Universidad del Bío-Bío
Facultad de Ciencias Empresariales
Departamento de Ciencias de la Computación y Tecnologías de la Información
Escuela de Ingeniería Civil en Informática

Implementación de un prototipo de la Extensión dqBP en BPMN

Memoria para optar al Título de
Ingeniero Civil en Informática

Autor: Guillermo Ignacio Fuentes Quijada

Profesor Guía: María Angélica Caro Gutiérrez
Chillán, Junio 2015

Resumen

Hoy en día las empresas prestan mucha atención a sus procesos de negocio, ya que se han vuelto un recurso valioso y permiten lograr ventajas sobre sus competidores. Consecuentemente con esto, se han creado diversos lenguajes y notaciones que permiten desarrollar el modelado de procesos de negocio, facilitando de esta manera, la identificación y el entendimiento de ellos. Por otra parte, las organizaciones consideran como un factor importante la calidad de datos, ya que ésta incide en el éxito o fracaso de las tareas realizadas por la empresa.

En este trabajo, se busca expresar la calidad de datos en los elementos de una de las principales notaciones de modelado de procesos de negocio, BPMN (Business Process Model and Notation), logrando de esta manera la especificación del proceso de negocios en conjunto con los requisitos de calidad.

Para lograr esta especificación se debe analizar las herramientas de modelado de código libre disponibles, para luego construir un prototipo que implemente la especificación de requisitos de calidad en el software seleccionado para este fin.

Abstract

Nowadays, the companies are concerned about their business process, because they have become a valuable resource and allow to gain advantages over their competitors. Consistent with this, they have created several languages and notations that develop modeling business process, thereby facilitating the identification and understanding of them. Moreover, the organizations regard as an important factor the data quality because this influences on the tasks success or failure performed by the companies.

On this research, we seek to express the data quality on the elements of one of the major modeling business processes notations, BPMN (Business Process Model and Notation), thus achieving the specification of the business process together with the quality requirements.

To achieve this specification we have to analyze the modeling tools free open source available, and then build a prototype that implements the specification quality requirements in the software selected for this purpose.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Toda empresa u organización, busca mejorar sus procesos, específicamente de sus negocios, ya que permiten mejorar su gestión y por tanto lograr ventajas respecto a sus competidores. Partiendo de este punto, se ha empezado a modelar los procesos de negocio para mejorarlos y hacerlos más eficientes.

Debido a lo anterior, se han especificado notaciones para modelar procesos de negocio, siendo una de las más usadas, Business Process Modeling and Notation (BPMN), esta notación permite modelar procesos de negocio y es la más cercana a los analistas de negocio, en las organizaciones. Otra de las notaciones disponibles para llevar a cabo esta tarea, son los Diagrama de Actividad, muy similar a la notación BPMN, e incluso podemos decir que son equivalentes.

Hoy en día, en el mercado nos encontramos con muchas herramientas, que modelan los procesos de negocio, con alguno de los lenguajes disponibles para este objetivo.

Por otra parte, en la actualidad, también podemos evidenciar que la Calidad de Datos de los elementos que iteractúan en las organizaciones se ha vuelto un factor incidente en el éxito general de estas.

Por tanto podemos señalar, que entre los factores claves para que una organización obtenga el éxito, se puede visualizar; el modelado de procesos de negocio y la Calidad de Datos en los elementos de datos, que interactúan en estos procesos. Por lo cual, algunos autores definen que el modelar la Calidad de Datos tempranamente en un proceso de negocio, facilitará la detección de problemas y, finalmente, el éxito de la organización.

En este trabajo, se busca extender una herramienta para el modelado de procesos de negocio, de tal forma, que permita especificar requisitos de Calidad de Datos en ellos. La elección de la herramienta a extender se basará en el análisis de diversos criterios, entre ellos, la cualidad de ser de código libre.

1.1. Objetivo General

Avanzar hacia la implementación de la extensión *dqBP* en una herramienta para modelar Procesos de Negocio, la cual use como notación BPMN 2.0 y de esta forma lograr el primer paso hacia la especificación de requisitos de Calidad de Datos.

1.2. Objetivos Específicos

- Investigar y seleccionar la herramienta (para modelar Procesos de Negocio) más adecuada para su posterior extensión.

- Modificar la herramienta seleccionada de modelado BPMN, generando un prototipo, el cual permita la extensión *dqBP* y lograr de esta forma, incorporar un flag para la representación de Calidad de Datos (en algunos de los elementos de la notación BPMN 2.0) y especificar algunas propiedades de los elementos asociados a la extensión.

- Permitir, mediante la utilización del prototipo generado, el procesado automático del modelo de Proceso de Negocio, con el objetivo de incorporar la Calidad de Datos, al código fuente del modelo.

1.3. Enfoque

Este trabajo, tiene un enfoque cualitativo, ya que en base a ciertos criterios definidos, se evalúan ciertas herramientas que permiten el modelado de procesos de negocio, de esta forma realizar la selección de una herramienta de modelado e implementación de un prototipo de la extensión dqBP.

1.4. Estructura del Trabajo

En el Capítulo 2 se relatan los antecedentes de este trabajo, así como la descripción a fondo de la extensión *dqBP*. Luego, el Capítulo 3 realiza un análisis de la tecnología disponible, que permite desarrollar esta extensión. En el Capítulo 4 se describe la herramienta seleccionada para implementar la extensión, con el objetivo de conocer las características y facilidades de implementación de la misma. En el Capítulo 5 se describe la implementación de la extensión *dqBP* en la herramienta seleccionada. El Capítulo 6, describe las pruebas, que demuestran el funcionamiento de la extensión. Finalmente, en el Capítulo 7 se entregan las Conclusiones de este trabajo.

Capítulo 2

Antecedentes

2.1. Conceptos Relacionados

Proceso de Negocio

Un Proceso de Negocio (en inglés *Bussines Process* (BP)) es un conjunto de uno o más procedimientos o actividades vinculadas, que son ejecutadas siguiendo un orden predefinido y que en conjunto persiguen una meta o política de empresa, normalmente en el contexto de una estructura organizativa que define los roles funcionales o relaciones (Chinosi and Trombetta, 2012).

Gestión Procesos de Negocio

La Gestión Procesos de Negocio (en inglés *Bussines Process Management* (BPM)) es la que proporciona gobernabilidad del entorno a los procesos de una empresa para mejorar la agilidad y el rendimiento operativo. Es un enfoque sistemático para mejorar los BP de cualquier organización. BPM no es una tecnología y no esta relacionada con los diagramas y arquitectura de los sistemas (Chinosi and Trombetta, 2012).

Modelado de Procesos de Negocio

El Modelado de Procesos de Negocio (en inglés *Bussines Process Modeling* (BPM)) se define como el período de tiempo cuando los flujos de trabajo (*workflow*) manuales y/o automatizados, que es descrito en un proceso, es definido y/o modificado electrónicamente. Dado que el Modelado de Procesos de Negocio y la Gestión de Procesos de Negocio comparten

el mismo acrónimo (*BPM*), estas actividades son confundidas entre sí. El Modelado de Procesos de Negocio es la actividad de representación de BP de una empresa, por lo cual el proceso actual puede ser analizado y mejorado en el futuro (Chinosi and Trombetta, 2012).

Notación y Modelado de Proceso de Negocio

El principal objetivo de la Notación y Modelado de Proceso de Negocio (en inglés *Bussines Process Modeling and Notation* (BPMN)) es proporcionar una notación que sea fácilmente comprensible por los usuarios de Negocio, los que van desde los analistas de negocio hasta los desarrolladores técnicos.

BPMN fue publicado originalmente en el 2004, por la iniciativa de *Business Process Modeling* como una notación gráfica (parcialmente inspirada en los Diagramas de actividad de UML (*Unified Modeling Language*, en español Lenguaje de Modelado Unificado) para representar la disposición gráfica de los Procesos de Negocio. La cada vez mayor adopción por parte de las empresas y el creciente interés en la notación, causaron la adopción de BPMN como estándar de la OMG (*Object Management Group*), en el 2006 (ver la Figura 2.1) (Chinosi and Trombetta, 2012).

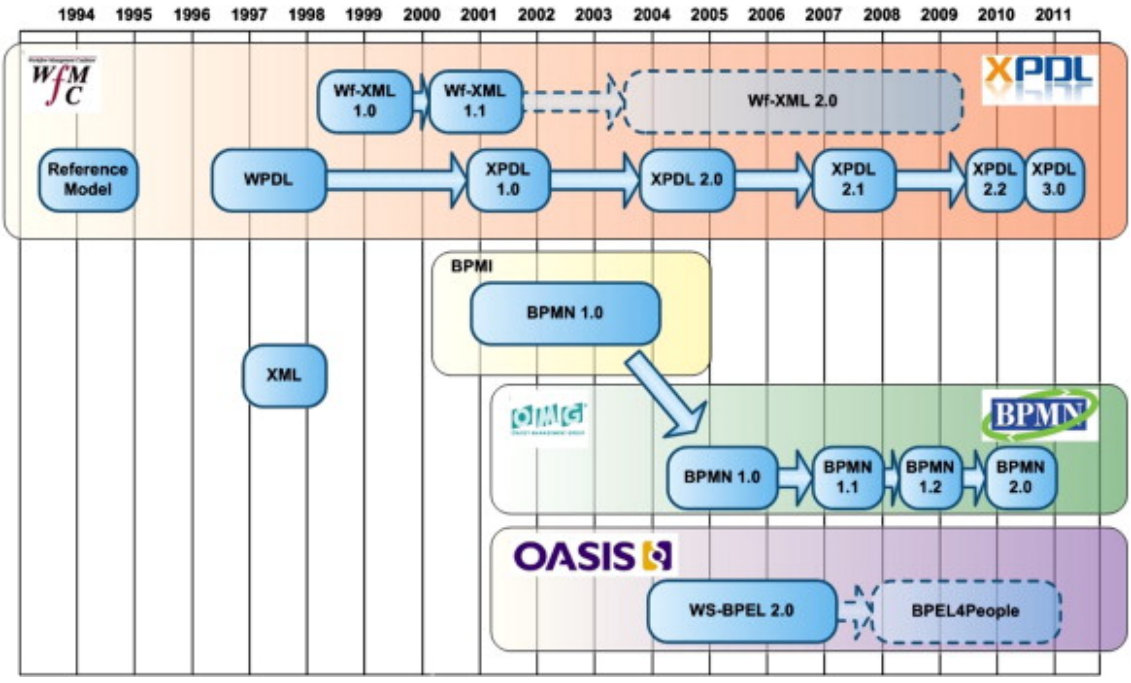


Figura 2.1: Línea de Tiempo del BPMN.

BPMN proporciona una notación gráfica para representar un proceso de negocio. Las versiones 1.x de BPMN no tenían una semántica claramente definida, ni un formato de serialización nativa. Sin embargo, BPMN 1.1 introdujo una descripción de la notación de los Diagrama de Clases UML, para dar una mejor formalización a la versin original, pero no fue suficiente para afirmar que BPMN 1.x tenía un meta-modelo bien definido.

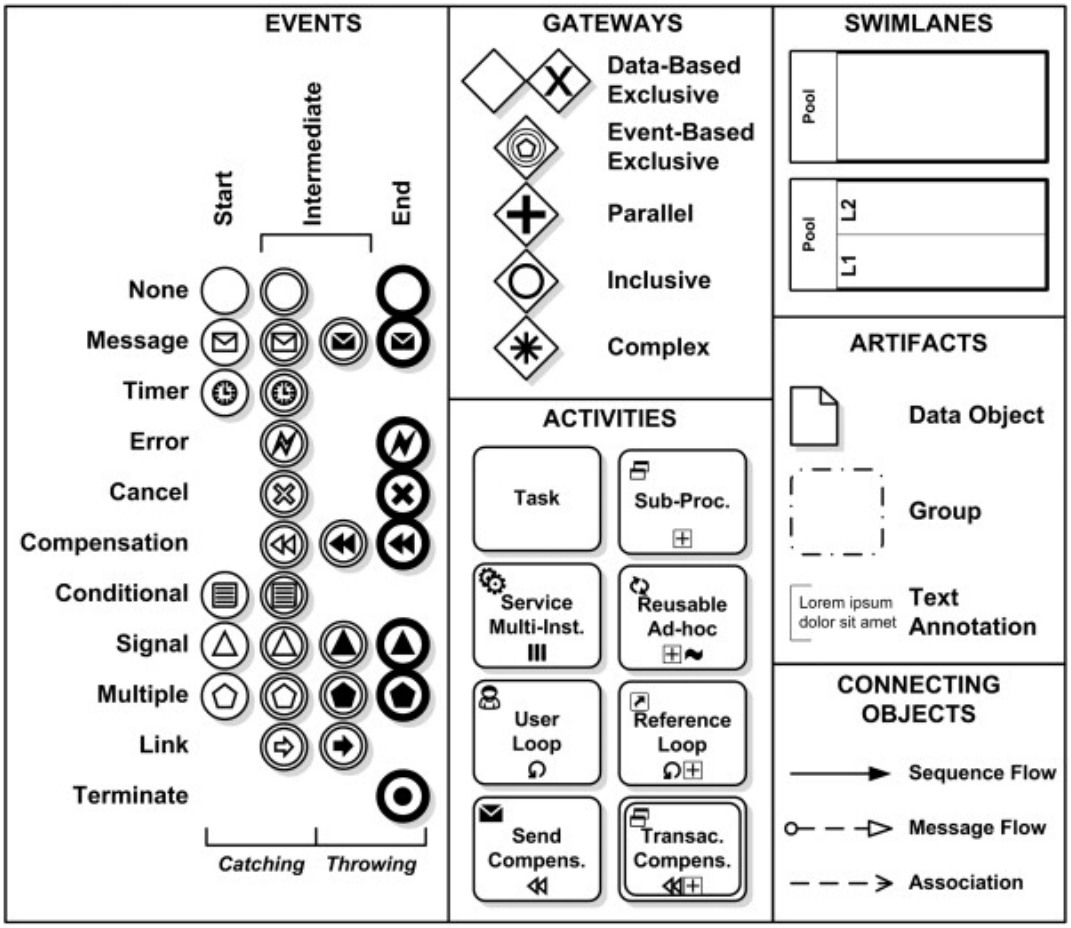


Figura 2.2: Resumen de Elementos de BPMN 1.2.

