# PROPUESTA DE MAPEO PARA LA TRANSFORMACIÓN DE MODELOS DE PROCESOS DE NEGOCIO A MODELOS DE ESPECIFICACIÓN DE SOFTWARE

**JUAN JOSÉ CADAVID GÓMEZ**

**UNIVERSIDAD EAFIT**

**Medellín**

**2008**

# PROPUESTA DE MAPEO PARA LA TRANSFORMACIÓN DE MODELOS DE PROCESOS DE NEGOCIO A MODELOS DE ESPECIFICACIÓN DE SOFTWARE

**JUAN JOSÉ CADAVID GÓMEZ**

**Proyecto de grado para optar al título de**
**Ingeniero de Sistemas**

**Asesor**
**JUAN BERNARDO QUINTERO**
**Docente Departamento de Informática y Sistemas**

**UNIVERSIDAD EAFIT**
**Medellín**
**2008**

A todos quienes trabajan

Por el progreso de la especie humana

## AGRADECIMIENTOS

Este trabajo no hubiera sido posible sin el apoyo incondicional de mis padres, Yolanda y Gustavo, quienes siempre me motivaron a encontrar mi camino en la vida. A mi asesor Juan Bernardo Quintero quien me condujo a través de la ruta que decidí seguir. A Raquel Anaya y el Grupo de Investigación en Ingeniería de Software de la Universidad EAFIT por abrirme el espacio para emprender éste camino del conocimiento, y a las profesoras del área de Sistemas de Información, Sonia Cardona y Berta Solórzano, por mostrarme cual debe ser el fin de ese camino.

A mi familia, Andrés, Santiago, Mónica, Valeria, Mariela y Hernando, y a mis amigos Carlos García, Cristian Cardona, Andrés Jiménez, Diana Palacio, Marcela Moreno, Marcela Sánchez, Laura López, Andrea Yanza, Magda Cárdenas, Jacob Enriquez y Kyoko Miki, junto a quienes he crecido, por brindarme su compañía y el constante apoyo incondicional a pesar de las circunstancias en cada caso.

A los amigos de Archetypus, Carlos Ospina, David López y Jesús Hincapié, quienes en el día a día de ir cristalizando estas ideas me hicieron reír y disfrutar la experiencia.

Al área de I+D Ingeniería de Avansoft, Luis Londoño, Lenin y Silvia Lozano, Nelson Moreno e Indira Cohen, con quienes el trabajo diario se convirtió en otra fuente de alimentación de la inspiración.

## CONTENIDO

**Pág.**

## LISTA DE TABLAS

**Pág.**

## LISTA DE FIGURAS

**Pág.**

## LISTA DE ANEXOS

**Pág.**

# INTRODUCCIÓN

Es común ver que las áreas de TI[1] de las organizaciones o proveedores "estratégicos" de soluciones de TI actúan solo cuando sus servicios son solicitados por un funcionario del negocio que responde a un problema identificado, limitándose a responder a este pero no se ocupan de conocer el contexto en el que éste se desenvuelve, de manera que actúan de forma reactiva. Este modo de operar conlleva muchos problemas; entre ellos la construcción de soluciones cuya finalización no es adecuada o es tardía, cuando las condiciones bajo la cual fue especificada han cambiado, ocasionando así una gran cantidad de proyectos fracasados, fallidas inversiones y cuantiosas pérdidas.

La realidad es que un llamado socio o proveedor estratégico de tecnologías de información no debería permanecer en una actitud pasiva mientras que la organización se encuentra en un eterno vaivén de cambio causado por factores internos y externos. Por el contrario, debe asumir una actitud proactiva, que en lugar de responder a situaciones de problemas se convierta en una verdadera plataforma de apoyo organizacional, en donde una visión conjunta lleve a acciones concretas de generación de valor y ventajas competitivas.

Evidenciamos entonces una situación actual que refleja las fallas percibidas en nuestro día a día y una situación deseada, que busca la continua creación de valor. Para lograr llegar allí, se hace necesario un puente, un encuentro semántico en donde el negocio y las tecnologías de información hablen el mismo idioma. Este empeño ha sido perseguido durante décadas por numerosas propuestas acogidas en el paraguas de la Ingeniería de Software, tales como la orientación a objetos, la ingeniería de requisitos, entre otras. Cada una de ellas ha venido

[1] Tecnologías de Información

cerrando mas y mas la brecha semántica TI-Negocio con su contribución, a pesar de que aún queda un largo camino por recorrer.

El trabajo que el lector tiene en sus manos es un esfuerzo más tras éste empeño. La presente propuesta ha tomado la ingeniería de modelos como la disciplina que otorga las técnicas, métodos, metodologías y herramientas que mejor satisfacen los requisitos de éste empeño, como se podrá evidenciar en los primeros capítulos del trabajo. Estamos convencidos de que el uso de modelos como artefactos de primera clase en el desarrollo de soluciones informáticas constituye una de las aproximaciones avanzadas con mayor generación de valor.

En el primer capítulo se realizará una presentación del proyecto de grado; en el segundo capítulo se hará un estudio del estado del arte de los tópicos relevantes a éste proyecto; en el tercer capítulo se presenta un estudio comparativo de los distintos enfoques de modelado de negocio más conocidos a la fecha; en el cuarto capítulo se presenta el conjunto de heurísticas de mapeo de CIM a PIM que conforman la propuesta; en el quinto capítulo se presenta la documentación técnica de la aplicación construida con base en la propuesta de transformación y finalmente, en el sexto y séptimo capítulo se exponen las conclusiones y trabajos futuros a partir del proyecto.

# 1. PRESENTACION DEL PROYECTO

## 1.1 Planteamiento del problema

Uno de las principales dificultades que se ha enfrentado en el uso de tecnologías de información a lo largo de la historia ha sido la falta de alineación de las soluciones tecnológicas con el problema para el cual están siendo construidas. En el contexto de las aplicaciones de software empresariales, nos damos cuenta de que a menudo los artefactos desarrollados no conforman con lo requerido para apoyar un proceso de negocio. Para abordar esta situación la ingeniería de software ha propuesto diversas técnicas; una de ellas es construir modelos que permitan capturar aspectos relevantes tanto del problema identificado como de la solución a construir. Alrededor de esto, han surgido estándares como UML[2], una notación para construir modelos que representan las diferentes vistas de un sistema, y MDA[3], un enfoque que propone la utilización de los modelos como los artefactos principales en la concepción y desarrollo de software.

Desde su especificación, MDA promete ser "otro paso pequeño en el largo camino de convertir nuestro arte [el desarrollo de software] en una disciplina de ingeniería" (OMG, 2003). Además de procurar una concordancia semántica entre el espacio y y el de la solución, esto implica grandes ventajas en el desarrollo de software, tales como la reutilización y la automatización de tareas de implementación. Debido a esto, numerosas investigaciones han surgido alrededor de éstas emergentes disciplinas, tanto en el escenario industrial y corporativo como en el académico. En (Quintero, et al., 2007) se propone como un posible trabajo futuro "Plantear estrategias para que las herramientas de transformación amplíen la cobertura del CIM[4], y de esta forma permitir el reuso desde etapas más

[2] *Unified Modeling Language,* Lenguaje Unificado de Modelado
[3] *Model Driven Architecture*, Arquitectura Dirigida por Modelos
[4] *Computation Independent Model:* Modelo Independiente de la Computación

tempranas". Este proyecto de grado busca definir unas heurísticas para implementar en una herramienta MDA con soporte a modelos de procesos de negocio de manera que se posibilite la obtención de un PIM[5] que exponga la especificación de la aplicación de software a ser construida.

## 1.2 Objetivos

Proponer e implementar una especificación para mapear un modelo de procesos de negocio a un modelo de especificación de software bajo el enfoque MDA.

### 1.2.1 Objetivos específicos

| Objetivo | Indicador verificable |
|---|---|
| a) Estudiar el estado del arte de los diversos estándares (ej. BPM, SOA), metodologías (ej. Proceso Unificado de Desarrollo) y herramientas (ej. CASE, análisis de negocios) existentes en la actualidad de relevancia para éste proyecto. | Estado del arte de los tópicos mencionados (Capítulo 2). |
| b) Realizar un estudio comparativo de enfoques de modelado y ejecución de procesos de negocio. | Evaluación y conclusiones del estudio; reportes técnicos (Capítulo 3). |
| c) Definir heurísticas de mapeo de un modelo independiente de la computación (CIM) a un modelo independiente de la plataforma (PIM). | Artículo con el conjunto propuesto de heurísticas (Capítulo 4). |
| d) Construir una implementación del mapeo propuesto, adaptando una herramienta *open source* de manera que permita el modelado de negocio y su transformación en modelos de arquitectura de sistemas, basado en la especificación de mapeo que se propone. | Aplicación ejecutable, con documentación técnica y guía de usuario (Capítulo 5 y aplicación anexa). |

**Tabla 1:** Indicadores verificables de los productos del proyecto.

[5] *Platform Independent Model:* Modelo Independiente de la Plataforma

## 1.3 Metodología

Inicialmente se hará un estudio del estado del arte alrededor de los tópicos en los que se desenvuelve el proyecto de grado, concretamente los estándares, metodologías, herramientas y propuestas de transformación existentes, el cual será materializado en el primer artículo a ser escrito. Se hará lectura de artículos e investigación en la web, y se probarán algunas aplicaciones para elegir la plataforma sobre la cual se implementará la especificación propuesta.

A continuación se hará un sondeo de las diferentes aproximaciones existentes para el modelado y ejecución de procesos de negocio, con las cuales se realizará un estudio comparativo basado en una serie de criterios pertinentes para el propósito del proyecto, permitiendo así la selección de la aproximación más apropiada para la definición de heurísticas.

Luego se procederá a construir la propuesta de heurísticas para la transformación de CIM a PIM, que constituye el núcleo central del proyecto. Con la bibliografía seleccionada, se trabajará conjuntamente con el asesor del proyecto, a la vez que se contará con el apoyo de otros miembros del Grupo de Investigación de Ingeniería de Software de la Universidad EAFIT.

Finalmente se realizará la implementación de la especificación de transformación sobre la plataforma seleccionada, la cual será probada. Seguidamente se desarrollará el estudio de un caso basado en una situación real para mostrar el funcionamiento y la utilidad de la propuesta.

## 1.4 Beneficiarios

El conocimiento obtenido en la actividad investigativa de éste proyecto de grado será de gran utilidad para la industria local del software interesada en dos

aspectos importantes. El primero de ellos es el desarrollo dirigido por modelos, que al usar el modelo como activo de valor en el proceso de desarrollo de software, permite el reuso desde etapas tempranas, incrementando así la eficiencia. Por otro lado, la disciplina del modelado de negocio le ha brindado mecanismos a los ingenieros de software para entender el dominio del problema sobre el cual están trabajando, debido a que la construcción de modelos inicia desde una perspectiva independiente de la computación; esto ha posibilitado cada vez más que las soluciones construidas sirvan de forma más efectiva su propósito. Concretamente hablando, serán beneficiados los usuarios de la herramienta construida, quienes obtendrán las ventajas que provee la propuesta metodológica presentada, como se expresan en el capítulo 4. Otros posibles beneficiarios son la academia, con el estudio de modelado de negocio para mostrar las diferentes alternativas para las actividades de modelado en las etapas tempranas de desarrollo de software.

## 1.5 Importancia del problema dentro de la carrera y el medio

Uno de los principales objetivos en la formación del ingeniero de sistemas corresponde a eficaz utilización de las TICs[6] en el medio. Hoy por hoy, la concentración en esta competencia cobra mayor importancia que las mismas fortalezas técnicas, debido a que la naturaleza de los negocios de nuestro medio requiere cada vez más nuestro entendimiento de ellos en toda su complejidad, no para aplicar la tecnología más avanzada en cada caso, sino siempre la más adecuada. Es por esta razón que durante el pregrado de Ingeniería de Sistemas se haga especial énfasis en dos áreas particulares:

- Los Sistemas de Información Organizacionales, que tratan de hacer caer en cuenta al estudiante de la importancia del valor que deben agregar las soluciones que construya para la organización en la que se desempeña como ingeniero de sistemas.
- La Ingeniería de Software, una disciplina que se propone unificar un conjunto de aproximaciones y buenas prácticas para establecer procesos de desarrollo de software, encaminados a construir productos efectivos de alta calidad y de forma eficiente.

Dentro de las ciencias computacionales las áreas de sistemas de información organizacionales e ingeniería de software se han mantenido aisladas. Este proyecto de grado pretende ser un punto de unión de éstas dos áreas, a fin de plantear una posible visión común en donde converjan los esfuerzos y se dé un paso adelante en la forma de utilizar las TICs.

[6] Tecnologías de Información y Comunicaciones

# 2. ESTADO DEL ARTE

El presente estudio del arte corresponde a la intención de hacer una examinación del entorno tanto académico como industrial como prerrequisito para la ejecución del proyecto. Los tópicos a examinar son el rol de los procesos de negocio en la ingeniería de software en la actualidad, el estado actual del desarrollo de aplicaciones de software empresariales y la ingeniería de modelos

## 2.1 El Modelado de Procesos de Negocio y su Importancia en el Desarrollo de Software

En esta sección se dará un vistazo de la actividad de construir modelos de procesos al interior de las organizaciones y su importancia dentro del quehacer organizacional.

### 2.1.1 Definición y Caracterización

A lo largo de la historia de las teorías administrativas, diferentes corrientes efímeras han surgido que proclaman diversas estrategias para las organizaciones. A pesar de su carácter volátil, todas procuraban el mismo objetivo para las organizaciones: Transformar lo que hacen para hacerlo mejor, más barato, mas rápido (Smith, 2003). Su motivación es que la globalización ha impuesto nuevos retos sobre las corporaciones, en las que los clientes demandan soluciones totales y personalizadas, las fronteras entre las industrias se hacen cada vez mas borrosas, la competición tradicional se transforma en nuevas formas de colaboración, las cadenas de valor se convierten en las principales ventajas competitivas y la única certeza es el cambio. Estas preocupaciones han sido consideradas por las diferentes estrategias administrativas tales como *downsizing*, reingeniería, *coaching*, costeo basado en actividades (ABC) entre otros. Sin embargo, todas abordan el problema de la misma forma: Administrar los procesos

de negocio de la organización, planteando metodologías para su análisis, diseño, implementación y ejecución.

Un proceso de negocio es una descripción del trabajo requerido para generar valor a los clientes de un negocio (Smith, 2005), presentando un conjunto completo y dinámicamente coordinado de actividades colaborativas y transaccionales (Smith, 2003). Las compañías los utilizan para entender sus propias prácticas, luego para medir su desempeño y subsecuentemente para mejorarlas. Los objetivos concretos varían según el proceso y la organización, entre ellos incrementar la eficiencia, alcanzar una productividad más alta, mayor confiabilidad, reducción de costos, lograr economías de escala, innovar, mejoras de calidad, y sobre todo, satisfacción del cliente. Los procesos de negocio implican un énfasis en el cómo, en contraste con los productos de negocio que enfatizan en el qué. Por lo tanto, un proceso es una ordenación específica de las actividades en un tiempo y lugar determinados, con un comienzo, un fin, y unas entradas y salidas claramente identificados, lo que constituyen una estructura para la acción (Davenport, 1993).

Según (Smith, 2003), los procesos de negocio presentan las siguientes características:

- Extensos y complejos, involucrando el flujo de extremo a extremo de materiales, información y compromisos de negocios.
- Dinámicos, respondiendo a las demandas de clientes y cambiando con las condiciones del mercado.
- Ampliamente distribuidos y personalizados a través de fronteras dentro y entre organizaciones, a menudo extendiendo múltiples aplicaciones en plataformas de tecnologías heterogéneas.
- De larga vida - Una simple instancia de un proceso como "desarrollar un producto" puede ejecutarse por meses o incluso años.

- Parcialmente automatizado. Actividades rutinarias son realizadas por computadores cuando es posible, por los beneficios en velocidad y confiabilidad.
- Son de naturaleza dual, "de negocios" y "técnica". Los procesos de TI son un subconjunto de procesos de negocio y proveen soporte a procesos de mayor escala que involucran personas y máquinas. Los procesos de negocio de extremo a extremo dependen de sistemas de computación distribuida que son ambos transaccionales y colaborativos.

### 2.1.2 La Administración de Procesos de Negocio (BPM[7])

Desde la década de 1910, Frederick Taylor observó grandes ineficiencias en el nivel operativo de las industrias dominantes en ese entonces, como por ejemplo las industrias de acero. Su conclusión fue que si hasta las más elementales tareas se planearan, grandes ahorros en tiempo y reducciones de costos podrían obtenerse a largo plazo, a medida que buenas prácticas se incorporaran a lo largo y ancho de las fábricas. Así, demostró que un enfoque más científico y de ingeniería podría ser más efectivo que los métodos tradicionales basados en la intuición e incentivos para motivar a los trabajadores. Medio siglo después, W. Edwards Deming demostraría con su método administrativo que el control estadístico de la calidad (SQC, Statistical Quality Control) le permitiría a jefes y trabajadores de fábricas sin conocimiento matemático alguno medir el desempeño de los procesos de manufactura. Ejemplos como éstos han demostrado que un enfoque estructurado y procedimental podría arrojar beneficios hasta antes desconocidos en la industria. Hoy, con las tecnologías de información, podemos imaginar nuevas innovaciones en la administración y mejoramiento de procesos de negocio. Sin embargo, es necesario conocer los desarrollos de las últimas

[7] *Business Process Management*, Administración o Gestión de Procesos de Negocio

décadas para conocer el estado del arte de la actual administración de procesos de negocio (Smith, 2005).

BPM, la administración de procesos, no es nada nuevo. Por décadas, analistas de negocio han tratado de implementar metodologías para optimizar la forma de trabajar al interior de sus organizaciones. Según (Smith, 2003), identificamos varios episodios en esta historia.

En los años 80, a las compañías se les prometió mejoras dramáticas en su desempeño gracias a la reingeniería de procesos. Esta primera ola de la administración de procesos proponía una mejora incremental de todos los procesos existentes, enfocándose en el flujo de trabajo, las entradas y las salidas de las etapas individuales de los procesos. Sin embargo, este enfoque bottom up proponía la automatización de los procesos con base a un rígido esquema de entradas y salidas de las actividades de trabajo, de manera que su alcance máximo eran las funciones de negocio. Además, un proyecto de reingeniería típicamente resultaba en grandes cantidades de trabajo manual y un gran costo en dinero y tiempo. Como resultado, pocas organizaciones realizaron reingeniería de sus procesos con una frecuencia menor a 5 años (Hagel, 2001), lo que no le permitía ser una alternativa de las organizaciones para adaptarse al constante cambio de su entorno.

Los años 90, la década de las suites ERP[8] y herramientas de workflow, se caracterizó por la adquisición de paquetes de software como una inversión estratégica para la administración de procesos de negocio. La idea era un producto "todo en uno", en donde cualquier componente requerido para la gestión organizacional estaba contenido en el paquete. Pero además de que su

[8] *Enterprise Resource Planning*

instalación podría tomar años, éstos eran inflexibles, de manera que la agregación de nuevos componentes o su personalización era bastante costosa.

En el pasado reciente, nuevas herramientas de modelado de procesos emergieron, las cuales podían capturar los procesos organizacionales en una forma gráfica entendible para los actores de negocio, y editable para una manipulación flexible y análisis subsecuente. Inicialmente, se utilizaron únicamente para producir documentación, y no proporcionaban un mecanismo de alineación de la implementación de éstos procesos con tecnologías de información. Adicionalmente, no existían estándares abiertos para el modelado, y en su lugar solo se contaban con herramientas propietarias.

En la actualidad, la tercera ola de la administración de procesos de negocio se hace evidente al contar con herramientas en las que los procesos pueden ser vistos por usuarios humanos mientras las máquinas lo ejecutan. La actividad de diseño de procesos se lleva a cabo desde arriba, el nivel de estrategia de negocio y diseño de procesos, hasta el bajo nivel, de apalancamiento de los sistemas de TI existentes. Estándares basados en metamodelos estructurados han surgido para apoyar el modelado de negocio y posibilitar la ruta hacia su ejecución con mínima intervención por parte de los desarrolladores de software.

Con base en las lecciones aprendidas de los sistemas de workflow y de las prácticas mal aplicadas de reingeniería, consorcios empresariales como BPMI[9] empezaron a visionar plataformas que permitieran el análisis, diseño, despliegue, ejecución y optimización de procesos de negocio de principio a fin, a los que denominarían BPMS[10] (Smith, 2005).

[9] *Business Process Management Initiative*
[10] *Business Process Management System* o *Business Process Management Suite*

### 2.1.3 El Ciclo de BPM

Las diferentes propuestas para hacer BPM han definido distintos ciclos de vida que definen las etapas por las que transcurre un proceso de negocio al interior de la organización. Sin embargo, en todas las propuestas identificamos tres macroactividades comunes:

- Diseño de Procesos: Se trata de la definición (en el caso de procesos nuevos) o descubrimiento (en el caso de procesos existentes) de procesos de negocio. Construir modelos de los procesos de negocio a gestionar al interior de la organización en alguna notación estándar de modelado para éste propósito (ver "Estudio comparativo de técnicas de modelado de negocio").
- Automatización de Procesos: También llamado ejecución de procesos de negocio. Se trata de construir una implementación tecnológica de apoyo al proceso. Por ejemplo, puede tratarse de una aplicación web orquestadora, de una instancia de BPEL[11] (OASIS WS-BPEL Technical Committee, 2007) o un ensamble de componentes de software.
- Monitoreo de Procesos: Obtención de indicadores de desempeño de los procesos en ejecución, para posibilitar un posterior análisis para llevar a cabo la optimización de procesos. Esto implica otra iteración del ciclo, empezando por un rediseño de los procesos.

En particular para éste trabajo, nos concierne principalmente la primera actividad, que directamente implica la construcción de modelos, que actuarán como artefactos insumo para el mapeo a modelos de especificación de aplicaciones de software, como se especifica el objetivo específico “c” de éste proyecto (ver sección 1.2.1), y así proveer un mecanismo para la segunda actividad, la

[11] *Business Process Execution Language*

automatización de procesos. El desarrollo de propuestas y herramientas para la actividad de monitoreo de procesos está fuera del alcance de este proyecto.

### 2.1.4 El Modelado de Procesos de Negocio

El modelado de procesos de negocio es la actividad de representar el estado actual (*as is*, como es) y el estado futuro (*to be*, como será) de los procesos de una empresa para su comparación y contraste, de tal forma que permitan el análisis y mejoramiento para llegar a la situación deseada. Los modelos son construidos por analistas de negocio y administradores (Business Modeling Forum, 2007). Hasta hace algunos años, los modelos de procesos de negocio eran simples gráficas que mostraban flujos de datos y no tenían otro propósito más que la documentación. Hoy en día, los procesos de negocio son modelos estructurados que muestran como los sistemas de información de la organización deben estar directamente alineados a los objetivos de ejecutivos de negocio, y gracias a los estándares abiertos actuales (revisados más adelante en este trabajo), hoy en día se pueden convertir en los artefactos que dirigen el desarrollo de las plataformas de TI que soportan su ejecución. Las representaciones gráficas de procesos ofrecidas por las herramientas de BPM les ofrecen a los gerentes y a los desarrolladores de sistemas de información una visión más amplia y un control sobre la lógica de su negocio, por lo tanto afianzando su eficiencia, innovación, flexibilidad y la posibilidad de adaptarse más naturalmente al cambiante entorno.

El modelado de procesos de negocio es la primera etapa dentro del ciclo de BPM, y su propósito es capturar la información clave de cada proceso. Esta descripción debe incluir el conjunto de actividades que componen el proceso, la secuencia lógica en la que éstas se ejecutan, diferentes participantes y sus responsabilidades, herramientas utilizadas, relación con objetivos corporativos y elementos de entrada (insumos de proceso) y salida (productos de proceso) (Lombardi Software, 2007). Adicionalmente, los modelos de procesos pueden

comprender modelos de redes, modelos de objetos, flujos de control, flujos de mensaje, reglas de negocio, métricas, excepciones, transformaciones y asignaciones (Smith, 2003).

A lo largo de las "olas" de la gestión de procesos de negocio, diferentes técnicas para el modelado del negocio han surgido. En el siguiente capítulo, realizamos un estudio comparativo de 5 técnicas de modelado a la luz de los criterios que son relevantes para todas las audiencias que las usan, desde analistas de negocio hasta arquitectos de software.

## 2.2 El Desarrollo de Aplicaciones de Software Empresariales en la Actualidad

Para lograr el objetivo de éste proyecto, es necesario dar un vistazo a las metodologías de desarrollo de software utilizadas en la actualidad. En la primera sección se analizan los principales problemas afrontados, con el fin de conocer cómo la ingeniería de modelos puede ayudar a mitigarlos. Luego se examinan las últimas tendencias en el desarrollo de software, haciendo énfasis en como éstas cuentan con un auténtico potencial para apoyar el negocio y cómo el modelado ayuda a lograr esta meta.

### 2.2.1 El Proceso Unificado de Rational (RUP)

RUP propone un proceso para el desarrollo de software que unifica varias disciplinas de trabajo (Kruchten, 2003). Fue creado durante los años 80 y 90 por Rational Software, que a partir de 2002 se convirtió en una filial de IBM. El proceso en sí mismo no es lineal; y presenta las actividades organizadas en dos dimensiones: una basada en su contenido y otra en su ordenación en el tiempo durante el desarrollo de una aplicación de software.

Las actividades organizadas según su contenido u objetivo específico dentro del proceso de desarrollo se denominan disciplinas o flujos de trabajo, y son: modelado de negocio, requisitos, análisis y diseño, implementación, pruebas y despliegue; y las disciplinas de apoyo que son: gestión del cambio y la configuración, gestión de proyectos y el entorno. Las actividades se pueden agrupar de acuerdo a su ordenación en el tiempo durante el proceso de desarrollo; en este caso se denominan fases, y son en su orden: Incepción, elaboración, construcción y transición (Kruchten, 2003).

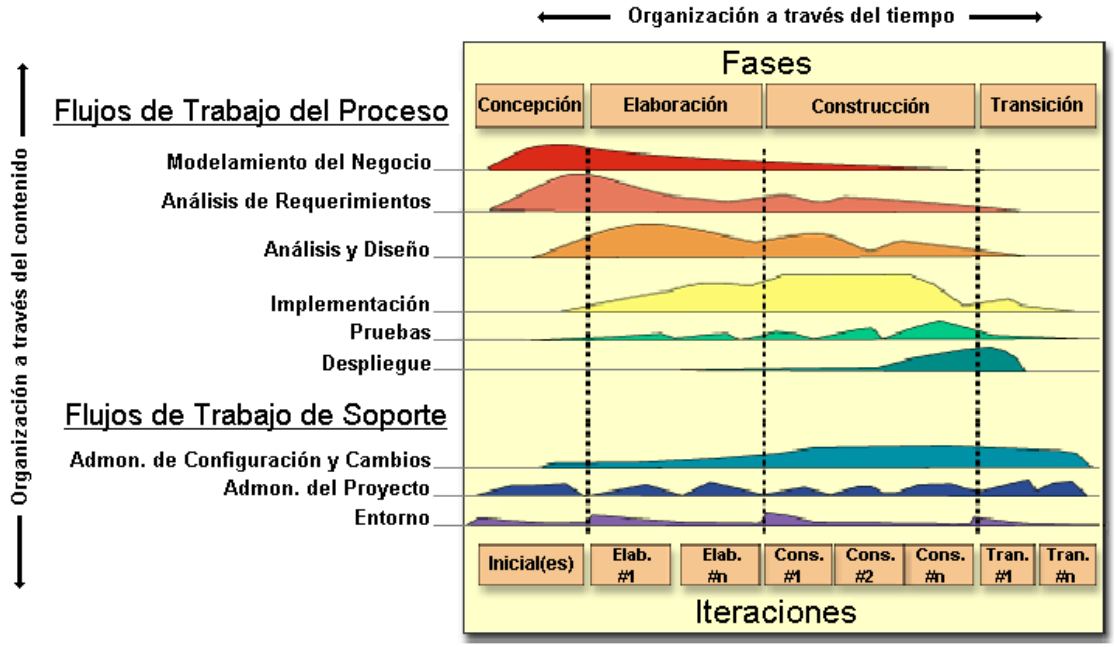


**Figura 1.** Proceso Unificado de Rational

RUP alcanzó grandes niveles de popularidad de aplicación en la industria, gracias a la estandarización de las mejores prácticas de ingeniería de software y su unificación en un proceso aplicable. Sin embargo, a través del tiempo se empezaron a destacar sus debilidades e insuficiencias. Una de ellas señalaba que el proceso era excesivamente robusto e incluso burocrático, pues generaba grandes volúmenes de documentación que podrían restarle agilidad al proceso de

desarrollo. Sin embargo, la crítica más fuerte se debió al hecho de que se trataba solo de un proceso de desarrollo que tiene como punto de partida una especificación de requisitos, y no una vista completa del contexto organizacional en el que la solución estará ubicada (Agile Manifesto Signatories, 2008).

