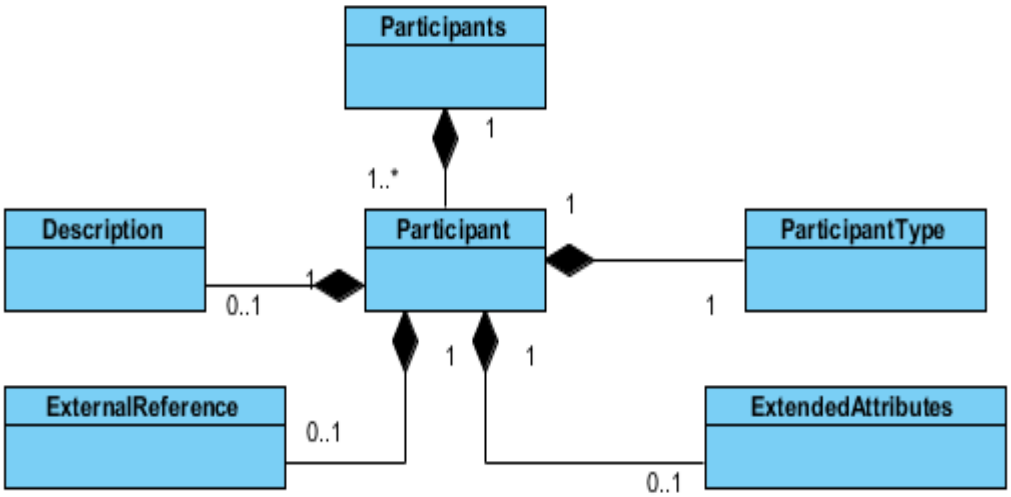



**Figura 3: Sub-diagrama de clases del diseño "Participants".**

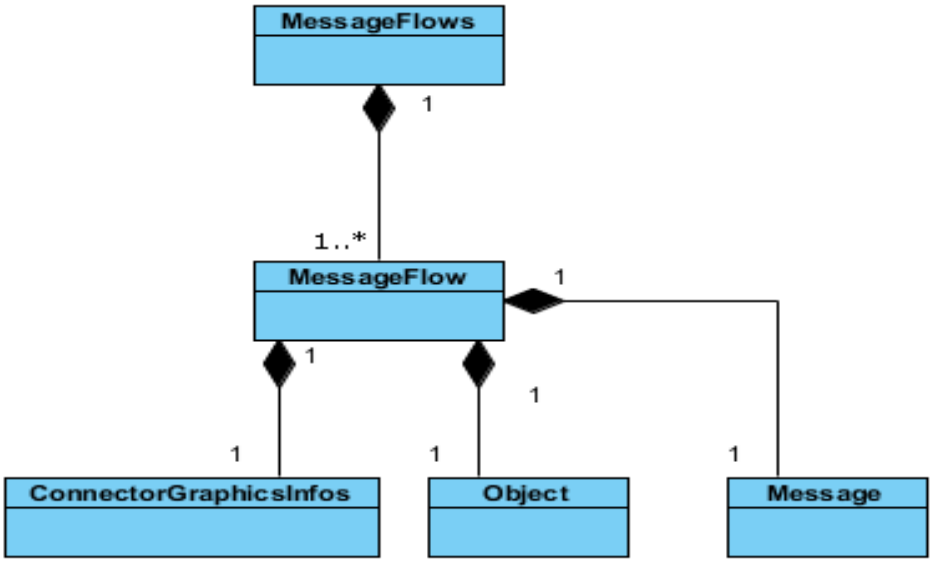



**Figura 4: Sub-diagrama de clases del diseño "MessageFlows".**

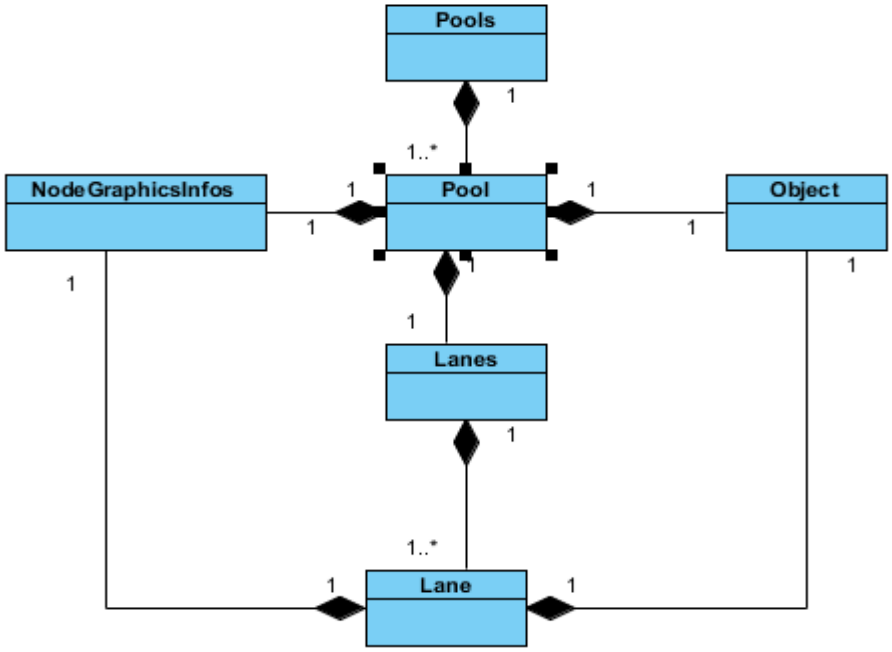



**Figura 5: Sub-diagrama de clases del diseño "Pools".**

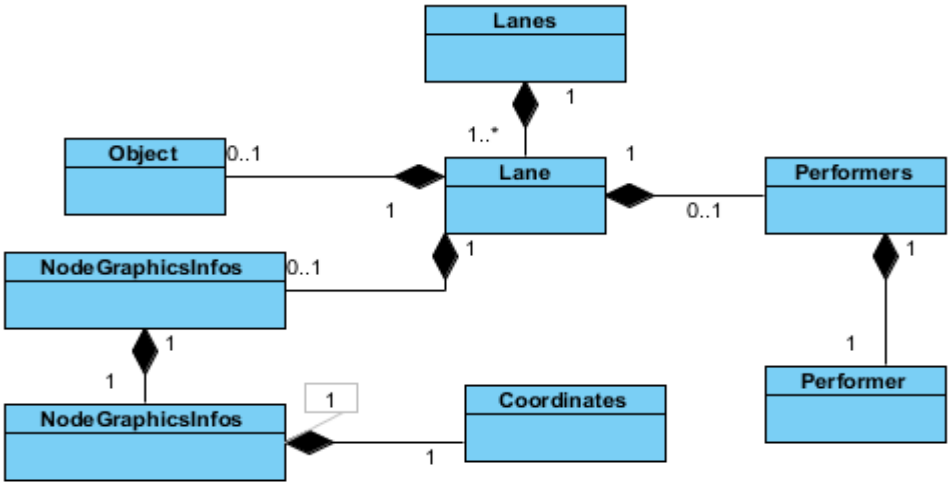



**Figura 6: Sub-diagrama de clases del diseño "Lanes".**

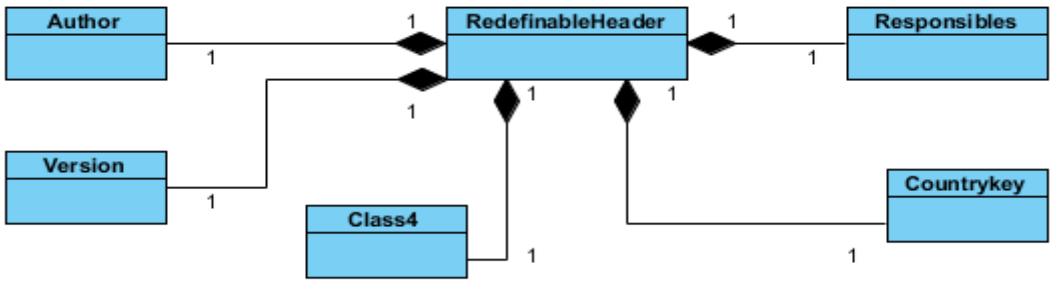



**Figura 7: Sub-diagrama de clases del diseño "RedefinableHeader".**

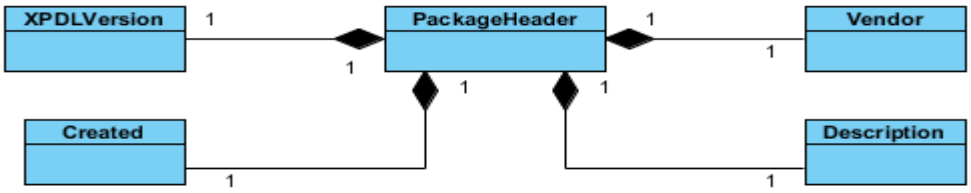



**Figura 8: Sub-diagrama de clases del diseño "PackageHeader".**

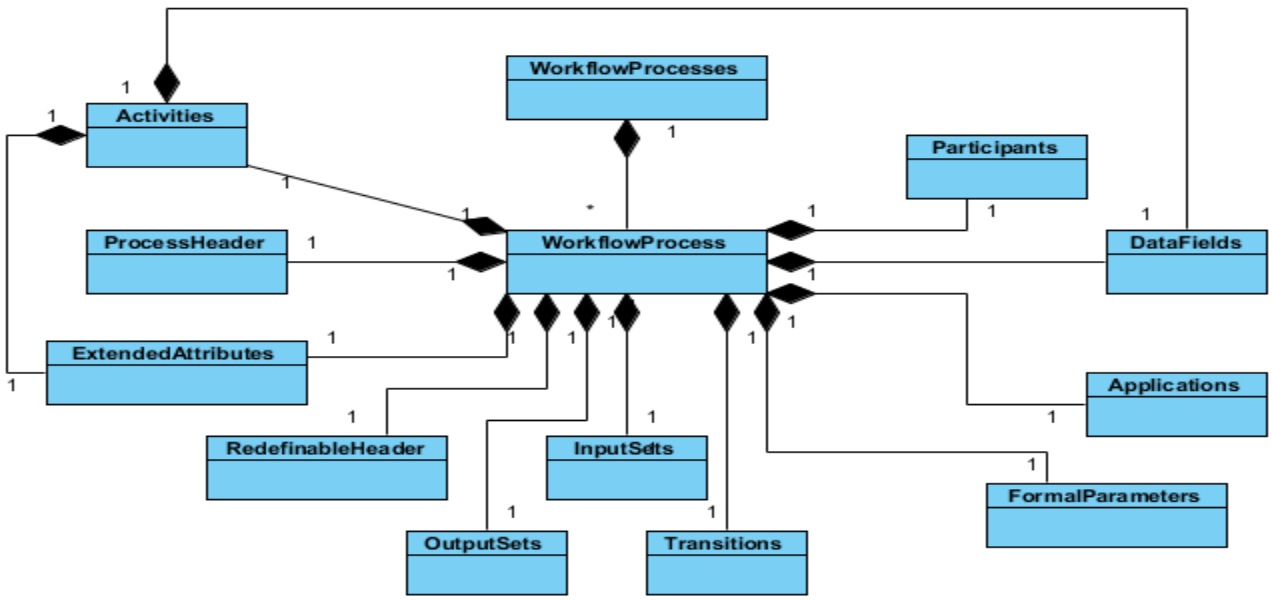



**Figura 9: Sub-diagrama de clases del diseño "WorkflowProcesses".**

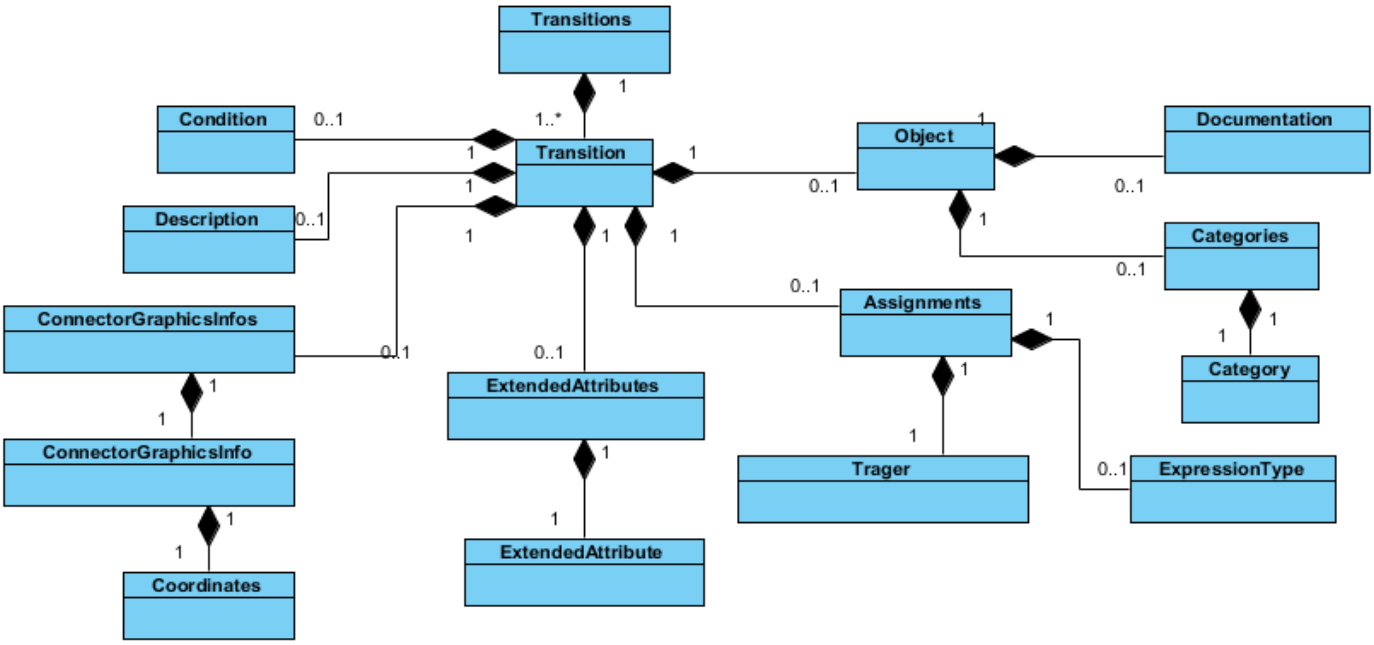



**Figura 10: Sub-diagrama de clases del diseño "Transitions".**

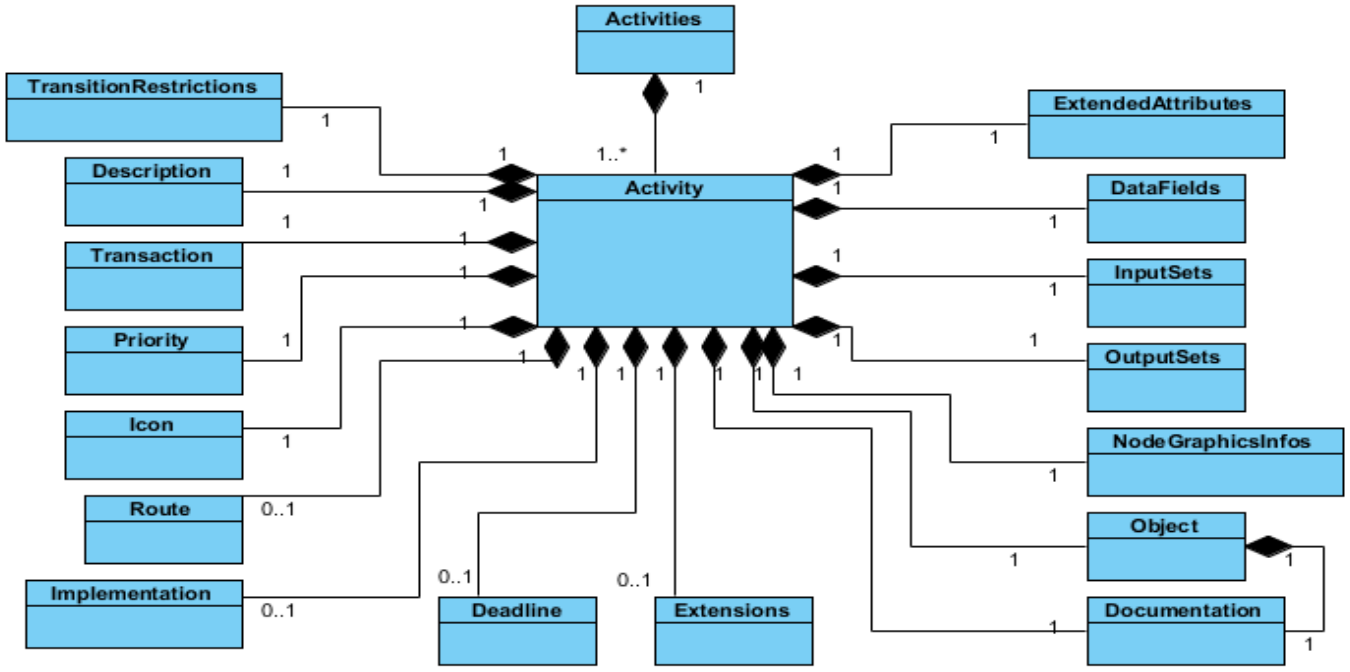



**Figura 11: Sub-diagrama de clases del diseño "Activities".**

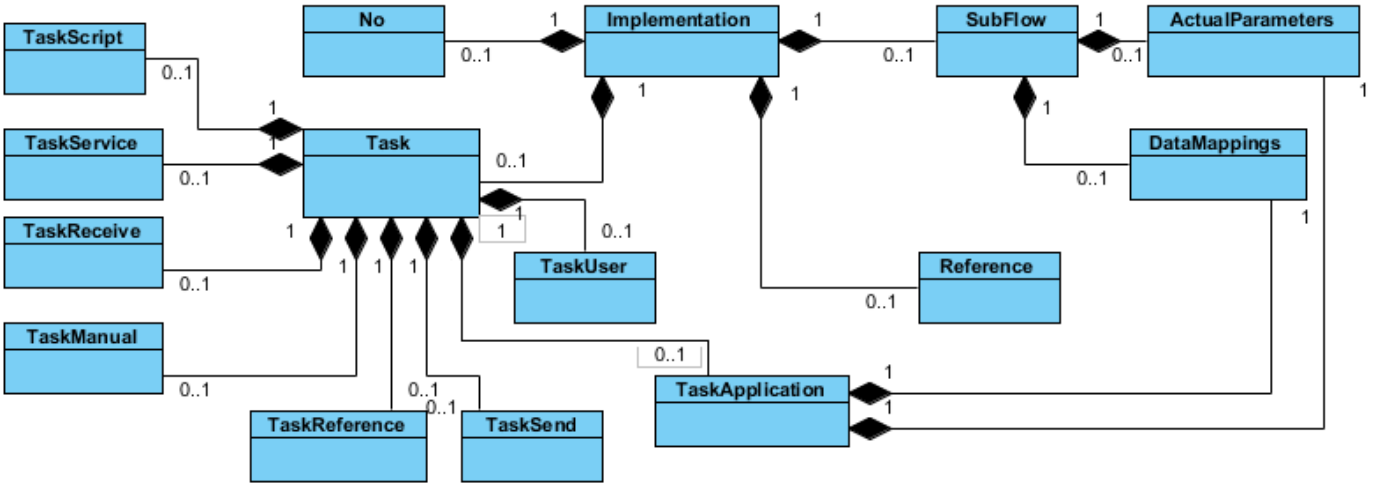



**Figura 12: Sub-diagrama de clases del diseño "Implementation".**

En este epígrafe se mostró el diagrama de clases del diseño el cual tuvo que ser separado, para un mejor entendimiento, en las clases más importantes que conforman el paquete.

## 1.7 Patrones de diseño.

El uso de patrones da la posibilidad de organizar las clases en estructuras comunes y bien probadas; modificando el sistema para mejorar su flexibilidad y extensibilidad. Emplear patrones de diseño ayuda a tener un lenguaje común con otros programadores.

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software.

En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Se debe tener presente los siguientes elementos de un patrón: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios).

Se clasifican como se muestra a continuación:

- Patrones Creacionales: Inicialización y configuración de objetos.

- Patrones Estructurales: Separan la interfaz de la implementación. Se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes.
- Patrones de Comportamiento: Más que describir objetos o clases, describen la comunicación entre ellos. [27]

Para que los componentes desarrollados cuenten con estas características se utilizaron una serie de patrones que se explican a continuación.

## 1.8 Patrones utilizados.

**Creador.**

El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Este patrón brinda un soporte a un bajo acoplamiento –patrón que será descrito más adelante lo que supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización. En el diseño está contenido en la clase Package, esta contiene a todas las etiquetas y es responsable de crearlas.

**Controlador.**

El patrón controlador es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto para aumentar la reutilización de código y a la vez tener un mayor control. Un Controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define además el método de su operación. En la solución se comprende en la clase Controladora, la cual tiene estas características.[28]

**Patrón Experto.**

Experto es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Con él no se pretende designar una idea oscura ni extraña; expresa simplemente la "intuición" de que los objetos hacen cosas relacionadas con la información que poseen.