BPMN 2.0

BPMN 2.0 es la última versión de *Business Process Modeling and Notation*, la cual difiere notablemente en muchos aspectos con las versiones anteriores, ya sea agregando nuevas características o cambiando las propiedades de elementos definidos en las versiones anteriores.

La especificación de BPMN 2.0 amplía el alcance y las capacidades del BPMN 1.2 en varias áreas: (i) se formaliza la semántica de ejecución de todos los elementos de BPMN, (ii) define un mecanismo de extensibilidad para ambas extensiones modelo de procesos y extensiones gráficas, (iii) refina la composición de eventos y su correlación, (iv) se extiende la definición de las interacciones humanas, (v) define los modelos de Coreografía y de conversación (un medio para una mejor interacción de modelado), y (vi) también resuelve conocidas inconsistencias y ambigüedades de BPMN 1.2. Además, BPMN 2.0 define un meta-modelo y un modelo de definición del diagrama, junto con el acompañamiento de formatos de intercambio, tanto XMI y XSD basado (Chinosi and Trombetta, 2012).

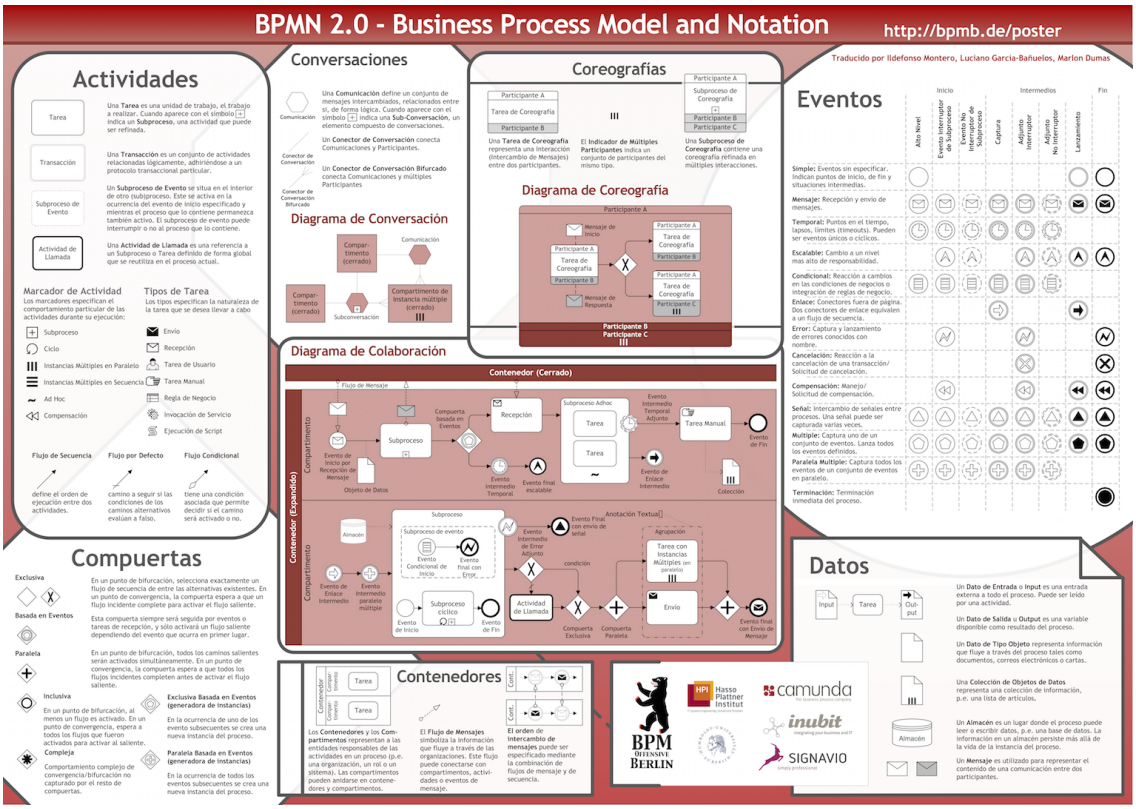


Figura 2.3: Poster BPMN 2.0 (Berlin, 2013).

Calidad de Datos

La Calidad de los Datos o Calidad de Información (suelen usarse indistintamente aunque sean conceptos diferentes) (en inglés Data Quality (DQ)) es un concepto multidimensional

y frecuentemente es definida como "datos apropiados para el uso". Esto quiere decir, que el usuario es quien determina si un conjunto de datos, usados en una determinada tarea y en un contexto específico, pueden ser usados para el objetivo previsto. La norma ISO/IEC 25012 define calidad de datos como "Grado en que las características de los datos satisfacen necesidades implícitas y establecidas cuando son usados en condiciones específicas". Lo anterior, da un papel relevante a la participación del usuario a la hora de definir si un conjunto de datos es de calidad (Caro et al., 2013).

Norma ISO/IEC 25012

Esta norma presenta un modelo genérico de calidad de datos (ISO/IEC-25012, 2008). Plantea que la gestión y mejora de los datos es importante para abordar situaciones como:

- Adquisición de datos en organizaciones donde la calidad del proceso de producción de datos es desconocido o débil.
- Existencia de datos defectuosos que contribuyen a generar información insuficiente, que provoca resultados inutilizables y clientes insatisfechos.
- Dispersión de datos entre varios propietarios y usuarios. Lo que puede implicar la falta de una visión coherente e integrada, necesaria para garantizar la interoperabilidad y la cooperación.
- La coexistencia de sistemas heredados con sistemas modernos. Sistemas de Información (SI) donde los datos cambian con frecuencia y su integración con otros datos es relevante (por ejemplo, SI en la Web).

Teniendo en cuenta estas situaciones, y dado que el ciclo de vida de los datos es a menudo más largo que el ciclo de vida del software, el modelo de DQ propuesto por la ISO pretende responder a estas necesidades contribuyendo a:

- Definir y evaluar los requisitos de DQ en procesos de adquisición, producción e integración de los datos.
- Identificar los criterios de garantía de DQ, útiles también para la re-ingeniería, evaluación y mejora de los datos.
- Evaluar la conformidad de los datos con la legislación y/o requisitos.

2.2. Contexto

El marco teórico en que se realiza este trabajo, es lo realizado por Rodríguez and Caro (2012), en donde desarrollan un método para la obtención de Requisitos de Software centrados en la Calidad de Datos, de aquí nace la extensión *dqBP* de BPMN, permitiendo la representación de requisitos de Calidad de Datos en un modelo de BP.

Este trabajo se enmarca en la implementación de la extensión *dqBP*. A continuación, se expone el trabajo de Rodríguez and Caro (2012).

2.2.1. BPiDQ*: Un método para la obtención de requisitos de software centrados en DQ desde especificaciones de BP

El objetivo de BPiDQ* es soportar la especificación temprana de requisitos de DQ en BP y, a partir de dicha especificación, obtener requisitos de software centrados en la DQ, expresados como casos de uso. En la Figura 2.4, en color gris, se resume el conjunto de elementos que forman parte de esta propuesta. Concretamente, en la parte central se muestra el nuevo método BPiDQ*, con sus cuatro etapas, la extensión *dqBP* que permite agregar requisitos de DQ en modelos de BP descritos con BPMN, repositorios tanto para las actividades de calidad de datos como para los casos de uso dedicados a representar los requisitos de calidad de datos y, finalmente, los casos de uso que se derivan de la aplicación del método BPiDQ*.

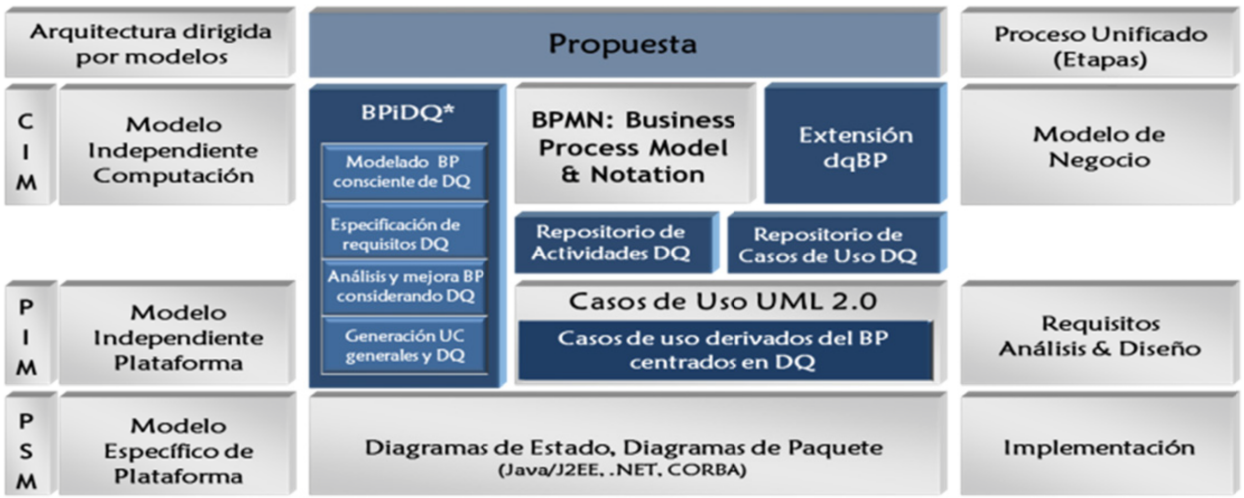


Figura 2.4: Vista General de BPiDQ*.

Componentes del método BPiDQ*

Para que la aplicación del método BPiDQ* sea posible es necesario contar con tres componentes que apoyan las etapas definidas. Estos componentes son la extensión de la notación BPMN, un conjunto de actividades relacionadas con el tratamiento de los requisitos de calidad de datos a nivel de proceso de negocio y un catálogo de casos de uso estándar que permiten abordar las especificaciones de DQ.

a) La extensión dqBP

La extensión dqBP tiene por objetivo agregar capacidad expresiva a la notación BPMN 2.0, permitiendo la representación de requisitos de DQ en un modelo de BP. En la Figura 2.5 se muestra el meta-modelo en que aparece la nueva clase dqFlag y el vínculo que ésta tiene con los elementos de BPMN.

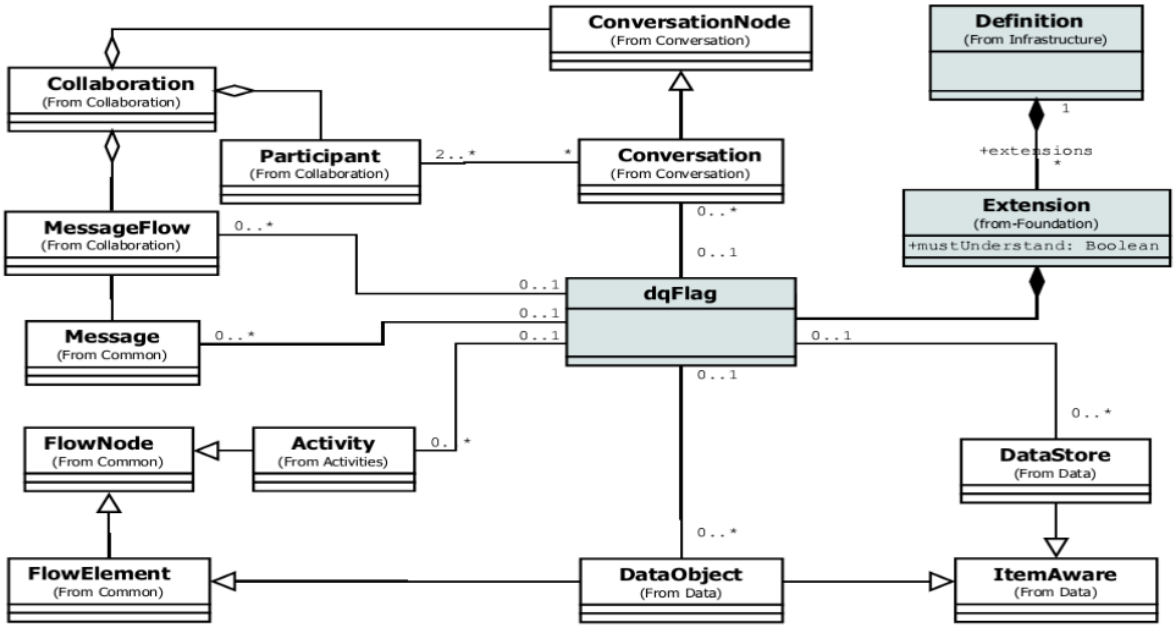


Figura 2.5: Extensión de BPMN 2.0 para incluir aspectos de calidad de datos.