RUP incluye como la disciplina inicial del proceso el modelado de negocio, pero no exige una representación integral de la organización, sino que en su lugar se limita a modelar algunos factores que pueden influir en el proceso de desarrollo; esto es una limitante pues un modelo completo del negocio es necesario para lograr el reuso a gran escala de activos de tecnología. Adicionalmente, al referirse al negocio utiliza el concepto de organización objetivo de un sistema, mostrando así claramente la tendencia de especificar y construir un sistema y luego implantarlo en una organización objetivo. Así evidenciamos que el proceso garantizaba un desarrollo de soluciones bajo buenas prácticas, pero no garantizaba que el resultado final iba a contribuir a la generación de valor en las organizaciones.

### 2.2.2 Los Requisitos y su Modelado con Casos de Uso

En el actual medio industrial, entre las compañías de software más maduras se puede reconocer procesos de desarrollo de software completamente guiados por la ingeniería de requisitos y el modelado de casos de uso, actividades a partir de las cuales se extraen los artefactos que dirigen las etapas posteriores del proyecto. Un requisito ha sido definido por (IEEE, 1993) como "Una condición o capacidad que un usuario necesita para solucionar un problema o lograr un objetivo", mientras que los casos de uso son una forma de representar los requisitos funcionales.

Para asegurar el cumplimiento de los objetivos de negocio del proyecto, se procura la conservación de la trazabilidad de dichos artefactos desde su especificación temprana hasta su implementación. De esta forma, la

especificación de un catálogo de requisitos y la construcción de modelos de casos de uso han probado ser técnicas muy útiles, además de resultar simples e intuitivas para el analista desarrollador.

Sin embargo, a pesar del avance que éstas técnicas han representado para la ingeniería de software, algunos autores han señalado las dificultades que éstas presentan, particularmente porque su uso frecuentemente se lleva a cabo en un ambiente de incertidumbre, en donde se carece de suficiente información acerca del contexto que rodea la aplicación de software a construir (es decir, la organización), y por lo tanto un alto componente de intuición por parte de los desarrolladores es requerido, lo que posiblemente acarrea a resultados imprecisos, alejados de lo que se esperaba (Ortín, et al., 2000). Específicamente, el problema consiste en que el modelado de los casos de uso por sí mismo no garantiza que éstos han sido identificados correctamente, y que van a brindar verdadero apoyo al negocio.


**Figura 2.** El desarrollo de aplicaciones de software empresariales en la actualidad.

En la figura 1 vemos una representación metafórica de la situación. Cada aplicación de software es una solución a un problema cuyo espacio hace parte de un contexto, pero éste permanece en la oscuridad. Es necesario tener una vista más amplia del entorno en donde cada uno de los actores se desenvuelve, e

identificar la motivación de negocio que impulsa la realización de sus casos de uso.

### 2.2.3 Análisis y Diseño Orientado a Objetos

El paradigma de la orientación a objetos traza sus raíces desde los años 60, pero solo hasta principios de los 90, con la aparición de los primeros lenguajes que se proclamaron "orientados a objetos", el paradigma se hizo realmente popular en la industria del desarrollo de software, al mismo tiempo que UML se empezaba a convertirse en la notación estándar para modelar soluciones construidas con este paradigma. Sin embargo, el conocimiento de los lenguajes de programación y las notaciones de modelado no era suficiente; era necesario adoptar el modo de "pensar en objetos", y aprender una metodología para traducir especificaciones de software a código.

Así nació el "arte" del Análisis y Diseño Orientado a Objetos (ADOO) (Larman, Octubre 2004), en donde en primer lugar los objetos -o conceptos- en el dominio del problema (análisis) son buscados y descritos, y subsecuentemente se definen los objetos de software y como éstos colaboran para satisfacer los requerimientos (diseño).

Cuando esta actividad del proceso ha sido llevada a cabo, se da lugar al inicio de la implementación, en donde los objetos son codificados en lenguajes como Java. (Larman, Octubre 2004) Adopta el ciclo de vida iterativo en donde las etapas de ADOO se repiten en cada iteración y a medida que avanzan, se enfatizan en los detalles y se pasa de los conceptos generales a lo específico.

Sin embargo, en la introducción de ésta publicación, se presenta un sumario de los pasos que componen el análisis orientado a objetos y el diseño orientado a objetos:

**Actividades de Análisis Orientado a Objetos**

- **Definición de casos de uso:** Son historias escritas que expresan lo que el usuario espera de la aplicación. Como tal, es una técnica que ayuda en la identificación, organización y gestión de requisitos funcionales.
- **Definición de un modelo de dominio:** La creación de una descripción del dominio desde la perspectiva de la clasificación por objetos. Una descomposición del dominio involucra una identificación de conceptos, atributos y asociaciones que son consideradas importantes. El resultado puede ser expresado en un modelo de dominio, que es ilustrado en un conjunto de diagramas que muestran los conceptos u objetos de dominio. Es importante notar que estos objetos no se refieren a objetos de software, sino a objetos del mundo real representados en un modelo.

**Actividades de Diseño Orientado a Objetos**

- **Definición de diagramas de interacción:** Luego de haber identificado los objetos del dominio, el diseño orientado a objetos se ocupa de definir los objetos de software y sus colaboraciones. Una notación común para ilustrar estas colaboraciones es un diagrama de interacciones que muestre el flujo de mensajes entre objetos de software y por lo tanto, la invocación de métodos.
- **Definición de diagramas de clases de diseño:** En adición a una vista dinámica de los objetos que colaboran entre sí, es útil crear una vista estática de las definiciones de clases con un diagrama de clases de diseño. Este ilustra los atributos y métodos de las clases. En contraste con el modelo de dominio, éste diagrama no ilustra conceptos del mundo real; en su lugar, exhibe clases de software.

Históricamente, el paradigma objetual representó un gran avance para la ingeniería de software, pues su abstracción principal invitaba a todos los

desarrolladores a que dejaran de pensar en los conceptos técnicos de la programación tales como variables, funciones, estructuras de datos entre otros, y en su lugar se enfocaran en "pensar en objetos" del mundo real, que luego podrían ser traducidos al código de un lenguaje de programación orientado a objetos como Java o C++.

Sin embargo, la experiencia demostró que el paradigma objetual para el desarrollo de software no siempre iba a permitir la representación del mundo real en términos de objetos y la implementación en código de éstos. Como expresa (Larman, Octubre 2004), "Los diseños y programas de objetos de software toman algo de inspiración de los dominios del mundo real, pero no son modelos directos o simulaciones del mundo real". Es por esto que éste mismo autor ha considerado que la disciplina de ADOO es un "arte", en el cual un buen trabajo solo puede ser realizado en la medida en que el analista/diseñador cuente con el grado de experiencia e intuición necesarios.

Adicionalmente, la definición de un modelo de dominio es el único artefacto del que se vale ADOO para entender el contexto que rodea el problema, el cual es insuficiente pues sólo ilustra los objetos y las características de cada uno sin tener en cuenta el contexto del problema, la organización. De esta forma, a pesar de que las metodologías de ADOO proponen unos pasos claros para ir desde la especificación del problema hasta la implementación de la solución en código, fallan a la hora de tener en cuenta el negocio y el valor agregado que se busca. Esto puede conllevar al desarrollo de aplicaciones inadecuadas que no generan un ROI (retorno de la inversión) para el negocio.

### 2.2.4 Un Cambio de Paradigma Requerido

De ésta forma, se hacen necesarios nuevos paradigmas, nuevas metodologías y tecnologías que posibiliten el desarrollo de aplicaciones de software empresariales

desde una perspectiva más amplia, que no se enfoque sólo en el problema a mano expresado en los requisitos y casos de uso, sino que sea consciente del amplio contexto del negocio. Este paradigma, más que preparar a los desarrolladores para pensar en objetos del mundo real, debe situarlos en un ambiente en donde sea posible una visión integral del negocio, y en lugar de operar intuitivamente bajo constante incertidumbre, se tenga toda la información requerida para construir aplicaciones que más que responder a un problema aislado, hacen parte de una verdadera plataforma completa de apoyo al negocio.
Algunas propuestas como EUP (Enterprise Unified Process) (Ambler, 2002) sugieren la inclusión más temprana de las actividades de modelado de negocio en los procesos de desarrollo de software, a fin de cerrar la brecha y obtener la visión requerida.

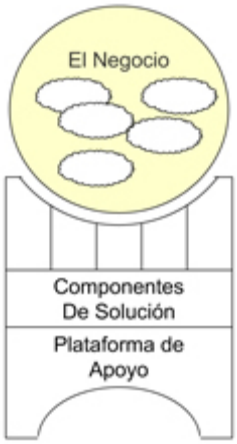


**Figura 3.** Un nuevo paradigma de vista integral al negocio.

En la figura 2 vemos una representación metafórica de la situación ideal. Una plataforma de apoyo que soporta componentes de solución, los cuales dan apoyo al negocio en su totalidad, y no a problemas específicos aislados. Para lograr esta visión y esta conciencia integral de la organización, es necesario recurrir a técnicas que nos brinden un modelo del negocio, una representación que ilustre los diferentes aspectos necesarios de la organización para la identificación de

posibilidades de valor agregado, la reducción de la incertidumbre y que además actúen como artefactos primarios que guíen el desarrollo de aplicaciones de software empresariales. En el siguiente capítulo se realiza un estudio comparativo de diversas técnicas de modelado, a la luz de diversos criterios, incluyendo los mencionados en esta sección y los requeridos para el cumplimiento de los objetivos de éste proyecto.

### 2.2.5 Hacia Una Arquitectura Orientada a Servicios (SOA[12])

La Arquitectura Orientada a Servicios (SOA) es una estrategia para la especificación, organización e implementación de activos de tecnologías de información (como aplicaciones de software, bases de datos y sistemas legados) en las organizaciones. Toma como principio fundamental el bajo acoplamiento; esto es, dos entidades se relacionarán entre sí solo mediante una interfaz de comunicación bien definida, garantizando así la flexibilidad requerida para las aplicaciones en las organizaciones, permitiendo así su reuso, fácil integración y fácil evolución a través del tiempo, en la medida en que el negocio cambie (Bea Systems, Inc., 2006) (Datz, 2004).

SOA logra su cometido a través del concepto de servicio. Este enfoque arquitectónico permite el apoyo de servicios de negocio a través de la creación de servicios de software o la exposición de éstos desde aplicaciones existentes. Un servicio de software es un componente que realiza una función distintiva que hace parte de un proceso de negocio (como por el ejemplo el servicio de recuperación de información de clientes de una base de datos) a través de una interfaz bien definida (una descripción electrónica de cómo llamar el servicio desde otros servicios) (WebMethods, Diciembre 2005). De ésta forma, los servicios colaboran

[12] *Service Oriented Architecture*, Arquitectura Orientada a Servicios

entre sí con contratos de responsabilidad bien definidos, gracias al principio de bajo acoplamiento.

Para la construcción de una SOA, es necesario un *stack* de tecnologías que brinden la interoperabilidad y permitan cumplir el requisito de bajo acoplamiento. Hoy en día, la tecnología más popular en este dominio es *Web Services*, un conjunto de estándares comprendido de XML[13], WSDL[14], SOAP[15] y UDDI[16].

### 2.2.6 La Relación de SOA con BPM

SOA constituye el complemento tecnológico por excelencia de la Gestión de Procesos de Negocio (BPM), debido a que proporciona la estrategia de desarrollo para cada fase del ciclo BPM (ver el capítulo "Los Procesos de Negocio y su Modelado"):

- **Diseño de Procesos:** En la construcción de un modelo de procesos de negocio, las actividades claves que lo conforman se identifican como servicios de negocio. De ésta forma, se crean servicios de software en alguna plataforma o se descubren y exponen servicios de aplicaciones de software existentes, según el enfoque de desarrollo (*top-down* o *bottom-up*).
- **Automatización de Procesos:** Una vez los servicios de software han sido creados o expuestos, éstos se despliegan y se publican en una plataforma de ejecución (por ejemplo, un contenedor de *Web Services*), en donde podrán ser accedidos por las entidades de orquestación que controlan la lógica de ejecución del proceso de negocio.

---

[13] *eXtensible Markup Language*, Lenguaje de Marcado Extensible
[14] *Web Services Description Language,* Lenguaje de Descripción de *Web Services*
[15] *Service Oriented Architecture Protocol*, Protocolo de Arquitectura Orientada a Servicio
[16] *Universal Description, Discovery and Integration*, Descripción, Descubrimiento e Integración Universales.

- **Monitoreo de Procesos:** Los servicios de software pueden proporcionar variedad de información, ya sea de resultados obtenidos en cada ejecución del proceso o indicadores de desempeño de infraestructura (tiempo, disponibilidad, etc.)

Esta comunión garantiza una alineación natural de los activos de TI con los procesos de negocio de la organización, convirtiendo así la arquitectura orientada a servicios en la verdadera plataforma de soporte integral a la creación de valor en el negocio. Como se expresa en los objetivos de éste proyecto, nuestro fin es propiciar esta unión utilizando para ello los métodos de la Ingeniería de Modelos.

## 2.3 La Ingeniería de Modelos

Desde los inicios mismos de la computación, la industria del software ha procurado crear abstracciones en las tecnologías para ocultar detalles irrelevantes a los usuarios y a los desarrolladores de soluciones. Un ejemplo de ello se puede evidenciar en la aparición de Fortran, el primer lenguaje de programación, con el cual los desarrolladores no tuvieron que enfrentarse directamente al código ensamblador de máquina, y por lo tanto haciendo de la labor de programación una tarea más accesible para un mayor número de personas. Igualmente, los sistemas operativos han sobrepuesto una capa de abstracción sobre los diferentes dispositivos de hardware que componen un computador, coordinando su funcionamiento para que el usuario pueda enfocarse en las tareas que le son verdaderamente relevantes (Schmidt, 2006).

Desde la aparición de la disciplina de la Ingeniería de Software en la década de los 60, diferentes paradigmas han surgido, cada uno de los cuales ha propuesto un conjunto de métodos, metodologías, técnicas y herramientas para el desarrollo de aplicaciones de software. Tal vez el más significativo de ellos ha sido la orientación por objetos, en la que se le incitaba al desarrollador a pensar que "todo

es objetos", un paradigma que aportó simplicidad al desarrollo de software, basándose en la modularización de las aplicaciones de software como medio para abordar su creciente complejidad, además de promover la reutilización de artefactos construidos y facilitar la mantenibilidad de las aplicaciones desarrolladas. Durante este periodo, la construcción de modelos fue una actividad que inició como un apoyo documental al proceso de desarrollo de software, en la que a través de diagramas se plasmaban los problemas identificados y las soluciones a implementar.

Posteriormente, con el surgimiento de las herramientas CASE[17], los modelos cobraron más importancia al convertirse en artefactos de entrada y salida para las operaciones de desarrollo de software. Estos permitían gestionar la trazabilidad de forma más natural desde el espacio del problema al espacio de la solución, para asegurar que los artefactos desarrollados estuvieran asociados y fueran consistentes en las diferentes fases del proceso. A partir de éste momento, la actividad de modelar se vuelve más estricta: se hace necesario construir modelos bien formados conformes a cierta especificación.

Luego surgieron las herramientas de transformación de modelos, que dieron pasó a una nueva posibilidad: la utilización de los modelos como artilugios fundamentales para el desarrollo de aplicaciones de software, que una vez más proponen un paradigma para abordar la creciente complejidad de los dominios de problemas enfrentados, así como del desarrollo de las soluciones. De ésta forma, se hace posible la automatización de la trazabilidad en el proceso de desarrollo de software. Así nace la Ingeniería de Modelos, en donde somos invitados a pensar que "todo es un modelo" (Bezivin, Abril 2004). Como consecuencia, marcos de trabajo han ido surgiendo en el frente de estándares con propuestas solidas como

[17] *Computer Aided Software Engineering*

MDA[18] de OMG (OMG, 2003), *Software Factories* (Greenfield, et al., 2004) o MDSD[19] (Völter, et al., 2006), y de herramientas como EMF[20] o *Microsoft Visual Studio Team System* (Microsoft, 2008) en donde los modelos toman el rol del artefacto principal de desarrollo (Bezivin, 2003).

### 2.3.1 Los Fundamentos de la Ingeniería de Modelos

Uno de los referentes bibliográficos con mayor relevancia alrededor de ingeniería de modelos es (Booch, et al., 2004), al tratarse de una publicación con miras a educar en la naciente propuesta de OMG, la Arquitectura Dirigida por Modelos (MDA). En ella, se definieron los tres fundamentos básicos del estándar, que mas allá de MDA, hoy son aplicables a todo el panorama de la ingeniería de modelos. Ellos dan cuenta de la justificación de la disciplina, así como de las ideas que rigen todos sus esfuerzos.

- Representación Directa: Los modelos deben proporcionan una representación estrictamente libre de detalles innecesarios para el modelador. Deben ser construidos dentro de un nivel de abstracción bien delimitado, como se define más abajo en la definición de niveles de abstracción de MDA.
- Automatización: La ingeniería de modelos busca la automatización de las actividades del proceso de desarrollo de software mediante la utilización de modelos, mediante técnicas como validación, transformación y visualización de modelos. Para lograrlo, éstos deben ser *bien formados*, conforme a una especificación estándar.
- Estándares Abiertos: La interoperabilidad y la amplia adopción de la disciplina son dos factores críticos de éxito para la ingeniería de modelos. Por ello, no es factible que las especificaciones de las distintas técnicas que

[18] *Model-Driven Architecture*
[19] *Model-Driven Software Development*
[20] *Eclipse Modeling Framework*

componen la disciplina sean definidas por entes propietarios, sino que en su lugar se conformen cuerpos de estándares (tales como OMG, W3C, OASIS, WFMC, OSOA, entre otros) para definirlos de manera abierta y por consenso.

### 2.3.2 Principios de la Ingeniería de Modelos: Modelo y Metamodelo

En (Bezivin, 2003) se puede encontrar un estudio completo desde una perspectiva general, así como una contextualización del uso histórico del modelado, dando cuenta así de que "todo es un modelo". Para nuestro propósito, basta con rescatar que un **modelo** es una representación abstracta de la realidad, que ilustra ciertos aspectos de relevancia para un determinado fin, y oculta los que no lo son. Por su lado, un **metamodelo** es una especificación formal de una abstracción comúnmente aceptada. Así, decimos que todos los modelos de un tipo específico conforman a cierto metamodelo.

Un ejemplo muy claro de estos principios lo podemos hallar en la cartografía. Un territorio geográfico puede estar *representado* por un mapa; existen diversos tipos de mapas, cada uno de los cuales muestra solo las características que le interesan al usuario de ese tipo específico de mapa. Por ejemplo, un mapa político básicamente muestra las divisiones políticas entre regiones, mientras que un mapa físico se ocupa de exhibir los accidentes geográficos de un territorio. En ambos casos, el tipo de mapa define unos lineamientos de acuerdo a los cuales el mapa es construido, para ocultar las características del territorio geográfico que no son relevantes e impidiendo así la saturación desde el punto de vista del usuario. Estos lineamientos, que comúnmente se expresan en forma de la leyenda del mapa, es lo que corresponde al *metamodelo*. Un mapa político es *conforme a* la leyenda del mismo que especifica el trazado de fronteras políticas, así como un mapa físico es conforme a la leyenda que define los elementos geofísicos a ser incluidos en el mapa. *Un metamodelo es un filtro precisamente definido expresado*

*en algún formalismo.* A su vez, los metamodelos de diferentes tipos pueden ser definidos conforme a un arquetipo específico, es decir, un *metametamodelo*.

### 2.3.3 Principios de la Ingeniería de Modelos: Transformación de Modelos

La transformación de modelos no es un concepto enteramente nuevo, pues las ciencias de la computación desde el principio han estudiado la generación de elementos de salida a partir de elementos de entrada, que normalmente corresponden a representaciones del mundo real. Este es el caso, por ejemplo, del procesamiento de datos, o de compilación de programas. Sin embargo, en la transformación de los modelos comúnmente no se utilizan conceptos formales como reescritura de términos, gramáticas de atributos y programación funcional, sino que se adopta un enfoque orientado a objetos para representar y manipular modelos, que en sí mismo es una estructura de clasificadores que proporciona una abstracción de un sistema o del contexto de éste. Las transformaciones pueden operar sobre todo tipo de modelos, como modelos UML, especificaciones de interfaces, esquemas de datos, descriptores de componentes, y código de programas. Esto lleva a que existan diferentes tipos de transformación.

Entre las diversas aplicaciones que tiene la transformación dentro de la ingeniería de modelos, (Czarnecki, et al., 2006) señala:

- La generación de modelos de bajo nivel de abstracción e incluso de código, a partir de modelos de alto nivel de abstracción.
- El mapeo y la sincronización entre modelos en el mismo nivel o diferentes niveles de abstracción.
- La creación de vistas del sistema basadas en criterios específicos.
- Automatización de tareas concernientes a la evolución de modelos, como refactorización de modelos.
- Ingeniería reversa de modelos de alto nivel desde modelos de bajo nivel o código.

En (Bezivin, 2003) y (Bezivin, Abril 2004) se encuentra lo que podría llamarse el arquetipo de la transformación de modelos, vislumbrado en la siguiente gráfica.

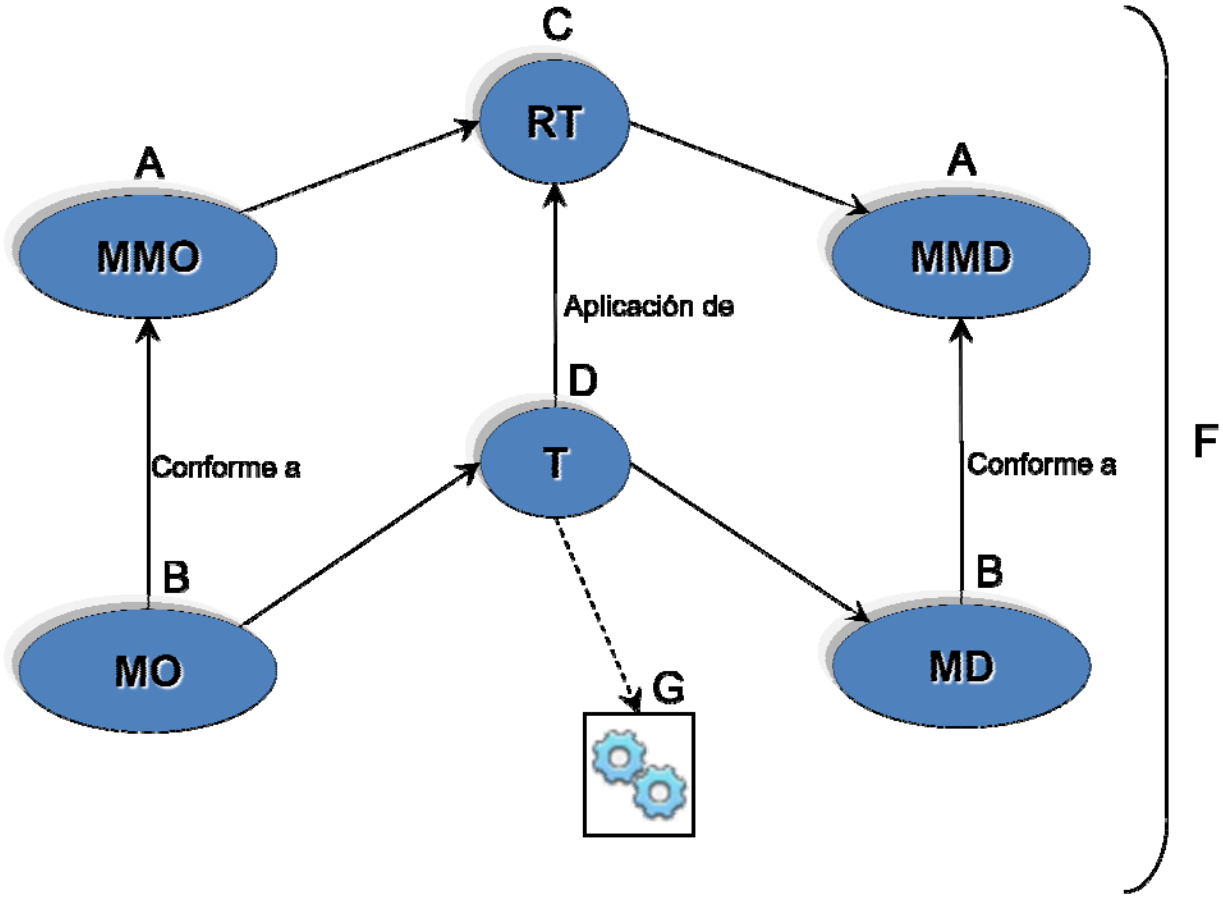


**Figura 4.** Arquetipo de la Transformación de Modelos

Aquí identificamos los siguientes componentes:

A. Editor de Metamodelos. Aquí se define la sintaxis abstracta del tipo de modelos en la transformación, tanto de origen (MMO, metamodelo de origen) como de destino (MMD, metamodelo de destino).
B. Editor de Modelos. Usualmente editores gráficos, en donde el usuario construye modelos de origen (MO) y de destino (MD) que son conformes a sus respectivos metamodelos.

C. Editor de Reglas de Transformación (RT). En donde el usuario define explícitamente las reglas de mapeo de los elementos de origen hacia los de destino.
D. Ejecutor de Transformaciones. El usuario ejecuta la transformación (T) del modelo de origen, de acuerdo con la estrategia de transformación utilizada.
E. Consola de Transformaciones. El usuario puede realizar seguimiento de sus procesos de transformación, así como la definición de nuevas transformaciones y la administración de modelos y metamodelos.
F. Motor de Transformación. Plataforma encargada de la interpretación de las reglas de transformación y la creación de modelos de salida.

### 2.3.4 Acercamientos a la Ingeniería de Modelos: MDA

OMG[21], la institución creadora de importantes estándares en la industria de software tales como UML y CORBA[22], ha propuesto MDA (*Model Driven Architecture*, Arquitectura Dirigida por Modelos) (OMG, 2003), con el propósito de separar la especificación de un sistema de los detalles de su implementación y la utilización de las capacidades de su plataforma. En la actualidad es tal vez el acercamiento más reconocido hacia la ingeniería de modelos, al proveer tres pilares requeridos para la naciente disciplina de Ingeniería de Modelos: Representación directa, automatización y estándares abiertos (Booch, et al., 2004).