El patrón fue usado con el objetivo de darle a las clases las responsabilidades necesarias siempre que contaran con la información para cumplirlas, por tal motivo cada clase tiene el mismo nombre de la etiqueta XPDL que tiene que crear, logrando así un mejor comportamiento entre las estas y haciendo que fueran más cohesivas, fáciles de comprender y mantener, permitiendo mayores facilidades de soporte.

**Bajo Acoplamiento.**

Este patrón es un principio que asigna la responsabilidad de controlar el flujo de eventos del sistema a clases específicas. Esto facilita la centralización de actividades. El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión. Un error muy común es asignarle demasiada responsabilidad y alto nivel de acoplamiento con el resto de los componentes del sistema.

Consiste en tener las clases lo menos ligadas entre sí, permitiendo que al producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

Mantener el bajo acoplamiento entre clases más que un patrón es una buena práctica del diseño dado que al realizar cambios no se afectan otros componentes, es más fácil de entender de manera aislada y de esta forma se pueden reutilizar mejor las clases. Este patrón se aplica en todas las clases definidas en el componente desarrollado.

**Alta Cohesión.**

Este patrón dice que la información que almacena una clase debe ser coherente y estar en la mayor medida de lo posible relacionada con la clase.

Realizando un diseño donde las clases mantengan una alta cohesión se mejora la claridad y facilidad con que se entiende el diseño, se simplifica el mantenimiento y las mejoras de funcionalidad, a menudo se genera un bajo acoplamiento, soportando mayor capacidad de reutilización.

Los patrones Bajo Acoplamiento y Alta Cohesión generalmente trabajan muy unidos. Un buen diseño requiere la correcta utilización de ambos patrones. En cada una de las clases del diseño de modelado en la solución se manifiestan ambos patrones, cada una de ellas implementando las responsabilidades específicas que le fueron asignadas, y delegando las demás.

## 1.9 La persistencia con el XPDL Generado.

Para lograr un buen diseño, se toma la decisión de que las clases creadas tendrán el mismo nombre que las etiquetas que construye. Dando así un mejor entendimiento de la solución, dado el gran número de clases existentes.

A continuación se muestra un ejemplo de una clase encargada de realizar una etiqueta XPDL.

```
function desassembleClass() {
    $bool=TRUE;

    $this->object->getEventType()->select(2);
    $objTypeEvent = $this->object->getEventType()->getSelectedItem();
    //print_r($objTypeEvent);die;
    if ($objTypeEvent instanceof ZendExt_WF_WFObject_Entidades_StartEvent) {
        $myStartEvent = new ZendExt_WF_WFTranslatorXPDL_Clases_StartEvent($objTypeEvent,$bool);
        $this->addAttribute($myStartEvent);
    }
    if ($objTypeEvent instanceof ZendExt_WF_WFObject_Entidades_IntermediateEvent) {
        //print_r('sdgdfg');die;
        $myIntermediateEvent = new ZendExt_WF_WFTranslatorXPDL_Clases_IntermediateEvent($objTypeEvent,$bool);
        $this->addAttribute($myIntermediateEvent);
    }
    if ($objTypeEvent instanceof ZendExt_WF_WFObject_Entidades_EndEvent) {
        //print_r($objTypeEvent);die;
        $myEndEvent = new ZendExt_WF_WFTranslatorXPDL_Clases_EndEvent($objTypeEvent,$bool);
        $this->addAttribute($myEndEvent);
    }
}

public function toXPDL($doc, &$objectTag) {
    $thisObjectTag = $doc->createElement("Event");

    foreach ($this->attributes as $value) {
        $value->toXPDL($doc, $thisObjectTag);
    }

    $objectTag->appendChild($thisObjectTag);
}
```

**Figura 13: Creando la etiqueta “Event” con su contenido en la clase Event.**

Seguidamente se muestra la etiqueta construida en el archivo XPDL.

```
<Event>
  <EndEvent>
    <TriggerResultMessage CatchThrow="">
      <Message Id="oxb6e2mbx754dCh" Name="Name" From="" To="" FaultName="EndEvent"/>
      <WebServiceOperation OperationName="">
        <Partner PartnerLinkId="PartnerLinkId" RoleType="RoleType"/>
        <Service ServiceName="ServiceName" PortName="PortName">
          <EndPoint EndPointType="ServiceName">
            <ExternalReference xref="" location="" namespace=""/>
          </EndPoint>
        </Service>
      </WebServiceOperation>
    </TriggerResultMessage>
  </EndEvent>
</Event>
```

**Figura 14: Etiqueta Event.**

De igual forma, en el proceso contrario, que sería construir un objeto a partir de esta etiqueta contenida en el archivo XPDL, se tiene funcionalidades que se encargan de esta tarea en la misma clase.

```
public function fromXPDL($objectTag) {
    $attribs = $objectTag->attributes;

    if ($attribs->length > 0) {
        for ($i = 0; $i < $attribs->length; $i++) {
            $node = $attribs->item($i);
            $prefFuncName = 'set';
            $fullFuncName = $prefFuncName . $node->nodeName;
            $this->object->$fullFuncName($node->nodeValue);
        }
    }

    for ($i = 0; $i < $objectTag->childNodes->length; $i++) {
        $Nonde = $objectTag->childNodes->item($i);
        $NodeName = $Nonde->nodeName;
        $Act_Type = $this->object->getEventType();
        $Act_Type->selectItem($NodeName);
        $cObj = $Act_Type->getSelectedItem();
        $prefClassName = 'ZendExt_WF_WFTranslatorXPDL_Clases_';
        $fullFuncName = $Nonde->nodeName;
        $ClassName = $prefClassName . $NodeName;
        $newTag = new $ClassName($cObj);
        $newTag->fromXPDL($Nonde);
    }
}
```

**Figura 15: Creando el objeto “Event” con su contenido en la clase Event.**

## 1.10 Conclusiones parciales.

En este capítulo fueron expuestos los artefactos generados durante el diseño de la solución propuesta, para así tener un mejor entendimiento del problema a resolver. En el mismo se expuso el modelo conceptual, los requisitos y el diagrama de clases del diseño propuesto por los autores. Una vez concluido el diseño de la solución puede darse paso a los flujos de implementación y prueba.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

## 2.1 Introducción.

En el capítulo se muestra el modelo de implementación que pone en práctica el diseño de la solución realizado en el capítulo anterior y se especifica el conjunto de validaciones y pruebas que evalúan la calidad del sistema.

## 2.2 Implementación.

El modelo de implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes. Describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados y cómo dependen los componentes unos de otros.

## 2.3 Diagrama de componentes.

La solución propuesta consta de un solo componente llamado WFTranslatorXPDL el cual interactúa con el componente WFValidator por medio de una interfaz, para la creación de los archivos XPDL y el paquete de clases de un modelo persistido en un archivo XPDL. A continuación se muestra el diagrama de componentes elaborado.

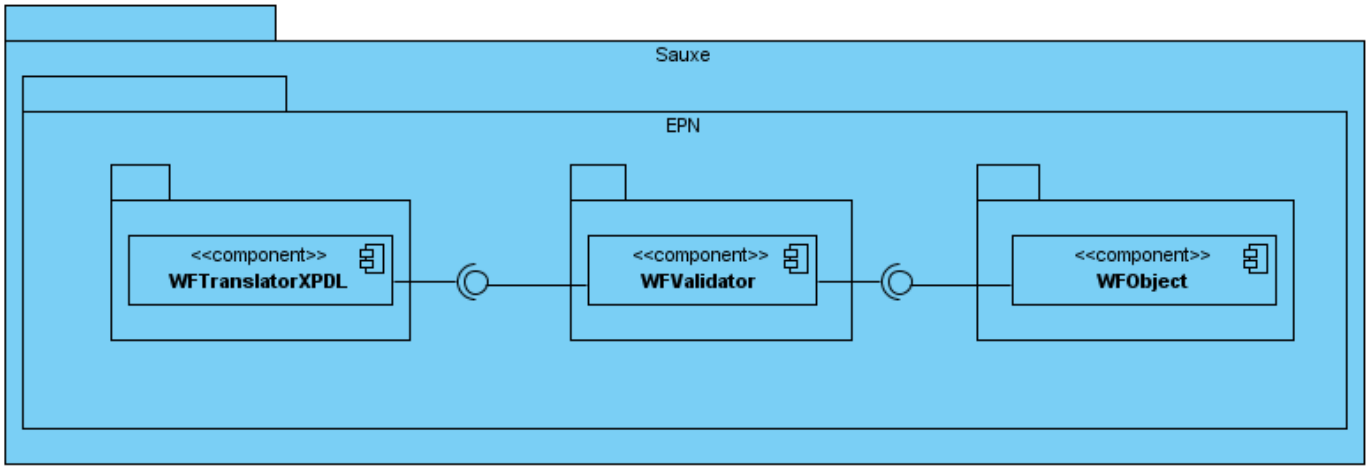



**Figura 16: Diagrama de Componentes.**

## 2.4 Estándares de codificación.

A continuación se muestra el estilo de código utilizado en la implementación, el cual sigue los estándares definidos para el marco de trabajo Sauxe.

El nombre de las clases, se escribe la primera letra con mayúsculas y las demás con minúsculas. En caso de ser un nombre compuesto el segundo tiene la misma estructura que el primer nombre. Ejemplo: Package, PackageHeader.

**Nomenclatura de las clases según el tipo.**

- Las Controladoras se encuentran dentro de la carpeta “app”: Las clases controladoras después de escribir el nombre según su nomenclatura se le añade “Controller”. Ejemplo: AplicationController.

**Nomenclatura de los métodos o funciones.**

El nombre de los métodos de una clase comienzan con minúsculas, en caso de que sea compuesto seguido del primer nombre comienza el segundo con la primera letra en mayúscula. Deben describir el propósito del mismo. Ejemplos: fromXPDL (), toXPDL ().Si es una función de una clase controladora después del nombre se le agrega la palabra “Action”. Ejemplo: salvarAction ().

**Nomenclatura de las variables.**

El nombre a emplear para los atributos se escribe con la primera palabra en minúscula, en caso de que sea un nombre compuesto seguido de la primara palabra se escribe la segunda con inicial mayúscula. Además en caso de ser un objeto se comienza con:”obj” y después se escribe el nombre. Ejemplo: intAtributo, objectTag.