Dado que BPMN es una notación en que se privilegia la representación simbólica de los distintos aspectos del negocio, se ha asociado un símbolo a la clase dqFlag que consiste en la fusión de las letras DQ (DQ). Este símbolo deberá ser usado para marcar los elementos de BPMN en los cuales es posible asociar requisitos de calidad de datos.

La forma en que se representa este nuevo símbolo en conjunto con los elementos de BPMN y el significado de dicha representación se muestran en la Tabla 2.1.

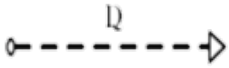
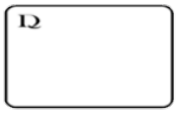
Vista gráfica	Significado
 ***Message***	Representa los datos contenidos en un mensaje, los cuales deberían satisfacer ciertos requisitos de DQ necesarios para el éxito del proceso de negocio. Por ejemplo, completitud y consistencia en una receta médica enviada por el médico al paciente
 ***Message Flow***	Representa los datos implícitos en un flujo de mensaje, los cuales deberían satisfacer ciertos requisitos de DQ. Por ejemplo, la vigencia de una autorización para una tarjeta de crédito.
 ***Conversation***	Representa los datos que están contenidos en una conversación compuesta por un conjunto de mensajes, los cuales deberían satisfacer ciertos requisitos de calidad de datos. Por ejemplo, seguridad y exactitud en los datos intercambiados entre los clientes y una línea aérea durante la compra de los billetes del vuelo.
 ***Data Object***	Representa los datos que se encuentran contenidos en un objeto de datos. Estos datos deberían satisfacer ciertos requisitos de calidad de datos, como por ejemplo, completitud, consistencia y exactitud de los datos (nombre, dirección) necesarios para entregar un paquete a un cliente.
 ***Data Store***	Representa los datos contenidos en una base de datos, los cuales deberían satisfacer ciertos requisitos de calidad de datos, como por ejemplo, completitud de los datos actualizados acerca de una venta de productos.
 ***Activity***	Representa los datos que son usados y/o producidos en una actividad. Estos datos deberían satisfacer ciertos requisitos de calidad de datos como por ejemplo, precisión y exactitud acerca de los presupuestos de gastos generados como salida de la actividad.

Tabla 2.1: Elementos de BPMN y la especificación de requisitos de DQ.

b) El repositorio de actividades de DQ

El segundo componente es un repositorio que contiene actividades en el nivel de BP orientadas a satisfacer requisitos de DQ. Un requisito de calidad de datos expresado en el modelo de BP con el símbolo (DQ-Flag) puede estar compuesto por una o más dimensiones de DQ. Cada una de las dimensiones de DQ se asocia a un conjunto de actividades de DQ contenidas en el repositorio. En la Tabla 2.2 se muestran, a modo de ejemplo, las dimensiones de DQ exactitud, oportunidad y completitud. Para cada una de ellas se entrega una definición de acuerdo con diferentes autores, un conjunto (no completo) de actividades que se podrían incluir en el modelo de BP para la mejora del mismo, teniendo en cuenta la DQ, y algunos ejemplos de la aplicación de estas actividades en el contexto de un BP.

Dimensiones DQ	Actividades de Mejora	Ejemplos
Exactitud: Grado en que los datos reflejan una vista del mundo real en un contexto y un proceso de negocio específico	Determinar el conjunto de datos que requieren exactitud. Definir el dominio válido para los datos. Verificar los datos con el dominio correcto. Verificar los datos en distintas fuentes. Limpiar las bases de datos para alcanzar los niveles de exactitud requeridos. Mejorar los datos hasta alcanzar la exactitud requerida.	El precio recibido por el cliente para una reserva de hotel debe ser exacto. En una prescripción médica, el nombre del medicamento puede ser confrontado con el Vademecum. El peso del paquete a entregar debe estar en el rango predeterminado.
Oportunidad: Grado en que los datos están lo suficientemente actualizados y disponibles como para ser útiles en un contexto y en un proceso de negocio específico.	Verificar si el dato tiene requisitos de vigencia para una determinada tarea. Para diferentes fuentes de datos, seleccionar una que provea los datos con la actualización requerida por el proceso. Verificar que el dato sea entregado en el tiempo requerido.	Comprobar si los mismos datos están en diferentes fuentes dentro de la empresa, y si es así elegir aquel dato con la actualización mas reciente. La verificación de validez de una tarjeta de crédito, por parte de una entidad financiera, debe obtenerse antes de confirmar una venta.
Completitud: Grado en que los datos tienen todos los valores necesarios para la ejecución exitosa de un proceso de negocio en un contexto y dominio específico	Especificar los datos que son obligatorios. Verificar que todos los datos obligatorios tengan valores. Completar los datos obligatorios con otras fuentes de datos. Usar un procedimiento para forzar la entrega de todos los datos obligatorios.	Los resultados de un examen de laboratorio debe contener valores para todas las pruebas solicitadas por el médico tratante. Para entregar un paquete, los datos relacionados con la dirección e identificación del cliente deben estar completos.

Tabla 2.2: Dimensiones de DQ y actividades de mejora asociadas.

c) Repositorio de casos de uso de DQ

El tercer componente del método es un repositorio que contiene los casos de uso estándar para cada dimensión de DQ que puede ser especificada como requisito de DQ en un BP. Estos casos de uso estándar han sido definidos en base a (i) la definición de cada dimensión de DQ, (ii) el conjunto de actividades que serán realizadas en función de los requisitos especificados (repositorio de actividades de DQ) y (iii) el conocimiento extraído de la literatura y de la experiencia de desarrolladores. En la Figura 2.6 se muestran algunos ejemplos de casos de uso estándar para las dimensiones de DQ exactitud y completitud.

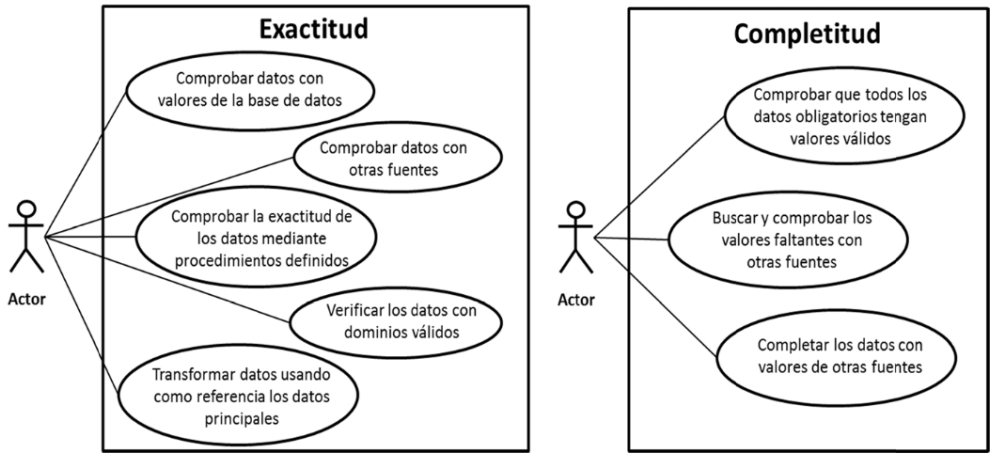


Figura 2.6: Ejemplos de Caso de Uso estándar.

Basados en estos casos de uso estándar de DQ, los trabajadores deberán hacer los ajustes necesarios de acuerdo a las características propias del BP y relacionarlos con los otros casos de uso obtenidos desde el BP propiamente dicho.

Etapas del metodo BPiDQ*

En las subsecciones siguientes se describen en detalle cada una de las etapas que componen el método BPiDQ*. Como se dijo anteriormente, el método sólo varía de la propuesta original en las dos últimas, que es cuando se pone énfasis en la obtención de artefactos útiles para el desarrollo de software. En la Figura 2.7 se muestra una vista completa del método BPiDQ*.

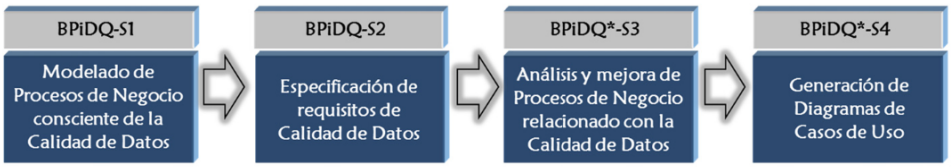


Figura 2.7: Las cuatro etapas del método BPiDQ*.

◇ **Etapa 1: Modelado de procesos de negocio consciente de la calidad de datos**

Esta etapa está dedicada a la captura temprana de requisitos de DQ, los que son representados en un modelo de BP a nivel descriptivo de BPMN. Durante el modelado se incorporan marcas (DQ-Flags) donde se estime que la calidad de los datos involucrados en el BP es relevante para el éxito del negocio. Los elementos de entrada de esta etapa son el estándar BPMN y la extensión que permite incluir requisitos de DQ. Los trabajadores de esta etapa son el experto del negocio y/o el analista de procesos de negocio, quienes tienen la responsabilidad de definir las necesidades del negocio y, desde esa perspectiva, la importancia que tiene la DQ para el desempeño del mismo. El resultado de esta etapa es una descripción del proceso de negocio en la cual se han incluido marcas (DQ-Flags) que denotan el interés de los expertos del negocio por profundizar en la definición de los requisitos de DQ que son importantes para el buen desempeño del proceso de negocio. Junto con ello, también se deben identificar los elementos de datos involucrados en las marcas y una estimación del nivel de influencia (baja, media o alta) que tienen los datos asociados a esas marcas en el desempeño total del BP.

◇ **Etapa 2: Especificación de requisitos de calidad de datos**

El principal objetivo de esta etapa es obtener una especificación detallada de los requisitos de DQ definidos en el proceso de negocio. El único elemento de entrada en esta etapa es el modelo de BP con requisitos de DQ (DQ-Flags). Los trabajadores involucrados en esta etapa son el analista de procesos de negocio y el experto en calidad de datos. Estos trabajadores determinan el conjunto final de DQ-Flags y especifican en forma detallada los requisitos de DQ asociados a cada uno de ellos. Las salidas de esta etapa son (i) el modelo del BP con requisitos de DQ (DQ-Flags) y (ii) para cada DQ-Flag, una especificación detallada que contiene: el elemento del BP en que se ha puesto el DQ-Flag, la importancia del requisito de DQ en el BP (alta, media

o baja), la probabilidad de ejecución de la actividad asociada a la especificación del requisito de DQ, las dimensiones de DQ asociadas, la sobrecarga para el BP debido a la incorporacion de nuevas actividades asociadas a las dimensiones de DQ, el nombre del elemento de dato involucrado en el requisito de DQ, su descripción, medio de soporte y origen.

- **Etapa 3: Análisis y mejora de Procesos de Negocio relacionado con la Calidad de Datos**

  En esta etapa se analiza y deciden las mejoras que se pueden hacer al modelo del BP teniendo en cuenta los requisitos de DQ especificados. Esta etapa ha variado respecto de la primera versión del método ya que originalmente estaba centrada en mejorar el modelo del BP en sí mismo (reorganización de actividades, inclusión de nuevas actividades, ajuste de los flujos de ejecución, etc.). No obstante, en la versión BPiDQ* sólo se ha considerado la introducción de las nuevas actividades relacionadas con las dimensiones de DQ derivadas de la especificación de los requisitos de DQ. Las entradas en esta etapa son: la descripción del BP con especificaciones de calidad de datos (DQ- Flags), un detalle de las especificaciones de DQ del BP y un repositorio con las actividades que se relacionan con las dimensiones de DQ. Los trabajadores involucrados en esta etapa son el diseñador de procesos de negocio y el experto en DQ. Las dimensiones de DQ son utilizadas para seleccionar el conjunto de actividades de DQ que se deberán agregar a la descripción del proceso de negocio. El resultado de esta etapa es una descripción del proceso de negocio en la cual se han incluido nuevas actividades que consideran los requisitos de DQ.

- **Etapa 4: Generación de Diagramas de Casos de Uso**

  Esta etapa también ha sido modificada respecto del método original, ya que ahora no sólo se generan los casos de uso relacionados con DQ (obtenidos desde el repositorio de casos de uso estándar de DQ) sino que también se obtendrán casos de uso generales que se corresponden con el resto de los requisitos representados en el BP y que, posteriormente, serán implementados como parte del sistema de información. Para ello se ha tenido en cuenta una propuesta que permite obtener casos de uso desde la descripción de un BP. Las entradas de esta etapa son: la descripción del BP con las actividades de calidad de datos agregadas en la etapa anterior y un repositorio con los casos de uso estándar que se relacionan con la DQ. Los trabajadores involucrados en

esta etapa son el analista de sistemas y el experto en calidad de datos. Las actividades relacionadas con calidad de datos se usan para seleccionar el conjunto de casos de uso de DQ estándar y el resto de los casos de uso se obtienen en forma directa desde la descripción del BP. El resultado de esta etapa es un conjunto de casos de uso que pueden ser usados en un proceso de desarrollo de software. Los casos de uso estándar no tienen asociados actores específicos debiendo ser integrados con los casos de uso que representan las actividades del BP (que representan todos los requisitos de la aplicación que soportará el BP). De manera que los casos de uso relacionados con las dimensiones de DQ serán considerados como casos de uso «include».