MDA define un **modelo de un sistema** como "una descripción o especificación de un sistema y su entorno para cierto propósito. Un modelo es a menudo presentado como una combinación de dibujos y texto. El texto puede estar en un lenguaje de modelado o en lenguaje natural" (OMG, 2003). Un ejemplo de estos lenguajes de modelado gráfico es UML, creado para el análisis, diseño y especificación de

[21] *Object Management Group*
[22] *Common Object Request Broker Architecture*

sistemas. Un modelo se encuentra siempre enmarcado en cierto nivel de abstracción, según se han delimitado en la especificación de MDA:

- Independencia de la Computación
- Independencia de la Plataforma
- Específico de la plataforma

Correspondiente a estos niveles de abstracción, MDA define el concepto de **punto de vista de un sistema** como "una técnica para la abstracción usando un conjunto selecto de conceptos de arquitectura y reglas estructurales, con el propósito de enfocarse en aspectos particulares al interior de ese sistema". Con base en este concepto, MDA constituye una arquitectura en la que especifica cierto tipo de modelos para ser utilizados y la relación entre ellos. Se proponen tres perspectivas (OMG, 2003):

- **Modelo independiente de la computación (CIM):** Es una representación del dominio del problema para el cual se construye el sistema, independiente de cualquier implementación en tecnologías de información.
- **Modelo independiente de la plataforma (PIM):** Es una representación de la solución a construir con un grado de independencia de la plataforma, con el fin de que la aplicación pueda ser desplegada en varias plataformas del mismo tipo.
- **Modelo específico de la plataforma (PSM):** Es un nivel más detallado del PIM, indicando los aspectos específicos para la implementación en una plataforma en particular (Java EE, Microsoft .NET, etc.)

MDA propone un modelo en cascada en el que el paso desde uno de estos niveles de abstracción al siguiente esta dado por un proceso de **transformación**. La transformación de modelos juega un importante papel en el desarrollo dirigido por modelos. Algunas de sus aplicaciones son la generación de modelos de bajo nivel, y eventualmente de código, a partir de modelos de alto nivel (como puede

evidenciarse en la siguiente figura), el mapeo y la sincronización entre modelos al mismo nivel o entre diferentes niveles de abstracción, la creación de vistas basadas en consultas de aspectos específicos, la modelación de procesos evolutivos de los sistemas como la refactorización desde modelos de bajo nivel o código (Czarnecki, et al., 2006).

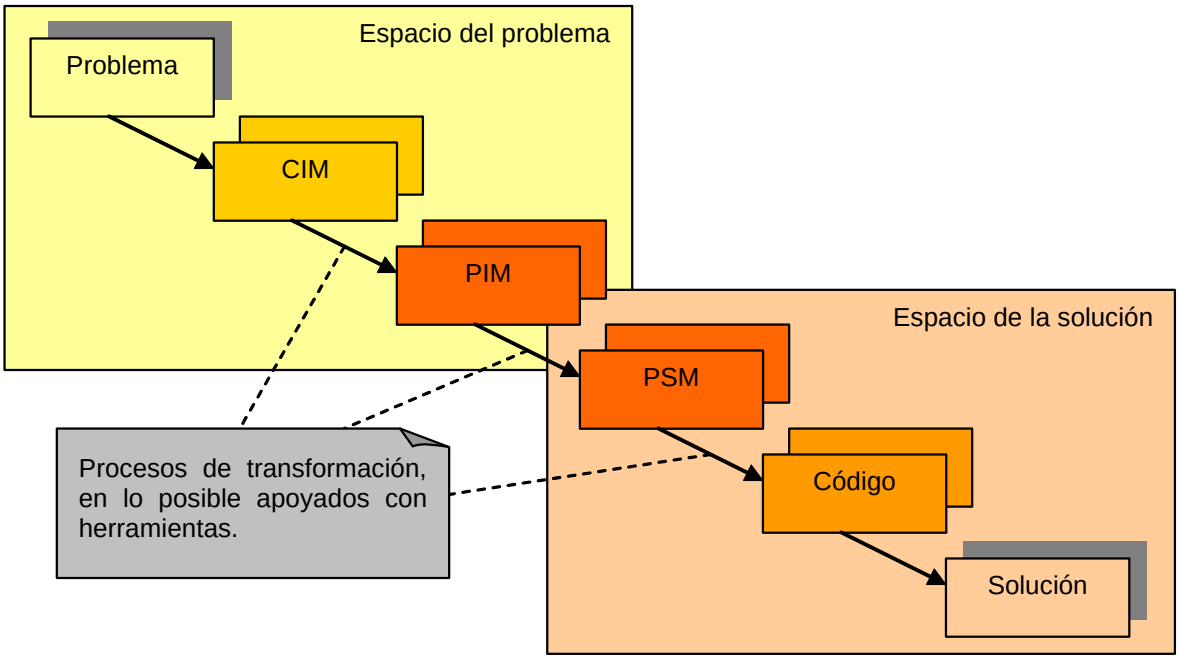


**Figura 5.** La transformación en el proceso basado en MDA **(Quintero, et al., 2007)**

La transformación de modelos es un proceso que se efectúa conforme a una **estrategia de mapeo**, o simplemente "mapeo", que especifica las reglas que asocian los elementos de un modelo origen con los elementos de un modelo destino. Estas reglas son elaboradas usando los metamodelos de origen y destino, posiblemente conformes a la especificación MOF. Además de estas reglas y metamodelos, la transformación toma como insumo de entrada el modelo origen *marcado*, es decir, con la información adicional detallando como se transformarán cada uno de sus elementos en el modelo destino. Las marcas pueden ser pensadas como una capa transparente que se le impone al modelo origen.

### 2.3.5 Meta-Object Facility

El estándar **MOF** (Meta-object Facility) de OMG es una de las especificaciones de meta-metamodelo mas utilizadas en ingeniería de modelos para la definición y utilización de metamodelos (OMG, 2001). MOF, en su versión actual 2.0, provee un framework de gestión de metadatos pera permitir el desarrollo dirigido por modelos. Ha contribuido significativamente al cumplimiento de los principios de la ingeniería de modelos, al proveer las líneas guía para la especificación de metamodelos, y así permitir la interoperabilidad y la transformación entre distintos tipos de modelos. Construido sobre las bases de UML, MOF introdujo el concepto de metamodelos formales y de mapeos entre éstos ubicados en diferentes niveles de abstracción. Para facilitar su adopción por parte de las herramientas y las técnicas de modelado, MOF ha sido definido en dos conjuntos: EMOF (Essential MOF), como punto de entrada, y CMOF (Complete MOF) para una conformidad más estricta al estándar.

A la fecha, la implementación de referencia de MOF por excelencia es **Ecore**, un metametamodelo autodescriptivo que toma un subconjunto de EMOF y hace parte de EMF (Eclipse Modeling Framework (Merks, Agosto, 2004.)). MDR (Metadata Repository) de Netbeans, es otra implementación de MOF.

### 2.3.6 MOF 2.0 Query/View/Transform

La especificación **QVT** de OMG para la transformación de modelos (OMG, 2007) define un estándar de naturaleza dual. En primer lugar es declarativo, centrándose en el “qué” y no en el “cómo” pues describe los elementos estructurales que componen una transformación, como los metamodelos (basados en MOF) y las relaciones de mapeo entre elementos de modelado. Y en segundo lugar, la especificación también se refiere al aspecto imperativo, centrándose en el “cómo” y no en el “qué”, en donde se define un lenguaje operacional para la construcción

de las propias reglas de transformación. Una de las implementaciones más populares de ésta especificación es ATL (Atlas Transformation Language), uno de los mecanismos para transformaciones modelo a modelo (M2M) de la comunidad de Eclipse, el cual permite la definición de reglas de mapeo entre metamodelos definidos en Ecore (Eclipse M2M, 2008). En 2007, Eclipse agregó una nueva propuesta por parte de Borland como implementación para el lenguaje operacional llamado Operational QVT (Eclipse M2M, 2008)

### 2.3.7 Otros Acercamientos a la Ingeniería de Modelos: Software Factories

En (Greenfield, et al., 2004) se hace una examinación de los problemas que han caracterizado históricamente la industria de software, como lo son la creciente complejidad de los problemas a enfrentar y el cambio en las organizaciones, un fenómeno cada vez más frecuente. Con el inminente paso de arquitecturas multi-capa a arquitecturas orientadas a servicios para el apoyo a procesos de negocio, el desarrollo de sistemas se hará cada vez más caro y complejo; de esta forma, la productividad se irá volviendo cada vez más un factor crítico.

El enfoque de estos autores se basa en el concepto de factorías de software. Una factoría de software es una configuración de cuatro componentes:

- **Frameworks**, que se desarrollan para posibilitar el desarrollo de productos basados en un estilo arquitectónico común por medio de la adaptación, configuración y en ensamble de componentes de framework.
- **Lenguajes**, que se definen para dar apoyo al proceso de ensamblaje.
- **Herramientas**, ayudan a los desarrolladores a comprometerse con los clientes, respondiendo rápidamente a cambios y desarrollando incrementalmente, a la vez que se automatizan tareas comunes para obtener productividad.

- **Patrones**, que proveen soluciones probadas a problemas frecuentes y comunes.

Con este enfoque, claman los autores, es posible construir productos de forma rápida y con bajo costo, dándole la bienvenida al cambio, en lugar de ser resistentes a él.

### 2.3.8 Otros Acercamientos a la Ingeniería de Modelos: Model-Driven Software Development

MDSD, un enfoque propuesto por Markus Völter y Thomas Stahl, se construye sobre la fundación de MDA atendiendo sus debilidades percibidas (Völter, et al., 2006). En lugar de requerir una estricta formalización en la construcción de modelos y la adherencia a especificaciones y metamodelos, se busca que los modelos sean abstractos y formales al mismo tiempo, de manera que su uso brinde una verdadera abstracción sobre la tecnología y enfocada sólo en el dominio del problema; el objetivo es lograr la *riqueza semántica*. Además de los modelos, MDSD se apoya en las plataformas de dominio específico, esto es, componentes y framework reusables y prefabricados que ofrecen poderosas bases que nativamente no son soportadas por lenguajes o plataformas técnicas como J2EE. Así, los motores de transformación o generadores se ocupan de mapear los modelos construidos en artefactos basados en éstas plataformas, que brindan APIs (Interfaces de Programación de Aplicaciones) de alta calidad para posibilitar el reuso a gran escala.

### 2.3.9 Aclaración acerca del uso de términos

En éste estudio comparativo se ha realizado la exploración de diversos conceptos relacionados como "MDA", "MDSD" e "Ingeniería de Modelos", y se han mostrado las diferencias entre sí. Sin embargo, a lo largo de éste trabajo dichos términos se

usan indistintamente para referirse a la disciplina que toma los modelos como artefactos centrales del proceso de desarrollo de software.

# 3. ESTUDIO COMPARATIVO DE TECNICAS DE MODELADO DE NEGOCIO

En el proceso de evolución de la gestión de procesos de negocio han surgido diferentes técnicas para el modelado de negocio. En este capítulo se realiza un estudio comparativo de 5 técnicas de modelado, usando criterios de interés para analistas de negocio, arquitectos de software, entre otros.

Históricamente, el aporte académico a la ingeniería de software se ha caracterizado por realizar un gran énfasis en las disciplinas de ingeniería de requisitos, análisis y diseño, pero pocas propuestas se han construido alrededor de actividades más tempranas desde una perspectiva independiente de la computación, como el modelado de negocio.

Este trabajo de investigación toma como antecedentes la taxonomías y comparaciones de técnicas presentadas en (Giaglis, 2001 ) y (Mendling, et al., 2004) en donde se evalúan y se comparan las propuestas de modelado de negocio más usadas. Algunas de éstas han servido como punto de partida en la elaboración reciente de estándares de modelado de negocio tales como BPMN (OMG, 2006).

El propósito de éste estudio comparativo dentro del contexto del proyecto corresponde a la necesidad de encontrar un referente industrial para la elaboración de modelos independientes de la plataforma (CIM) ya que al parecer no existe un consenso en la actualidad acerca de cómo construir estas representaciones del espacio del problema (el negocio).

Es importante notar que a diferencia de lo estipulado en los objetivos del proyecto, el estudio comparativo se centra en técnicas de modelado mas no en ambientes de ejecución de procesos de negocio, debido a que nuestra propuesta como tal

presenta una alternativa para la automatización y ejecución. De todas maneras, éstos han sido tenidos en cuenta en los criterios de evaluación del estudio.

## 3.1 Caso de Estudio

Con el fin de ilustrar las diferentes técnicas de modelado de negocio, proponemos el siguiente escenario. Una empresa de servicios tiene un proceso de Quejas y Reclamos que hace parte fundamental de su cadena de valor, cuyo objetivo de negocio es asegurar la satisfacción de sus clientes, según se incluye en las políticas organizacionales. Los principales actores identificados en este proceso son los siguientes:

- Cliente inconforme: Una persona (o empresa, o persona representante de una empresa) a la cual la empresa le ha prestado un servicio, y ésta quiere manifestar una queja o reclamo a la compañía.
- Usuario interno: Cualquier empleado de la compañía que haya recibido una notificación de inconformidad por parte del cliente, ya sea personalmente, telefónicamente, a través de correo electrónico o algún otro medio.
- Gerente de cuentas corporativas: Es el encargado principal en la compañía de gestionar la relación con los clientes, y es el responsable de velar por su satisfacción. En el evento de una queja o reclamo, debe asignar un responsable y asegurar la efectividad de la solución.
- Encargado de la solución: Persona encargada de implementar una solución a la queja presentada.

## 3.2 Técnicas de Modelado de Negocio Evaluadas

La siguiente tabla presenta las técnicas de modelado de negocio que por su relevancia y popularidad fueron consideradas para este trabajo evaluativo.

| Ref. | Nombre | Año | Dirección Sitio |
|---|---|---|---|
| (Eriksson, et al., 2000) | **EyP**: Extensiones de Ericsson y Penker para UML | 2000 | http://wiley.com/compbooks/catalog/29551-5.htm |
| (OMG, 2006) | **BPMN**: Notación para el Modelado de Procesos de Negocio | 2004 | http://www.bpmn.org |
| (Sprint Framework, 2006) | **RAD**: Diagramas de Actividades de Roles | 1995 | http://www.sprint.gov.uk/pages.asp?id=77 |
| (Mayer, Septiembre 1995) | **IDEF3**: Captura de Descripción de Procesos | 1989 | http://www.idef.com |
| (Ortín, et al., 2000) | **UMurcia**: Propuesta Universidad de Murcia | 2000 | http://www.lsi.us.es/~amador/JIRA/Ponencias/JIRA_Ortin.pdf |

**Tabla 2:** Técnicas de modelado de negocio estudiadas.

### 3.2.1 Extensiones de Ericsson y Penker para UML

La motivación de ésta técnica se encuentra en la problemática alrededor de las actividades de análisis que tradicionalmente se utilizan en las etapas tempranas del desarrollo de software, que no siempre presentan una adecuada alineación al negocio que le traerá beneficios a éste último. Concretamente, se refieren de la siguiente forma (Eriksson, et al., 2000):

> "Muchos aficionados a UML predican que el ciclo de desarrollo debe iniciar con el modelado de casos de uso para definir los requerimientos funcionales de un sistema y el uso específico por uno o más actores. Las descripciones de casos de uso son usadas para analizar y diseñar una arquitectura robusta del sistema que realice los casos de uso (Esto es lo que se conoce como desarrollo 'dirigido por casos de uso'). Pero como saber que todos los casos de uso o incluso los casos de uso correctos que mejor apoyan el negocio en el cual el sistema opera están identificados, para dar respuesta a estas inquietudes es necesario modelar y entender lo que rodea al sistema.
>
> Modelar el contexto de un sistema involucra responder preguntas tales como:
>
> - Cómo interactúan los diferentes actores.
> - Qué actividades hacen parte del trabajo de cada actor.

- Cuáles son los objetivos de su trabajo.
- Qué otras personas, sistemas o recursos están involucrados y no aparecen como actores en éste sistema específico.
- Qué reglas guían y gobiernan actividades y estructuras.
- Existen formas para que los actores se desempeñen más eficientemente sus actividades”

Aunque UML fue creado para especificar la arquitectura y el diseño de un sistema, éste presenta la característica de la extensibilidad para crear aditamentos para cubrir otras necesidades de modelado. La propuesta de (Eriksson, et al., 2000) sugiere el uso del diagrama de actividades, pues de acuerdo a la definición dada en (Davenport, 1993), un proceso de negocio es esencialmente una secuencia de actividades. Para este diagrama, se propone un conjunto de extensiones para el modelado de negocio, a través de los siguientes elementos de modelado:

- Procesos: Describen como se realiza el trabajo y se transforman los recursos, para alcanzar los objetivos y están determinados por las reglas.
- Recursos: Personas, materiales, información y productos que son usados o producidos en el negocio.
- Reglas: Una declaración que define o restringe algún aspecto del negocio. Se representan a través de OCL (Object Constraint Language) (OMG, 2003).
- Metas: El propósito del negocio que se desea alcanzar. Se asocian con un proceso u objeto en particular.

Adicionalmente, un arquitecto de negocio puede definir un conjunto de estereotipos y propiedades para su dominio particular.

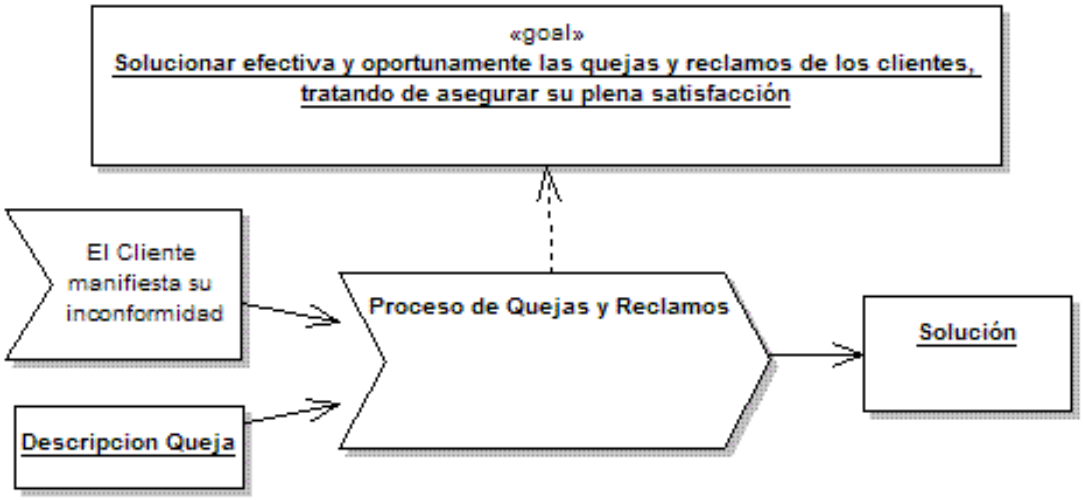


**Figura 6.** Proceso de quejas y reclamos modelado con la técnica Extensiones de Ericsson y Penker para UML

### 3.2.2 Modelado de Negocio con BPMN

El organismo BPMI propuso el estándar BPMN (OMG, 2006), una notación gráfica altamente intuitiva que le permite a los individuos con mayor conocimiento sobre el negocio construir modelos sin requerir conocimiento alguno de estándares o mecanismos formales involucrados en las soluciones tecnológicas para soportar la gestión de procesos de negocio.

BPMI realizó un estudio de las técnicas de modelado de negocio más usadas para su creación en 2002, afianzando así su credibilidad y facilitando su adopción como estándar por parte del público, entre ellas el diagrama de Actividades de UML (posiblemente incluyendo la propuesta de Ericsson y Penker), IDEF, los diagramas de Flujo de Actividades y Decisiones y Cadenas de Procesos de Eventos (EPC). La consolidación de éstas diversas técnicas condujo a la creación de la versión inicial de BPMN (Business Process Management Initiative, 2002).

La notación fue construida para ser instantáneamente entendible por todos los usuarios del negocio, desde los analistas que crean los bocetos iniciales de los

procesos, hasta los desarrolladores técnicos responsables de implementar la tecnología que conducirá a la realización de estos procesos, y finalmente, a los encargados del negocio de administrar y monitorear los procesos (OMG, 2006).

El estándar comprende solo un tipo de diagrama, el BPD (*business process diagram*), cuya notación se asemeja en gran medida a un diagrama de flujo tradicional. Cuenta con un número de elementos que permite modelar el curso de actividades a lo largo de la organización (y de los actores externos involucrados), divididos en cuatro categorías (Una descripción más detallada, incluyendo su significado y su representación gráfica, se puede encontrar en (OMG, 2006)):

- Objetos de flujo: Elementos básicos para representar el comportamiento de un proceso de negocio. Incluye actividades, eventos y puntos de decisión.
- Objetos de conexión: Elementos para conectar los objetos de flujo entre sí. Incluyen flujos de secuencia, flujos de mensajes y asociaciones.
- Carriles: Hay dos formas de agrupar los elementos primarios: A través de "calles" y "carriles".
- Artefactos: Elementos para proveer información adicional al modelo de procesos. Incluyen anotaciones, objetos de datos y grupos.

De esta forma, se permite la ilustración de la secuencia de actividades o tareas que se realizan al interior de una organización para alcanzar un determinado fin. Adicionalmente, a diferencia de otras técnicas, BPMN ofrece una "ruta directa a ejecución". Entre las características enunciadas en su más reciente especificación (OMG, 2006) se habla de mapeo a lenguajes de menor nivel de abstracción, como es el caso de BPEL4WS (*Business Process Execution Language for Web Services*) (OASIS WS-BPEL Technical Committee, 2007), haciendo posible de esta forma la *representación directa* a entornos de ejecución de procesos; adicionalmente se tiene la gran ventaja de estar concebido teniendo en cuenta los últimos estándares y aproximaciones basadas en el desarrollo de sistemas de

información organizacionales, como Arquitectura Orientada a Servicios (SOA), como se puede evidenciar en (Emig, et al., 2006) y (Emig, et al., 2007).

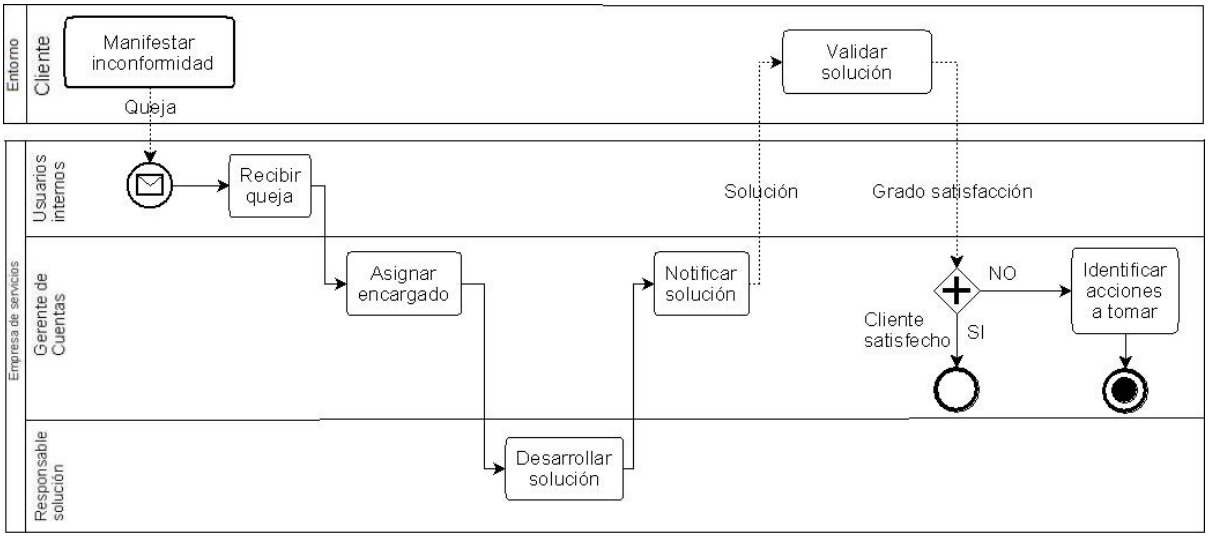


**Figura 7.** Proceso de quejas y reclamos modelado con la técnica BPMN

### 3.2.3 Modelado de Negocio con RADs (*Role Activity Diagrams*)

Los diagramas de actividades de roles toman como su elemento de modelado principal los individuos o grupos en una organización, y el rol que asumen dentro de un proceso, sus actividades de componentes y sus interacciones, junto con eventos externos y la lógica que determina cuales actividades se ejecutan y cuando. Esto es a diferencia de otros enfoques como BPMN o el diagrama de actividades de UML, que toman la actividad como elemento de modelado principal. Por esta razón, los RADs son más adecuados cuando el elemento humano es más crítico en las abstracciones a realizar del negocio (Sprint Framework, 2006).

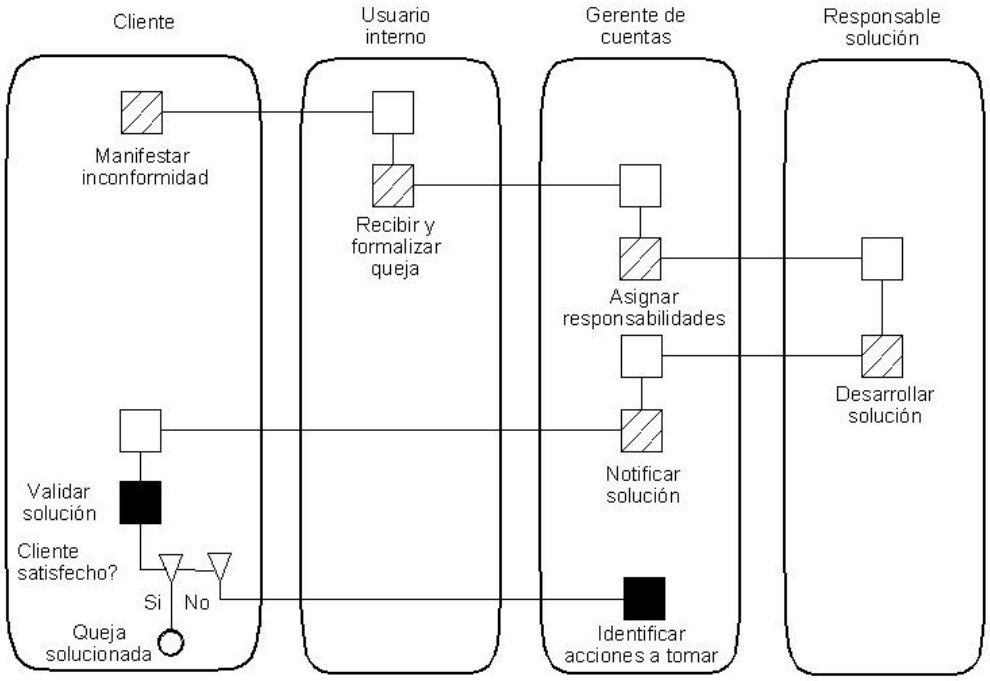


**Figura 8.** Proceso de quejas y reclamos modelado con la técnica diagrama de actividades de roles

Los diagramas de actividades de roles son dibujados con el flujo principal a la izquierda y con condiciones de elección y refinamientos posteriores a la derecha. Sus elementos de modelado son, entre otros:

- Roles: Un rol realiza un conjunto de actividades que juntas alcanzan un objetivo particular.
- Actividades: Los items de trabajo que las personas llevan a cabo.
- Ordenaciones: Actividades ordenadas por estado.
- Interacción: Puntos en los que un rol interactúa con otro para cumplir un objetivo.
- Elecciones: Condiciones bajo las cuales diferentes actividades toman lugar.