## 2.5 Métricas de software.

Las métricas de software son una medida cuantitativa que permite a los desarrolladores tener una visión profunda de la eficacia del proceso del software y de los proyectos que dirigen utilizando el proceso como un marco de trabajo. Se reúnen los datos básicos de calidad y productividad. Estos datos son entonces analizados, comparados con promedios anteriores, y evaluados para determinar las mejoras en la calidad y productividad. Las métricas son también utilizadas para señalar áreas con problemas de manera que se puedan desarrollar mejoras en el proceso del software.[26]

Las métricas escogidas para realizárseles al diseño son dos de las propuestas en el modelo de desarrollo del centre CEIGE y están contenidas en el libro de métricas realizado por Lorenz y Kidd, estas son:

## Tamaño Operacional de Clases (TOC).

El tamaño general de una clase se puede determinar empleando las medidas siguientes:

- El número total de operaciones (tanto operaciones heredadas como privadas de la instancia) que están encapsuladas dentro de la clase.
- El número de atributos (tanto atributos heredados como atributos privados de la Instancia) que están encapsulados en la clase.

Si existen valores grandes de TOC éstos mostrarán que una clase puede tener demasiada responsabilidad, lo cual reducirá la reutilización de la clase y complicará la implementación y la comprobación, por otra parte cuanto menor sea el valor medio para el tamaño, más probable es que las clases existentes dentro del sistema se puedan reutilizar ampliamente. [29]

**Tabla 19: Atributos de calidad que se evalúan en la métrica TOC.**

| Atributo de calidad | Modo en que lo afecta |
|---|---|
| **Responsabilidad** | Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase. |
| **Complejidad de implementación** | Un aumento del TOC implica un aumento de la complejidad de implementación de la clase. |
| **Reutilización** | Un aumento del TOC implica una disminución del grado de reutilización de la clase. |

**Tabla 20: Criterios de evaluación para la métrica TOC.**

| Atributo | Categoría | Criterio |
|---|---|---|
| **Responsabilidad** | Baja | <=Promedio |
| | Media | Entre Promedio y 2*Promedio |
| | Alta | >2*Promedio |
| **Complejidad implementación** | Baja | <=Promedio |
| | Media | Entre Promedio y 2*Promedio |
| | Alta | >2*Promedio |
| **Reutilización** | Baja | >2*Promedio |
| | Media | Entre Promedio y 2*Promedio |
| | Alta | <=Promedio |

## Relaciones entre Clases (RC).

Esta métrica está dada por el número de relaciones diferentes con que cuenta y evalúa los siguientes atributos de calidad:

**Tabla 21: Atributos de calidad que se evalúan en la métrica RC.**

| Atributo de calidad | Modo en que lo afecta |
|---|---|
| **Acoplamiento** | Un aumento del RC implica un aumento del Acoplamiento de la clase. |
| **Complejidad de mantenimiento** | Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase. |
| **Reutilización** | Un aumento del RC implica una disminución en el grado de reutilización de la clase. |
| **Cantidad de pruebas** | Un aumento del RC implica un aumento de la Cantidad de pruebas de unidad necesarias para probar una clase. |

**Tabla 22: Criterios de evaluación para la métrica RC.**

| Atributo | Categoría | Criterio |
|---|---|---|
| **Acoplamiento** | Ninguno | 0 |
| | Bajo | 1 |
| | Medio | 2 |
| | Alto | >2 |
| **Complejidad de mantenimiento** | Baja | <=Promedio |
| | Media | Entre Promedio y 2*Promedio |
| | Alta | >2*Promedio |
| **Reutilización** | Baja | >2*Promedio |
| | Media | Entre Promedio y 2*Promedio |
| | Alta | <=Promedio |
| **Cantidad de pruebas** | Baja | <=Promedio |
| | Media | Entre Promedio y 2*Promedio |
| | Alta | >2*Promedio |

Las métricas empleadas están diseñadas para evaluar los siguientes atributos de calidad:

- Responsabilidad. Consiste en la responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.

- Complejidad de implementación. Consiste en el grado de dificultad que tiene implementar un diseño de clases determinado.
- Reutilización. Consiste en el grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de software.
- Acoplamiento. Consiste en el grado de dependencia o interconexión de una clase o estructura de clase, con otras, está muy ligada a la característica de Reutilización.
- Complejidad del mantenimiento. Consiste en el grado de esfuerzo necesario a realizar para desarrollar un arreglo, una mejora o una rectificación de algún error de un diseño de software. Puede influir indirecta, pero fuertemente en los costes y la planificación del proyecto.
- Cantidad de pruebas. Consiste en el número o el grado de esfuerzo para realizar las pruebas de calidad (Unidad) del producto (componente, modulo, clase, conjunto de clases, etc.) diseñado.

A continuación se realizará el análisis de los resultados obtenidos al aplicar las métricas escojidas al diseño planteado anteriormente.

### 2.5.1 Tamaño Operacional de Clases (TOC).

Los valores grandes para esta métrica, indican que la clase debe tener bastante responsabilidad. Esto reducirá la reutilización de esta clase y complicará la implementación y las pruebas. Se pueden calcular los promedios para el número de atributos y operaciones de clase. Cuando menor sea el valor del promedio para el tamaño será más posible que las clases dentro del sistema puedan ser reutilizadas.

El tamaño general de una clase se puede determinar empleando medidas para saber el número total de operaciones (tanto operaciones heredadas como privadas de la instancia) que están encapsuladas dentro de la clase así como encontrando el número de atributos (tanto atributos heredados como atributos privados de la Instancia) que están encapsulados en la clase. Si existen valores grandes de TOC éstos mostrarán que una clase puede tener demasiada responsabilidad, lo cual reducirá la reutilizabilidad de la clase y complicará la implementación y la comprobación, por otra parte cuanto menor sea el valor medio para el tamaño, más probable es que las clases existentes dentro del sistema se puedan reutilizar ampliamente.

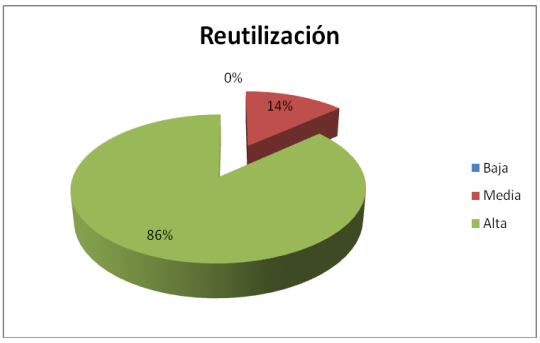
La evaluación técnica y su influencia en los parámetros de calidad Responsabilidad, Complejidad de Implementación y Reutilización son mostrados en el **Anexo 1: Tabla de la evaluación técnica (TOC).**.

Teniendo en cuenta las medidas o umbrales de referencia en lo que respecta al número de operaciones y/o atributos de las clases se establece que un tamaño de clase pequeño es aquel que tiene un valor menor o igual que 4, un tamaño medio para aquellas clases que tengan 5 operaciones y un tamaño de clase grande es aquel que es mayor o igual que seis (6). Así se concluye que el componente cuenta con 170 clases, para un promedio de cantidad de operaciones de 4.14. Los valores de tamaño quedan distribuidos de la siguiente manera:

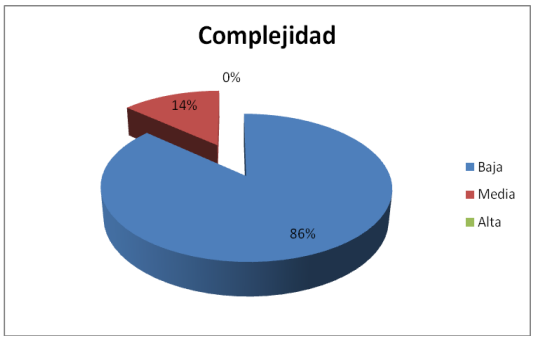
**Tabla 23: Umbrales definidos para el tamaño de clases.**

| Umbral | Tamaño | Cantidad de Clases |
|---|---|---|
| Pequeño | <=4 | 143 |
| Medio | =5 | 22 |
| Grande | >=6 | 1 |

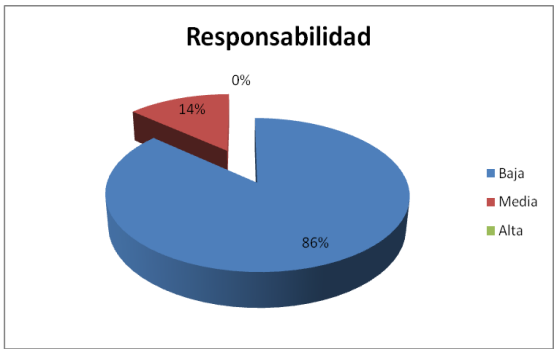
Como se puede observar en la tabla anterior el 84.1% de las clases están clasificadas en pequeñas, el 12.9% están en el umbral de clases de tamaño medio y solo el 0.5% en el umbral de clases grandes, lo cual brinda un resultado positivo según los parámetros de calidad Responsabilidad, Complejidad y Reutilización propuestos para esta métrica.




**Gráfica 1: Reutilización.**




**Gráfica 2: Complejidad.**




**Gráfica 3: Responsabilidad.**

| Intervalos | Cantidad de Clases |
|---|---|
| Entre 1 y 5 procedimientos | 169 |
| Entre 6 y 10 procedimientos | 1 |
| Entre 11 y 15 procedimientos | 0 |
| Entre 16 y 20 procedimientos | 0 |
| Entre 21 y 25 procedimientos | 0 |
| Entre 26 y 30 procedimientos | 0 |
| Entre 31 y 35 procedimientos | 0 |

**Tabla 24: Intervalos definidos para los procedimientos.**

### 2.5.2 Relaciones entre Clases (RC).

Los valores grandes de RC implican un aumento del Acoplamiento, en la complejidad del Mantenimiento de la clase y en la cantidad de Pruebas de unidad necesarias para probar una clase por lo que traería consigo una mayor complejidad. Un valor grande de RC implicaría una disminución en el grado de Reutilización de la clase.

La evaluación técnica y su influencia en los parámetros de calidad Acoplamiento, Complejidad de Mantenimiento, Cantidad de Pruebas y Reutilización son mostrados en el **Anexo 2: Tabla de la evaluación técnica (RC).**

Teniendo en cuenta el número de dependencias de las clases se establece que un valor de relaciones pequeño es aquel que tiene un valor menor o igual que uno 1, un valor medio para las que están compendiadas entre uno y menores o incluyendo 3 dependencias y un valor de relaciones grande es aquel que es mayor que 3. Así se concluye que el componente cuenta con 170 clases, para un promedio de cantidad de relaciones de 2.86.