Bajo el contexto de está investigación, se desarrollará la extensión *dqBP*, contemplando los pasos uno y dos del método BPiDQ*, en búsqueda de la implementación de las bases para que una herramienta de modelado de Procesos de Negocio, con Notación BPMN 2.0, pueda especificar requisitos de Calidad de Datos.

Capítulo 3

Tecnología Disponible

Un interesante acercamiento a la tecnología existente para la implementación de la extensión *dqBP*, es el trabajo realizado por Ortega et al. (2014), en donde se abordan las herramientas para el modelado de procesos de negocio, las cuales permiten su extensibilidad y son de código libre.

En base a este trabajo (Ortega et al., 2014), se ha decidido analizar las 9 herramientas identificadas, las cuales son de código libre, cumplen con el estándar BPMN 2.0 y tienen posibilidad de ser extendidas. En base a este análisis, se tendrán los argumentos necesarios para poder seleccionar la herramienta con mayores facilidades para ser extendida.

3.1. Herramientas

3.1.1. *Yaoqiang BPMN Editor*

Yaoqiang BPMN Editor es un editor gráfico de diagramas de procesos de negocio, que cumple con las especificaciones de la OMG para el estándar más reciente BPMN 2.0 .

El editor permite importar y exportar archivos en formato BPMN 2.0. Dentro de sus funcionalidades tiene la validación de la sintaxis BPMN en tiempo real y generación automática de información para diagramas de intercambio. Además es importante señalar que tiene una arquitectura extensible para crear Plug-in y una biblioteca para la extensibilidad. También es posible ejecutar simulaciones de BPMN, permitiendo el montaje y desmontaje de subprocesos de una forma flexible. Implementa directamente a un motor de BPMN 2.0, como soporte para

el modelado de procesos de negocios. Tiene la propiedad de soportar la importación desde Microsoft Visio Professional y GraphML hacia archivos BPMN 2.0 (Inc, 2010).

3.1.2. *Modelio - entorno de modelado*

Modelio es una herramienta de modelado de código abierto que proporciona soporte para el último estándar de UML y BPMN 2.0.

Se puede extender mediante la adición de módulos que añaden nuevas funcionalidades. Un gran conjunto de estos módulos son gratuitos y de código abierto, lo que facilita la gestión y generación de código. Modelio, sigue los estándares de modelado TOGAF, SysML, SoaML y permite la generación de documentos. Esta herramienta, además se caracteriza por lo siguiente: (i) permite la exportación e importación a archivos con extensión XMI (ii) incorpora la posibilidad de extensión mediante la adición de módulos (iii) posee soporte para scripting lenguaje (Jython) y (iv) proporciona soporte a la propuesta Model Driven Architecture (MDA) (Modeliosoft, 2011).

3.1.3. *Bonita BPM: Open Source BPM*

Bonita BPM es una herramienta de código abierto para el modelado, gestión de procesos de negocio y flujos de trabajo (Bonitasoft, 2009). Esta posee los tres siguientes componentes:

(i) Bonita Estudio: permite al usuario crear y modificar gráficamente los procesos de negocio con el estándar BPMN 2.0. Además Bonita Estudio, que está basado en Eclipse, permite trabajar con procesos diseñados con otros estándares y tecnologías como XPDL o jBPM.

(ii) Bonita BPM Motor: es una API de Java, que le permite interactuar mediante programación con el motor de BPMN y ejecutar los procesos de negocio, este esta disponible bajo licencia GPL. Se basa en Hibernate.

(iii) Bonita Portal: es una interfaz de Webmail que permite gestionar las tareas. Este es de código abierto y se puede descargar bajo licencia GPL.

3.1.4. *MyBPMN*

MyBPMN una solución Open Source BPM que permite modelar, ejecutar y optimizar procesos de negocio a través de un entorno gráfico y sin necesidad de programación.

Esta herramienta tiene tres componentes principales que son: (i) MyBPMN Designer que es una aplicación de modelado BPMN basado en Eclipse 3.5, (ii) MyBPMN Engine que es totalmente compatible con BPMN 2.0 y (iii) MyBPMN Web que es una consola de administración (MyBPMN, 2009).

3.1.5. *Signavio-CORE-componentes*

Signavio-CORE-componentes es una herramienta de software que permite crear modelos de proceso de negocio que pueden cargar motores de proceso gracias al formato de intercambio XML estandarizado.

Dentro de las características que posee Signavio-CORE-componentes se pueden mencionar: (i) la utilización de un formato de intercambio XML estandarizado y (ii) la capacidad para exportar a los siguientes formatos ARIS©, XPDL, PDF, Visio y Excel (Signavio, 2010).

3.1.6. *BPMNX*

BPMNX es una herramienta desarrollada en la plataforma Eclipse para el desarrollo de extensiones para el meta-modelo BPMN 2.0 (Stroppi, 2010).

La herramienta consta de tres componentes principales que son:

(i) BPMN+X: que es un componente que se basa en la especificación del mecanismo de extensión BPMN.

(ii) QVT (Query/View/Transformation): que es un lenguaje que permite la transformación entre modelos.

(iii) JET Model-to-Code Transformation: este componente ayuda a la producción de documentos de esquema XML que pueden ser procesados por herramientas BPMN.

3.1.7. *Camunda Modeler*

Camunda Modeler es un plug-in de modelado BPMN 2.0 para Eclipse que se centra en el modelado transparente de procesos y diagramas de colaboración. Este es parte del conjunto de software Camunda BPM. La herramienta tiene la propiedad de leer y escribir archivos de diagrama BPMN 2.0 (GmbH, 2013).

3.1.8. *BPMN2 Visual Editor for Eclipse*

BPMN2 Visual Editor for Eclipse es uno de los plug-in para el modelado de procesos de negocio sobre Eclipse. Es compatible con el estándar BPMN 2.0 que se complementa de buena forma con jBPM5 y permite, además, que los modelos sean almacenados en el formato BPMN 2.0 XML (Codehoop, 2012).

3.1.9. *BPMN 2.0 Modeler Project*

BPMN 2.0 Modeler Project proporciona una herramienta de modelado gráfico que permite la creación y edición de diagramas BPMN. La herramienta se basa en Eclipse Graphiti y utiliza BPMN 2.0 EMF (Eclipse Modeling Framework) meta model, está desarrollada actualmente en el contexto del proyecto Eclipse Model Development (MDT). Este meta-modelo es compatible con la especificación de BPMN 2.0 propuesto por la Object Management Group (Foundation, 2013).

3.2. Análisis comparativo de las Herramientas

El punto de partida para iniciar este análisis, es que las herramientas contarán con el soporte del Meta-Modelo BPMN 2.0, siendo uno de los criterios para descartar las posibles herramientas a seleccionar como tecnología disponible para el desarrollo de esta extensión.

Posterior a esto, es necesario identificar las herramientas que contarán con licencia de proyecto Open Source (Código Libre), de esta manera, poder realizar un gran barrido con las herramientas de modelado BPMN y finalmente, luego de analizar otros criterios, poder obtener la herramienta candidata para el desarrollo de la extensión *dqBP*.

A continuación, en la Tabla 3.1, se muestra un resumen de la comparación realizada por Ortega et al. (2014), en donde por cada herramienta se indican las características que posee.

Criterios de Comparación	Yaoqiang BPMN Editor	Modelio entorno de modelado	Bonita BPM: Open Source BPM	MyBPMN	Signavio CORE components	BPMNX	Camunda Modeler	BPMN2 Visual Editor for Eclipse	BPMN 2.0 Modeler Project
Soporte a BPMN 2.0	Sí	Sí	Sí	Sí	Sí	Sí	Sí	Sí	Sí
Plataforma compatible	Windows, Mac Os y Linux	Windows, Mac Os y Linux	Windows, Solaris y Linux	Windows, Mac Os y Freebsd	Windows, Mac Os y Linux	Windows, Mac Os y Linux	Windows, Mac Os y Linux	Windows, Mac Os y Linux	Windows, Mac Os y Linux
Provee arquitectura para plug-in o extensiones	Sí	Sí	No	No	No	Sí	No	No	Sí
Tipo de licencia	GPLv3	GPLv3	GPLv2	Eclipse Public License	GPLv3	Apache License 2.0	Eclipse Public License	Eclipse Public License	Eclipse Public License
Lenguaje de programación	Java	Java	Java	Java	Java	Java	Java	Java	Java
Utiliza formato de salida estándar	BPMN 2.0 file	XMI	BPMN 2.0 file	BPMN 2.0	ARIS, XPDL	BPMN 2.0	BPMN 2.0	BPMN 2.0	BPMN 2.0
Nivel de documentación	Medio	Medio	Alto	Bajo	Bajo	Bajo	Medio	Medio	Medio

Tabla 3.1: Análisis Comparativo de las Herramientas seleccionadas.

En base a la Tabla 3.1, se puede deducir, que una de las mejores herramientas para realizar la extensibilidad es *Bonita BPM: Open Source BPM*. Esta herramienta nos entrega la factibilidad de ser (i) Open Source, ya que cuenta con licencia *GPLv2* (General Public License versión 2), (ii) cumplir con el estándar BPMN 2.0, (iii) ser una herramienta Multiplataforma (disponible para *Windows, Mac Os y Linux*).

Si bien tenemos un candidato, es necesario abordar las herramientas desde otro punto de vista para obtener un veredicto final.

3.2.1. Análisis desde el punto de vista de la Implementación

A pesar, de tener como base el trabajo realizado por Ortega et al. (2014), es necesario analizar cada una de las herramientas desde el punto de vista de la implementación, esto como forma de analizar algunos criterios que no abordó el trabajo antes mencionado.

Estos criterios se definieron, previo análisis, de cada una de las herramientas estudiadas en el punto anterior; en la búsqueda de factores claves para el éxito de la extensión, como por ejemplo, que la herramienta implemente el 100 % de los elementos de BPMN 2.0, entre otros, que se definirán a continuación.

Criterios de Análisis

a) Elementos de BPMN

Contar con todos los elementos básicos de BPMN 2.0 expuestos en la Figura 2.3 y especialmente, los elementos que podrán contar con la marca de Calidad de Datos DQ (DQ), estos elementos son expuestos en la Tabla 2.1. Este criterio será evaluado de la siguiente forma:

- **Completo:** Se considerará como *completo*, a aquel Modelador de Procesos de Negocio, que posea la paleta completa de BPMN 2.0, y en especial consideración, los elementos afectados por la extensión dqBP, analizados en el capítulo anterior.
- **Incompleto:** Se considerará como *incompleto*, a aquel Modelador de Procesos de Negocio, que tenga la paleta incompleta de los elementos de BPMN 2.0, o en su defecto, a aquel que no posea alguno de los elementos afectados por la extensión dqBP.

b) Meta-Modelo de Herramienta

Contar con un Meta-Modelo del Modelador BPMN, el cual permita identificar claramente cómo abordar la implementación de la Extensión de la Herramienta. Este criterio será evaluado con un *"Sí"*, si posee un Meta-Modelo que represente al Modelador de Procesos de Negocio y en su defecto, será evaluado con un *"No"*, si el Modelador no tiene un Meta-Modelo que lo represente.

c) Interfaz de Programación de Aplicaciones (API)

Contar con un API del Modelador BPMN, de la Herramienta en sí, y no de sus módulos o extensiones previas, el cual facilite el proceso de implementación de la extensión de la herramienta. Este criterio será evaluado con un *"Sí"*, si posee una API, que indique los métodos para realizar cambios en el Modelador de Procesos de Negocio y en su defecto, será evaluado con un *"No"*, si el Modelador no tiene una API.

d) Foro de Discusión

Contar con una Comunidad activa (considerando como activa, que esta cuente con actividad durante el año 2015), en donde se pueda realizar consultas y se aborden los temas de cómo modificar o extender el Modelador BPMN. Este criterio será evaluado de la siguiente forma:

- **Alta:** Será calificado con el concepto *alta*, a aquel Modelador de Procesos de Negocio, que tiene un foro de discusión acerca de la modificación de la herramienta, que tenga una actividad de sus usuarios durante el presente año (2015), y además, la generación de respuestas a sus usuarios, no sea superior a 7 días.
- **Media:** Será calificado con el concepto *media*, a aquel Modelador de Procesos de Negocio, que tiene un foro de discusión acerca de la modificación de la herramienta, que tenga una actividad de sus usuarios durante el presente año (2015), y la generación de respuestas a sus usuarios, es realizada en lapsos de semanas e incluso meses.
- **Baja:** Será calificado con el concepto *baja*, a aquel Modelador de Procesos de Negocio, que tiene un foro de discusión acerca de la modificación de la herramienta y tenga una actividad de sus usuarios anterior al presente año (2015) o aquel Modelador de Procesos de Negocio, que no tiene un foro de discusión acerca de la modificación de la herramienta.

e) Repositorio de Control de Versiones de la Herramienta

Disponer de un repositorio web, que cuente con las modificaciones del Modelador de Procesos de Negocio (Open Source), y se detallen periódicamente las correcciones sufridas por la Herramienta. Este criterio será evaluado con un *"Sí"*, si posee un repositorio web, que contenga los cambios y/o modificaciones del Modelador de Procesos

de Negocio y en su defecto, será evaluado con un *"No"*, si el Modelador no tiene un Repositorio.

f) Usabilidad

Usabilidad, vista cómo la facilidad del usuario para generar los diagramas con el Modelador de Procesos de Negocio. Este criterio será evaluado al realizar, en cada una de las herramientas seleccionadas, el diagrama expuesto en la Figura 3.1 (White and Miers, 2009) y posteriormente calificada, en alguno de los siguientes ítem:

- **Alta:** Será calificada con usabilidad *alta*, la herramienta que demuestre facilidad en uso y la vinculación de los elementos del diagrama expuesto en la Figura 3.1, así como, la facilidad de disponer los elementos.
- **Media:** Será calificada con usabilidad *media*, la herramienta que demuestre facilidad en uso y la vinculación de los elementos del diagrama expuesto en la Figura 3.1, pero demuestra dificultad para disponer de los elementos del diagrama.
- **Baja:** Será calificada con usabilidad *baja*, la herramienta que no demuestre facilidad en uso y genere dificultad para la creación del diagrama expuesto en la Figura 3.1.