### 3.2.4 Modelado de Negocio con IDEF3: Captura de Descripción de Procesos

La familia IDEF de técnicas de modelado fue desarrollada como un conjunto de formalismos de notaciones para representar y modelar procesos y estructuras de datos de una forma integrada. La suite IDEF consiste de un número de técnicas independientes, de las cuales las más conocidas son IDEF0 (modelado de funciones), IDEF1x (modelado de datos) e IDEF3 (captura de descripción de procesos). Para el caso específico del modelado de negocio, las más relevantes son IDEF0 e IDEF3, siendo IDEF1x más adecuado en el campo de modelado de sistemas de información (Mayer, Septiembre 1995).

El método IDEF0 está diseñado para modelar las decisiones, acciones y actividades de una organización u otros sistemas y, como tal, es orientado principalmente hacia la perspectiva de modelado funcional. Como herramienta de comunicación, su fortaleza reside en que solo utiliza un elemento de modelado, el **ICOM (*Input-Control-Output-Mechanism*)** (Giaglis, 2001 ). El modelado de procesos es soportado descomponiendo ICOMs de alto nivel en niveles más detallados que muestran la descomposición jerárquica de las actividades. Sin embargo, la gran limitante de los modelos IDEF0 es que son estáticos, y no muestran el comportamiento a través del tiempo de las actividades que conforman los procesos.

Para remediar estas limitaciones, el estándar IDEF3 fue creado por la Fuerza Aérea de EEUU. IDEF3 describe procesos como secuencias ordenadas de eventos o actividades. Su objetivo es formular el *cómo*, mientras que IDEF0 se encargaba de mostrar el *qué* de las actividades de negocio.

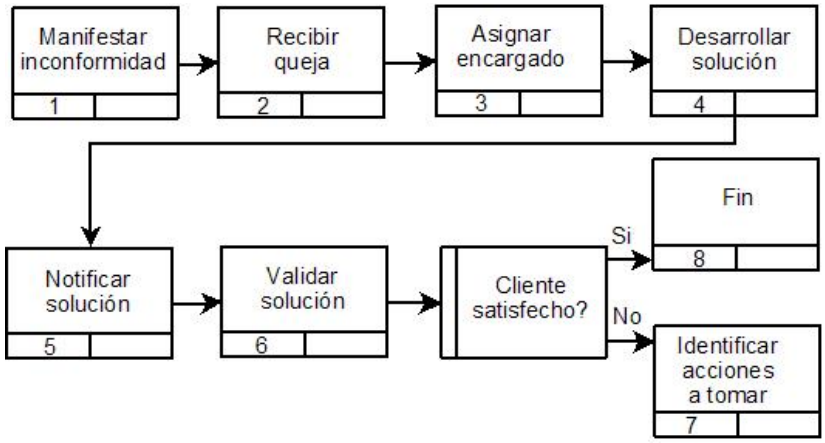


**Figura 9.** Proceso de quejas y reclamos modelado con la técnica IDEF3

### 3.2.5 Modelado de Negocio con UML: Propuesta Universidad de Murcia

En (Ortín, et al., 2000) se hace un análisis de las actividades de modelado en las etapas tempranas del proceso de desarrollo de software, y señala algunos problemas con los enfoques metodológicos actuales. Este estudio indica que utilizar el modelado de casos de uso como punto de partida, como suele hacerse comúnmente, sin tener antes una representación del dominio, puede conllevar a la identificación, especificación e implementación de funcionalidades no adecuadas para el dominio en el cual se construye la solución. Por lo tanto, se basan en trabajos que han demostrado la importancia de conocer el dominio como prerrequisito para modelar casos de uso, para lo cual es necesario construir un modelo de negocio.

La propuesta justifica la relevancia del *flujo de tareas* que compone el proceso de negocio, además considera importante incluir en el modelo otros elementos como los *datos* que son producidos y manipulados por las actividades del proceso, los *roles* que ejecutan las tareas del proceso y las *reglas de negocio* que gobiernan el comportamiento de los flujos. Se propone una metodología, en una serie de pasos, para construir el modelo del negocio.

El primer paso es la identificación de los procesos, a partir de los objetivos estratégicos de la organización. A menudo estos objetivos presentan un alto nivel de abstracción, por lo que puede ser necesario descomponerlos en objetivos más específicos. Cada uno de estos objetivos descompuestos, es modelado como un *caso de uso del negocio*. Aquí se ilustran los diferentes procesos como casos de uso, asociados con los actores responsables de su ejecución, ya sea al interior de la compañía (funcionarios en áreas de negocio) o al exterior de ella (proveedores y clientes).

El siguiente paso es realizar la descripción extendida de los casos de uso de negocio. Para esto se utiliza una plantilla en la que por cada caso de uso se especifica su objetivo (qué se intenta conseguir), descripción (especificación informal de lo que hace el proceso), prioridad, riesgos (por ejemplo errores o fallos que pueden ocurrir al ejecutar este proceso), posibilidades (cambios o mejoras futuras del proceso), tiempo y coste aproximados de ejecución.

Para ilustrar la interacción colaborativa de los diferentes roles en los procesos, la propuesta sugiere construir un modelo de roles mediante un diagrama de clases UML, en el cual cada rol es representado por una clase estereotipada. Así, se pueden expresar las relaciones de colaboración entre ellos mediante asociaciones, como también se pueden especifican atributos tales como el área de negocio correspondiente y sus responsabilidades.

Los flujos de cada proceso se ilustran a través de un diagrama UML de secuencias, donde los objetos denotan las instancias de los roles que interactúan en el proceso, y los mensajes son las actividades que lo componen. Sin embargo, una representación más detallada se realiza a través de un diagrama de actividades, el cual es la piedra angular de la propuesta. Este muestra la secuencia de actividades estructuradas según el rol que las realiza, para lo cual el diagrama hace uso de "calles" (*swim lanes*), y muestra la información requerida y

producida por cada actividad en la forma de objetos, llamados *objetos de información* (OMG, 2003).

Las reglas de negocio son especificadas en un glosario externo al modelo. Estas se especifican de acuerdo a los objetos de información y a las actividades que componen al proceso.

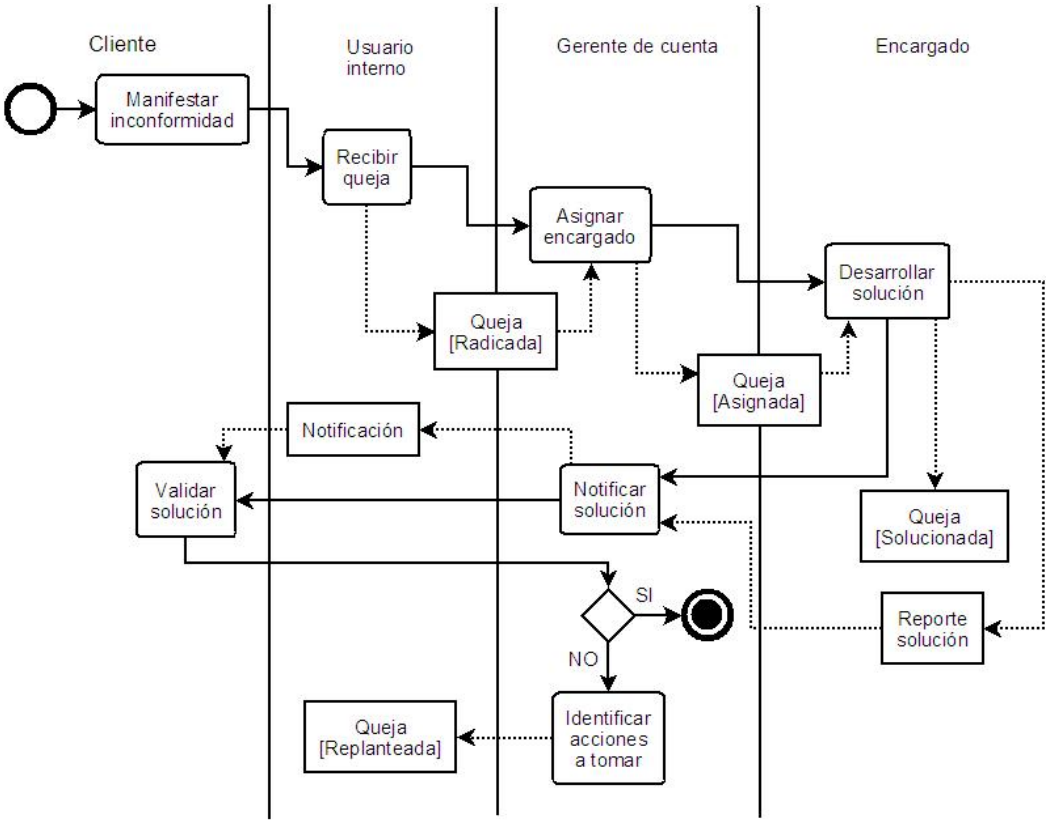


**Figura 10**. Proceso de quejas y reclamos modelado con la técnica de diagrama de actividades de UML de la Universidad de Murcia

## 3.3 Criterios de Comparación

Los criterios de comparación son definidos en tres categorías que constituyen los pilares de MDA propuestos en “An MDA Manifesto” (Booch, et al., 2004): (a) *representación directa* para enfocarse en el dominio del problema más que en la

tecnología, (b) *automatización* de las tareas mecánicas que no requieren intervención humana y (c) *estándares abiertos* que posibiliten la interoperabilidad de las herramientas y plataformas.

### 3.3.1 Representación Directa

Se refiere a reducir la distancia semántica entre el dominio del problema y su representación en un modelo, de manera que se permita un acoplamiento directo de las soluciones hacia los problemas para las cuales fueron construidas. Estos criterios evalúan cada técnica, en la medida en que redirija el foco de atención hacia las ideas y conceptos del dominio del problema, y lo aleje del dominio de soluciones de tecnología (Arsanjani, 2005).

- **R1. Adopción de CIM**: modelado desde un punto de vista independiente de la computación según lo definido en (OMG, 2003): Un CIM es un modelo de un sistema que muestra el sistema en el entorno en el cual operará, y como tal ayuda a presentar lo que se espera que el sistema realizará.
- **R2. Estructura y Comportamiento:** capacidad para representar las vistas estructural y de comportamiento de la organización a través de procesos de negocios.
- **R3. Reglas de Negocio:** Soporte para el modelado de reglas de negocio, debido a que estos son elementos altamente cambiantes en el devenir de los procesos de negocio, y los usuarios de los modelos deberían poder visualizarlos y manipularlos claramente (Debevoise, 2005).
- **R4. Roles:** Capacidad para representar los diferentes roles que ejecutan las diferentes funciones en los procesos de negocio. Es necesario, tanto para la administración y control de procesos, como para la implementación de aplicaciones de soporte a procesos, conocer las responsabilidades de cada una de las actividades.

- **R5. Objetivos y E/S:** Capacidad para representar objetivos de negocio, entradas y salidas de información en las actividades del proceso, ya sea en la forma de documentos (información estructurada) o mensajes (información sin estructurar).
- **R6. B2B:** Capacidad para representar interacciones negocio a negocio (B2B) a través de la cadena de abastecimiento, de manera que las organizaciones externas con las que se colabora asumen un rol externo dentro del proceso.
- **R7. Usabilidad:** Usabilidad por parte de stakeholders no técnicos, tales como analistas de negocio, administradores o diseñadores de procesos. Estas personas son quienes más conocen el negocio, y a menudo hace parte de los mismos procesos.

### 3.3.2 Automatización

En esta categoría se mide la capacidad de las técnicas de modelado de negocio para permitir la automatización de las tareas de desarrollo de software que son mecanizadas y no dependen de la intuición humana, de manera que se incremente la productividad y se reduzca el esfuerzo requerido. Este es uno de los propósitos fundamentales de la naciente disciplina de Ingeniería de Modelos (ver (Bezivin, 2003), (Bezivin, Abril 2004), y (Völter, et al., 2006)).

- **A1. Respaldo Metodológico:** Existencia de un respaldo metodológico para el modelado de negocio y la ejecución de procesos que proporcione una guía clara y concisa acerca de cómo construir un modelo que represente al negocio en todos sus aspectos relevantes.
- **A2. Brecha modelado-ejecución:** Brecha entre la técnica de modelado y la ejecución de procesos de negocio. Los modelos de negocio deben convertirse en activos que posibiliten la automatización de los procesos de negocio.

- **A3. Motor de ejecución:** Existencia de un ambiente de infraestructura (motor de ejecución) o estándar de ejecución de los procesos de negocio modelados. Los modelos de procesos deberían contar con un mapeo directo al entorno de producción en donde darán soporte a los procesos.
- **A4. SOA:** Compatibilidad con estrategias actuales para arquitecturas de aplicaciones empresariales compuestas, como SOA (*Service Oriented Architecture*), que se caracterizan por ser distribuidas, de bajo acoplamiento y sobre todo, de apoyo directo a los procesos de negocio.

### 3.3.3 Estándares Abiertos

En esta categoría se evalúa cada técnica de modelado en la medida en que esté definida alrededor de un estándar. Los estándares industriales no solo ayudan a eliminar la heterogeneidad de las diversas alternativas, sino que también fomentan el desarrollo de un ecosistema de proveedores de herramientas interoperables para diversos propósitos, siendo así más atractivo para la adopción en el sector industrial.

- **E1. Respaldo Consorcio Industrial:** Respaldo por parte de algún consorcio de estándares abiertos reconocido por la industria, debido a que éstos representan un consenso entre las compañías con más experiencia en este tema.
- **E2. Metamodelo disponible:** Existencia y disponibilidad del metamodelo para posibilitar la transformación de modelos, ya que de ésta forma se aprovechará su potencial como activos de conocimiento y se obtendrán los beneficios que proporciona la ingeniería de modelos. Un ejemplo característico de dicho metamodelo se puede evidenciar en (Ghalimi, 2007).
- **E3. Framework de Modelado:** Implementación del metamodelo en un framework de modelado como el propuesto en (Eclipse EMF, 2007), con el propósito de usarlo en herramientas de transformación de manera que se

facilite el mapeo hacia otros tipos de modelos en otros niveles de abstracción. Un estudio de dichas herramientas puede encontrarse en (Quintero, et al., 2007).

- **E4. Herramientas *open source*:** Existencia de herramientas *open source* que soporten la técnica, ya que de ésta forma se podrá utilizar el beneficio de comunidad que provee el desarrollo de software *open source*.

## 3.4 Escala de Comparación

En la Tabla 2 se presenta la escala de valores usada en el análisis comparativo. La escala se basa en el nivel de soporte que tienen las técnicas para cada uno de los criterios de comparación.

| Valor | Nombre | Nivel de Soporte |
|---|---|---|
| 1 | Nulo | No soportado, no documentado. |
| 2 | Pobre | Soportado con artificios, poca o ninguna documentación. |
| 3 | Regular | Soportado y documentado, pero difícilmente aplicable. |
| 4 | Bueno | Soportado, documentado y fácilmente aplicable. |
| 5 | Nativo | Soportado, documentado y fácilmente aplicable desde la definición inicial de la técnica. |

**Tabla 3:** Escala de Comparación

## 3.5 Análisis Comparativo

En la Tabla 3 se presentan las calificaciones dadas a cada ítem propuesto en los criterios de comparación descritos en la sección 5, de igual forma se muestra un promedio para cada uno de los ítems y una calificación total para cada una de las técnicas.

| Cód. | Criterio | EyP | BPMN | RAD | IDEF3 | UMurcia | Prom. |
|---|---|---|---|---|---|---|---|
| R1 | Adopción de CIM | 4 | 5 | 5 | 5 | 5 | 4.8 |
| R2 | Estructura y Comportamiento | 4 | 4 | 4 | 3 | 5 | 4 |
| R3 | Reglas de Negocio | 3 | 3 | 1 | 1 | 3 | 2.2 |
| R4 | Roles | 2 | 5 | 5 | 1 | 5 | 3.6 |
| R5 | Objetivos y E/S | 5 | 4 | 2 | 1 | 5 | 3.4 |
| R6 | B2B | 3 | 5 | 3 | 1 | 4 | 3.2 |
| R7 | Usabilidad | 3 | 5 | 3 | 3 | 3 | 3.4 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Promedios Representación Directa | | 3.4 | 4.4 | 3.3 | 2.1 | 4.3 | **3.5** |
| A1 | Respaldo Metodológico | 4 | 3 | 3 | 4 | 5 | 3.8 |
| A2 | Brecha modelado-ejecución | 3 | 5 | 1 | 1 | 5 | 3 |
| A3 | Motor de ejecución | 2 | 5 | 1 | 2 | 3 | 2.6 |
| A4 | SOA | 1 | 5 | 1 | 2 | 3 | 2.4 |
| Promedios Automatización | | 2.5 | 4.5 | 1.5 | 2.3 | 4.0 | **3.0** |
| E1 | Respaldo Consorcio Industrial | 3 | 5 | 4 | 4 | 2 | 3.6 |
| E2 | Metamodelo disponible | 3 | 5 | 1 | 3 | 5 | 3.4 |
| E3 | Framework de Modelado | 2 | 5 | 1 | 1 | 5 | 2.8 |
| E4 | Herramientas *open source* | 1 | 5 | 1 | 1 | 5 | 2.6 |
| Promedios Estándares Abiertos | | 2.3 | 5.0 | 1.8 | 2.3 | 4.3 | **3.1** |
| **Totales** | | **43** | **69** | **36** | **33** | **63** | **48.8** |

**Tabla 4:** Análisis Comparativo de Técnicas de Modelado de Negocio

En la tabla el ranking de los totales de las columnas ilustra el orden de cumplimiento de los criterios por parte de las técnicas evaluadas, mientras que el promedio de las filas ilustra el nivel de adopción de cada criterio por parte de todas las técnicas en una escala de uno a cinco. Los promedios por categoría ilustran la forma como las técnicas de modelos de negocio están evaluaciones en los frentes de la representación directa, automatización y estándares abiertos.

Para cada una de las técnicas evaluadas está disponible como anexo un reporte técnico en donde se ilustra el montaje del caso de aplicación en la respectiva técnica con su descripción y la justificación para la calificación de cada uno de los criterios.

## 3.6 Conclusiones

- BPMN y la propuesta de modelado de negocio de la Universidad de Murcia con diagramas UML de actividades, constituyen estrategias atractivas para el modelado de negocio a la luz de los criterios evaluados
- La principal fortaleza de la propuesta de la Universidad de Murcia es la robustez y completitud para representar el negocio y la utilización purista de diagramas UML de actividades.

- La fortaleza de BPMN es su acercamiento al entorno de negocio, convirtiéndose en una técnica simple y usable para personas sin conocimientos técnicos.
- La propuesta de Ericsson y Penker permite el modelado de procesos, pero en su artefacto principal (el modelo de negocio) proporciona una visa de caja negra que no otorga una vista completa y relevante de las actividades al interior de cada proceso.
- Los diagramas de actividades de roles (RADs) proporcionan un buen mecanismo para representar las responsabilidades de los roles en cada proceso, pero la técnica no tienen un apoyo en la automatización de soluciones y no cuenta con herramientas para soportar su modelado. Esto puede deberse a que el propósito de la técnica es apoyar la reingeniería de procesos, en lugar de la implementación de aplicaciones de apoyo a procesos.
- IDEF3 es una técnica que provee un nivel de detalle considerable en el modelado de procesos, pero carece de capacidad para representar otros aspectos importantes, lo que puede deberse a que su propósito principal es la captura y documentación de procesos. Además, su gran limitación es la evolución de la técnica altamente acoplada con productos propietarios, haciendo difícil el uso de ésta técnica para la construcción de soluciones de ingeniería de modelos.

Con base en los resultados obtenidos, se ha seleccionado BPMN como la notación de modelado del CIM más adecuada para éste proyecto.

Buscando compatibilidad con la técnica de modela de negocio elegida, la especificación del PIM en la transformación será SCA, por ser destino típico para la automatización de modelos construidos con BPMN, además de otras razones expresadas más adelante en la sección 4.3.

# 4. HEURISTICAS DE MAPEO DE CIM A PIM

## 4.1 Introducción: Una vista General a la Transformación de CIM a PIM

En el panorama de trabajos alrededor de la ingeniería de modelos tanto a nivel industrial como académico, reinan predominantemente la construcción de herramientas de transformación de modelos a código, debido al inmediato beneficio que podemos obtener de la generación automática de artefactos nativos en los procesos de desarrollo de software actuales vistos en la mayoría de las organizaciones. En los términos de clasificación de modelos ofrecidos en (OMG, 2003), explicados a fondo en capítulo 2 “Estado del Arte”, se trata de transformaciones de PIM a PSM[23], de PSM a código, y en la mayoría de los casos, directamente de PIM a código.

La transformación de CIM a PIM no ha sido un terreno muy explorado, y algunas de las razones que esto explica se enuncian a continuación.

- La práctica de hacer modelos independientes de la computación no siempre es llevada a cabo con disciplina, a pesar de que es necesaria para entender el contexto del problema para el cual se construirá una solución. Comúnmente se debe a la escasez de tiempo o recursos en los proyectos de desarrollo de software.
- A pesar de que su nombre expresa sugestivamente que los modelos no deben exponer ningún aspecto de implementación tecnológica, la definición de CIM no expresa con claridad qué tipos de modelos deberíamos construir. Tradicionalmente, en los procesos de desarrollo de software se ha hablado de casos de uso, así como de modelos de dominio y de

[23] *Platform Specific Model*, Modelo Específico de Plataforma

procesos de negocio. Estos no son mutuamente excluyentes, sino que aspectos del contexto de forma diferente.

En nuestros procesos de desarrollo actuales, ya sea en el paradigma de la orientación a objetos, o de la orientación a servicios, nos referimos a la disciplina de pasar del espacio del problema al espacio de la solución como análisis y diseño. De acuerdo a factores como el problema, el paradigma usado, entre otros, ésta disciplina es llevada a cabo de forma diferente cada vez. En lugar de tener un método bien definido a usar en todos los casos, hacer un buen análisis y diseño es más bien un arte, en el cual múltiples heurísticas son usadas. La transformación de CIM a PIM se encarga de automatizar las fases del proceso de desarrollo involucradas con esta disciplina, pero debido a las infinitas alternativas de hacerlo, es necesario determinar un conjunto bien definido de heurísticas que garantice un análisis y diseño efectivo. En éste trabajo se presentan las caracterizaciones de los modelos CIM y PIM que se utilizarán en la propuesta de transformación, y luego se definirán dichas heurísticas para el mapeo entre dichos tipos de modelos.

## 4.2 Caracterización del tipo de CIM elegido

En ésta sección se hace un recuento del trabajo de selección realizado para elegir un tipo de modelo cuya perspectiva fuera independiente de la computación. Se analizan los criterios evaluados, una contextualización del tipo de CIM escogido y una ilustración de su uso con un caso práctico.

### 4.2.1 Criterios para la Selección del tipo de CIM

En (OMG, 2003) se define el CIM como "una vista del sistema desde la perspectiva independiente de la computación. Un CIM no exhibe los detalles de la estructura de los sistemas. Un CIM a veces es llamado un modelo de dominio y en su especificación se usa un vocabulario que es familiar a los practicantes del

dominio en cuestión. Se asume que el usuario primario del CIM, el practicante del dominio, no tiene conocimiento acerca de los modelos o artefactos usados para lograr la funcionalidad para la cual los requerimientos son articulados en el CIM. El CIM juega un rol importante en acortar la brecha entre los expertos en el dominio y sus requerimientos en un lado, y los expertos en el diseño y la construcción de los artefactos que juntos satisfacen los requerimientos del dominio, en el otro lado".

Según esta definición, los requerimientos analizados en (Arsanjani, 2005) y lo examinado en el estado del arte de este trabajo, el tipo de CIM que debemos utilizar debe cumplir con los siguientes requisitos:

- Los modelos deben estar orientados hacia su usuario principal, los expertos del dominio. Concretamente hablando, nos referimos a los analistas de negocio, quienes se encargan de construir representaciones del estado actual del negocio, con el fin de identificar problemas y proponer soluciones de mejora.
- Los modelos deben expresar una descripción de la funcionalidad realizada por los actores del negocio, en términos de las tareas que realizan como parte de su trabajo.
- El tipo de modelos que constituyan la representación del dominio del negocio deben ser los procesos de negocio. Como se demostró en el estado del arte de este trabajo, los procesos de negocio proveen una perspectiva de la dinámica de la organización de extremo a extremo, respondiendo al cuestionamiento de como cumplen con el objetivo para el que fueron diseñados.
- El tipo de modelo para el CIM debe cumplir con los principios básicos de la Ingeniería de Modelos, según lo concluido en el estado del arte a partir de (Bezivin, 2003). Concretamente, debe presentar la característica de poseer un metamodelo que especifique formalmente los elementos de representación que se utilizan.

- Debe cumplir con alguna implementación presente en una herramienta de modelado, idealmente *open source*, para garantizar su accesibilidad por las diferentes audiencias de éste trabajo.
- Debe cumplir con los pilares requeridos expresados en (Booch, et al., 2004): Automatización, representación directa y estándares abiertos.

### 4.2.2 Tipo de CIM elegido: BPMN (*Business Process Modeling Notation*)

Además de la presentación que se ha hecho de ésta técnica en la sección 3.1.2, no es necesario ahondar más en sus características. En su lugar, en esta sección nos concentraremos en lo requerido de ésta técnica para su uso en la transformación de modelos; es decir, una descripción formal de la técnica y un metamodelo.

A la fecha han surgido dos alternativas para la formalización de los elementos de modelado de BPMN: BPDM (*Business Process Definition Metamodel*) (Adaptive Inc., 2007), un nuevo estándar de OMG que se ha ocupado de definir un metamodelo, orientado tanto hacia la especificación formal de los modelos de negocio como su representación con una notación de modelado como BPMN (OMG, 2006). Al estar basado en MOF (*Meta-Object Facility*), es posible construir un esquema XML del metamodelo para permitir la interoperabilidad entre herramientas de modelado. El estándar es independiente tanto de las notaciones de representación como de las metodologías de definición de procesos.

La otra alternativa existente a la fecha es el esquema BPMN del proyecto *open source* Eclipse STP-BPMN (http://www.eclipse.org/stp/bpmn). Ha sido construido utilizando tecnologías del proyecto EMF (*Eclipse Modeling Framework*) de Eclipse Foundation (http://www.eclipse.org). Específicamente, se han basado en Ecore, una implementación del estándar MOF (*Model-Object Facility*) de OMG, y presenta una alternativa mucho más simple que BPDM (Ghalimi, 2007).

Cabe anotar que BPMN tiene como uno de sus requisitos principales proveer un puente a través de la brecha semántica entre el modelado de procesos de negocio y la ejecución de los mismos. Es por esto que una de sus capacidades esperadas es la capacidad de mapear directamente a un lenguaje de ejecución, como BPEL4WS. El metamodelo propuesto en el proyecto Eclipse STP-BPMN soporta esta característica de forma inherente, lo cual se puede evidenciar en la herramienta descargable de la dirección anteriormente mencionada. A pesar de no ser una implementación estricta de la especificación de la notación, es uno de los pocos artefactos *open source* disponibles y usables para la ingeniería de modelos. A continuación se presenta una versión reducida de dicho metamodelo, exponiendo sólo aquellos elementos que son de interés para la transformación CIM a PIM.