**Tabla 25: Cantidad de Dependencias por Clases.**

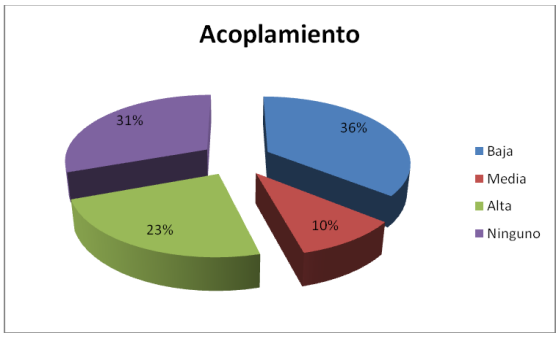
| Intervalos | Cantidad de Clases |
|---|---|
| 0 dependencias | 52 |
| 1 dependencias | 61 |
| > 1 dependencias | 57 |

Los valores de cantidad de relaciones quedan distribuidos de la siguiente manera:

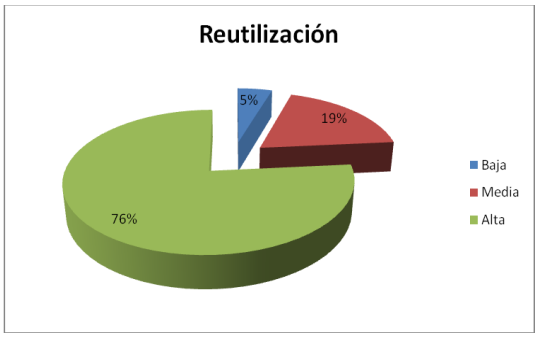
**Tabla 26: Umbral para la cantidad de relaciones de las clases.**

| Umbral | Relaciones | Cantidad de Clases |
|---|---|---|
| Pocas | <=1 | 113 |
| Medias | >1 y <= 3 | 29 |
| Muchas | >3 | 28 |

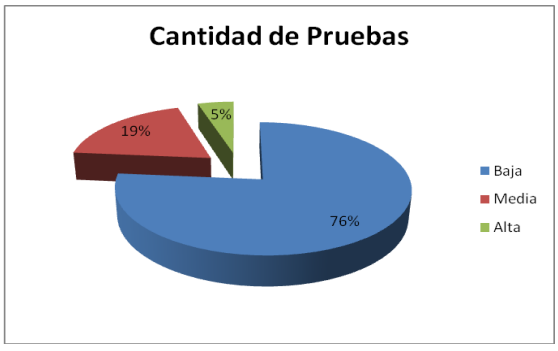
Como se puede observar en la tabla anterior el 66.4% de las clases están clasificadas con pocas relaciones, el 17.09% están en el umbral de clases con una media cantidad de relaciones y el 16.4% en el umbral de las clases con muchas relaciones, lo cual brinda un resultado positivo según los parámetros de calidad Complejidad del Mantenimiento, Cantidad de Pruebas y Reutilización, y el Acoplamiento.



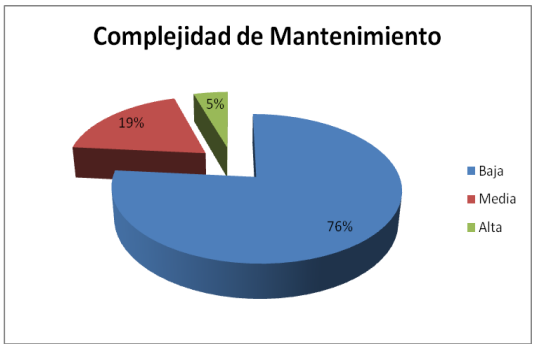

**Gráfica 4: Acoplamiento.**




**Gráfica 5: Reutilización.**




**Gráfica 6: Cantidad de Pruebas.**




**Gráfica 7: Complejidad de Mantenimiento.**

## 2.6 Matriz de cubrimiento de los parámetros de calidad evaluados.

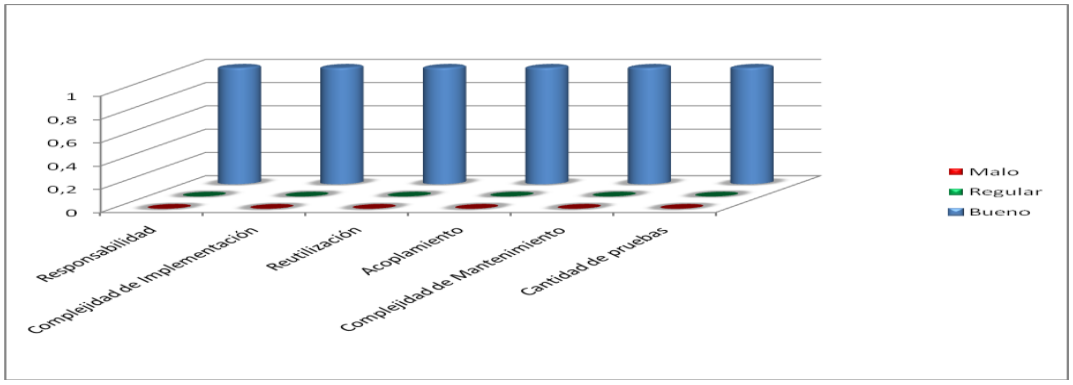
La matriz de cubrimiento o matriz de inferencia de indicadores de calidad, es el resumen de los resultados obtenidos al aplicar las métricas mencionadas en el epígrafe anterior. Esta matriz es una representación estructurada de los atributos de calidad para evaluar la calidad del diseño de la solución propuesta.[30]

Esta matriz tomará valores de 0 o 1 según los criterios evaluativos de Bueno. Malo o Regular para cada uno de los atributos de calidad.

Para tener una evaluación de Bueno, el criterio predominante debe ser de Alto en el atributo de calidad Reutilización, o que la suma de los criterio Medio y Alto sea mayor que el de Bajo o Ninguno y mayor que el 50 % del total, y Bajo o Ninguno en los atributos Responsabilidad, Complejidad de Implementación, Acoplamiento, Complejidad de Mantenimiento y Cantidad de Pruebas, o que la suma de estos dos sea mayor que el 50% del total de los criterios obtenidos en el atributo.

Para obtener una evaluación de Regular, deberá predominar el criterio de Medio, o que sea mayor que el 50% del total en el atributo Reutilización y en los atributos Responsabilidad, Complejidad de Implementación, Acoplamiento, Complejidad de Mantenimiento y Cantidad de Pruebas que Medio o Bajo sea más que el 50% del total en el atributo o la suma de Bajo y Medio, mayor que el 50%.

Para obtener la evaluación de Malo, tendrá que predominar el criterio Bajo o Ninguno o que la suma de estos dos sea mayor que el 50% del total de los criterios obtenidos en el atributo Reutilización, y en Responsabilidad, Complejidad de Implementación, Acoplamiento, Complejidad de Mantenimiento y Cantidad de Pruebas que Alto o Medio sea el predominante, o la suma de estos mayor que el 50%.




**Gráfica 8: Matriz de cubrimiento para las métricas realizadas al diseño.**

Se puede observar en esta gráfica que todos los atributos toman resultados positivos en la matriz de cubrimiento, quedando todos en el umbral de Bueno, lo que quiere decir que las clases se diseñaron con un bajo acoplamiento, una baja complejidad del mantenimiento y de la implementación, una baja responsabilidad y una alta reutilización, lo que permite la portabilidad del componente, el fácil entendimiento y que si en algún momento alguna de las clases se afectan el diseño no sufrirá de grandes cambios que afectaran su propósito.

## 2.7 Pruebas de software.

Las pruebas del software son un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación. Dichas pruebas son realizadas con el objetivo de detectar errores en el sistema, por lo que se llevan a cabo durante todo el ciclo de vida del producto. Los casos de prueba especifican una forma de probar el sistema, incluyendo las entradas con las que se ha de probar, las condiciones bajo las que ha de probarse, así como los resultados esperados.[31]

### 2.7.1 Pruebas estructurales o de caja blanca.

La prueba de caja blanca se basa en el diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivarlos. Mediante la prueba de la caja blanca el ingeniero del software puede obtener casos de prueba que:

- Garanticen que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo, programa o método.
- Ejerciten todas las decisiones lógicas en las vertientes verdadera y falsa.
- Ejecuten todos los bucles en sus límites operacionales.
- Ejerciten las estructuras internas de datos para asegurar su validez.

Es por ello que se considera a la prueba de Caja Blanca como uno de los tipos de pruebas más importantes que se le aplican a los software, logrando como resultado que disminuya en un gran porciento el número de errores existentes en los sistemas y por ende una mayor calidad y confiabilidad.[31]
A partir del diseño o del código fuente, se dibuja el grafo de flujo asociado.[31] En este caso se muestra uno de los métodos más importantes de la solución.