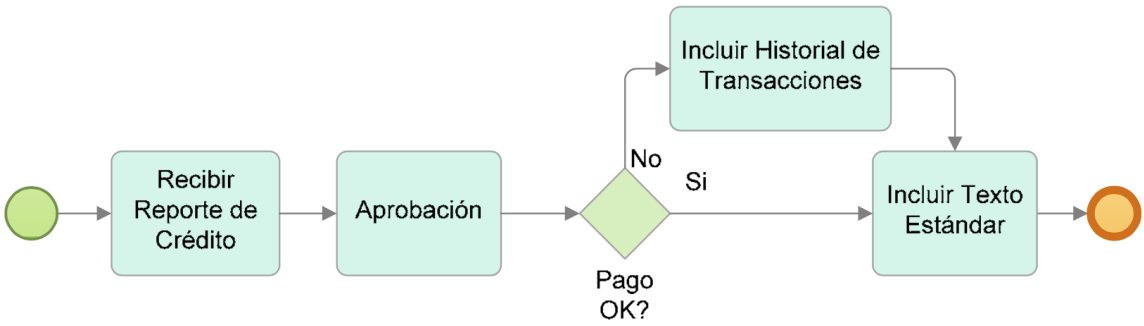


Figura 3.1: Ejemplo de BPMN.

g) Especificación de Propiedades de los elementos

La generación de Modelos de Negocio, no solo consta de la vinculación de elementos de BPMN, sino que también de la especificación de las propiedades de cada uno de los elementos que componen dicho modelo, de esta manera lograr un correcta abstracción

en la representación del negocio. Debido a esto, es importante tomar como criterio, la facilidad de especificación de las propiedades de los elementos, siendo este criterio calificado con una de las siguientes formas:

- **Alta:** La herramienta no presenta dificultades para especificar propiedades de los elementos del Diagrama, así como, tampoco es dificultoso encontrar el módulo para generar la especificación de las propiedades de los elementos.
- **Media:** La herramienta presenta dificultades para la especificación de propiedades y/o encontrar el módulo para generar la especificación de las propiedades de los elementos es complicado para el usuario.
- **Baja:** La herramienta no tiene un módulo para especificar las propiedades de los elementos del diagrama.

h) Estado Proyecto

Este es un criterio de relevancia, ya que es importante que el proyecto se encuentre activo, debido a que la extensión necesite continuidad y proyección. Este criterio será clasificado como *"Activo"*, si este se encuentra aún en modificación por parte de los creadores, y como *"No activo"*, si los creadores no han realizado más modificaciones o han declarado el proyecto como cerrado.

i) Extensión Multiplataforma

Este criterio evalúa el formato para la extensión, y si este es aceptado por todas las plataformas para las cuales se diseñó la herramienta. Si bien es cierto, todas las herramientas evaluadas son multiplataformas, lo cual quiere decir que están disponibles para más de una plataforma (Windows, Mac OS y Linux) en sus diferentes versiones, estas herramientas se adaptan al sistema donde van a ser ejecutadas, por lo cual se generan diferentes tipos de ejecutables. Lo anterior quiere decir que las extensiones tendrán, en algunos casos, tener que ser construidas para distintas plataformas.

Se define que aquella herramienta, que al extenderla, genere un extensión apta para todos los sistemas operativos soportados por la herramientas será evaluada con un *"Sí"* y aquella herramienta que genere una extensión, individualizada, para cada sistema operativo soportado, será evaluada con un *"No"*.

Comparación de las Herramientas

En la Tabla 3.2 se expone un resumen de análisis realizado a cada una de las herramientas Open Source.

Cabe mencionar, que una de las herramientas evaluadas, al momento de calificar los distintos criterios, ya no se encontraban el proyecto activo e incluso el código fuente se encontraba con desperfectos y no se logró ensamblar el proyecto para realizar la evaluación. En este caso, no se evaluaron los criterios que fueron planteados y se les designó la leyenda *"Herramienta no disponible"*.

Criterios de Comparación	Yaoqiang BPMN Editor	Modelio entorno de modelado	Bonita BPM: Open Source BPM	MyBPMN	Signavio CORE components	BPMNX	Camunda Modeler	BPMN2 Visual Editor for Eclipse	BPMN 2.0 Modeler Project
Elementos de BPMN	Completo	Incompleto	Incompleto	Completo	Herramienta no disponible	Completo	Completo	Completo	Completo
Meta-Modelo de la Herramienta	No	Sí	No	No	Herramienta no disponible	No	No	No	No
API	No	Sí	No	No	Herramienta no disponible	Sí	Sí	No	Sí
Foro de Discusión	Activo	Activo	Activo	Baja	Herramienta no disponible	Baja	Activo	Baja	Activo
Repositorio de Control de Versiones	Sí	Sí	Sí	No	Herramienta no disponible	Sí	Sí	Sí	Sí
Usabilidad	Baja	Media	Alta	Baja	Herramienta no disponible	Baja	Alta	Media	Alta
Especificación de Propiedades	Alta	Alta	Alta	Media	Herramienta no disponible	Baja	Alta	Alta	Alta
Extensión Multiplataforma	Sí	No	No	Sí	Herramienta no disponible	No	No	Sí	Sí
Estado Proyecto	Activo	Activo	Activo	No Activo	Herramienta no disponible	No Activo	Activo	No Activo	Activo

Tabla 3.2: Análisis Comparativo de las Herramientas seleccionadas desde el punto de vista de la Implementación.

Al comparar las herramientas, se descartan algunas de ellas, esta acción se realiza debido a que ciertas herramientas, no cumplen factores claves, como la implementación de todos los elementos de BPMN 2.0 o que el proyecto, actualmente, se encuentre activo.

Luego, con el análisis comparativo, se puede evidenciar que las mejores herramientas, pese a no contar con un Meta-Modelo de construcción, son *Camunda Modeler* y *BPMN 2.0 Modeler Project* (estas herramientas fueron denotadas con un color gris claro en la Tabla 3.2). Estas dos herramientas contemplan buenas bases para la extensión del Modelador de BPMN, ya que cumplen con los criterios de evaluación. Por ejemplo, las dos herramientas poseen la totalidad de los elementos de BPMN 2.0 y cuentan con una alta usabilidad, vista como usuario final.

Cabe destacar, que al generar la extensión de la herramienta, sólo *BPMN 2.0 Modeler Project*, genera un archivo compatible con Eclipse, en todos los Sistemas Operativos en que está disponible.

La herramienta mejor evaluada es *BPMN 2.0 Modeler Project*, la cual bajo los criterios anteriormente mencionados, es uno de los Modeladores BPMN 2.0 Open Source con la capacidad de extensibilidad y genera un plug-in multiplataforma. Es por esta última capacidad, y la buena valoración en cada uno de los criterios evaluados, siendo algunos de ellos, el poseer la totalidad los Elementos de BPMN, contar con una API y tener un Foro de Discusión donde plantear posibles problemas que se presenten; hace que sea la herramineta seleccionada y utilizada para desarrollar e implementar un prototipo de la extensión *dqBP*.

Capítulo 4

Descripción de la Herramienta Seleccionada

BPMN 2.0 Modeler es una herramienta de modelado gráfico, que permite la creación y edición de diagramas BPMN. Esta herramienta que se basa en Eclipse Graphiti (Framework de gráficos basado en Eclipse, para el desarrollo de diagramas) y utiliza BPMN 2.0 EMF meta model (Framework de modelado BPMN 2.0).

El propósito específico de este proyecto es proporcionar una herramienta de modelado intuitiva para el analista de negocio. BPMN 2.0 Modeler, ofrece edición gráfica y la posibilidad de creación de archivos de BPMN 2.0, compatibles para el dominio de BPMN, así como los modelos de diagrama de intercambio.

Fundación Eclipse busca que los alcances de esta herramienta sean los siguientes:

- Capacidades de creación y edición de archivos compatibles BPMN 2.0 - Básico.
- Modelado de Procesos, Procesos de Ejecución y Coreografía de Modelado.
- Puntos de extensión para el Plug-in, que permitan al editor la personalización de la herramienta, para aplicaciones específicas.
- Ejecución de procesos BPEL.
- El despliegue de recursos BPMN en un tiempo de ejecución adecuado.
- Simulación y soporte para depuración de Procesos de Negocio.

Hoy en día, sólo se ha logrado implementar los primeros 3 puntos de los alcances de BPMN 2.0 Modeler, pero la comunidad de desarrolladores se encuentra trabajando para la implementación de los puntos que aún faltan.

4.1. Aspectos de Operación

Esta herramienta presenta una interfaz gráfica simple e intuitiva, que permite al analista de negocio plasmar sus diseños sin problemas en el lienzo.

En la Figura 4.1, se aprecia la simpleza de la herramienta de modelado, teniendo todos los elementos necesarios para diseñar modelos de Procesos de Negocio, siendo estas una paleta de herramientas que contiene todos los elementos de la notación BPMN 2.0 y una sección para la especificación de las propiedades de los elementos.

La paleta de herramientas, por defecto, se encuentra a lo largo del borde derecho del lienzo de dibujo (apreciable en la Figura 4.1). Se compone de varios cajones, los cuales contienen las "herramientas"que se arrastran en el lienzo de dibujo para crear elementos BPMN 2.0.

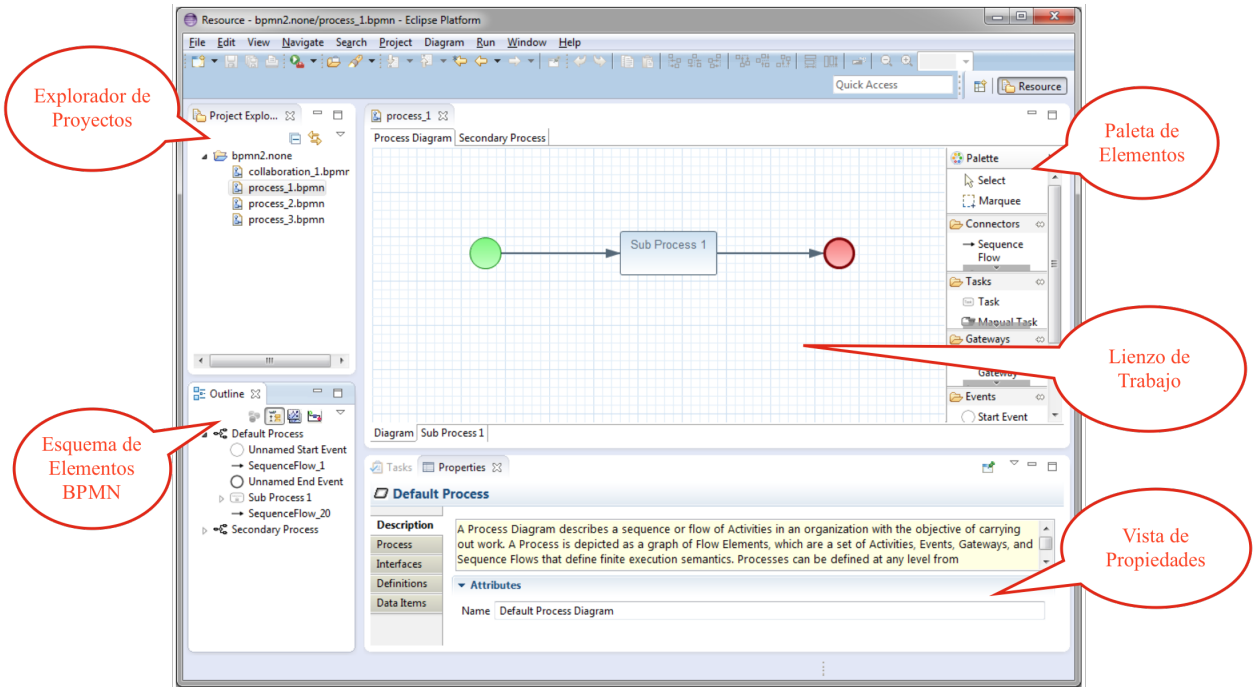


Figura 4.1: Interfaz gráfica de BPMN 2.0 Modeler.