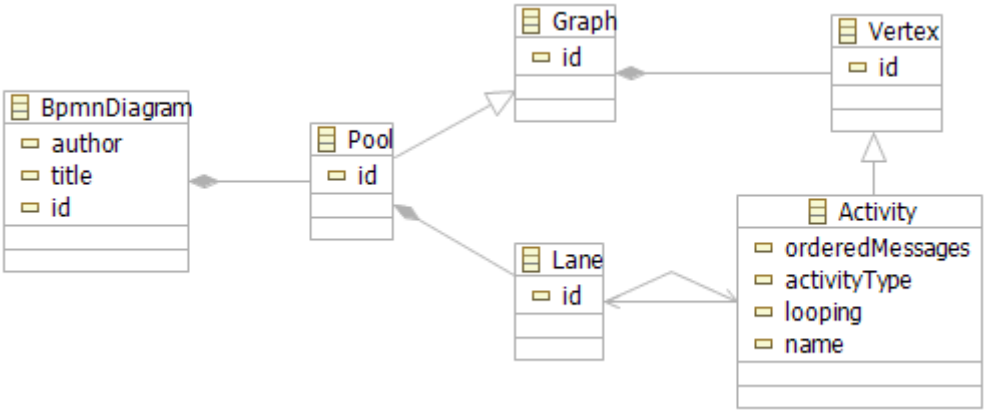


**Figura 11.** Metamodelo reducido de BPMN

.

Un **diagrama BPMN** se encuentra fundamentalmente estructurado por una o más ***pools*** o piscinas, compuestas en sí mismas por ***lanes*** o carriles. Un carril se encuentra asociado con un conjunto de **actividades** que representan los pasos de ejecución del proceso de negocio, y ésta relación se encuentra generalizada con la asociación de **grafos** compuestos de **vértices**. Cabe anotar que elementos como compuertas lógicas (*gateways*) y eventos no han sido plasmados en ésta versión del metamodelo, ya que no son relevantes para la transformación de CIM

a PIM que propone éste trabajo. Estos conciernen más para etapas posteriores en el ciclo de vida transformacional, por ejemplo la generación de artefactos nativos como interfaces WSDL o BPEL.

### 4.2.3 Caso de aplicación: Un CIM de un proceso de Quejas y Reclamos

El siguiente diagrama es un modelo que representa el proceso de Quejas y Reclamos de una compañía de servicios, del caso de estudio presentado en la sección 3.1; corresponde al modelo presentado en el estudio comparativo pero en una versión más detallada para estudiar los elementos de BPMN como técnica de modelado y su utilización en el proceso de transformación.. Este es un ejemplo del modelo origen requerido para la transformación de CIM a PIM.

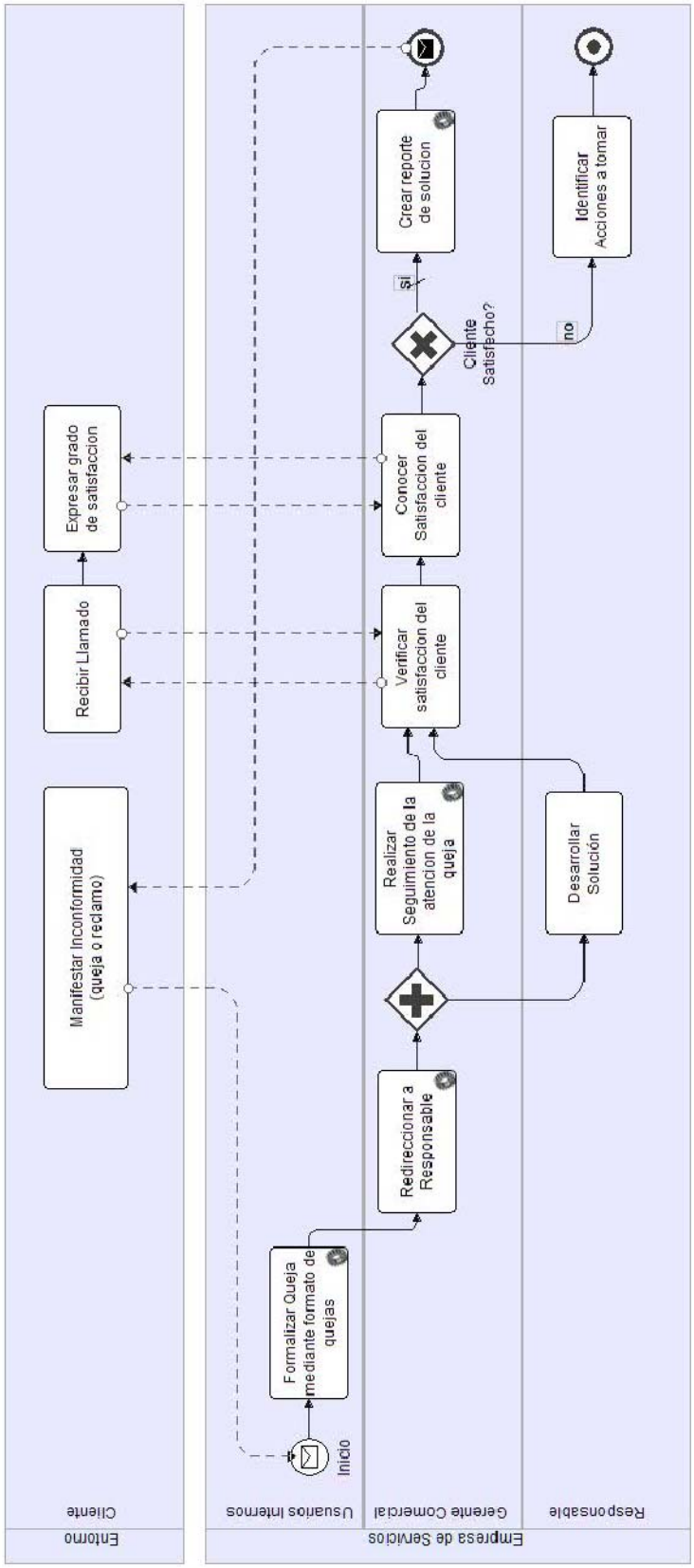


**Figura 12.** Modelo BPMN de un proceso de quejas y reclamos en una compañía de servicios

El proceso inicia cuando un usuario interno es contactado por el cliente para manifestar una inconformidad. En este sentido, es necesario que la queja sea formalizada mediante un documento siguiendo el formato de quejas y reclamos disponible en la intranet de la organización. Una vez formalizada la queja, en el área comercial se procede a procesar la queja. Esto incluye asignar un responsable de la compañía de acuerdo al cliente o proyecto del que se esté tratando, quien se encarga de identificar la causa de la queja, plantear alternativas de solución y el nuevo compromiso con el cliente y llevarlo a cabo. A lo largo de esas actividades, en el área comercial se está velando por la eficacia de la solución, que le proporcione al cliente el resultado esperado.

Al final, se hace una visita personal al cliente para conocer su nivel de satisfacción (esto es representado con una compuerta de flujos paralelos, el rombo con signo de adición). Si el cliente se encuentra satisfecho, se procede a elaborar un reporte de la solución a la queja y las implicaciones que ésta trae para el proyecto, si hay alguna. Si no hay satisfacción, se procede a dar un tratamiento especial, en el que se programan una serie de reuniones para conocer la expectativa del cliente frente a la compañía y crear un espacio para negociar (Estos flujos se encuentran delimitados por una compuerta lógica, el rombo con una equis).

## 4.3 Caracterización del tipo de PIM elegido

En ésta sección se hace un recuento del trabajo de selección realizado para elegir un tipo de modelo para la construcción de aplicaciones de software cuya perspectiva fuera independiente de cualquier plataforma de implementación. En primer lugar se analizan los criterios evaluados que debe cumplir la técnica de modelado, luego una contextualización del tipo de PIM escogido y una ilustración de su uso con el caso práctico definido para el proyecto.

### 4.3.1 Criterios para la Selección del PIM

Tal como se especificó en el capítulo del Estado del Arte, la Arquitectura Orientada a Servicios se constituye como la estrategia de construcción de software por excelencia para el apoyo a procesos de negocio a través de la organización, en lugar de los clásicos estilos de arquitectura de aplicaciones monolíticas. Es por esto que el tipo de modelo usado para la transformación CIM a PIM debe cumplir con los siguientes criterios alrededor de SOA:

- Permitir plasmar completamente en un modelo aplicaciones compuestas que den apoyo a procesos de negocio. Estas aplicaciones utilizan servicios y componentes tanto nuevos como existentes, de aplicaciones legacy externas, que deben diferenciarse.
- El modelo PIM de aplicaciones compuestas debe permitir representar tanto la forma de crear componentes individuales como los mecanismos para describir cómo esos componentes trabajan juntos.
- El tipo de PIM seleccionado debe permitir modelar toda la información necesaria para las transformaciones posteriores en el ciclo de vida, como a modelos específicos de la plataforma (PSM) y artefactos nativos (código).
- Otros criterios expresados en el capítulo de estudio comparativo de técnicas de modelado de negocio, en las categorías de Automatización y Estándares Abiertos.

### 4.3.2 Tipo de PIM elegido: SCA (*Service Component Architecture*)

Para la selección del tipo de PIM no se realizó un estudio comparativo como el empleado para decidir sobre el tipo de CIM. Esto se debe a la falta de madurez y reconocimiento de las diferentes técnicas de modelado de sistemas bajo el enfoque de SOA, lo que impide la ejecución de un estudio imparcial y concluyente. La técnica elegida corresponde a la que hasta la realización de este trabajo parece

más promisoria, además del cumplimiento de los criterios enumerados en la anterior sección.

Open SOA es un consorcio de proveedores de soluciones para aplicaciones empresariales conformado por IBM, Oracle, Sun Microsystems, SAP, BEA Systems entre otros. En marzo de 2007, publicaron la primera versión de SCA (Service Component Architecture). Conformado por varias especificaciones, SCA define un enfoque para modelar aplicaciones compuestas y llevarlas a la ejecución, mediante la creación y ensamble de componentes en aplicaciones compuestas, todo bajo una visión SOA. Como técnica de modelado, nos permite representar en un modelo una aplicación compuesta que da soporte a un proceso de negocio, conformada por componentes orquestados a través de servicios. De ésta forma, se procura conservar el bajo acoplamiento, lo que posibilita el reuso a gran escala de componentes de servicio (Chappell, Julio 2007).

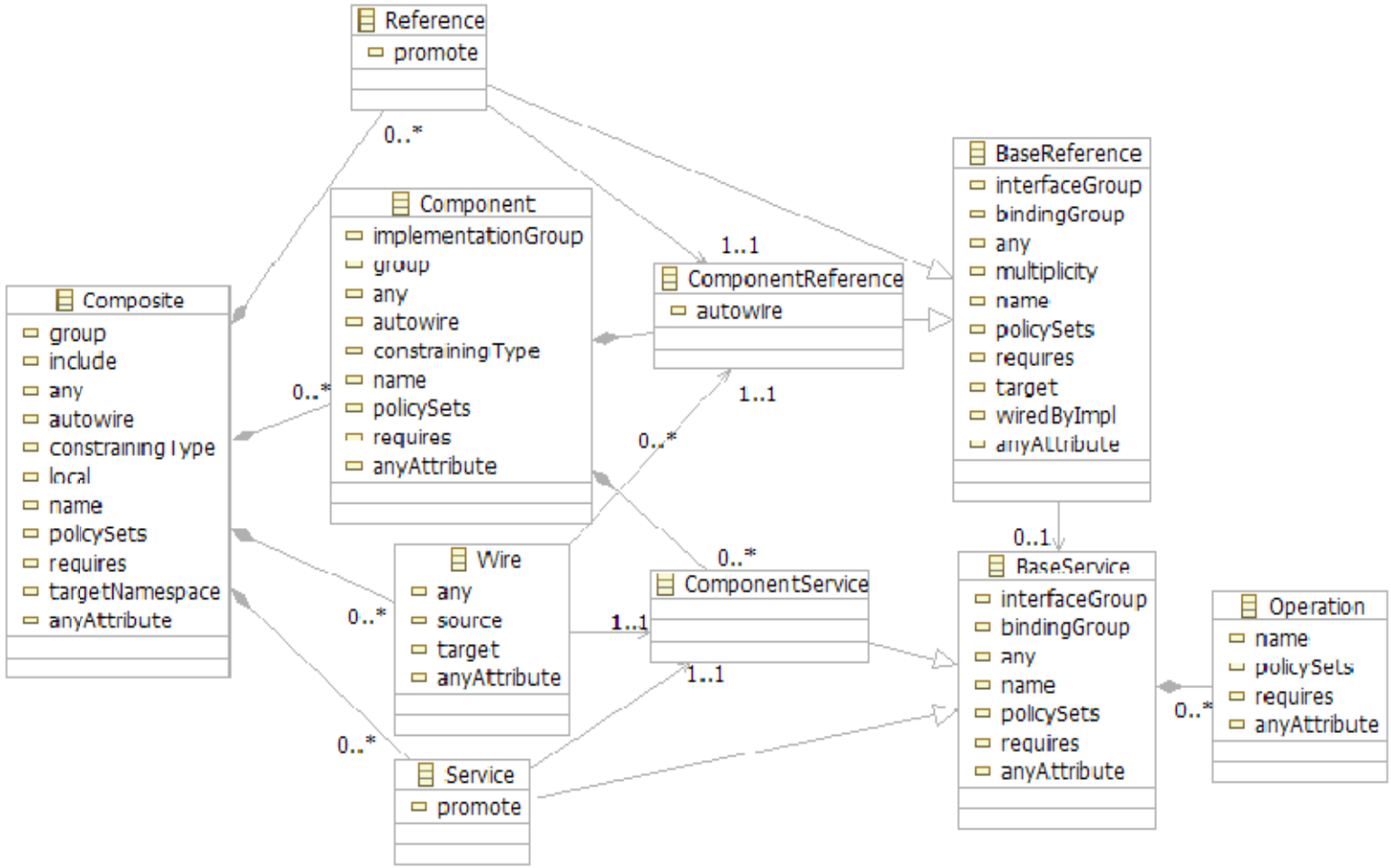


**Figura 13**. Metamodelo de SCA.

El elemento principal de SCA es el ***composite***, que corresponde a una aplicación compuesta, la cual se encuentra conformada por uno o más **componentes**. Un componente ofrece su funcionalidad a otros componentes a través de **servicios**, y acceden a la funcionalidad de otros a través de **referencias**. Cuando un servicio de un componente es ofrecido externamente como un servicio de la aplicación compuesta, se dice que es promovido, a través de un ***wire***, al igual que cuando las referencias de un componente invocan un servicio externo al *composite*, que también son promovidas. Como componentes funcionales, un servicio posee un conjunto de **operaciones**.

El metamodelo distingue los servicios que ofrecen los componentes y los *composite*, llamándolos ComponentService y Service, respectivamente. Ambos descienden del tipo BaseService que contiene los atributos comunes para los dos. Este mismo caso se presenta para las referencias.

### 4.3.3 Caso de aplicación: Un PIM de un proceso de Quejas y Reclamos

El siguiente diagrama es un modelo que representa la aplicación compuesta que le da soporte al proceso de Quejas y Reclamos, ilustrado en el capítulo anterior. Este es un ejemplo del modelo destino que se espera obtener con las heurísticas de transformación de CIM a PIM.

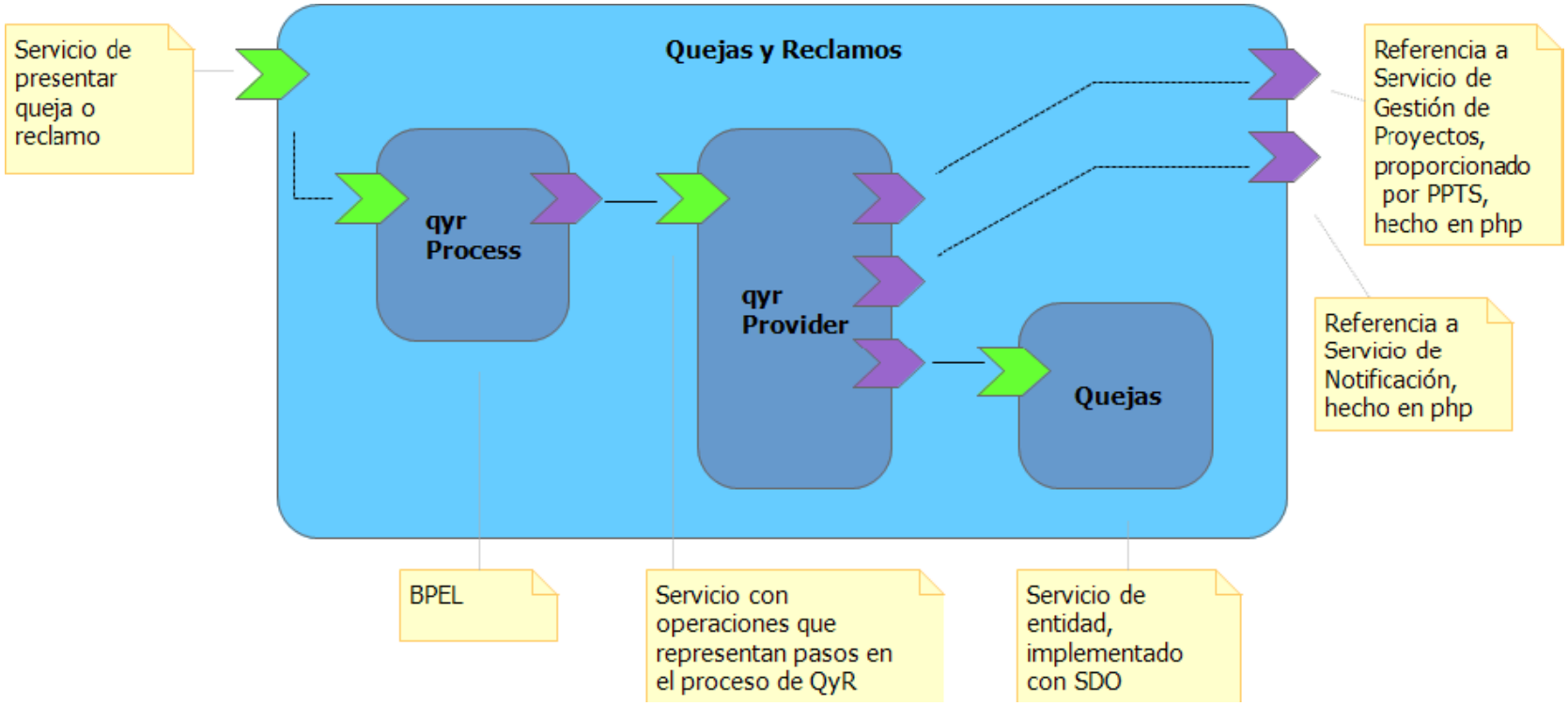


**Figura 14.** Modelo en SCA de la aplicación compuesta de quejas y reclamos

A continuación se describen los elementos del modelo, de acuerdo a la jerarquía que propone el metamodelo.

***Composite* "Quejas y Reclamos".** Describe la agregación de componentes que conforman la aplicación compuesta que da soporte al proceso de quejas y reclamos.

- Servicio promovido "Presentar Queja o Reclamo".
- Componente "qyr Process". Es el proceso de BPEL que orquesta las operaciones del servicio que provee "qyr Provider"
    - Servicio "Presentar Queja o Reclamo". Asociado mediante un *wire* con el servicio del *composite* "Presentar Queja o Reclamo". Se trata del proceso BPEL como un todo expuesto como un servicio.
    - Referencia "qyrOperations". Invoca el servicio "qyrOperations" del componente "qyr Provider".
- Componente "qyr Provider".
    - Servicio "qyr Operations". Contiene las operaciones que representan pasos en el proceso de quejas y reclamos.
    - Referencia "Gestión de Proyectos". Contiene un *wire* a la referencia del *composite* "Gestión de Proyectos".
    - Referencia "Notificación". Contiene un *wire* a la referencia del *composite* "Notificación".
    - Referencia "Gestión Quejas". Invoca el servicio "Gestión Quejas" del componente "Quejas".
- Componente "Quejas"
    - Servicio "Gestión Quejas". Implementado con SDO, para obtener información de la entidad Quejas y Reclamos de negocio.
- Referencia "Gestión de Proyectos". Invoca el servicio externo de gestión de proyectos, el cual es proporcionado por la aplicación PPTS (*Project Planning and Tracking System*), el cual se encuentra construido en PHP.

- Referencia "Notificación". Invoca los servicios de notificación, entre ellos el de envío de correos electrónicos. Pueden ser implementados en PHP.

## 4.4 Heurísticas de Mapeo

En esta sección se ilustran las heurísticas que se plantean para la transformación de CIM a PIM. Como se enunció en la introducción, este tipo de transformación corresponde a la disciplina de análisis y diseño, alrededor de la cual existen varias propuestas metodológicas para guiar su ejecución. De esta forma, es necesario presentar la contextualización metodológica a partir de la cual se realiza la propuesta de heurísticas para el mapeo de CIM a PIM.

### 4.4.1 Criterios para la Selección de Enfoque Metodológico

Como puede evidenciarse en los diversos criterios expuestos en las últimas dos secciones para la elección de tipos de CIM y de PIM, no es de extrañarse que el trasfondo metodológico usado para la propuesta de heurísticas para mapear entre estos tipos de modelos sea basado en SOA como paradigma de desarrollo. Aquí recalcamos los criterios más importantes que son cumplidos por el enfoque elegido:

- Debe permitir una alineación directa de los aspectos del negocio, que son plasmados en el CIM, con los aspectos de la implementación tecnológica, que son plasmados en el PIM.
- Debe ser un enfoque de análisis y diseño de sistemas, que sea probado industrialmente y se garantice su efectividad, y se caracterice por ser una solución al problema de alinear TI con el negocio.
- Debe proporcionar un método bien definido para pasar de CIM a PIM de acuerdo a los tipos de modelos escogidos, es decir, BPMN a SCA, respectivamente.

### 4.4.2 Enfoque metodológico elegido: Identificación de Servicios

Se ha escogido la propuesta de Thomas Erl (Erl, 2007) para hacer el proceso de pasar del espacio del problema al espacio de la solución en el contexto de SOA, proceso que el autor ha bautizado como identificación de servicios. La razón por la que se ha tomado esta metodología como referencia es que provee una secuencia de pasos para pasar de un modelo de procesos de negocio a un modelo de servicios identificados, lo que corresponde con el mapeo que nos proponemos definir, entre las notaciones BPMN como CIM y SCA como PIM.

### 4.4.3 Heurísticas de transformación

Thomas Erl propone una serie de 12 pasos para la identificación de servicios. Para nuestro caso particular, la propuesta resulta un poco extensa, por lo cual la reducimos a los siguientes cinco pasos:

1. **Filtrar los pasos del proceso que no son ejecutables.** Los modelos de procesos de negocio son descripciones abstractas totalmente independientes de aspectos de computación, como aplicaciones de software o infraestructura de hardware. En su lugar, el analista de negocio los construye con actividades que dicen cómo lograr el objetivo del proceso, pero algunas de éstas actividades no son automatizables, como por ejemplo "Llamar a cliente". Otras, por ejemplo como "Registrar formulario", son operaciones relacionadas con la captura y manipulación de datos, por lo tanto son candidatas para convertirse en operaciones de servicios. Concretamente, la entrada de éste paso es el listado de todos los pasos del proceso, y la salida es la lista de sólo los servicios candidatos.
2. **Definir actividades identificadas del proceso de negocio.** Para cada uno de los pasos identificadas en el paso anterior, se define:

- Definición: Describir qué hace ésta actividad como un paso dentro del proceso de negocio que lo contiene.
- Entradas: Cuales son las entradas de información y eventos que requiere el servicio para llevar a cabo su tarea.
- Salidas: Información o producto de la ejecución de la actividad. Puede tratarse tanto de datos manipulados como de una simple notificación de confirmación.
- Historias de usuario: Describir cual es la forma en la que el usuario que participa en el proceso de negocio lleva a cabo ésta actividad. Por ejemplo, a través de un formulario, una notificación por correo electrónico, entre otras.
- Implementación: Detalles de la implementación tecnológica de la actividad. Plataforma, lenguaje de programación, entre otros detalles relevantes.
- Servicios de entidad, aplicación e infraestructura utilizados: La ejecución de las actividades de negocio pueden requerir de sistemas legados, aplicaciones externas u otros servicios, que aquí se detallan.

3. **Agrupación de las actividades en contextos lógicos.** En el modelo de servicios, las actividades identificadas anteriormente corresponden a operaciones de servicios. Estas son agrupadas de acuerdo a contextos reutilizables que permitan su identificación y posterior reutilización. Estos contextos se llaman servicios de negocio, y residen lógicamente dentro de la aplicación compuesta.
4. **Definir servicios de entidad.** Se refiere a la gestión de ciertos objetos de negocio que son manipulados durante los procesos; por ejemplo, 'Clientes', 'Proyectos', 'Quejas', entre otros, que son requeridos por las actividades enunciadas en el paso 2. También es necesario definir su ubicación lógica, es decir, si hace parte de la aplicación compuesta que se está desarrollando para soportar el proceso de negocio o si hace parte de otros sistemas externos.

5. **Definir servicios de aplicación e infraestructura.** Se refieren a servicios orientados a la tecnología, como por ejemplo 'Imprimir documento' o 'Enviar email', que son requeridos por las actividades enunciadas en el paso 2. También es necesario definir su ubicación lógica, es decir, si hace parte de la aplicación compuesta que se está desarrollando para soportar el proceso de negocio o si hace parte de otros sistemas externos.

### 4.4.4 Caso de aplicación: Identificación de Servicios para el proceso de quejas y reclamos

A continuación se presenta la ilustración de dicha metodología con el caso presentado en las últimas dos secciones. Basados en el modelo en BPMN del proceso de quejas y reclamos, procedemos a la ejecución de los tres pasos expuestos.

**Paso 1: Filtrar los pasos del proceso que no son automatizables luego de revisar el modelo BPMN**

- Formalizar queja mediante formato de quejas
- Redireccionar a responsable
- Realizar seguimiento de la atención de la queja
- Crear reporte de solución

**Paso 2: Definir actividades identificadas del proceso de negocio.**

**ACTNEG01: Formalizar queja mediante formato de quejas**

Definición: La queja presentada queda formalizada dentro de la organización, lista para que la empresa asuma su responsabilidad frente al cliente y actúe.

Entradas: El Formato de Quejas diligenciado

Salidas: Visto bueno.

Historia de usuario: El usuario interno recibe la queja del usuario y diligencia el formato en un formulario web.

Implementación: La información del formato de quejas y reclamos es ingresada en una base de datos o repositorio XML centralizado.
Servicios de entidad usados: Gestión de Quejas

**ACTNEG02: Redireccionar a responsable**

Definición: Con la queja formalizada, el gerente de cuentas corporativas determina el equipo del proyecto correspondiente y asigna un responsable de la solución, a quien se le comunican los detalles de la queja.
Entradas: Queja formalizada.
Salidas: Detalle de la asignación y de la responsabilidad a la persona elegida, con los detalles de la queja formalizada.
Historia de usuario: El gerente comercial obtiene la información de todos los proyectos y sus equipos de trabajo, y elige a quien asignar para resolver la queja, a quien le envía los detalles y le informa de la responsabilidad de su solución y su compromiso con el cliente.
Implementación: Se prepara una notificación que se le envía al responsable elegido con la entrada del servicio y el mensaje personalizado del gerente de cuentas.
Servicios de entidad utilizados: Gestión de clientes y proyectos
Servicios de infraestructura utilizados: Notificación

**ACTNEG03: Realizar seguimiento de la queja**

Definición: Tanto el responsable de la solución como el gerente comercial necesitarán llevar seguimiento de cada queja y su solución y evolución, para procurar que la queja está siendo resuelta de forma eficiente.
Entradas: Queja a la cual se le realizará seguimiento. Comentarios (novedades) si son necesarios.
Salidas: Detalle del historial del seguimiento de la queja.