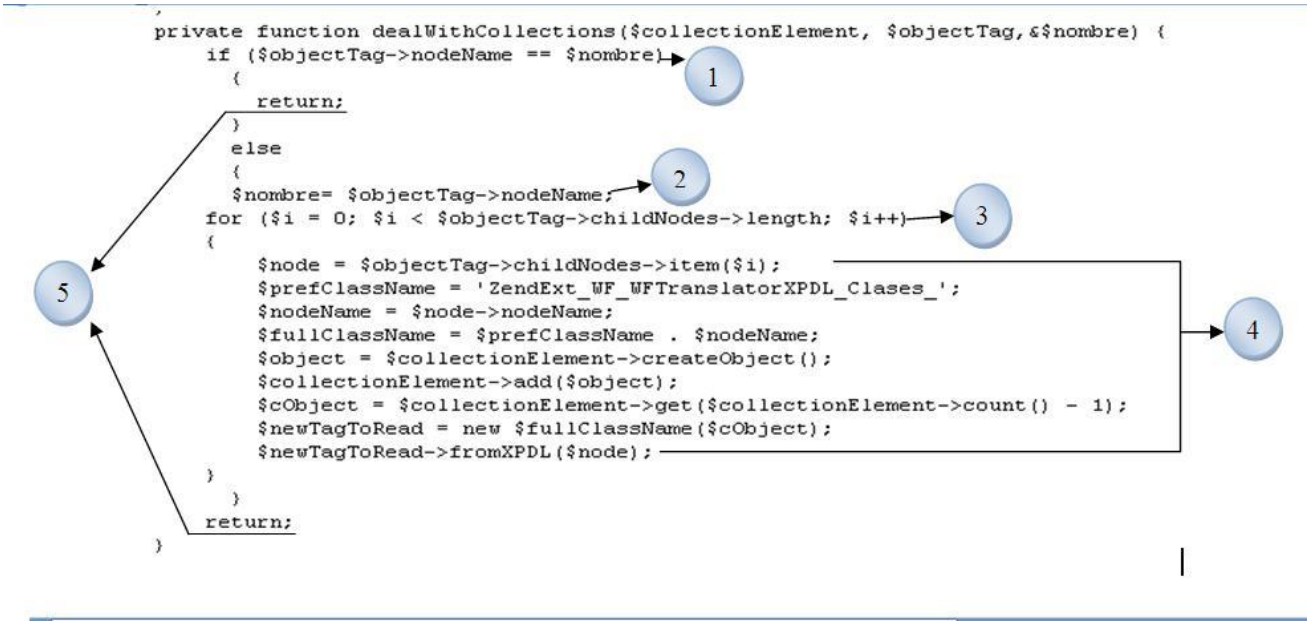



**Figura 17: Código fuente de la funcionalidad convertir elementos de tipo colección (dealWithCollections).**

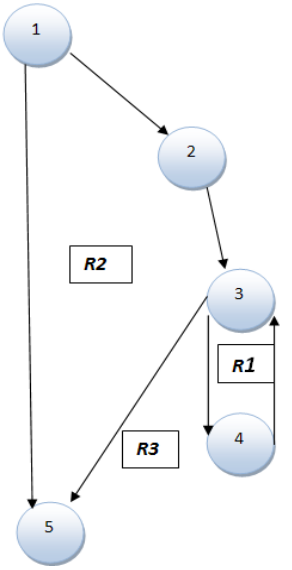



**Figura 18 Grafo de flujo asociado a la funcionalidad convertir elementos de tipo colección (dealWithCollections).**

El valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y nos da un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez. Un camino independiente es cualquier camino

del programa que introduce por lo menos un nuevo conjunto de sentencias de procesamiento o una nueva condición. Los valores del cálculo de la complejidad efectuado a los demás métodos de la solución se pueden ver en el **Anexo 3: Tabla de valores del cálculo de la complejidad.**

**Complejidad ciclomática de la funcionalidad elementos de tipo colección (dealWithCollections).**

Hay tres formas fundamentales de calcular la complejidad[32]:

**1. V (G) = (A – N) + 2**

V (G) = (6-5) + 2

V (G) =3

Donde A es el número de aristas del grafo y N es el número de nodos.

**2. V (G) = P + 1**

V (G) =2+1

V (G) =3

Donde P es el número de nodos predicado contenido en el grafo G.

**3. V (G) = R**

Donde “R” es la cantidad total de regiones, para cada formula “V (G)”.

V (G) =3

Con el cálculo se obtuvo valor 3, seguidamente se representan los caminos básicos por los que puede transitar el flujo:

Camino #1: (1-5).

Camino #2: (1-2-3-4-3--5).

Camino #3: (1-2-3-5).

**Derivación de casos de prueba de la funcionalidad elementos de tipo colección (dealWithCollections).**

Luego de tener elaborados los Grafos de Flujos y los caminos a recorrer, se preparan los casos de prueba que forzarán la ejecución de cada uno de esos caminos. Se escogen los datos de forma que las condiciones de los nodos predicados estén adecuadamente establecidas, con el fin de comprobar cada camino.

Casos de prueba para cada camino.

- Camino 1: 1-5.

Escoger dos veces el mismo nodo, para que:

`$objectTag->nodeName == $nombre`.

- Camino 2: 1-2-3-4-3-5.

Escoger un nodo que no haya sido tratado y que tenga al menos un hijo, es decir que:

`$i < $objectTag->childNodes->length` donde:

`$i = 0` y `$objectTag->childNodes->length>=1`.

- Camino 3: 1-2-3-5.

Escoger un nodo que no haya sido tratado y que no tenga hijos, es decir que:

`$i < $objectTag->childNodes->length` donde:

`$i = 0` y `$objectTag->childNodes->length=0`.

Luego de haberse aplicado las pruebas de caja banca, en especifico las pruebas del camino básico, seleccionando diferentes caminos a través del cálculo de la complejidad ciclomática, se ha arribado a la conclusión de que los resultados obtenidos fueron aceptables ya que se pudo comprobar que el flujo de trabajo de la función esta correcto ya que cumple con las condiciones necesarias que se habían planteado.

### 2.7.2 Pruebas de rendimiento.

En la Ingeniería del Software, las pruebas de rendimiento son aquellas que son realizadas para determinar qué tan rápido un sistema realiza una tarea bajo ciertas condiciones pre-planificadas de trabajo. Estas pruebas también son utilizadas para validar y verificar diferentes aspectos de la calidad de software, como por ejemplo, escalabilidad, fiabilidad y el buen uso de los recursos. Las pruebas de rendimiento constituyen un subconjunto de la Ingeniería de Pruebas, la cual se esfuerza en mejorar el rendimiento, basándose en el diseño y la arquitectura de un sistema, antes de la realización del proceso de codificación.[31]

Las pruebas de rendimiento pueden servir para diferentes propósitos. Pueden demostrar que el sistema cumple los criterios de rendimiento. Pueden comparar dos sistemas para encontrar cuál de ellos funciona

mejor o medir que partes del sistema o de cargas de trabajo provocan que el conjunto ofrezca bajo rendimiento. Para su diagnóstico, los ingenieros de software utilizan herramientas que permiten monitorear y medir qué partes de un dispositivo o software contribuyen a disminuir el rendimiento. [31]

**Pruebas de carga.**

Este es el tipo más sencillo de pruebas de rendimiento. Una prueba de carga se realiza generalmente para observar el comportamiento de una aplicación bajo una cantidad de peticiones esperada. Esta carga puede ser el número esperado de usuarios concurrentes utilizando la aplicación y que realizan un número específico de transacciones durante el tiempo que dura la carga. Esta prueba puede mostrar los tiempos de respuesta de todas las transacciones importantes de la aplicación. Si la base de datos, el servidor de aplicaciones, y otros componentes también se monitorizan, entonces esta prueba puede mostrar cual es el cuello de botella en la aplicación.[33]

**Pruebas de estrés.**

Esta prueba se utiliza normalmente para romper la aplicación. Se va doblando el número de usuarios que se agregan a la aplicación y se ejecuta una prueba de carga hasta que se rompe. Este tipo de prueba se realiza para determinar la solidez de la aplicación en los momentos de carga extrema y ayuda a los administradores para determinar si la aplicación rendirá lo suficiente en caso de que la carga real supere a la carga esperada.[33]

A la solución se le aplicaron las pruebas de rendimiento de carga y estrés, utilizando para ello la herramienta JMeter. Estas pruebas fueron realizadas por el departamento de calidad del centro CEIGE.

Resultados obtenidos después de aplicadas las pruebas de Estrés y Carga al componente WFTranslatorXPDL.

Para la opción Salvar.

**Tabla 27: Resultados obtenidos por la herramienta JMeter para la opción Salvar.**

| Label | # M... | Media | Mediana | Lín... | Mín | % Error | Máx | Kb/sec | Rendimiento |
|---|---|---|---|---|---|---|---|---|---|
| | 50 | 6299 | 6779 | 9970 | 1002 | 0,00% | 11663 | 9,5 | 3,7/sec |
| | 100 | 4476 | 3165 | 7170 | 1013 | 0,00% | 18191 | 1,0 | 46,6/min |
| | 50 | 12709 | 12399 | 14... | 11441 | 0,00% | 15314 | 3,1 | 1,7/sec |
| | 50 | 6924 | 6937 | 10... | 4898 | 0,00% | 10821 | 3,4 | 1,9/sec |
| gin/ | 50 | 204 | 198 | 370 | 5 | 0,00% | 449 | 9,3 | 2,4/sec |
| sponse/index.php | 50 | 56235 | 58672 | 78... | 22488 | 0,00% | 79935 | 4,3 | 34,5/min |
| | 50 | 17720 | 19763 | 24... | 553 | 0,00% | 31035 | ,0 | 36,9/min |
| rdominio | 50 | 5396 | 5489 | 6632 | 2396 | 0,00% | 9729 | 1,9 | 40,8/min |
| ralsistema | 50 | 6162 | 6293 | 8233 | 1320 | 0,00% | 10172 | 1,8 | 39,0/min |
| l | 50 | 6660 | 7000 | 8583 | 1446 | 0,00% | 8916 | 1,6 | 35,9/min |
| arperfil | 50 | 6928 | 7755 | 8568 | 1444 | 0,00% | 8755 | 1,6 | 33,5/min |
| aretiquetas | 100 | 7340 | 7720 | 8522 | 2398 | 0,00% | 10067 | 2,7 | 58,1/min |
| ardatostabpanel | 50 | 7272 | 7652 | 8233 | 4958 | 0,00% | 8982 | 1,5 | 29,5/min |
| ardesktop | 50 | 6417 | 6562 | 7950 | 4299 | 0,00% | 8359 | 1,4 | 29,9/min |
| s/images/desktop/shared/icons/fam/user.gif | 50 | 148 | 100 | 409 | 9 | 100,00% | 628 | ,2 | 32,1/min |
| php/aplicacion/aplicacion | 50 | 5879 | 5533 | 8429 | 3476 | 0,00% | 10135 | 1,5 | 31,1/min |
| s/aplicacion/config.json | 50 | 127 | 124 | 305 | 7 | 0,00% | 380 | 1,1 | 33,3/min |
| s/aplicacion/config.json | 50 | 127 | 124 | 305 | 7 | 0,00% | 380 | 1,1 | 33,3/min |
| s/components/toolbar/caxtor.ide.toolbar.js | 50 | 41 | 40 | 79 | 3 | 0,00% | 157 | 3,4 | 33,4/min |
| s/components/gui-designer/caxtor.ide.gui-designer.js | 50 | 38 | 37 | 77 | 4 | 0,00% | 106 | 6,6 | 33,4/min |
| s/components/canvas/caxtor.ide.canvas.js | 50 | 38 | 32 | 78 | 4 | 0,00% | 101 | 6,3 | 33,4/min |
| s/components/tool-box/caxtor.ide.tool-box.js | 50 | 34 | 32 | 58 | 8 | 0,00% | 116 | 2,4 | 33,5/min |
| s/components/tool-box/caxtor.ide.component-view.js | 50 | 35 | 31 | 64 | 2 | 0,00% | 148 | 1,1 | 33,5/min |
| s/components/object-inspector-tree/caxtor.ide.object-insp... | 50 | 40 | 37 | 80 | 5 | 0,00% | 175 | 6,2 | 33,5/min |
| php/aplicacion/salvar | 50 | 4874 | 4595 | 7682 | 1379 | 0,00% | 9725 | 1,6 | 3,1/sec |
| | 1350 | 6439 | 4931 | 12... | 2 | 3,70% | 79935 | 25,4 | 7,0/sec |

Para la opción Cargar.