La herramienta cuenta con una sección para la visualización del código fuente del modelo BPMN, el cual se encuentra en formato XML (del inglés eXtensible Markup Language ("lenguaje de marcas extensible")). En la Figura 4.2, se puede apreciar la sección del código fuente del modelo.

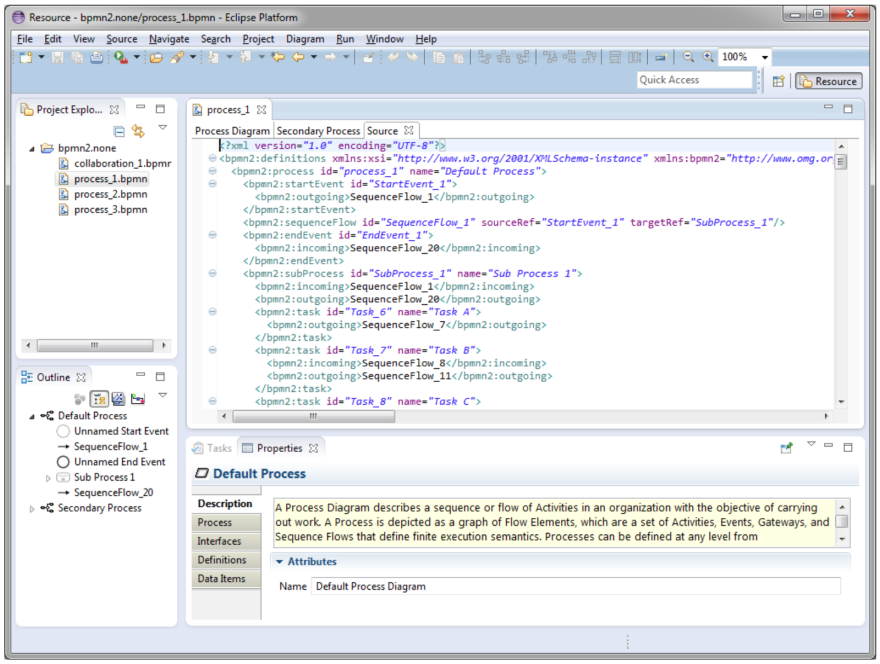


Figura 4.2: Vista del Código BPMN 2.0 Modeler.

La sección del esquema (Figura 4.3) es independiente del editor y tiene la intención de mostrar un árbol orientado a la vista jerárquica del archivo. Este punto está sincronizado con el lienzo de dibujo; cuando se selecciona un elemento en el lienzo, se destaca en la sección del esquema. A la inversa, cuando se selecciona un elemento en el esquema, también se destaca en el lienzo de dibujo.

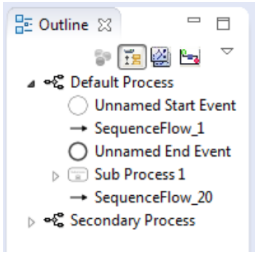


Figura 4.3: Esquema de los elementos de un Modelo BPMN.

La sección de propiedades (Figura 4.4), se utiliza para editar los atributos de un elemento seleccionado. Esta sección también se sincroniza con la sección del esquema, de tal manera que cuando se selecciona un elemento de árbol del Esquema, sus atributos se muestran en la sección propiedades.

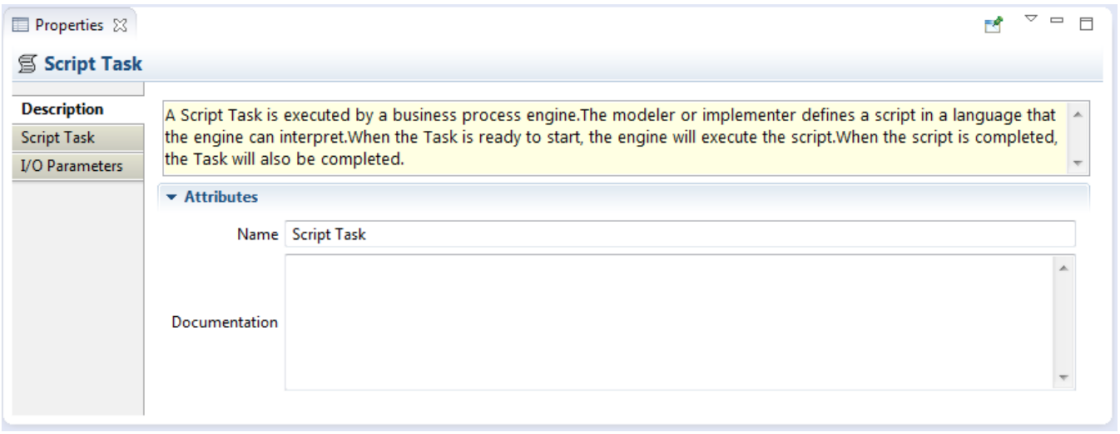


Figura 4.4: Sección de Propiedades de BPMN 2.0 Modeler.

Además la herramienta cuenta con un menú desplegable (Figura 4.5), el cual se visualiza cuando el cursor pasa sobre un elemento del modelo o recorre las cercanías de este. Este menú refleja opciones por defecto de todos los elementos BPMN, y además opciones dependientes del elemento y/o el contexto en que se encuentre el mismo.

Figura 4.5: Menú desplegable de los elementos BPMN.

4.2. Aspectos de Implementación

BPMN 2.0 Modeler fue diseñado bajo la existencia de otros proyectos de la Fundación Eclipse, estos proyectos le otorgan gran parte de las características implementadas en este Plug-in.

Esta herramienta tiene dependencia de los siguientes proyectos:

- **MDT Project,** el cual incluye el meta modelo de BPMN 2.0, el cual es un modelo subyacente de BPMN 2.0 Modeler.
- **EMF Project,** necesario para acceder al meta modelo de BPMN 2.0.
- **EMF Validation Framework,** el cual se utiliza para aplicar reglas de validación adicional.
- **Graphiti Project,** el cual es el framework para la implementación de la gráfica en BPMN 2.0 Modeler.
- **Mangrove Project,** se utiliza para conectar el BPMN 2.0 Modeler con otros editores del proyecto SOA (de la Fundación Eclipse), así como, para el transporte de información de ejecución, como los datos de seguimiento, a los diagramas BPMN 2.0.

El proyecto de BPMN esta implementado con codificación JAVA y distribuido en cinco sub-proyectos:

- **Core**
- **Help**
- **Runtime JBoss jbpm5**
- **UI** (User Interface)
- **WSIL** (Web Services Inspection Language)

El paquete “Core” del proyecto BPMN 2.0 Modeler implementa las características de los elementos de BPMN, las operaciones lógicas y validaciones de los diagramas.

El paquete "Help" del proyecto implementa las secciones de ayuda y permite a los usuarios tener una guía, al efectuar operaciones de creación o edición de diagramas BPMN

El paquete "Runtime JBoss Jbpm5" implementa las características necesarias para utilizar el Plug-in bajo la plataforma JBoss, utilizando elementos ya implementados en el Plug-in.

El paquete "UI" implementa las capacidades gráficas de la herramienta, como por ejemplo: lienzo y sus características, y la paleta de elementos de la notación BPMN.

El paquete "WSIL" implementa las capacidades técnicas para recrear la implementación de servicios web, los cuales permitan la validación de los procesos de negocio, implementados en el diagrama.

4.2.1. Facilidades de Extensión

Al plantear las facilidades de extensión, es necesario partir por el lenguaje en que se encuentra codificado el proyecto, y en este caso particular el proyecto fue realizado en el Lenguaje de Programación Orientado a Objetos llamado "JAVA". A partir de este punto podemos identificar una de las principales facilidades de extensión.

Siguiendo con las facilidades, de extensión esta es una herramienta que fue construida bajo EPL (Eclipse Public License), la cual permite a los desarrolladores; copiar, adaptar y distribuir código de fuente, que este bajo esta licencia.

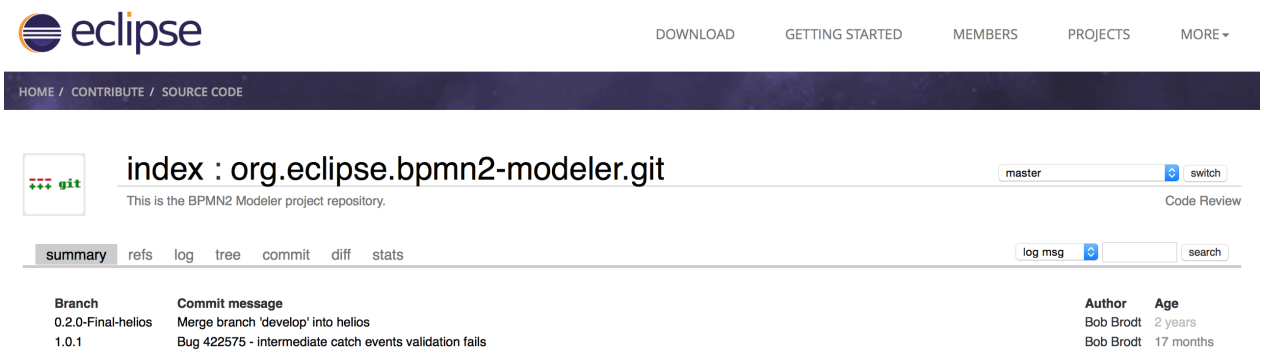


Figura 4.6: Repositorio y Control de Versiones GIT.

Además, este proyecto cuenta con un repositorio actualizado por la comunidad de la fundación eclipse, de donde podemos obtener las versiones más depuradas del código fuente

del proyecto y de esta forma poder obtener versiones limpias con las posibles modificaciones que le puede realizar un desarrollador independiente e incluso los desarrolladores pueden subir sus branch para exponer sus cambios a la comunidad del proyecto.

La Fundación Eclipse tiene para cada uno de sus proyecto una Wiki (librería de contenidos, acerca del proyecto). En este caso particular, esta Wiki contiene información en relación a las diferentes formas de extender la herramienta, planteando ejemplos de estos procesos y explicando los puntos de extensión que posee la herramienta

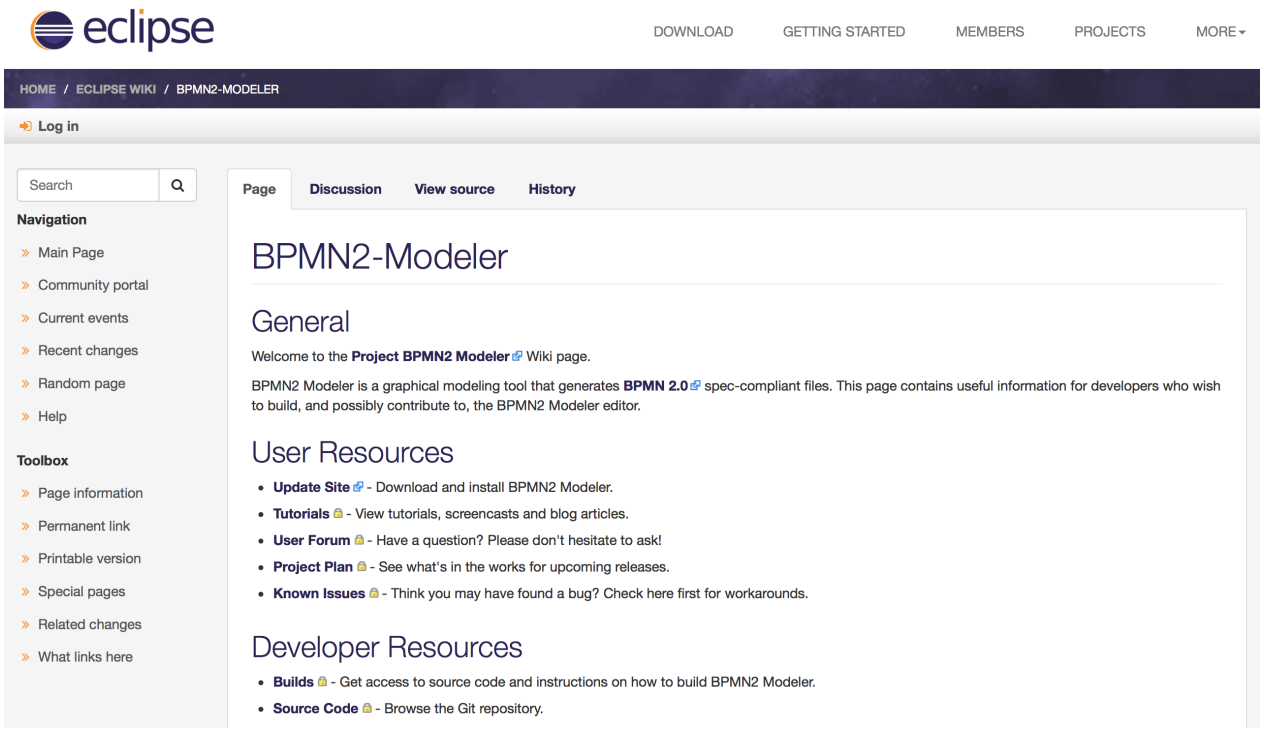


Figura 4.7: Wiki de BPMN 2.0 Modeler.