Historia de Usuario: El usuario selecciona la queja o el cliente a la cual le desea hacer el seguimiento, de la cual obtiene su historial de seguimiento y puede agregar novedades.
Implementación: El historial del seguimiento se almacena en el mismo repositorio en donde residen las quejas.
Servicios de entidad usados: Gestión de Quejas

**ACTNEG04: Crear reporte de solución**

Definición: Al haber sido solucionada la queja para el cliente, se prepara un reporte de la solución para dejar constancia de la respuesta y el compromiso de Avansoft con su satisfacción.
Entradas: Detalle de la queja, y el seguimiento de su solución.
Salidas: Reporte de la solución de acuerdo al formato diligenciado.
Historia de Usuario: El gerente comercial obtiene el detalle de la queja y su solución con el historial de seguimiento, y determina si la resolución fue exitosa (si la solución fue oportuna, si fue efectiva, si el cliente quedó satisfecho).
Implementación: Se crea el reporte el cual se le envía al cliente a través de un servicio de notificación. El reporte se almacena en el repositorio central junto con el detalle de la queja y su historial de seguimiento.
Servicios de entidad usados: Gestión de Quejas
Servicios de infraestructura utilizados: Notificación.

**Paso 3: Agrupación de las actividades en contextos lógicos**

En éste caso podemos ver que las cuatro actividades pueden agruparse bajo el mismo contexto lógico de quejas y reclamos.

**SVCNEG01: Proveedor de operaciones de quejas y reclamos**

Operaciones:
ACTNEG01: Formalizar queja mediante formato de quejas
ACTNEG02: Redireccionar a responsable

ACTNEG03: Realizar seguimiento de la queja

ACTNEG04: Crear reporte de solución

**Paso 4: Definir Servicios de Entidad**

**SVCAPL001: Gestión de Proyectos**

Ubicación lógica: Externo a la aplicación compuesta de QyR

Operaciones:

SVCAPL001-1: Consultar proyectos dado un cliente

SVCAPL001-2: Consultar todos los proyectos en progreso de la compañía, agrupados por cliente

SVCAPL001-3: Consultar los proyectos anteriores de la compañía, agrupados por cliente

SVCAPL001-4: Consultar todos los empleados de la compañía y sus asignaciones a proyectos

SVCAPL001-5: Buscar un empleado de la compañía y sus asignaciones a proyectos

SVCAPL001-6: Consultar los empleados asignados a un proyecto específico

**SVCAPL004: Gestion de Quejas**

Operaciones:

Ubicación lógica: Interno en la aplicación compuesta de QyR

SVCAPL004-1: Consultar descripción y seguimiento de queja

SVCAPL004-2: Almacenar una nueva queja

SVCAPL004-3: Agregar comentario de seguimiento a una queja

SVCAPL004-4: reporte de solución de una queja

**Paso 5: Definir servicios de aplicación e infraestructura**

**SVCINF001: Notificación**

Ubicación lógica: Externo a la aplicación compuesta de QyR

Operaciones:

SVCINF001-1: Enviar una nueva notificación por correo electrónico

SVCINF001-2: Enviar una nueva notificación por mensajería instantánea

SVCINF001-3: Enviar una nueva notificación por VoIP

## 4.5 Reglas de Transformación

En esta sección definimos con precisión cuales son las reglas que permitirán a un motor de transformación tomar cada uno de los elementos del modelo origen y transformarlo en un elemento del modelo destino. No obstante, es necesario conocer la estrategia de transformación empleada, a la luz de las alternativas que se proponen en (OMG, 2003).

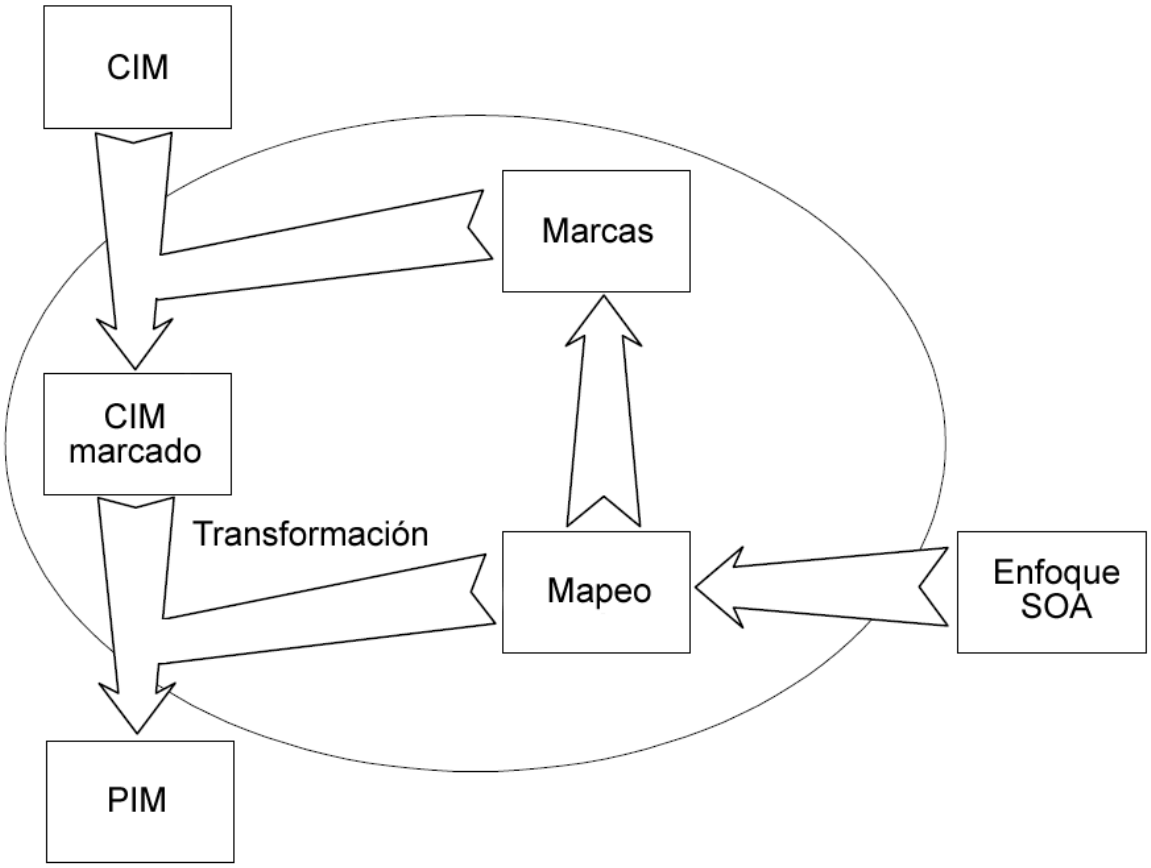


**Figura 15**. Estrategia de transformación empleada

Como se muestra en la figura 14, se utilizó un enfoque basado en marcas sobre el modelo para agregar la información requerida para la transformación. Para definir cuál es ésta información y como se estructura, se define un perfil que contiene el conjunto de marcas y sus propiedades (comúnmente conocidas como *tagged values*). En la siguiente sección se define el perfil para éste proyecto.

Esta definición de marcas es tenida en cuenta en la construcción del mecanismo de mapeo, comúnmente llamadas reglas de transformación. Dicho mapeo es definido de acuerdo a un enfoque o propuesta metodológica, que sugiere cómo los elementos semánticos de un modelo origen son mapeados a un modelo destino. En éste caso, dicho enfoque es SOA con la metodología de identificación de servicios propuesta por Thomas Erl (Erl, 2007), como se justificó en la sección 4.4.2. El proceso de transformación puede verse como un algoritmo procedimental para facilitar su entendimiento, como puede verse en la sección 4.5.2, y concretamente las reglas de transformación definidas en el motor de transformación de modelos se encuentran en la sección 4.5.3.

### 4.5.1 Perfil de Marcado de Modelos BPMN

Para nuestra propuesta de mapeo, sólo es necesaria una marca, la cual tiene por nombre "Automatable". Las actividades sobre el modelo BPMN que el analista de negocio considere automatizables llevarán ésta marca, y se agregará información mediante los siguientes tagged values:

- ServiceProvider : String.
- InternalEntityServicesRequired: String (Lista de nombres separados por comas).
- ExternalEntityServicesRequired: String (Lista de nombres separados por comas).
- InfrastructureServicesRequired: String (Lista de nombres separados por comas).

### 4.5.2 Algoritmo de Transformación

1. Crear *composite* con el nombre del proceso
2. Crear servicio del *composite*
3. Crear componente del proceso
   3.1. Crear componente con nombre del proceso + "-process" y descripción "Implementación del proceso en BPEL de " + nombre del proceso
   3.2. Crear servicio del componente
   3.3. Promover servicio a servicio de *composite*
4. Crear componente(s) de proveedores de servicios
   4.1. Por cada proveedor de servicios identificado en las actividades con la marca “automatable”, crear componente de proveedor de servicios
   4.2. Crear servicio por cada componente
   4.3. De cada actividad con automatable=true, agregar operaciones a cada servicio respectivo
   4.4. Crear referencias al componente del proceso y conectarlas con cada servicio.
5. Crear componente(s) internos de entidad
   5.1. Por cada entidad interna requerida en los servicios de negocio, crear un componente de entidad
   5.2. Crear servicios y conectarlos con los componentes proveedores de servicios que los requieren.
6. Crear referencias a entidades externas
   6.1. Por cada entidad externa requerida en los servicios de negocio, crear una referencia en el *composite* y promoverla.
7. Crear referencias a servicios de infraestructura
   7.1. Por cada servicio de infraestructura requerido en los servicios de negocio, crear una referencia en el *composite* y promoverla.

### 4.5.3 Reglas de Transformación

Estas reglas de mapeo de cada elemento de origen a su respectivo destino serán definidas en el lenguaje de transformación elegido para construir la herramienta.

| Regla | Elementos Modelo Origen | Elementos Modelo Destino |
|---|---|---|
| R1 | BPMNDiagram | Composite de SCA<br>Componente de Proceso: al interior de éste composite (nombre:= [nombre del proceso en el modelo origen]+'-process')<br>Servicio de éste componente, promovido.<br>Referencia, a cada uno de los servicios creados en los providers. |
| R2 | <automatable>Activity<br>Tag: ServiceProvider :: String | Componente de Proveedor de Servicio (nombre:=[actividad.serviceProvider], servicio.operaciones+=[actividad.nombre])<br>{excluye los componentes ya existentes con este nombre}<br>Wire desde la referencia del componente del proceso hacia servicio. |
| R3 | <automatable>Activity<br>Tag: InternalEntityServicesRequired :: String | Componentes de Entidad Internos (nombre:=[actividad.internalEntityName], servicios conectado a Componente de Proveedor de Servicio)<br>Referencia desde Componente Proveedor de Servicio a las entidades internas (no promovidas) |
| R4 | <automatable>Activity<br>Tag: ExternalEntityServicesRequired :: String | Reference del Composite ServiceReference desde Componente Proveedor de Servicio a las entidades externas (promovidas) |
| R5 | <automatable>Activity<br>Tag: InfrastructureServicesRequired :: String | Reference del Composite ServiceReference desde Componente Proveedor de Servicio a los servicios de infraestructura (promovidas) |

**Tabla 5:** Reglas de transformación de CIM a PIM.

# 5. HERRAMIENTA DE TRANSFORMACIÓN DE CIM A PIM “PLUG-IN BPMN2SCA”

Para el adecuado cumplimiento de los objetivos de éste proyecto, y con el ánimo de darle un sentido práctico que trascienda los lustros de la academia hacia la industria y la generación de valor, se ha desarrollado un plug-in que emplea la propuesta descrita para la transformación de modelos. El objetivo de la herramienta es apoyar construcción de software en la etapa de la etapa de definición de espacio del problema y espacio de la solución, comúnmente conocida como análisis y diseño, bajo un enfoque MDA, apoyados en los tres pilares descritos en el Estado del Arte (representación directa, automatización y estandarización), bajo una estrategia SOA (arquitectura orientada a servicios) que permita un apoyo directo a los procesos de negocio.

La plataforma para la cual se desarrollará el plug-in es Eclipse. Más que un entorno de desarrollo, es una plataforma muy flexible en la que es posible construir herramientas para diferentes propósitos e integrarlas muy fácilmente, gracias a su arquitectura de componentes en donde los plug-ins pueden ser extendidos por otros plug-ins. Esta arquitectura está respaldada por una de las comunidades open source más sólidas en el momento, respaldada por los gigantes de la industria de tecnología. En ella, se han desarrollado otros proyectos alternos a la plataforma pero basados en ella, que proveen soluciones de software con tecnologías y enfoques de punta. Entre ellos, podemos encontrar:

- **Eclipse Modeling Project.** Tiene como misión promover el uso y evolución de herramientas basadas en el enfoque de desarrollo dirigido por modelos, por medio de la construcción de un conjunto unificado de herramientas, frameworks e implementaciones de estándares industriales. El subproyecto principal es EMF (Eclipse Modeling Framework).

- **Eclipse SOA Tools Platform.** Tiene como misión la construcción de frameworks y herramientas para el diseño, configuración, ensamblaje, despliegue, monitoreo y administración de aplicaciones bajo el enfoque SOA.

La aplicación completa se encuentra anexa a éste trabajo. Información acerca de cómo utilizarla se encuentra en la guía de usuario anexa. A continuación detallamos las arquitecturas lógica y física de la herramienta.

## 5.1 Arquitectura Lógica

Al ser una herramienta de transformación de modelos, la arquitectura lógica del plug-in exhibe los elementos propuestos en el arquetipo de la transformación de modelos propuesto en (Bezivin, Abril 2004).

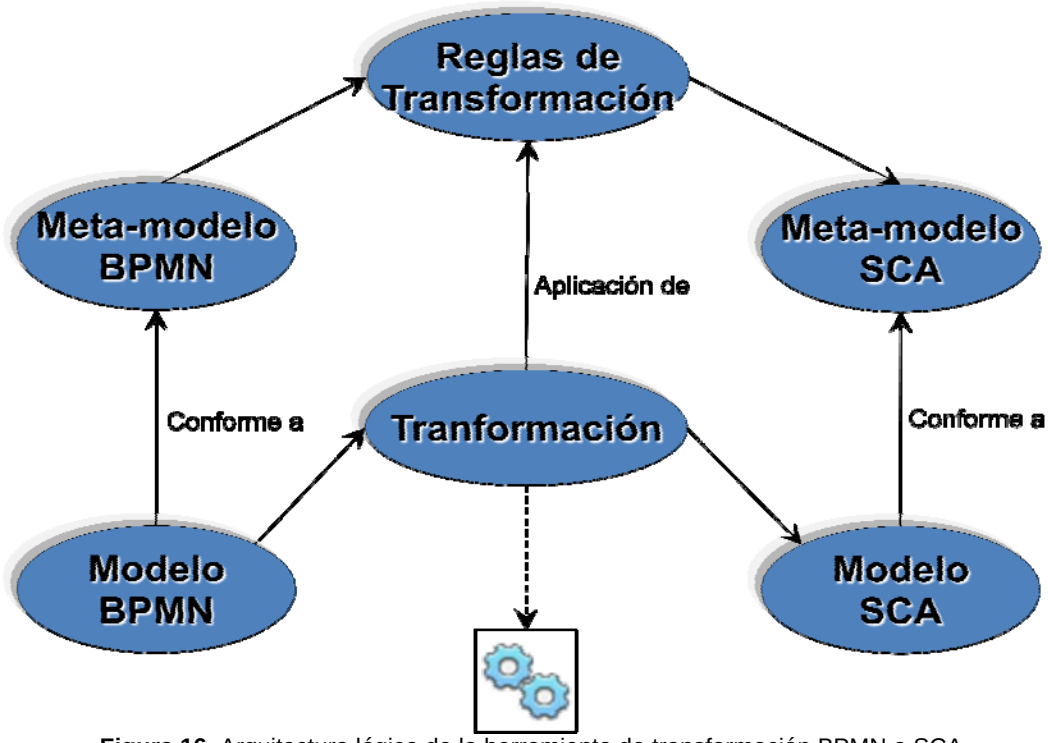


**Figura 16.** Arquitectura lógica de la herramienta de transformación BPMN a SCA.

A continuación se detallan cada uno de los elementos de la arquitectura lógica.

**Metamodelo del CIM: BPMN.** Define la sintaxis abstracta de BPMN como notación de modelado para procesos de negocio. Será utilizado el metamodelo mostrado en el capítulo 3, en "Caracterización del CIM".

**Editor de modelos CIM: Eclipse STP-BPMN.** Editor gráfico que define la sintaxis concreta de BPMN para permitir al usuario la creación de modelos de procesos de negocio, que son la entrada del proceso de transformación (acompañado de las marcas sobre el modelo).

**Reglas de Transformación.** Módulo donde han sido definidas las cinco reglas de mapeo como se ha expuesto en el capítulo 3, en "Heurísticas y Reglas de Transformación".

**Ejecución de la Transformación.** Módulo en donde se definen las anotaciones sobre el modelo de entrada que proporcionan la información adicional requerida para el proceso de transformación de modelos.

**Metamodelo del PIM: SCA.** Define la sintaxis abstracta de SCA como notación de modelado arquitecturas de componentes de servicios, bajo el enfoque SOA. Será utilizado el metamodelo mostrado en el capítulo 3, en "Caracterización del PIM".

**Editor de modelos PIM: Eclipse STP-SCA.** Editor gráfico que define la sintaxis concreta de BPMN para permitir al usuario la visualización de modelos de componentes de servicios, como salida del proceso de transformación. Así mismo, en este modelo resultante se puede completar para agregar nueva información para las posteriores etapas de transformación hacia tener artefactos nativos desplegando/ejecutando.

## 5.2 Arquitectura Física

A continuación se muestran los componentes que conforman la arquitectura física de la herramienta de transformación BPMN a SCA “Plug-in BPMN2SCA”.

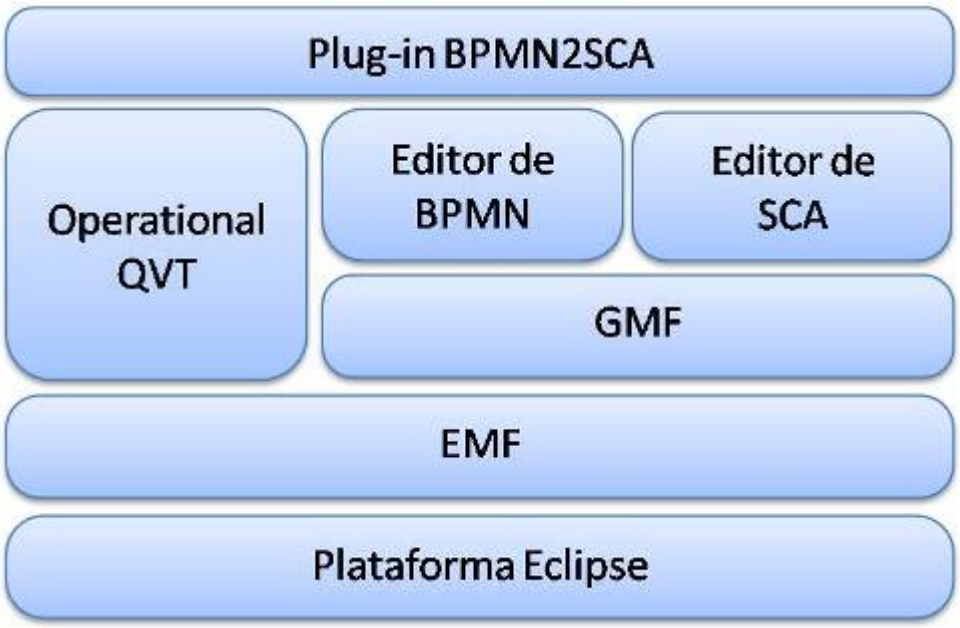


**Figura 17.** Arquitectura física de la herramienta de transformación de CIM a PIM

**Plataforma Eclipse.** La plataforma define un conjunto de frameworks y servicios comunes que colectivamente conforman el ambiente de integración para dar soporte al uso de eclipse como un modelo de componentes, como una plataforma de aplicaciones enriquecidas y de integración de herramientas. En sí mismo, Eclipse se apoya en Equinox, una implementación del estándar OSGi de modelo de componentes, que provee el concepto de plug-in para Eclipse (Eclipse, 2007).

**EMF.** El proyecto EMF (*Eclipse Modeling Framework*) es un framework de modelado para construir herramientas y otras aplicaciones basadas en un modelo estructurado (Eclipse EMF, 2007). A partir de un metamodelo, EMF genera un conjunto de herramientas extendibles para la creación y edición de instancias de

ese metamodelo. Este metamodelo puede estar descrito en XMI, o en Ecore, una implementación minimalista de MOF (*Meta-Object Facility*) (OMG, 2001).

**GMF.** GMF (*Graphical Modeling Framework*) complementa a EMF proporcionando herramientas para la creación de editores gráficos para la creación y edición de modelos (Eclipse GMF, 2007). Provee facilidades para la definición de elementos gráficos que conforman la sintaxis concreta de un lenguaje de modelado.

**Plug-in STP-BPMN.** Contiene el metamodelo de BPMN definido en Ecore, la notación de metamodelos de EMF y el editor de BPMN, basado en GMF.

**Plug-in STP-SCA.** Contiene el metamodelo de SCA definido en Ecore, la notación de metamodelos de EMF y el editor de SCA, basado en GMF.

**Plug-in Operational QVT.** Motor de transformación utilizado para transformar modelos conformes a metamodelos definidos en Ecore (Eclipse M2M, 2008). Es una implementación del enfoque operacional para transformación de modelos definido en el estándar (OMG, 2007).

## 5.3 Vistazo de la Herramienta Construida

A continuación presentamos dos figuras exhibiendo el modelo origen construido por el usuario y el modelo obtenido a partir de la transformación, respectivamente. En el anexo F se puede encontrar una guía para su instalación y uso.

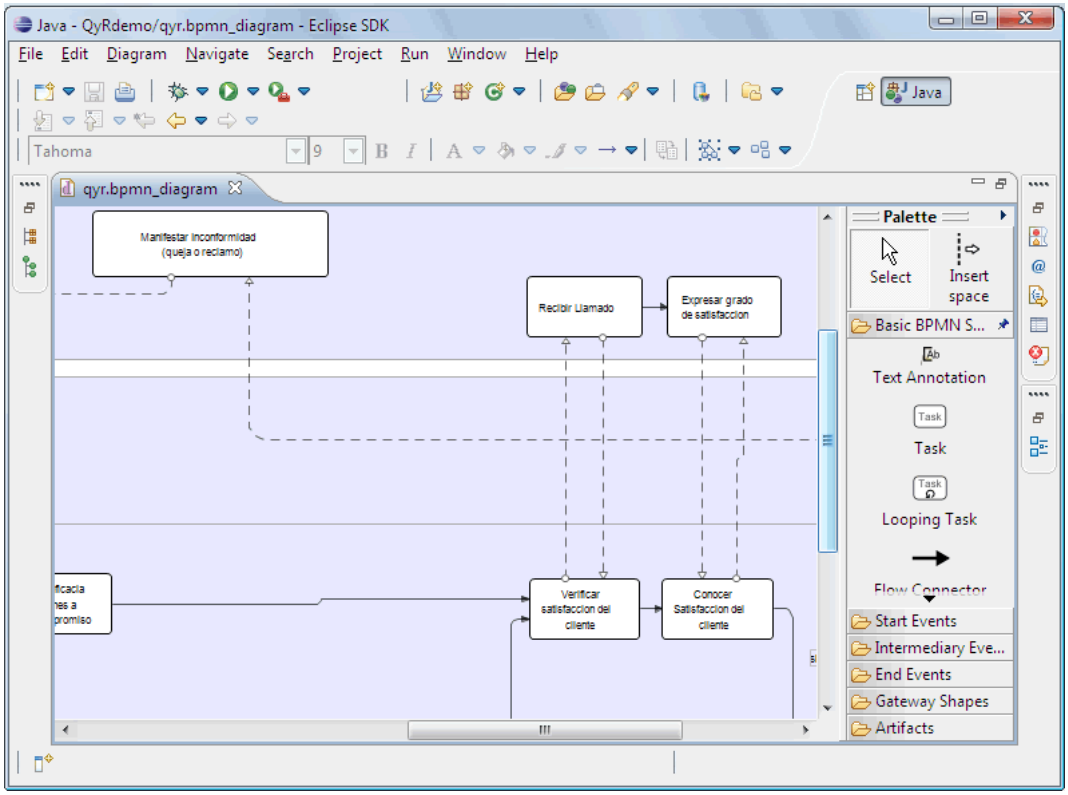


**Figura 18.** Modelo origen, construido por el usuario

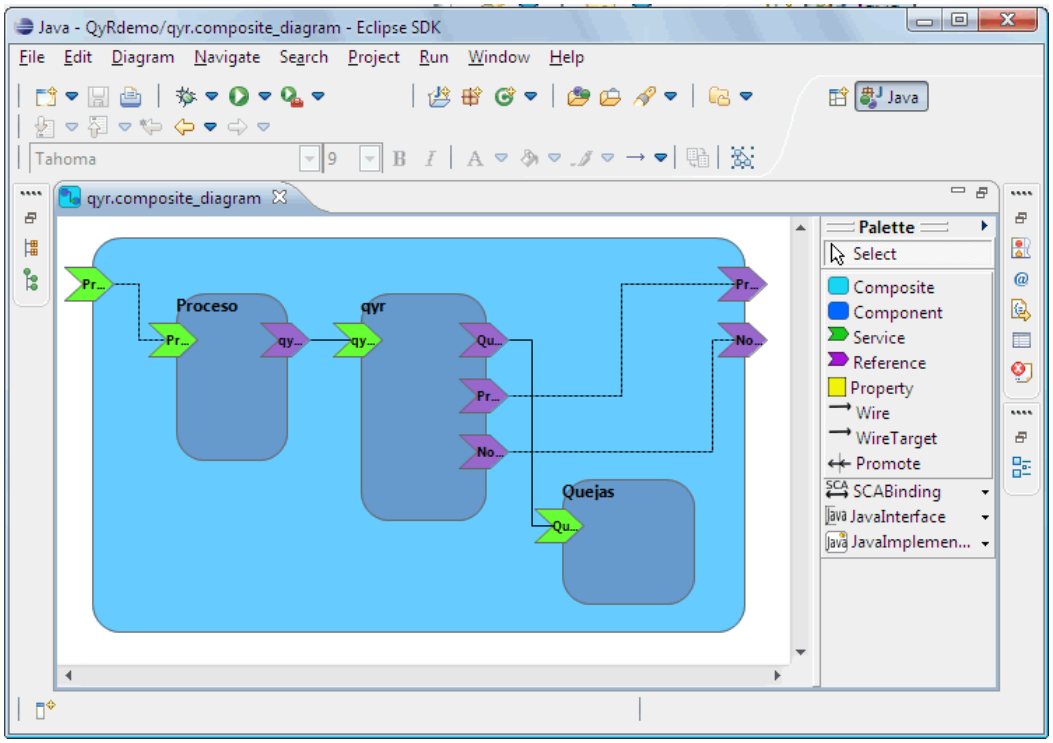


**Figura 19.** Modelo de SCA resultante

# CONCLUSIONES

A continuación quedan consignadas algunas conclusiones obtenidas luego de la ejecución del proyecto, en cuanto al cumplimiento de objetivos, la experiencia en el trabajo, la importancia del trabajo como contribución al estado del arte en las disciplinas que con éste se relacionan, y su aplicación inmediata en la industria en el entorno local, nacional e internacional.