**Tabla 28: Resultados obtenidos por la herramienta JMeter para la opción Cargar.**

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| /canvas/resources/images/bg.gif | 50 | 13 | 11 | 39 | 2 | 41 | 0,00% | 42,7/min | ,6 |
| xt/resources/images/default/panel/light-hd.gif | 50 | 10 | 10 | 21 | 2 | 39 | 0,00% | 42,7/min | ,6 |
| xt/resources/images/default/panel/white-top-b... | 50 | 10 | 10 | 23 | 2 | 35 | 0,00% | 42,7/min | ,6 |
| xt/resources/images/default/panel/tool-sprites.... | 50 | 12 | 10 | 23 | 2 | 131 | 0,00% | 42,7/min | 3,1 |
| xt/resources/images/default/toolbar/bg.gif | 50 | 26 | 11 | 26 | 2 | 681 | 0,00% | 42,7/min | ,6 |
| xt/resources/images/default/button/btn.gif | 50 | 10 | 11 | 21 | 2 | 40 | 0,00% | 42,8/min | 3,0 |
| /canvas/resources/images/send-back.png | 50 | 12 | 11 | 24 | 3 | 40 | 0,00% | 42,8/min | 14,7 |
| /canvas/resources/images/duplicate.png | 50 | 27 | 10 | 30 | 3 | 729 | 0,00% | 42,8/min | 14,9 |
| xt/resources/images/default/tree/folder.gif | 50 | 9 | 7 | 19 | 2 | 37 | 0,00% | 42,8/min | ,7 |
| /canvas/resources/images/delete.png | 50 | 14 | 12 | 29 | 3 | 93 | 0,00% | 42,8/min | 14,6 |
| /toolbar/resources/images/save.png | 50 | 16 | 9 | 27 | 2 | 174 | 0,00% | 42,8/min | ,4 |
| /toolbar/resources/images/new.png | 50 | 25 | 10 | 29 | 1 | 672 | 0,00% | 42,8/min | ,2 |
| /canvas/resources/images/send-front.png | 50 | 17 | 9 | 38 | 3 | 110 | 0,00% | 42,8/min | 15,0 |
| /toolbar/resources/images/upload.png | 50 | 25 | 9 | 46 | 1 | 669 | 0,00% | 42,8/min | ,5 |
| /toolbar/resources/images/export.png | 50 | 12 | 10 | 26 | 3 | 32 | 0,00% | 42,8/min | ,4 |
| /toolbar/resources/images/undo.png | 50 | 9 | 8 | 21 | 2 | 46 | 0,00% | 42,8/min | ,4 |
| /toolbar/resources/images/redo.png | 50 | 15 | 9 | 35 | 1 | 141 | 0,00% | 42,8/min | ,4 |
| /toolbar/resources/images/open.png | 50 | 11 | 10 | 22 | 2 | 32 | 0,00% | 42,8/min | ,4 |
| xt/resources/images/default/window/left-corner... | 50 | 16 | 8 | 64 | 2 | 87 | 0,00% | 42,8/min | ,1 |
| xt/resources/images/default/window/icon-ques... | 50 | 17 | 10 | 48 | 1 | 86 | 0,00% | 42,8/min | 1,1 |
| xt/resources/images/default/window/left-right.p... | 50 | 13 | 11 | 32 | 1 | 50 | 0,00% | 42,9/min | ,1 |
| xt/resources/images/default/window/right-corn... | 50 | 14 | 12 | 34 | 2 | 117 | 0,00% | 42,9/min | ,2 |
| xt/resources/images/default/shadow.png | 50 | 11 | 8 | 30 | 2 | 37 | 0,00% | 42,9/min | ,2 |
| xt/resources/images/default/progress/progres... | 50 | 17 | 10 | 62 | 2 | 93 | 0,00% | 42,9/min | ,6 |
| xt/resources/images/default/window/top-botto... | 50 | 13 | 12 | 23 | 2 | 48 | 0,00% | 42,9/min | ,1 |
| xt/resources/images/default/qtip/bg.gif | 50 | 15 | 10 | 52 | 2 | 57 | 0,00% | 43,0/min | ,8 |
| xt/resources/images/default/shadow-c.png | 50 | 15 | 13 | 39 | 1 | 49 | 0,00% | 43,0/min | ,1 |
| xt/resources/images/default/shadow-lr.png | 50 | 11 | 12 | 23 | 2 | 25 | 0,00% | 43,0/min | ,1 |
| editorft/index.php/aplicacion/salvar | 50 | 1801 | 1459 | 3145 | 108 | 7042 | 0,00% | 3,3/sec | 2,1 |
| xt/resources/images/default/window/icon-info.gif | 50 | 37 | 24 | 97 | 2 | 205 | 0,00% | 44,5/min | 1,1 |
| | 8800 | 769 | 13 | 2049 | 1 | 99600 | 2,27% | 55,7/sec | 2219,8 |

Como se puede observar para la opción Salvar el rendimiento arrojó un resultado de 3.1 segundos y para la opción Cargar de 3.3 segundos, los cuales son resultados positivos ya que son menores del requerido

por el Requisito no Funcional de Rendimiento, el cual plantea que los tiempos de respuesta y velocidad de procesamiento de la información serán rápidos, no mayores de 5 segundos para las actualizaciones.

## 2.8 Conclusiones parciales.

En el presente capítulo se realizó una validación del diseño propuesto mediante la aplicación de métricas para la evaluación de los atributos de calidad Reutilización, Responsabilidad, Complejidad de Implementación, Acoplamiento, Complejidad de Mantenimiento y Cantidad de Pruebas y dio resultados positivos para todos estos, lo cual permitió valorar el diseño propuesto de muy bueno dado los excelentes resultados obtenidos. Además se describen las pruebas estructurales y funcionales realizadas que permitieron comprobar el correcto funcionamiento del sistema.

# CONCLUSIONES

Con la culminación de la presente investigación se arriba a las siguientes conclusiones:

- Se estudiaron los principales conceptos asociados con el tema da la investigación para asentar las bases del marco teórico, también se realizó un estudio de herramientas reconocidas por la WfMC con el objetivo de encontrar posibles reutilizaciones y se concluyó que estas no se pueden integrar con el marco de trabajo Sauxe debido a las características peculiares de este.
- Se le aplicaron métricas contenidas entre las propuestas por Lorenz y Kidd y arrojaron resultados positivos en los atributos de calidad evaluados.
- Las pruebas de caja blanca y de rendimiento realizadas a la aplicación, validaron el correcto funcionamiento de la misma.
- La Herramienta permite al Editor de Proceso de Negocio salvar los procesos de negocio al estándar XPDL, para la persistencia de los mismos, y permite cargar un proceso estandarizado en un XPDL, logrando así la estandarización y posibilitando la gestión de estos procesos a conveniencia.

## RECOMENDACIONES

Una vez concluido el proceso de desarrollo se recomienda llevar a cabo la construcción de una segunda versión donde se incorporen las siguientes funcionalidades:

- Si en un futuro se consumen servicios web en el marco de trabajo Sauxe, esta herramienta necesita que se le realicen cambios, para lograr la persistencia adecuada de estos en el estándar XPDL, ya que se le ejecutaron cambios para lograr persistir los servicios de integración que son los que manipula hoy Sauxe.

Utilizar la documentación generada en futuras versiones del traductor.

BIBLIOGRAFIA


1. Soto, A.R.d. and E.C. Fernández, *Nuevas Tendencias en Sistemas de Información: Procesos y Servicios1*.
2. Jorge Cardoso, R.P.B.a.A.S., *Workflow Management Systems vs. ERP Systems: Differences, Commonalities, and Applications*
3. Rodríguez, U.A., *Implantacion de Systemas ERP en la PYMES.* 2010.
4. Pedro Solana González, M.A.M., Daniel Pérez González, *Análisis y Modelado con redes de Workflow del Proceso de Tratamiento de Experiencias Operativas.*
5. Claudia Jiménez Quintana, L.F.V., Francisco Pinto, *Análisis de Modelos de Procesos de Negocios en relación a la dimensión informática.*
6. *The Workflow Management Coalition Specification.* 2008.
7. Valencia., D.d.S.I.y.C.U.P.d., *Proceso de desarrollo de software.* 2010.
8. Padilla, I.L., *PLANEACION DE LOS RECURSOS DE LA EMPRESA: ERP*. **02**.
9. *Herramientas informáticas de ayuda a la gestión por procesos en la empresa.*
10. Ing. Oiner Gómez Baryolo1, I.Y.M.B. and I.D.G. Tejo3, *ARQUITECTURA TECNOLÓGICA PARA EL DESARROLLO DE SOFTWARE.*
11. Villacis, C.G.S., *Elaboración de un componente de software reutilizable para la interacción de aplicaciones cliente y sistemas de gestión de flujos de trabajo.* p. 72,73.
12. Pérez, M.S., *Propuesta de modelo de desarrollo de software tecnológico del Centro de Soluciones de Gestión.* 2009.
13. PHP, G.d.d.d., *Manual de PHP*. 2001.
14. Esser, S., *Secure Programming with the Zend-Framework.* 2009.
15. Morrison, M., *Xml Al Descubierto.* 2000.
16. Coalition, W.M., *The Workflow Management Coalition Specification,Workflow Management Coalition Workflow Standard,Process Definition Interface -- XML Process Definition Language.* 2008.
17. *¿Qué es NetBeans?*
18. *Visual Paradigm For Uml.* . 2011.
19. *Área temática de Linux. Portal de Linux* 2011.
20. Aquino, Y., *Implementación de módulo de Contabilidad General del sistema integral de gestión Cedrux.* 2009.
21. Mancha, C.d.e.d.s.l.d.C.L., *Análisis de aplicación: Geany.* 2011.
22. mozilla.org, *De todos. Para todos.* 2012.
23. PortalProgramas, *"Interfaz grafica para Subversion."*.
24. Sawyer, I.S.y.P., *Requirements Engineering: A good practice guide.* 1997.
25. María José Escalona, N.K., *Ingeniería de Requisitos en Aplicaciones para la Web – Un estudio comparativo.* 2002.
26. Pressman, R.S., *Ingenieria del Software - Un Enfoque Practico.* 2002. . 8448132149**.**
27. Tedeschi, N., *¿Qué es un Patrón de Diseño?* .
28. Astudillo, M.V.y.H., *Fundamentos de Ingeniería de Software.*
29. Gonzalez, D.H., *Métricas en el desarrollo del Software.*

30. DesarrolloWeb.com, *Métricas de Software.*
31. Usaola, D.M.P., *Curso de doctorado sobre Proceso software y gestión del conocimiento. Pruebas del Software.* Ciudad Real, febrero de 2006.
32. Germán Hüttemann, A.G., Alberto Amadeo, *Trabajo de Pruebas de Software del Sistema de Exámenes Web del grupo 3, desarrollado en la materia Ingeniería de Software II.*
33. Portal Corporación Sybven, Corporación Sybven, C.A., 2012

GLOSARIO DE TÉRMINOS

**Android:** Es un sistema operativo móvil basado en Linux, que está enfocado para ser utilizado en dispositivos móviles como teléfonos inteligentes, tabletas, Google TV y otros dispositivos.