Por otra parte, cabe destacar que al ensamblar el proyecto en un IDE (del inglés Integrated Development Environment o un Ambiente de Desarrollo Integrado), como por ejemplo Eclipse IDE, se puede apreciar que la construcción de la herramienta se encuentra bien definida, lo que se logra con la utilización de paquetes para diferenciar funciones a cumplir, por cada uno de ellos.

Otro punto, que representa una facilidad de la implementación de esta herramienta, es al momento de construir el Plug-in (del IDE Eclipse), ya que el proyecto cuenta con archivos Maven (es una herramienta de software para la gestión y construcción de proyectos JAVA),

los cuales efectúan automáticamente la construcción del proyecto; finalmente, generando un archivo con la extensión ".zip" (extensión perteneciente a un formato de compresión de archivos). Este archivo puede ser leído por cualquier IDE Eclipse, en cada una de sus últimas plataformas (Windows, Linux y MAC OS), y de esta forma implementar la herramienta en el IDE Eclipse, para realizar y editar diagramas de Procesos de Negocio.

Capítulo 5

Desarrollo del Prototipo de Extensión (Plug-in)

Este capítulo describe el proceso de implementación del prototipo de la extensión dqBP en la plataforma seleccionada, siendo en este caso BPMN 2.0 Modeler, la cual funciona bajo el entorno y soporte de Eclipse, en su IDE de Modelado.

5.1. Análisis y Diseño

En esta etapa inicial, se realizó la captura de requisitos y el diseño de la solución, la cual tiene por objetivo plantear un prototipo para implementar la extensión dqBP, analizada anteriormente en el capítulo 2. Esta extensión contempla la incorporación de una marca gráfica a ciertos elementos de la notación BPMN (Tabla 2.1) y, por otra parte, la especificación de características de Calidad de Datos; esto, posteriormente visto, como una propiedad de cada elemento que cuente con la marca, la que posteriormente será reflejada en el código BPMN (XML) de exportación del modelo.

5.1.1. Descripción de requisitos

En el capítulo 1 fueron enunciados los objetivos que debe cumplir el prototipo, a continuación se procede a describir cada uno de estos objetivos, como un requisito funcional, para la implementación de la herramienta:

- Modificar la herramienta de modelado de Procesos de Negocio "BPMN 2.0 Modeler", para generar un prototipo, el cual permita la extensión *dqBP* y lograr de esta forma, incorporar una marca gráfica (Figura 5.1) para la representación de Calidad de Datos en 6 elementos de la notación BPMN (Figura 2.3), los cuales fueron descritos en la Tabla 2.1 y especificar algunas propiedades de los elementos asociados a la extensión.
- Permitir que el prototipo procese automáticamente el modelo de Proceso de Negocio, que tiene incorporada la Calidad de Datos en algunos de sus elementos para obtener, finalmente, un archivo con formato XML del modelo de proceso de negocio enriquecido.

Figura 5.1: Marca gráfica *dqBP*.

En particular, en el primero de los requisitos especificados, se tiene que, en base a las características que posee la herramienta (descritas en el capítulo 4), se debe concebir una forma para incorporar la marca gráfica seleccionada, en el entorno de modelado que posee la herramienta, permitiendo que el usuario, integre la Calidad de Datos a sus procesos de negocio.

En cuanto al segundo punto, este requerimiento describe la necesidad de procesar el modelo automáticamente e incorporar las características de Calidad de Datos en el código fuente de la herramienta, para que luego puedan ser exportadas e identificadas las características de Calidad de Datos incorporadas. Este código fuente del modelo, es un archivo XML, en donde se describe el modelo y sus elementos, además, sus características y tiene como extensión de archivo ".bpmn".

5.1.2. Diseño de la Solución

Para el desarrollo de la solución, se tomaron en cuenta cada una de las características presente en la herramienta de Eclipse, como por ejemplo, la presencia de un menú contextual (ver Figura 4.5) y la generación de un código XML para el proceso de negocio.

En este caso, se tomó en cuenta que BPMN 2.0 Modeler no utiliza imágenes predefinidas de los elementos de la notación BPMN 2.0 (Figura 2.3), sino que dibuja cada uno de los elementos para ser dispuestos en el modelo de Procesos de Negocio. Esto lo realiza gracias a Graphiti Proyect, el cual provee recursos para la generación de elementos gráficos.

Figura 5.2: Ejemplo de marca gráfica en las Task.

Llegado a este punto, es posible determinar que en el caso específico de las "Task", el Modelador es capaz de incorporar elementos gráficos específicos (pequeñas imágenes), para identificar, por ejemplo, "Task" que sean clasificadas como "Manuales" (véase Figura 5.2).

En base a lo anterior, se identificaron las clases que realizan esta tarea, las cuales interactúan con la generación de cada uno los elementos de BPMN 2.0. En la Figura 5.3, se refleja, un caso particular, para decorar los elementos "Task" del modelo de Procesos de Negocio. En este proceso, la clase `AbstractAddDecoratedTaskFeature` solicita la marca gráfica a `ImageProvider`, para luego utilizar los métodos de la clase `ShapeDecoratorUtil`, los cuales permiten la incorporación de la marca; esta última clase es la que interactúa con la API de Graphiti Proyect.

Luego, de la identificación de las clases, se requirió definir una forma en que el prototipo asimilara la incorporación de la marca, en otras palabras, incluya un elemento (botón) para agregar la marca en los elementos que incorporen la Calidad de Datos. Para esto, se utilizó el menú desplegable del modelador, el cual aparece al superponer el cursor en los elementos del modelo.

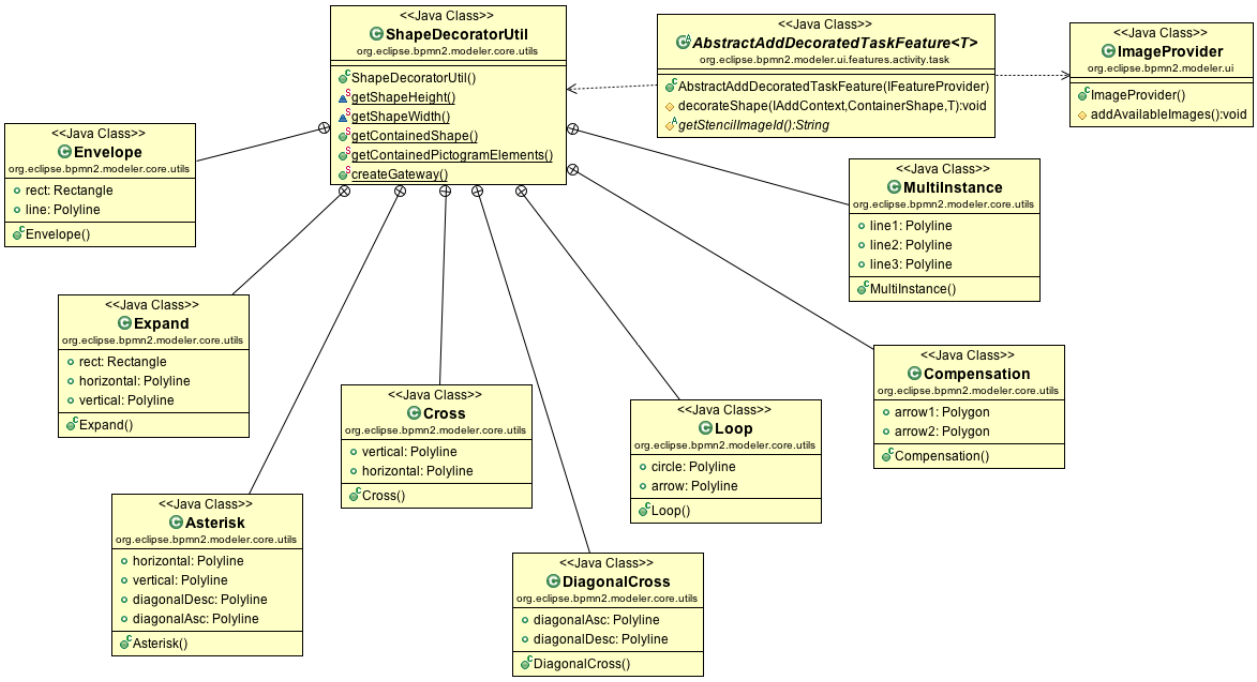


Figura 5.3: Interacción entre clases para incorporar una marca gráfica.

Lo que se buscaba conseguir era que la marca puediera ser incluida en este menú y luego de seleccionar esta opción se incorporase la marca en el elemento BPMN, como se refleja en la recreación de la Figura 5.4.

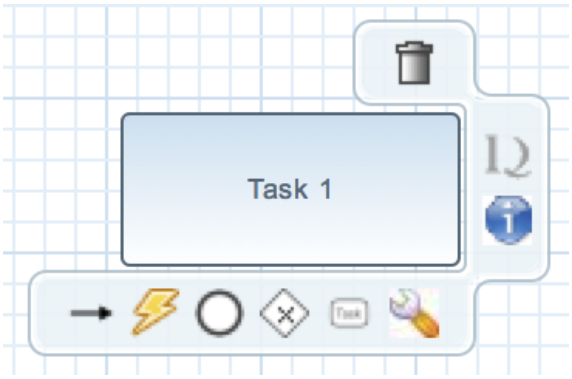


Figura 5.4: Recreación del Menú desplegable con la Marca.

En la Figura 5.5 se puede ver cómo luego de seleccionar la marca en el menú desplegable, esta se incorpora en el elemento de BPMN, en cual está permitido en la especificación de Calidad de Datos.

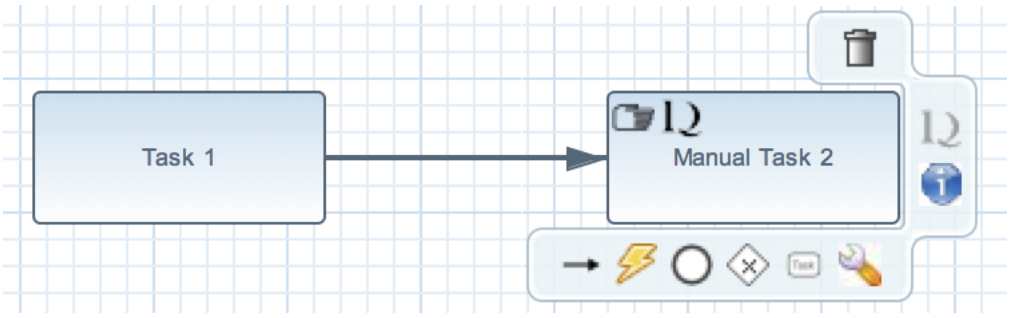


Figura 5.5: Menú despegable con la marca.

En la Figura 5.6, se recrea la incorporación de la marca, de manera alternativa a la descrita anteriormente, luego de hacer click derecho sobre los elementos que integran la marca *dqBP*.

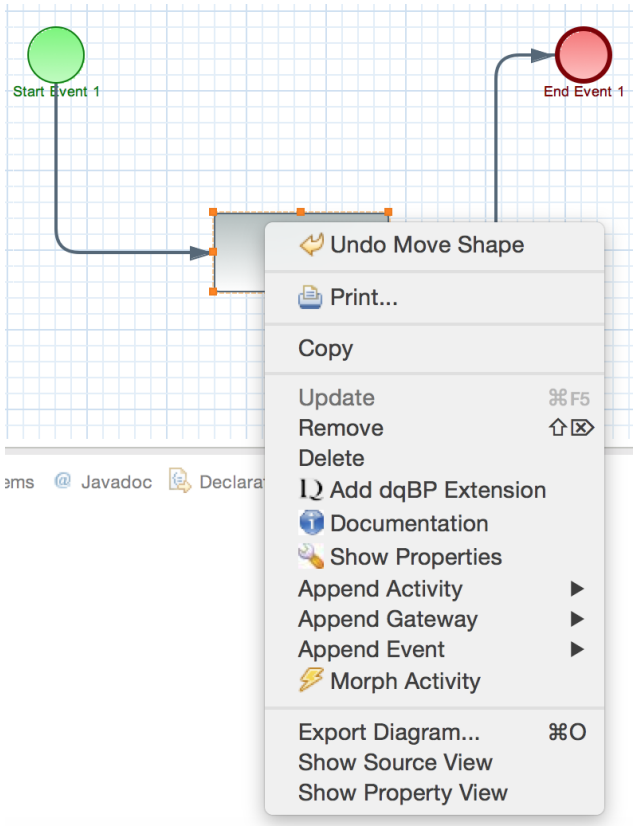


Figura 5.6: Menú despegable con la marca (Click derecho).

La clase `Bpmn2ToolBehaviorProvider` (véase Figura 5.7), es en donde se especifican las opciones que se despliegan en los elementos de BPMN, y es aquí en donde se deben realizar

las validaciones para que la opción de la marca sólo la contemplen los elementos que fueron especificados en la Tabla 2.1.