## Los pilares de MDA funcionan

Como se estipuló en el anteproyecto, el objetivo general del trabajo fue "Proponer e implementar una especificación para mapear un modelo de procesos de negocio a un modelo de especificación de software bajo el enfoque MDA (Arquitectura Dirigida por Modelos)". Haber tomado ésta directriz hizo necesario un referente bibliográfico que definiera los principios básicos de MDA sobre los que se apoyara éste proyecto. Este fue encontrado en (Booch, et al., 2004), en donde los autores definieron los tres pilares (*tenets*) que conforman la base para MDA: Representación directa, Automatización y Estándares Abiertos. Para tener éxito, nuestra propuesta debería seguir dichos estándares.

Para una propuesta de mapeo de modelos de procesos de negocio a un modelo que especifica las características funcionales de una aplicación de software, la **representación directa** es crucial; nos presentó el reto de ofrecer un mecanismo en donde el usuario pudiera plasmar directamente la realidad de su problema la introducción de términos técnicos, además de otras necesidades complejas. Es por esto que se realizó un estudio comparativo de técnicas de modelado de negocio, teniendo en cuenta éstos requerimientos, del cual se concluyó que BPMN es la alternativa más procedente.

Sin embargo, también fue necesario tener en cuenta la verdadera propuesta de valor de MDA e Ingeniería de Modelos como conjunto de disciplinas y estándares, que promete la **automatización** de actividades a través de las diferentes etapas del proceso de desarrollo de software. Con la propuesta de éste trabajo, se procuró que el modelo destino de la transformación de modelos fuera un artefacto que no tuviera exclusivamente fines documentales, sino que sirviera como un elemento dentro de generación automática de componentes ejecutables. Es por esto que se eligió SCA como notación para el modelo destino, ya que a pesar de ser un estándar muy reciente sin implementaciones estables a la fecha, permite la especificación arquitectónica de aplicaciones compuestas de software, y muy pronto existirán soluciones open source para la generación automática de aplicaciones en plataformas como Java o PHP. Además de esto, SCA ha sido constituido alrededor de SOA como estrategia, de la que se habla hoy por hoy como el futuro de las aplicaciones de software empresariales.

Para lograr los objetivos del proyecto se aprovecharon al máximo los recursos de la comunidad Open Source, específicamente de Eclipse. Además de la credibilidad que aporta el uso de herramientas contribuidas por los líderes industriales pertenecientes a ésta comunidad, el factor más ventajoso es el soporte que ellas dan a los diferentes **estándares abiertos**, como BPMN, SCA y QVT, que fueron usados en éste proyecto. Estos estándares, concebidos en el seno de consorcios industriales de alto rango como OMG y Open SOA, aseguran que las soluciones nunca tendrán un carácter propietario y que estarán disponibles tanto para el aprovechamiento industrial como para la investigación y la academia, quienes también podrán contribuir a su mejoramiento.

De esa forma, concluimos que seguir éstos principios básicos de MDA trasciende el simple requisito de cumplirlos al igual que toda herramienta construida bajo éste enfoque, sino que por el contrario guían a los constructores a desarrollar

soluciones que de verdaderamente le aportan valor agregado a sus respectivos usuarios. Este es el caso de éste proyecto.

**Todo es experimental, pero es necesario la temprana adopción**

Durante la ejecución del proyecto, nos pudimos dar cuenta de que muchas de las herramientas y estándares utilizados no corresponden a soluciones probadas industrialmente, sino que corresponden a ideas que muchas veces aún se encuentran en un periodo de experimentación, pero cuya promesa de valor es tan grande que se hace imperativo la temprana adopción para disfrutar de sus beneficios en el futuro. Este es el caso por ejemplo del motor de transformación Operational QVT, descrito en la sección de arquitectura física del capítulo "Plug-in para la Transformación de Modelos", cuyo primer *release* experimental fue publicado en noviembre de 2007.

Debido al carácter experimental, la existencia de una comunidad activa de miembros dispuestos a apoyar se hace indispensable para lograr resultados satisfactorios para el desarrollo y adopción de nuevas tecnologías. En este caso, Eclipse nos ha mostrado el valor de comunidad que se desarrolla en el *open source*, pues el apoyo de sus miembros y la constante interacción posibilitan que nuevas ideas de carácter integrador como la de éste proyecto surjan.

**Todo es modelos**

Citando la frase de (Bezivin, 2003) y (Bezivin, Abril 2004), es definitivo el valor que tienen los modelos como artefactos dentro del proceso de desarrollo de software, tanto que podríamos hablar de llegar un nivel de madurez de éstos en los que la actividad de desarrollo consista exclusivamente en hacer modelos. Si bien en un punto de la historia se llegó a la afirmación global de que "todo es objetos" como mecanismo para modelar los problemas del mundo real, trabajos como éste nos

demuestran ahora que "todo es modelos", puesto que además de permitir plasmar la realidad y las diferentes vistas de una solución a través de ellos, son las semillas un proceso cada vez más automatizado de soluciones de software.

**SOA es la clave para el apoyo directo a los procesos de negocio**

Uno de los retos enfrentados en éste proyecto era hallar el enfoque o tipo de modelo para los modelos destino de la transformación, es decir, modelos de especificación de software, debido a las innumerables propuestas que han existido a lo largo de la historia. Sin embargo, el estudio del estado del arte nos enseño que el desarrollo de aplicaciones de software empresariales presenta hoy nuevas demandas, que las metodologías tradicionales de desarrollo, basadas exclusivamente en RUP y análisis y diseño orientado a objetos no pueden suplir. En la Arquitectura Orientada a Servicios (SOA), se encontró la solución a dichas demandas, debido a su naturaleza de ser explícitamente conducida por el modelo del negocio.

Una vez elegido éste paradigma, nos encontramos con el siguiente reto de encontrar un tipo de modelo que se ajustara a ésta metodología, y que cumpliera con los principios básicos del desarrollo de software dirigido por modelos. Al hacer un recorrido por las diferentes propuestas de consorcios industriales dedicados a SOA, encontramos SCA, la cual se acoplaba directamente al proyecto. Se puede concluir entonces que la importancia de SOA hoy por hoy no es estricta al ámbito académico o experimental, o que reside exclusivamente en los métodos que una empresa puede utilizar para apoyarse en las tecnologías de información, sino que cobra relevancia en el entorno general de la tecnología al integrarse con disciplinas emergentes como el desarrollo de software dirigido por modelos.

Durante el desarrollo del plug-in de la plataforma Eclipse que se construyó con el propósito de probar la propuesta de éste trabajo, se encontraron en los círculos

del software *open source* adeptos a la posibilidad de integrar SOA con la ingeniería de modelos, y el beneficio que esto le traería en términos de productividad a los desarrolladores de aplicaciones bajo éste paradigma. Es por lo tanto plausible concluir que SOA es el futuro del desarrollo de aplicaciones software empresariales.

**La transformación de modelos no es un proceso lineal**

En el transcurso del desarrollo de éste proyecto se ha visto que la transformación de modelos que propone MDA no es un proceso lineal que va desde un modelo CIM, hacia un PIM, luego un PSM y luego artefactos nativos de la plataforma, con un tipo de modelo en cada nivel; sino que en su lugar varios tipos de modelo pueden intervenir en un mismo nivel; y un modelo puede convertirse en el origen de dos procesos de transformación produciendo cada uno un modelo de tipo diferente, al igual que un modelo puede ser el resultado conjunto de dos modelos de transformación distintos.

**El sueño es realidad**

MDA y los enfoques relacionados surgieron a partir de los sueños de unos cuantos veteranos en el campo de las tecnologías y desarrollo de software al vislumbrar hacia donde se dirige la disciplina. Vieron posible que a través de los modelos era posible cerrar las actuales brechas que se pueden ver en las actuales metodologías de desarrollo de software, y aún más importante, automatizar hasta el nivel que sea posible. Sin embargo, la incógnita persiste: ¿es posible llegar a una solución de software ejecutable a partir de modelos independientes de la computación?

La gran conclusión a la que llegamos con éste trabajo es que éste sueño sí es posible; si bien aún nos falta para llegar allí, los pasos se están dando en la

dirección correcta. Los trabajos futuros sugeridos señalan otros esfuerzos por realizar que también aportarían hacia la realización de éste sueño, siempre y cuando se basen en los pilares de MDA y se encuentren en el marco de una comunidad que comparta similares objetivos.

# TRABAJOS FUTUROS

## Mapeos correspondientes a etapas posteriores de desarrollo

El actual proyecto se centró en el mapeo de un CIM (modelo independiente de la computación) a un PIM (modelo independiente de la plataforma). Sin embargo, para llegar a una solución de software ejecutable hacen falta los mapeos a posteriores etapas, los cuales serían de PIM a PSM (modelo específico de la plataforma), de PSM a artefactos nativos (código ejecutable) o incluso desde PIM a artefactos nativos. Se podría recomendar seguir la estructura de éste trabajo, al hacer un estudio del arte de las herramientas, metodologías y técnicas actuales, hallar los tipos de modelo más adecuados para el origen y el destino y elegir la estrategia de transformación más adecuada. En el contexto de éste trabajo, una propuesta concreta consistiría en el mapeo de modelos SCA (generados en éste trabajo) hacia modelos específicos de plataforma (PSMs) que soporten SCA, como Java o PHP en la actualidad.

## Ciclos completos transformacionales

Uno de los hechos que se vislumbraron en éste artículo es que el desarrollo de software dirigido por modelos no es un proceso de transformación lineal, no más bien multidimensional en donde modelos de diferentes tipos describen vistas a diferentes niveles de abstracción. Se puede hacer una propuesta conceptual acerca de éste modelo multidimensional de transformaciones, y como éste funcionaría en la práctica para cumplir el ciclo transformacional completo.

## Divulgación

En trabajos alrededor de tópicos emergentes en la industria de tecnologías de información como MDA, Ingeniería de Modelos, SOA, BPM, Modelado de Negocio

entre otros es crucial la divulgación. A la fecha, éste trabajo ha sido divulgado ante la comunidad industrial y académica internacional en dos ocasiones:

- Ponencia en el evento IDEAS 2008: XI Workshop Iberoamericano de Ambientes de Software e Ingeniería de Requisitos. “Estudio Comparativo de Técnicas de Modelado de Negocio”. Febrero 2008, Recife, Pernambuco, Brasil.
- Participación en Google Summer of Code 2008. Proyecto “Transformation of BPMN/BPEL to SCA models for STP (SOA Tools Platform)” como estudiante asociado a la organización Eclipse. Se está trabajando con Adrian Mos, investigador de INRIA, Grenoble, Francia.

## BIBLIOGRAFIA


**Adaptive Inc. 2007.** *Business Process Definition MetaModel (BPDM) (Final submission).* s.l. : OMG Document (http://modeldriven.org/web/bpdm), 2007.

**Agile Manifesto Signatories. 2008.** *Independent Signatories of The Manifesto for Agile Software Development.* s.l. : http://agilemanifesto.org/sign/display.cgi?ms=all, 2008.

**Ambler, Scott. 2002.** *Introduction to the Enterprise Unified Process (EUP).* s.l. : http://enterpriseunifiedprocess.com, 2002.

**Arsanjani, A. 2005.** *Empowering the Business Analyst for On Demand Computing.* s.l. : IBM Systems Journal, vol 44, no. 1, 2005.

**Bea Systems, Inc. 2006.** *Extending the Business Value of SOA through BPM.* 2006.

**Bezivin, Jean. Abril 2004.** *In Search of a Basic Principle for Model Driven Engineering.* s.l. : UPGRADE-Cepis (http://www.upgrade-cepis.org/issues/2004/2/up5-2Bezivin.pdf), Abril 2004.

**—. 2003.** *On The Unfication Power of Models.* s.l. : ATLAS Group, Universidad de Nantes, Francia (http://www.sciences.univ-nantes.fr/lina/atl/), 2003.

**Booch, Grady, y otros. 2004.** *An MDA Manifesto.* s.l. : Business Process Trends/MDA Journal, 2004.

**Business Modeling Forum. 2007.** s.l. : http://www.businessmodelingforum.com [Citada en Junio 2, 2007], 2007.

**Business Process Management Initiative. 2002.** *Business Process Modeling Notation (BPMN) Version 1.0 .* s.l. : http://www.bpmn.org/Documents/BPMN%20V1-0%20May%203%202004.pdf, 2002.

**Chappell, David. Julio 2007.** *Introducing SCA.* s.l. : DavidChappel & Associates, Julio 2007.

**Czarnecki, K y Helsen, S. 2006.** *Feature-based survey of model transformation approaches.* s.l. : IBM Systems Journal. Vol 45, No. 3, 2006.

**Datz, Todd. 2004.** *What You Need To Know About Service Oriented Architecture.* s.l. : Revista CIO, 2004. Enero 2004.

**Davenport, Thomas H. 1993.** *Process Innovation: Reengineering Work through Information Technology.* s.l. : Harvard Business School Press, 1993.

**Debevoise, Tom. 2005.** *Business Process Management with a Business Rule Approach.* s.l. : Business Knowledge Architects, 2005.

**Eclipse. 2007.** *About the Eclipse Project.* s.l. : http://www.eclipse.org/eclipse/, 2007.

**Eclipse EMF. 2007.** *Eclipse Modeling Framework.* s.l. : Eclipse, 2007.

**Eclipse GMF. 2007.** *Eclipse Graphical Modeling Framework.* s.l. : http://www.eclipse.org/gmf/, 2007.

**Eclipse M2M. 2008.** *ATL: Atlas Transformation Language.* s.l. : http://www.eclipse.org/m2m/atl/, 2008.

**—. 2008.** *Operational QVTO.* s.l. : http://wiki.eclipse.org/M2M/Operational_QVT_Language_%28QVTO%29, 2008.

**Emig, Christian, Weisser, Jochen y Abeck, Sebastian. 2007.** *Development of SOA-Based Software Systems – an Evolutionary Programming Approach.* s.l. : Universität Karlsruhe, Alemania, 2007.

**Emig, Christian, y otros. 2006.** *Model-Driven Development of SOA Services.* s.l. : Universität Karlsruhe, Alemania, 2006.

**Eriksson, HE y Penker, M. 2000.** *Business Modeling with UML.* 2000.

**Erl, Thomas. 2007.** *SOA: Principles of Service Design.* s.l. : Prentice Hall, 2007.

**Ghalimi, Ismael. 2007.** *Get Your BPMN Schema Today.* s.l. : IT Redux (http://itredux.com/blog/2007/04/04/get-your-bpmn-schema-today/), 2007.

**Giaglis, George. 2001 .** *A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques.* s.l. : International Journal of Flexible Manufacturing Systems, 2001 .

**Greenfield, J y Short, K. 2004.** *Software factories: Assembling Applications with Patterns, Models, Frameworks and Tools.* s.l. : Addison Wesley, 2004.

**Hagel, John III, Seely Brown, John. 2001.** *Cut Loose from Old Business Processes.* s.l. : Revista Optimize, 2001.

**HE Eriksson, M Penker. 2000.** *Business Modeling with UML.* 2000.

**IEEE. 1993.** *La Práctica recomendada para las Especificaciones de Requerimientos de Software.* s.l. : IEEE/ANSI Standard 830–1993, 1993.

**J Mendling, G Neumann, M Nuttgens. 2004.** *A Comparison of XML Interchange Formats for Business Process Modeling.* s.l. : Memorias de EMISA, 2004 (http://wi.wu-wien.ac.at), 2004.

**Kruchten, Philippe. 2003.** *The Rational Unified Process: An Introduction.* 2003.

**Larman, Craig. Octubre 2004.** *Applying UML and Patterns.* s.l. : Prentice Hall. Tercera Edicion. , Octubre 2004.

**Lombardi Software. 2007.** *Process Discovery: The First Step of BPM.* 2007.

**Mayer, Richard. Septiembre 1995.** *IDEF3 Process Description Capture Method Report.* s.l. : Knowledge Based Systems, Inc, Septiembre 1995.

**Merks, Ed. Agosto, 2004..** *Eclipse Modeling Framework.* s.l. : The Eclipse Series (http://www.eclipse.org/emf)., Agosto, 2004.

**Microsoft. 2008.** *Visual Studio Team System.* s.l. : http://msdn2.microsoft.com/en-us/teamsystem/default.aspx, 2008.

**OASIS WS-BPEL Technical Committee. 2007.** *Web Services Business Process Execution Language Version 2.0.* 2007.

**OMG. 2006.** *Business Process Modeling Notation Specification.* 2006.

**—. 2007.** *Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification.* s.l. : http://www.omg.org/cgi-bin/doc?ptc/2007-07-07, 2007.

**—. 2001.** *Meta Object Facility (MOF) Core Specification Version 2.0 [Documento Electrónico].* s.l. : http://www.omg.org/docs/omg/06-01-01.pdf., 2001.

**—. 2003.** *Object Management Group. UML 2.0 OCL Specification. .* s.l. : (Citada: 29 septiembre 2007) http://www.omg.org/docs/ptc/03-10-14.pdf [Documento Electrónico], 2003.

**—. 2003.** *Object Management Group: "Model Driven Architecture (MDA) Guide" Version 1.0.1 [Documento Electrónico]. http://www.omg.org/docs/omg/03-06-01.pdf.* 2003.

**—. 2003.** *UML 2.0 Superstructure Specification.* s.l. : OMG, 2003. (Citada: 10 agosto 2006) [Documento Electrónico] http://www.omg.org/docs/formal/05-07-04.pdf, 2003.

**Ortín, MJ, y otros. 2000.** *El Modelo de Negocio como Base del Modelo de Requisitos.* s.l. : Grupo de Investigación de Ingeniería del Software, Universidad de Murcia, España., 2000.

**Quintero, Juan Bernardo y Anaya, Raquel. 2007.** *Marco de Referencia para la Evaluación de Herramientas Basadas en MDA.* s.l. : Grupo de Investigación en Ingeniería de Software, Universidad EAFIT, 2007.

**Schmidt, Douglas. 2006.** *Model-Driven Engineering.* s.l. : Vanderbilt University, Nashville TN, USA., 2006.

**Smith, Howard. 2005.** *BPM Works.* s.l. : BPTrends (http://www.bptrends.com), 2005.

**—. 2005.** *From CIO to CPO via BPM.* s.l. : Computer Sciences Corporation (http://www.csc.com), 2005.

**Smith, Howard. Fingar, Peter. 2003.** *Business Process Management: The Third Wave.* s.l. : Meghan-Kiffer Press, 2003.

**Sprint Framework. 2006.** *A Guide to Role Activity Diagrams.* s.l. : http://www.sprint.gov.uk/pages.asp?id=77, 2006.

**Völter, M y Stahl, T. 2006.** *Model-Driven Software Development.* s.l. : John Wiley & Sons, 2006.

**WebMethods. Diciembre 2005.** *SOA: Revolutionizing IT Systems Architecture.* s.l. : http://www.webmethods.com, Diciembre 2005.

## ANEXO A: REPORTE TÉCNICO EVALUACIÓN MODELADO DE NEGOCIO CON EXTENSIONES DE ERICSSON Y PENKER PARA UML

La siguiente es una evaluación de los criterios definidos en la sección 3.2 para la comparación de técnicas de modelado de negocio, con la escala definida en la sección 3.3.

| Criterio | Justificación | Valor |
|---|---|---|
| R1 | El diagrama proporciona una vista del contexto del proceso de negocio exhibiendo sus entradas y salidas, los objetivos y las reglas que lo gobiernan, así como los eventos que disparan su ejecución. Sin embargo, la falta de detalle en la composición de los procesos de negocio no otorga el conocimiento requerido acerca del negocio. | 4 |
| R2 | En su artefacto principal proporciona una vista estructural, más no dinámica. Para esto se requiere añadir un artefacto de diagrama de secuencia. | 4 |
| R3 | Aunque la propuesta sí considera las reglas como un elemento importante de un modelo de negocio, éstas no son representadas en ningún diagrama. En su lugar, especifica que las reglas pueden ser declaradas en OCL (Object Constraint Language) en cualquiera de las vistas. | 3 |
| R4 | Al ofrecer un enfoque de caja negra, la propuesta no incluye el modelado de roles. Sin embargo, al modelar varios procesos que interactúan entre sí, es posible el uso de *swimlanes* para representar las áreas de negocio involucradas. | 2 |
| R5 | A través de objetos con estereotipos se expresan los artefactos que son insumos o productos (entradas y salidas) de los procesos. | 5 |
| R6 | La técnica no habla de interacciones B2B específicas, pero éstas se podrían modelar con swimlanes al igual que los roles. | 3 |
| R7 | A pesar de ser independiente de la computación, la propuesta presenta un conjunto de elementos de modelado que no resultan familiares para | 3 |

| | | |
|---|---|---|
| | una persona sin conocimiento de modelado de software, como objetos de estado con estereotipos. | |
| A1 | En su libro, Ericsson & Penker exponen metodológicamente como identificar los aspectos relevantes del negocio y modelarlos con su propuesta. | 4 |
| A2 | El modelo de negocio se expone como artefacto precedente a un modelo de casos de uso, pero no especifica cómo llegar hasta él. | 3 |
| A3 | La propuesta provee un precedente para el modelado de casos de uso, para dirigir la construcción de aplicaciones de software con un proceso de desarrollo tradicional. | 2 |
| A4 | El estándar no realiza ningún tipo de consideraciones sobre plataformas o estilos arquitectónicos. | 1 |
| E1 | Aunque la técnica está basada en UML, la propuesta por sí misma no constituye un estándar. Su medio de difusión es a través de la publicación de Ericsson & Penker, [Ericsson2000], y éste no representa un consenso industrial. | 3 |
| E2 | El metamodelo no es de dominio público, que es un perfil de UML, aunque puede consultarse en [Ericsson2000]. | 3 |
| E3 | No se encontró ninguna implementación del metamodelo de la técnica, pero debido a que corresponde a extensiones de UML, podría implementarse con una herramienta de metamodelado como EMF. | 2 |
| E4 | La técnica ha sido implementada en herramientas como *Enterprise Architect* o M1, pero no se encontró ninguna alternativa disponible *open source* para ver su estructura o para su manipulación, aunque ésta fácilmente se podría construir con los mecanismos de extensibilidad de UML (perfiles y estereotipos) con herramientas open source como ArgoUML. | 1 |
| **Total** | | **43** |

## ANEXO B: REPORTE TÉCNICO EVALUACIÓN MODELADO DE NEGOCIO CON BPMN

La siguiente es una evaluación de los criterios definidos en la sección 3.2 para la comparación de técnicas de modelado de negocio, con la escala definida en la sección 3.3.

| Criterio | Justificación | Valor |
|---|---|---|
| R1 | Exhibe los procesos de negocio con las actividades que lo componen según son modeladas por los analistas de negocio, otorgando así una representación independiente de la computación del dominio alrededor del cual se construirá una solución. | 5 |
| R2 | Se centra mucho en la dinámica, exhibiendo el flujo de ejecución de las actividades, pero en cuanto a la estructura solo muestra aspectos básicos como la definición de roles con *swimlanes* y organizaciones con *calles.* | 4 |
| R3 | BPMN no cubre directamente reglas de negocio, debido a que OMG ha propuesto otro estándar relacionado al respecto, SBVR (*Semantics of Business Vocabulary and Business Rules*). | 3 |
| R4 | Los roles, tanto al interior de la organización como en las interacciones negocio a negocio, son ilustrados a través de *swimlanes* y *calles,* y su comunicación es expresada a través de mensajes. | 5 |
| R5 | A través de mensajes entre participantes o roles, se definen los insumos o productos de las actividades del proceso. | 4 |
| R6 | Cada organización es representada a través de una *calle*, y al interior de ella las áreas o roles responsables son modelados con *swimlanes.* | 5 |
| R7 | La usabilidad por parte de analistas de negocio no técnicos fue pilar fundamental en la especificación de ésta técnica. | 5 |
| A1 | El estándar fue construido para ser "agnóstico de la metodología", de manera que no existe una propuesta metodológica concreta para construir modelos con ésta técnica. | 3 |
| A2 | Esta técnica, creada en el contexto de nuevas aproximaciones en TI | 5 |

| | | |
|---|---|---|
| | para la organización, provee mapeo directo hacia estándares de ejecución de procesos basados en tecnologías de carácter abierto, como Web Services. | |
| A3 | Los estándares de ejecución de procesos de negocio asociados a BPMN, tales como BPEL (*Business Process Execution Language*) y XPDL (*XML Process Definition Language*), posibilitan un acoplamiento directo a motores de ejecución. | 5 |
| A4 | Esta técnica fue propuesta con los principios de SOA (Arquitectura Orientada a Servicios) en donde las aplicaciones de software que soportan los procesos de negocio son un conjunto de componentes de servicio. Este paradigma se basa en la naturaleza distribuida de las aplicaciones empresariales. | 5 |
| E1 | BPMN fue inicialmente propuesto por BPMI (Business Process Management Initiative), organización que posteriormente se fusionó con OMG, cuyos estándares han sido el referente industrial del modelado en las últimas décadas con especificaciones como UML. La especificación está disponible en http://www.bpmn.org. | 5 |
| E2 | Actualmente hay dos propuestas de metamodelos para BPMN: El esquema BPMN de Intalio y BPDM. | 5 |
| E3 | Una implementación de ésta técnica de modelado es Eclipse STP-BPMN (http://www.eclipse.org/stp/bpmn), en donde el metamodelo ha sido implementado con EMF (Eclipse Modeling Framework), de una forma en que se puede visualizar gráficamente y extraer para uso en herramientas de transformación de modelos. | 5 |
| E4 | La herramienta anteriormente mencionada, es *open source*, como todos los proyectos de Eclipse. | 5 |
| **Total** | | **69** |

## ANEXO C: REPORTE TÉCNICO EVALUACIÓN MODELADO DE NEGOCIO CON DIAGRAMAS DE ACTIVIDADES DE ROLES

La siguiente es una evaluación de los criterios definidos en la sección 3.2 para la comparación de técnicas de modelado de negocio, con la escala definida en la sección 3.3.

| Criterio | Justificación | Valor |
|---|---|---|
| R1 | Permite modelar las diferentes actividades que los roles llevan a cabo para los procesos de negocio de la organización, obteniendo así una representación del dominio de negocio independiente de la computación. | 5 |
| R2 | Además de representar las actividades, exhibe las responsabilidades de los roles que las ejecutan, expresando así el aspecto estructural, sin embargo algunos elementos quedan por fuera. | 4 |
| R3 | Las reglas de negocio no son mencionadas en la propuesta. | 1 |
| R4 | Los roles son el aspecto fundamental de la propuesta, mostrando sus actividades, orden de ejecución, iteraciones, estados, entre otros elementos. | 5 |
| R5 | A través de estados e interacciones entre roles se mencionan las entradas y salidas de las actividades, pero no se hacen específicos en el diagrama. | 2 |
| R6 | Aunque el enfoque por roles predomina, no hay distinción entre interacciones al interior de la organización y entre negocios (B2B) | 3 |
| R7 | La técnica no es atractiva gráficamente para motivar su uso por parte de analistas de negocio no técnicos. | 3 |
| A1 | Aunque proporciona un conjunto de elementos de modelado bastante intuitivos, no presenta una guía concreta para construir el modelo. | 3 |
| A2 | El propósito de la técnica es la documentación y descripción de procesos, no su ejecución. | 1 |
| A3 | La técnica es utilizada para reingeniería de procesos, lo cual es un proceso robusto cuya ejecución no puede automatizarse en gran | 1 |

| | | |
|---|---|---|
| | medida. | |
| A4 | El estándar no realiza ningún tipo de consideraciones sobre plataformas o estilos arquitectónicos. | 1 |
| E1 | El ente que da el principal respaldo a esta técnica es SPRINT, un framework de herramientas del gobierno del Reino Unido para reingeniería de procesos. | 4 |
| E2 | No existe un metamodelo formal, ya que la técnica no fue diseñada para transformación de modelos. | 1 |
| E3 | No se encontró ninguna implementación del metamodelo. | 1 |
| E4 | No se encontró ninguna implementación de la técnica. | 1 |
| **Total** | | **36** |