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

**Arquitectura de SW:** La arquitectura de software define, de manera abstracta, los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación entre ellos. Toda arquitectura debe ser implementada en una arquitectura física, que consiste simplemente en determinar qué computadora tendrá asignada cada tarea.

**BPEL:** (Business Process Execution Language) es un lenguaje estandarizado para la composición de servicios web. Básicamente, consiste en un lenguaje basado en XML diseñado para el control centralizado de la invocación de diferentes servicios Web.

**BPMN:** (Business Process Modeling Notation) es una notación gráfica estandarizada que permite el modelado de procesos de negocio, en un formato de flujo de trabajo (workflow).

**C:** Es un lenguaje orientado a la implementación de Sistemas Operativos, concretamente Unix. C es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.

**CASE: (**Ingeniería de Software Asistida por Computadora) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero.

**CSS:** (Cascading Style Sheets) es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML, es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores.

**Framework:** Es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar.

**Gecko:** Es un motor de renderizado libre escrito en C++, es una plataforma para aplicaciones multiplataforma, es decir: permite ejecutar aplicaciones sobre su motor.

**HTML:** HyperText Markup Language (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de páginas web.

**IDE:** (Integrated Development Environment), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien poder utilizarse para varios.

**InforMix:** Es un sistema gestor de bases de datos de IBM, adquirida en 2001 a una compañía (también llamada Informix o Informix Software) cuyos orígenes se remontan a 1980.

**InterBase:** Es un sistema de gestión de bases de datos relacionales (RDBMS) desarrollado y comercializado por la compañía Borland Software Corporation y actualmente desarrollado por su ex-filial CodeGear.

**IOC:** Es un método de programación en el que el flujo de ejecución de un programa se invierte respecto a los métodos de programación tradicionales, en los que la interacción se expresa de forma imperativa haciendo llamadas a procedimientos o funciones.

**J2EE:** Es una plataforma de programación para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java con arquitectura de N capas distribuidas y que se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones.

**Java JDK:** Es un software que provee herramientas de desarrollo para la creación de programas en Java. Puede instalarse en una computadora local o en una unidad de red.

**Java Script:** Es un lenguaje de programación interpretado. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

**Java:** Es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

**Linux:** Es un núcleo libre de sistema operativo basado en Unix. Es uno de los principales ejemplos de software libre.

**Mac OS:** Es el nombre del sistema operativo creado por Apple para su línea de computadoras Macintosh. Es conocido por haber sido el primer sistema dirigido al gran público en contar con una interfaz gráfica compuesta por la interacción del mouse con ventanas, Icono y menús.

**MathML:** (Mathematical Markup Language) es un lenguaje de marcado basado en XML, cuyo objetivo es expresar notación matemática de forma que distintas máquinas puedan entenderla, para su uso en combinación con XHTML en páginas web, y para intercambio de información entre programas de tipo matemático en general.

**MVC:** Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de negocio en tres componentes distintos.

**mySQL:** Es un sistema de gestión de bases de datos relacional, multihilo y multiusuario.

**Oracle:** Es un sistema de gestión de base de datos objeto-relacional (o ORDBMS por el acrónimo en inglés de Object-Relational Data Base Management System), desarrollado por Oracle Corporation. Se considera a Oracle como uno de los sistemas de bases de datos más completos.

**Payton:** Es un lenguaje de programación de alto nivel cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible.

**Perl:** Es un lenguaje de programación diseñado por Larry Wall en 1987.

**RSS:** (Really Simple Syndication) es un formato XML para señalar o compartir contenido en la web. Se utiliza para difundir información actualizada frecuentemente a usuarios que se han suscrito a la fuente de contenidos.

**RUP:** (Rational Unified Process en inglés, habitualmente resumido como RUP) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

**SGML:** (Standard Generalized Markup Language) consiste en un sistema para la organización y etiquetado de documentos.

**Solaris:** Es un sistema operativo de tipo Unix desarrollado desde 1992 inicialmente por Sun Microsystems y actualmente por Oracle Corporation como sucesor de SunOS.

**SQL:** El lenguaje de consulta estructurado o SQL (por sus siglas en inglés Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en estas.

**SVG:** Los Gráficos Vectoriales Escalables (del inglés Scalable Vector Graphics) o SVG es una especificación para describir gráficos vectoriales bidimensionales, tanto estáticos como animados, en formato XML.

**TIC:** Las tecnologías de la información y la comunicación (TIC o bien NTIC para nuevas tecnologías de la información y de la comunicación) agrupan los elementos y las técnicas usados en el tratamiento y la transmisión de la información, principalmente la informática, Internet y las telecomunicaciones.

**UML:** Lenguaje Unificado de Modelado (LUM o UML, por sus siglas en inglés, *Unified Modeling Language*) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group).

**W3C:** El World Wide Web Consortium, abreviado W3C, es un consorcio internacional que produce recomendaciones para la World Wide Web.

**Wf-XML:** Es un estándar desarrollado por la WfMC, que ofrece una forma estándar para un motor de BPM para invocar un proceso en el otro motor de BPM, y esperar a que se complete.

ANEXOS

**Anexo 1: Tabla de la evaluación técnica (TOC).**

| Clase | Cantidad de Procedimientos | Responsabilidad | Complejidad | Reutilización |
|---|---|---|---|---|
| Controlador | 5 | Media | Media | Media |
| Base | 6 | Media | Media | Media |
| Package | 4 | Baja | Baja | Alta |
| PackageHeader | 4 | Baja | Baja | Alta |
| RedefineableHeader | 4 | Baja | Baja | Alta |
| WorkflowProcesse | 5 | Media | Media | Media |
| Activities | 5 | Media | Media | Media |
| Activity | 4 | Baja | Baja | Alta |
| Transitions | 5 | Media | Media | Media |
| Pool | 4 | Baja | Baja | Alta |
| Pools | 4 | Baja | Baja | Alta |
| Lanes | 4 | Baja | Baja | Alta |
| Lane | 4 | Baja | Baja | Alta |
| XPDLVersion | 4 | Baja | Baja | Alta |
| WorkingTime | 4 | Baja | Baja | Alta |
| WorkflowProcesses | 4 | Baja | Baja | Alta |
| WebServiceOperation | 5 | Media | Media | Media |
| WebServiceFaultCatch | 4 | Baja | Baja | Alta |
| WaitingTime | 4 | Baja | Baja | Alta |
| Version | 4 | Baja | Baja | Alta |
| VendorExtensions | 4 | Baja | Baja | Alta |
| VendorExtension | 4 | Baja | Baja | Alta |
| Vendor | 4 | Baja | Baja | Alta |
| ValidTo | 4 | Baja | Baja | Alta |
| ValidFrom | 4 | Baja | Baja | Alta |
| UnionType | 4 | Baja | Baja | Alta |
| TypeDeclarations | 4 | Baja | Baja | Alta |
| TypeDeclaration | 4 | Baja | Baja | Alta |
| TriggerTimer | 5 | Media | Media | Media |
| TriggerRule | 4 | Baja | Baja | Alta |

**Anexo 2: Tabla de la evaluación técnica (RC).**

| Clase | Cantidad de Relaciones de Uso | Acoplamiento | Complejidad Mantenimiento | Reutilización | Cantidad de Pruebas |
|---|---|---|---|---|---|
| Controlador | 1 | Bajo | Baja | Alta | Baja |
| Base | 168 | Alto | Alta | Baja | Alta |
| Package | 17 | Alto | Alta | Baja | Alta |
| PackageHeader | 10 | Alto | Alta | Baja | Alta |
| RedefineableHeader | 5 | Alto | Medio | Medio | Medio |
| WorkflowProcess | 15 | Alto | Alta | Baja | Alta |
| Activities | 1 | Bajo | Baja | Alta | Baja |
| Activity | 24 | Alto | Alta | Baja | Alta |
| Transitions | 1 | Bajo | Baja | Alta | Baja |
| Pool | 3 | Alto | Medio | Medio | Medio |
| Pools | 1 | Bajo | Baja | Alta | Baja |
| Lanes | 1 | Bajo | Baja | Alta | Baja |
| Lane | 4 | Alto | Medio | Medio | Medio |
| XPDLVersion | 0 | Ninguno | Baja | Alta | Baja |
| WorkingTime | 0 | Ninguno | Baja | Alta | Baja |
| WorkflowProcesses | 1 | Bajo | Baja | Alta | Baja |
| WebServiceOperation | 5 | Alto | Medio | Medio | Medio |
| WebServiceFaultCatch | 2 | Medio | Baja | Alta | Baja |
| WaitingTime | 0 | Ninguno | Baja | Alta | Baja |
| Version | 0 | Ninguno | Baja | Alta | Baja |
| VendorExtensions | 1 | Bajo | Baja | Alta | Baja |
| VendorExtension | 3 | Alto | Medio | Medio | Medio |
| Vendor | 0 | Ninguno | Baja | Alta | Baja |
| ValidTo | 0 | Ninguno | Baja | Alta | Baja |
| ValidFrom | 0 | Ninguno | Baja | Alta | Baja |
| UnionType | 1 | Bajo | Baja | Alta | Baja |
| TypeDeclarations | 1 | Bajo | Baja | Alta | Baja |
| TypeDeclaration | 3 | Alto | Medio | Medio | Medio |
| TriggerTimer | 2 | Medio | Baja | Alta | Baja |
| TriggerRule | 0 | Ninguno | Baja | Alta | Baja |
| TriggerMultiple | 4 | Alto | Medio | Medio | Medio |

**Anexo 3: Tabla de valores del cálculo de la complejidad.**

| **Nombre del Método** | **V (G) = (A – N) + 2** | **V (G) = P + 1** | **V (G) = R** |
|---|---|---|---|
| desassembleXPDL | 3 | 3 | 3 |
| toXPDL | 2 | 2 | 2 |
| fromXPDL | 6 | 6 | 6 |
| myGenerarDocumento | 1 | 1 | 1 |
| myParcearXPDL | 2 | 2 | 2 |
| dealWithChoises | 3 | 3 | 3 |