## ANEXO D: REPORTE TÉCNICO EVALUACIÓN MODELADO DE NEGOCIO CON IDEF3

La siguiente es una evaluación de los criterios definidos en la sección 3.2 para la comparación de técnicas de modelado de negocio, con la escala definida en la sección 3.3.

| Criterio | Justificación | Valor |
|---|---|---|
| R1 | La característica principal de ésta técnica es la descomposición de los procesos de negocio en unidades funcionales, logrando así una captura de conocimiento independiente de la computación. | 5 |
| R2 | Tiene un gran énfasis en la vista de dinámica gracias a su propuesta de descomposición funcional, pero no hace representación alguna de la vista estructural. | 3 |
| R3 | La propuesta no reconoce reglas de negocio como un elemento de modelado, sin embargo habla de "reglas organizacionales" soportadas a través de bifurcaciones de flujos de actividades. | 1 |
| R4 | No se modelan los roles responsables de las actividades. | 1 |
| R5 | La propuesta no modela los productos o insumos de las unidades funcionales. | 1 |
| R6 | La propuesta no contextualiza las actividades de los procesos, ubicándolas en las organizaciones que las ejecutan. | 1 |
| R7 | Presenta un concepto muy simple para modelar procesos, el de unidades funcionales, pero no carece de otros elementos necesarios para el analista de negocio. | 3 |
| A1 | La metodología para construir el modelo de negocio es bastante simple - la descomposición de modelos. | 4 |
| A2 | El propósito de la técnica es la documentación y descripción de procesos, no su ejecución. | 1 |
| A3 | Las capacidades de la técnica se centran alrededor de la captura y documentación de procesos, pero en cuanto a automatización solo presenta capacidades para simulación, según las herramientas. | 2 |

| | | |
|---|---|---|
| A4 | Define un entorno de simulación de modelos para la captura de datos, pero no se refiere a las plataformas de ejecución de procesos. | 2 |
| E1 | La familia de estándares fue propuesta por la Fuerza Aérea de EEUU, siendo hoy de dominio público. Sin embargo, quienes se han ocupado de difundir el estándar han sido las entidades que fabrican herramientas de modelado con su especificación. | 4 |
| E2 | La especificación contiene una descripción formal de la técnica, pero no de una forma estructurada que constituya un metamodelo. | 3 |
| E3 | No se encontró una implementación de la técnica que proporcionara el metamodelo de forma accesible. | 1 |
| E4 | Aunque es un estándar abierto, desde sus inicios la firma KBSI (Knowledge Based Systems Inc., http://www.kbsi.com) ha desarrollado las únicas herramientas que actualmente lo soportan, PROCAP (captura de procesos) y PROSIM (simulación de procesos), las cuales son propietarias. | 1 |
| **Total** | | **33** |

## ANEXO E: REPORTE TÉCNICO EVALUACIÓN MODELADO DE NEGOCIO CON UML: PROPUESTA DE LA UNIVERSIDAD DE MURCIA

La siguiente es una evaluación de los criterios definidos en la sección 3.2 para la comparación de técnicas de modelado de negocio, con la escala definida en la sección 3.3

| Criterio | Justificación | Valor |
|---|---|---|
| R1 | La técnica provee una vista independiente de la computación, desplazando detalles de implementación tecnológica a otros artefactos de la propuesta (modelos de dominio y casos de uso). | 5 |
| R2 | A través del uso de actividades y objetos de información, la propuesta exhibe ambas vistas estructural y dinámica. | 5 |
| R3 | Especifica las reglas de negocio en un artefacto externo al diagrama: El glosario de reglas. | 3 |
| R4 | La propuesta hace obligatorio el uso de *swimlanes* del diagrama de actividades de UML, ilustrando así la interacción de roles a lo largo del proceso. | 5 |
| R5 | A través de objetos con estado, un elemento del diagrama de actividades de UML, se pueden modelar los artefactos relevantes con sus propiedades en cada paso del proceso. | 5 |
| R6 | Las interacciones entre organizacionales se representan con *swimlanes*, pero éstas no especifican las fronteras del negocio. | 4 |
| R7 | A pesar de ser independiente de la computación, la propuesta presenta un conjunto de elementos de modelado que no resultan familiares para una persona sin conocimiento de modelado de software, como objetos de estado con estereotipos. | 3 |
| A1 | La propuesta incluye una metodología que parte de los objetivos de la organización hasta la construcción de diagramas de procesos. | 5 |
| A2 | Utiliza el modelo de negocio para fabricar la especificación de sistemas que apoyen el negocio, con modelos y especificación de casos de uso, | 5 |

| | | |
|---|---|---|
| | reglas de negocio y requisitos no funcionales. | |
| A3 | La propuesta no presenta una solución estándar para la ejecución de procesos, sino que en su lugar presenta una base para la construcción de aplicaciones de software de con un proceso de desarrollo tradicional, basado en casos de uso. | 3 |
| A4 | El modelo de negocio es aplicable en el desarrollo de cualquier sistema, pero los artefactos posteriores propuestos (modelos conceptual y de casos de uso) son propios del paradigma orientado a objetos, que no toma en cuenta la naturaleza distribuida de las aplicaciones empresariales. | 3 |
| E1 | A pesar de encontrarse basada en UML, y de estar referenciada en varios trabajos, la propuesta de la Universidad de Murcia no representa un consenso industrial, sino más bien un trabajo académico, y por esta razón su uso y divulgación puede verse limitado. | 2 |
| E2 | Debido a su uso purista del diagrama de actividades UML, se puede utilizar el metamodelo MOF encontrado en la especificación de UML. | 5 |
| E3 | Debido a que solo es necesario el metamodelo del diagrama de actividades de UML, cualquier implementación de éste podría usarse para propósitos de transformación de modelos. Uno de ellos es UML2, de Eclipse (http://www.eclipse.org/modeling/mdt/?project=uml2). | 5 |
| E4 | La propuesta se puede implementar en cualquier herramienta open source UML que soporte el diagrama de actividades, como ArgoUML o Eclipse UML2 Tools. | 5 |
| **Total** | | **63** |

## ANEXO F: GUÍA DE USUARIO HERRAMIENTA DE TRANSFORMACIÓN DE CIM A PIM

Este plug-in le permite a un usuario analista de negocio modelar un proceso de negocio en BPMN y transformarlo en un modelo de arquitectura de componentes de servicio en SCA, como base para la construcción de una aplicación compuesta que soporte dicho proceso de negocio. Para abrir la aplicación, en el CD adjunto, dentro de la carpeta "Plug-in BPMN2SCA" abrir el ejecutable "eclipse.exe". Es necesario tener una instalación de la máquina virtual de Java (Java Runtime Environment v1.5 o posterior).

Si es la primera vez que se abre esta instalación de Eclipse, se va a preguntar la ubicación del *workspace*. Si desea ver el ejemplo usado en éste trabajo (proceso de Quejas y Reclamos), puede seleccionar la carpeta “demoworkspace/” incluida en el cd. Si desea construir un modelo desde cero, elegir la ubicación en donde se guardarán los modelos y continuar con los siguientes pasos.

**Paso 1: Construir un modelo de procesos de negocio**

1. Una vez abierta la plataforma Eclipse, crear un nuevo proyecto mediante el menú File, seleccionar New, y luego Project.
2. En el diálogo que aparece, seleccionar la categoría General y luego Project. Se le asigna un nombre (por ejemplo, "QyRdemo") y una ubicación al proyecto (puede dejarse la que coloca por defecto), y se hace click en Finish. El proyecto ha sido creado.
3. Para crear un nuevo modelo BPMN de procesos de negocio, se selecciona el proyecto creado, se hace click derecho y se selecciona New, y luego Other.

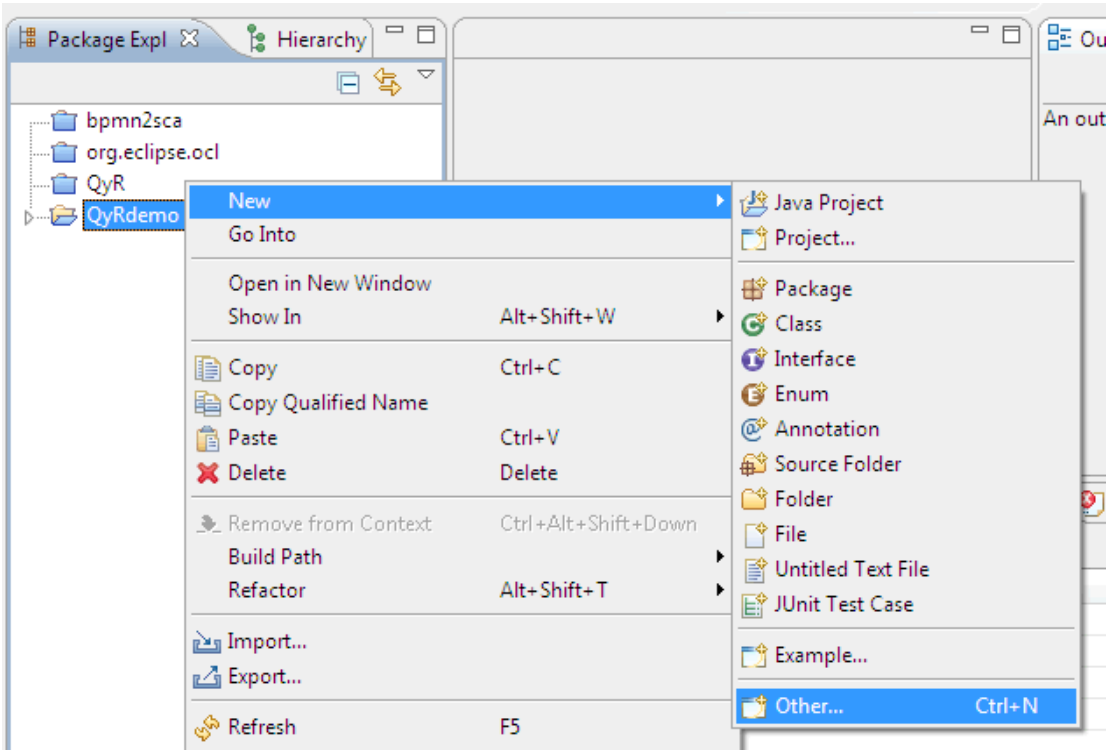


**Figura 20.** Crear un nuevo elemento para el proyecto

4. Del dialogo que se abre, seleccionar la categoría Other, y luego Bpmn Diagram

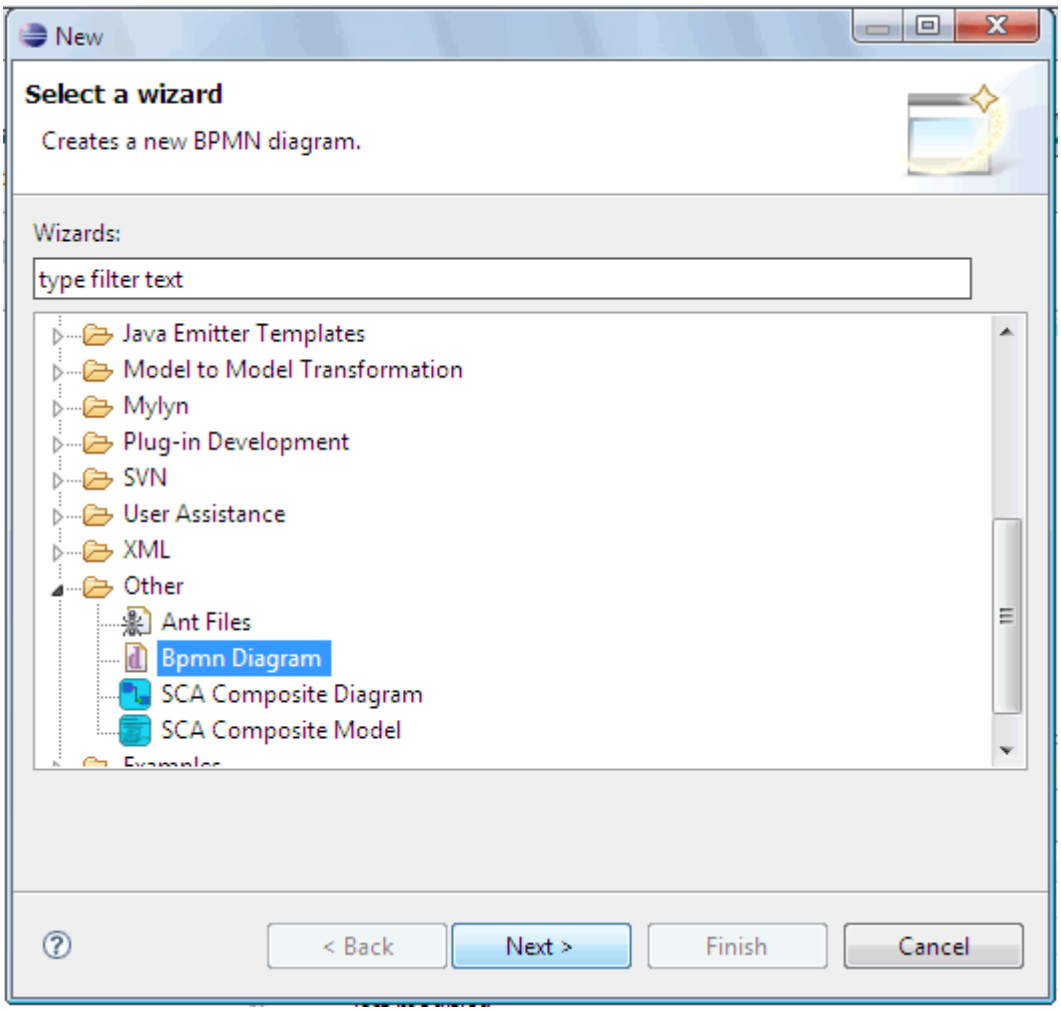


**Figura 21.** Crear un nuevo diagrama BPMN

5. Asignarle un nombre y click en Finish.
6. El editor de modelos BPMN se abre. El usuario puede comenzar a modelar su proceso ubicando elementos de la paleta BPMN sobre el canvas.

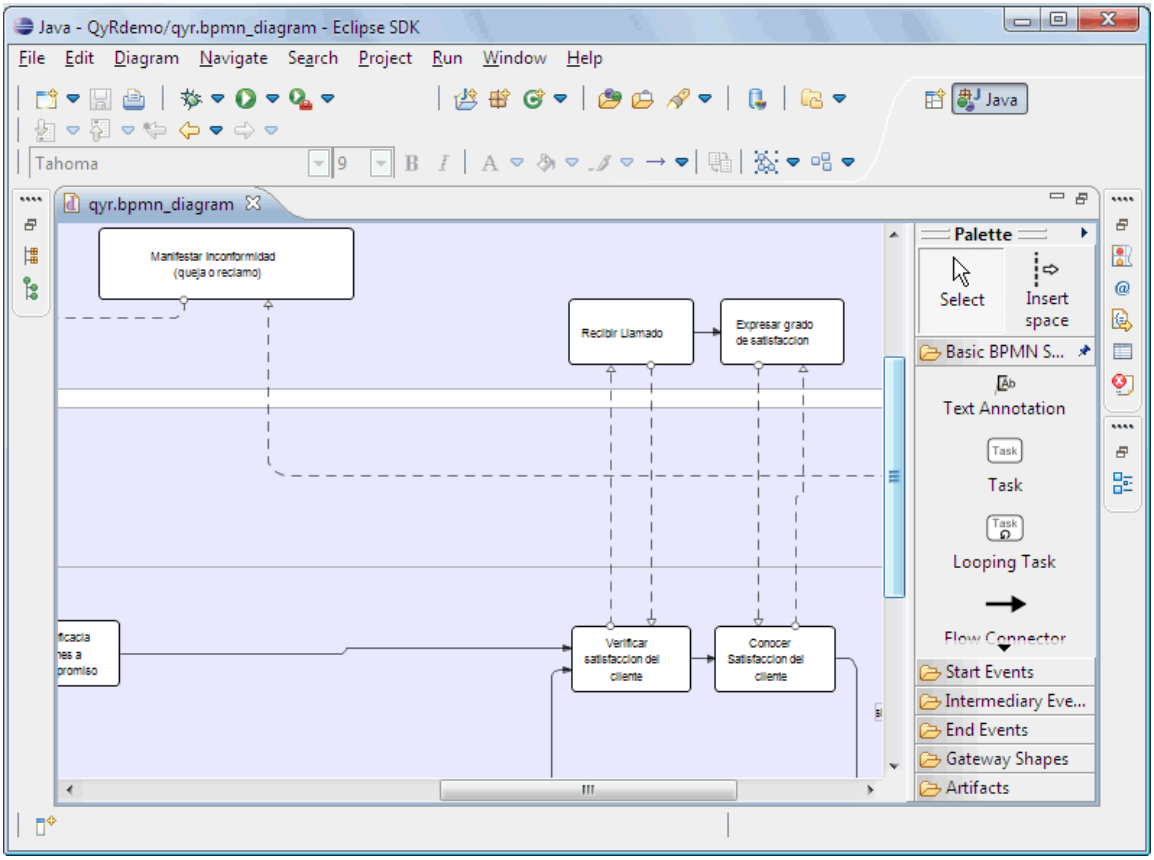


**Figura 22.** Editor de BPMN

Para más información acerca del uso del BPMN editor, consultar http://www.eclipse.org/stp/bpmn/.

**Paso 2: Anotar en el modelo las actividades automatizables**

1. A la luz de la propuesta metodológica de la transformación descritas en la sección "Heurísticas de Mapeo de CIM a PIM", el usuario debe decidir cuál de las actividades del proceso de negocio son automatizables y serán soportadas por componentes de software. En cada una de estas actividades, hacer click derecho y elegir "Set as Automatable". La actividad queda marcada con icono de un engranaje.

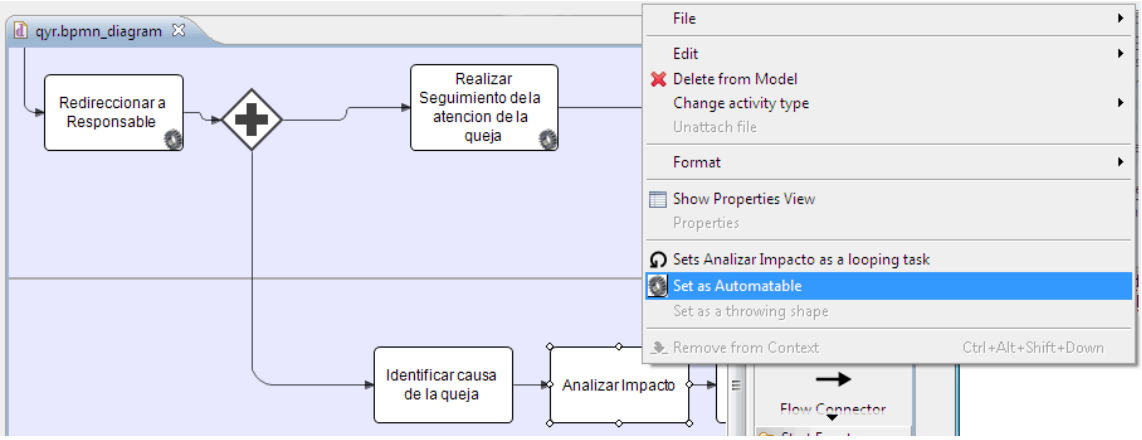


**Figura 23.** Configurando una actividad como automatizable

A continuación, para cada tarea automatizable:

2. Hacer click derecho sobre la actividad, y seleccionar Show Properties View.
3. En la vista Properties que aparece, seleccionar la pestaña Annotations.

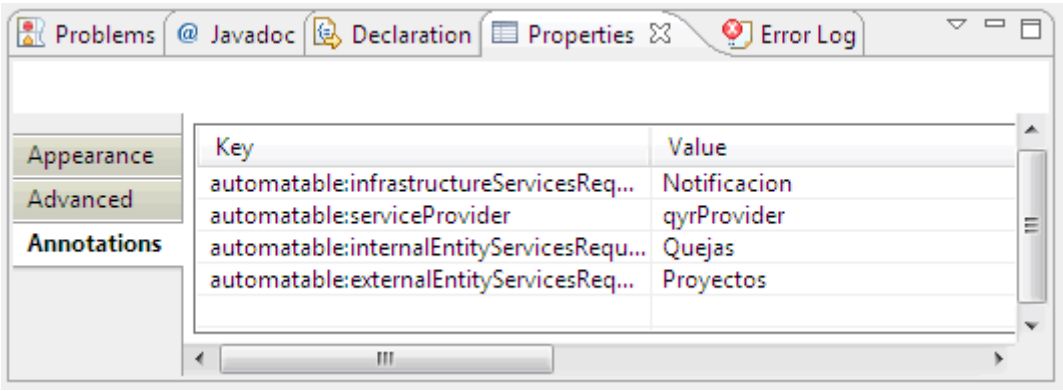


**Figura 24.** Vista de propiedades

4. Llenar la información requerida de acuerdo a la propuesta de heurísticas:
   - automatable:serviceProvider: Obligatorio. El nombre del componente proveedor de servicios que encapsula y expone la funcionalidad que da soporte a ésta actividad (ejemplo, QyRprovider).
   - automatable:internalEntityServicesRequired: Opcional. Una cadena separada por comas, nombrando los servicios de entidad internos

requeridos por ésta actividad. De acuerdo a la propuesta metodológica, son los objetos de negocio que requiere la implementación de ésta actividad definidos en la aplicación compuesta a construir (ejemplo, Quejas).

- automatable:externalEntityServicesRequired: Opcional. Una cadena separada por comas, nombrando los servicios de entidad internos requeridos por ésta actividad. De acuerdo a la propuesta metodológica, son los objetos de negocio que requiere la implementación de ésta actividad definidos en aplicaciones compuestas externas a la que se va a construir (ejemplo, Proyectos).
- automatable:infrastructureServicesRequired: Opcional. Una cadena separada por comas, nombrando los servicios de infraestructura requeridos por la implementación de ésta actividad (ejemplo, Notificación).

5. Guardar los cambios sobre el diagrama, seleccionando el menú File y luego la opción Save.

**Paso 3: Transformar el modelo a SCA**

Una vez el modelo de BPMN ha sido creado y anotado, puede transformarse a SCA. En el proyecto, hacer click derecho sobre el archivo .bpmn correspondiente al modelo creado (por ejemplo, qyr.bpmn), seleccionar BPMN to SCA Transformation y luego Transform to SCA model.

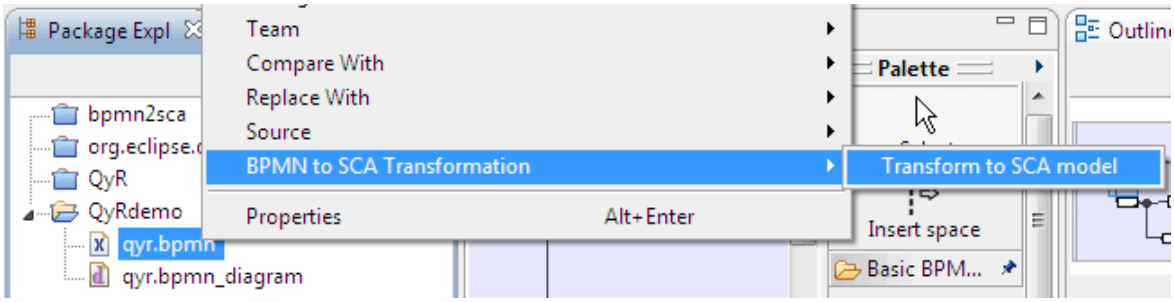


**Figura 25.** Transformando a SCA

El modelo SCA es obtenido. Es posible que se necesiten organizar los componentes gráficos manualmente.

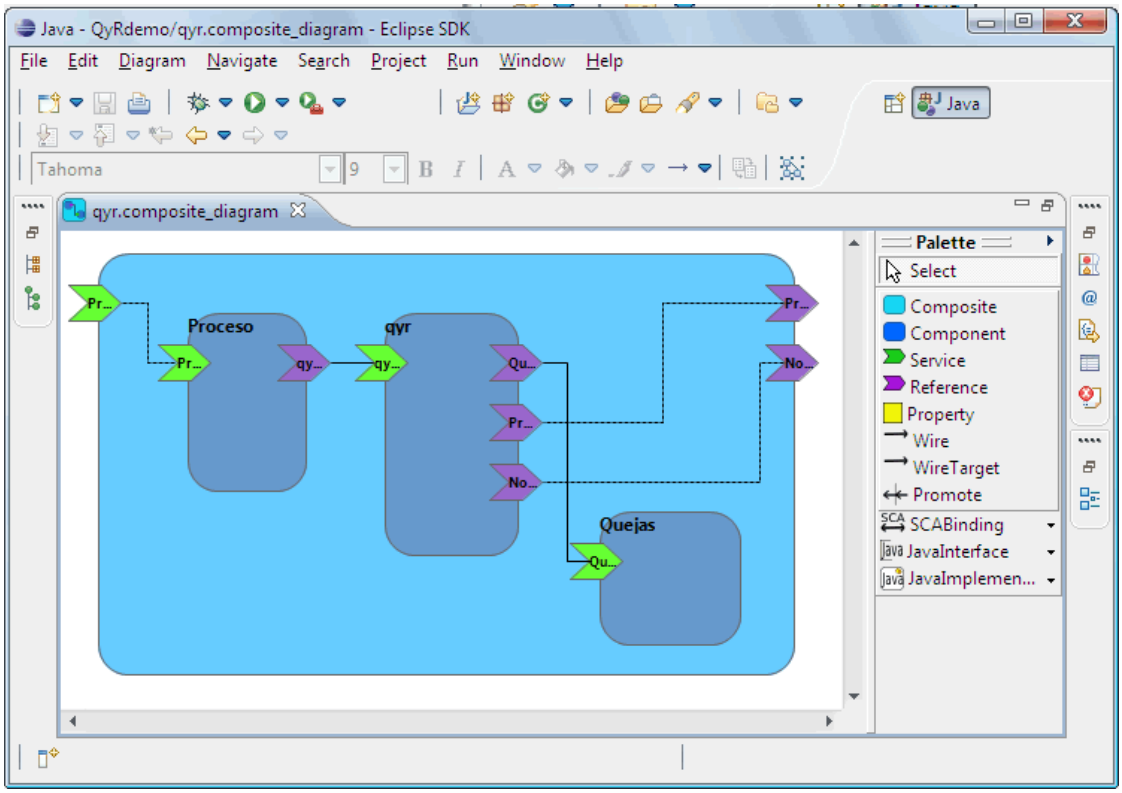


**Figura 26.** Modelo de SCA resultante