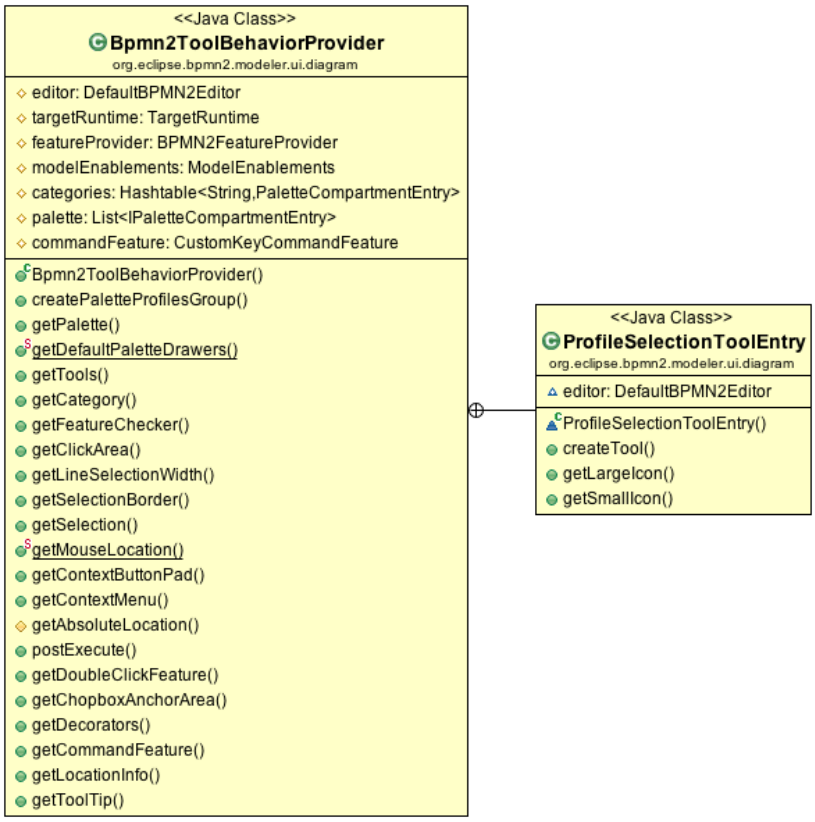


Figura 5.7: Clases que regulan el Funcionamiento del Menú desplegable.

Luego de analizar y contemplar el diseño de toda la parte gráfica del prototipo, se contempla el diseño de la ventana de incorporación de propiedades, correspondientes a la extensión de la herramienta. Estas propiedades contemplan los siguientes atributos:

- Nombre.
- Descripción.
- Influencia (*Alta, Media, Baja*).
- Fuente (*Interno, Externo*).
- Soporte (*Electrónica, No-Electrónica*).

Para el diseño de esta ventana se contempló como guía a la clase `ShowPropertiesFeatures` (véase Figura 5.8), la cual se encarga de levantar el menú de propiedades de los elementos, mostrando los atributos propios de cada elemento, como se puede ver en la Figura 5.9.

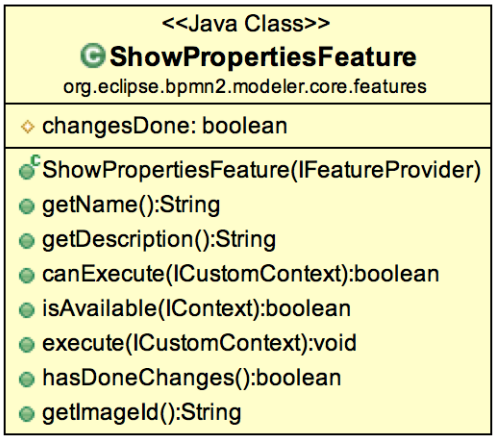


Figura 5.8: Clase que regula el Funcionamiento de la Ventana de Propiedades de los Elementos.

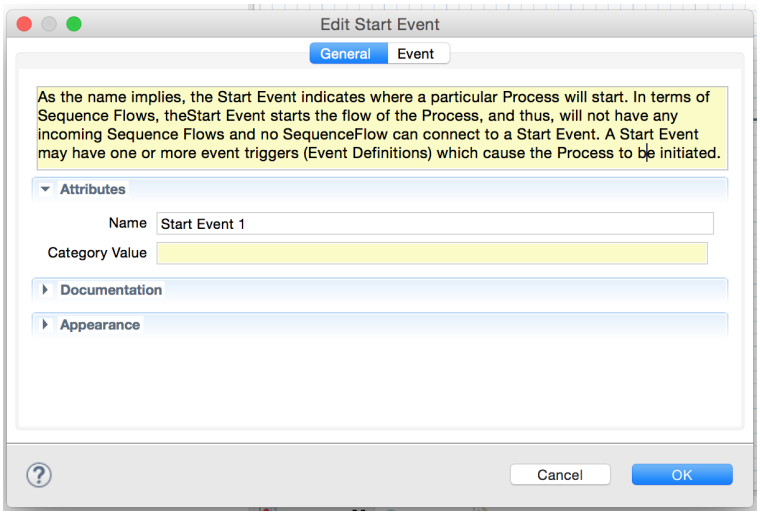


Figura 5.9: Propiedades de los Elementos de la Herramienta.

Esta clase, contempla la incorporación de los recuadros de texto propios a los atributos de cada elemento de BPMN presente en la herramienta y luego realizar la incorporación de estos atributos al código XML del modelo de proceso de negocio.

5.2. Implementación

Para realizar la implementación del plug-in se siguieron las pautas entregadas en el diseño del plug-in, por ello en esta sección se relatará los pasos que se siguieron para la construcción de esta extensión.

Para que la marca DQ se pueda integrar al menú del elemento (el que se muestra al superponer el cursor sobre el elemento y el que se muestra al realizar click derecho en elemento), es necesario crear una clase similar a `ShowPropertiesFeatures` (vista en la sección anterior), la cual tendrá por misión integrar la marca en los menú a través de la clase `Bpmn2ToolBehaviorProvider`.

Debido a lo anterior, se creó una clase llamada `ShowPropertiesFeatures`, la cual está encargada de introducir la marca a los menú, insertar la marca en los elementos de BPMN y, además, iniciar la ventana de propiedades de *dqBP*.

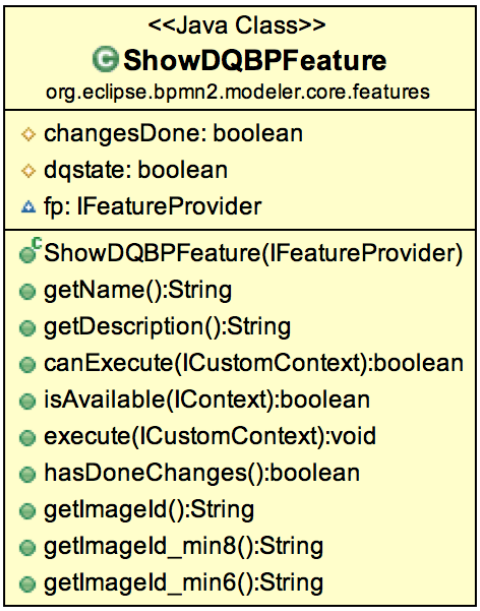


Figura 5.10: Clase `ShowDQBPFeatures`.

En la Figura 5.10, se aprecia la estructura de la clase creada, partiremos por analizar los métodos encargados de entregar la dirección de la imagen, para luego seguir con la disponibilidad de la marca en los métodos, posteriormente la inserción de la marca en los

elementos permitidos y, finalmente, cómo iniciar la ventana de propiedades de los elementos con DQ.

Los siguientes métodos son los encargados de entregar la dirección de la imagen, para esto se agrega una variable de entrada en la clase `IConstants`, para luego realizar la llamada. El motivo de disponer de 3 métodos para las imágenes es incorporar cada imagen en proporción al tamaño del elemento.

```
1  public String getImageId() {
2    return IConstants.ICON_DQBP_16;
3  }
4  public String getImageId_min8() {
5    return IConstants.ICON_DQBP_8;
6  }
7
8  public String getImageId_min6() {
9    return IConstants.ICON_DQBP_6;
10 }
```

Para restringir la disponibilidad de la extensión a ciertos elementos de BPMN, los cuales fueron definidos en el capítulo 2, se especifica un método llamado `isAvailable()`, el cual se utiliza en la clase `Bpmn2ToolBehaviorProvider`, para determinar los elementos en los cuales se puede disponer de la utilización de la marca.

```
1  public boolean isAvailable(IContext context) {
2    if(context instanceof ICustomContext){
3      PictogramElement pes[] = ((ICustomContext)context).getPictogramElements();
4      DiagramEditor editor = (DiagramEditor)getDiagramBehavior()
5                .getDiagramContainer();
6      editor.setPictogramElementForSelection(pes[0]);
7      getDiagramBehavior().refresh();
8      EObject obj = BusinessObjectUtil
9                .getBusinessObjectForPictogramElement(pes[0]);
10     if(obj instanceof Message || obj instanceof DataStoreReference
11       || obj instanceof DataObject|| obj instanceof Task
12       || obj instanceof MessageFlow || obj instanceof Conversation
13       || obj instanceof DataInput || obj instanceof DataOutput)
14         return true;
```

```
15      else return false;
16    }
17    return false;
18  }
```

En cuanto a la inserción de la marca en cada uno de los elementos permitidos, se necesitó conocer el tamaño con el cual la herramienta los dibuja. Es por esto que se llegó al archivo `plugin.xml`, perteneciente al paquete `org.eclipse.bpmn2.modeler.ui`, en el cual se dispone de los tamaños de cada elemento, como se puede visualizar en la Figura 5.11.

```
<style object="SHAPES" shapeBackground="D4E7F8" defaultWidth="110" defaultHeight="50"/>
<style object="DATA" shapeBackground="D4E7F8" defaultWidth="36" defaultHeight="50"/>
<style object="DataInput" shapeBackground="D4E7F8" defaultWidth="36" defaultHeight="50"/>
<style object="DataObject" shapeBackground="D4E7F8" defaultWidth="36" defaultHeight="50"/>
<style object="DataObjectReference" shapeBackground="D4E7F8" defaultWidth="36" defaultHeight="50"/>
<style object="DataOutput" shapeBackground="D4E7F8" defaultWidth="36" defaultHeight="50"/>
<style object="DataStoreReference" shapeBackground="D4E7F8" defaultWidth="50" defaultHeight="50"/>
<style object="Message" shapeBackground="D4E7F8" defaultWidth="30" defaultHeight="20"/>
```

Figura 5.11: Extracto del archivo `plugin.xml`.

Para realizar el proceso de inserción de la imagen, se crearon dos funciones que imitan el funcionamiento de uno de los métodos clase `ShapeDecoratorUtil`, visto en la sección anterior, los cuales por estructuración del proyecto, se crean en la clase antes mencionada.

```
1  public static Image createImageDQ(GraphicsAlgorithmContainer p, String imageId,
2      int x, int y){
3    if (imageId!=null && !imageId.trim().isEmpty()) {
4      Image img = gaService.createImage(p, imageId.trim());
5      gaService.setLocationAndSize(img, x, y, TASK_IMAGE_SIZE, TASK_IMAGE_SIZE);
6      return img;
7    }
8    return null;
9  }
10
11 public static Image createImageDQ(GraphicsAlgorithmContainer p, String imageId,
12     int x, int y, int tam){
13   if (imageId!=null && !imageId.trim().isEmpty()) {
14     Image img = gaService.createImage(p, imageId.trim());
15     gaService.setLocationAndSize(img, x, y, tam, tam);
16     return img;
17   }
```

```
18    return null;
19  }
```

El primer método permite insertar las imágenes con el tamaño normal de 16x16 pixeles y el segundo, permite integrar imágenes con tamaños diferentes.

Para la ejecución de la inserción, se diseña el método `execute()`, perteneciente a la clase `ShowDQBPFeatures`, en donde se realiza la ejecución del métodos descritos anteriormente, incorporando la posición de cada elemento, teniendo presente el tamaño.

```
1  public void execute(ICustomContext context) {
2    PictogramElement[] pes = context.getPictogramElements();
3    DiagramEditor editor = (DiagramEditor)getDiagramBehavior()
4        .getDiagramContainer();
5    editor.setPictogramElementForSelection(pes[0]);
6    getDiagramBehavior().refresh();
7    EObject businessObject = BusinessObjectUtil
8        .getBusinessObjectForPictogramElement(pes[0]);
9
10   if(!dqstate){
11     ContainerShape x = (ContainerShape) fp
12         .getPictogramElementForBusinessObject(businessObject);
13     if(businessObject instanceof Message){
14       ShapeDecoratorUtil.createImageDQ(fp
15         .getPictogramElementForBusinessObject(businessObject)
16         .getGraphicsAlgorithm(), getImageId_min6(), 12, 1, 6);
17     }
18     if(businessObject instanceof DataStoreReference){
19       ShapeDecoratorUtil.createImageDQ(context.getInnerPictogramElement()
20         .getGraphicsAlgorithm(), getImageId_min8(), 22, 8, 6);
21     }
22     if(businessObject instanceof Conversation){
23       ShapeDecoratorUtil.createImageDQ(x.getChildren().get(0)
24         .getGraphicsAlgorithm(), getImageId(), 25, 2);
25     }
26     if(businessObject instanceof DataInput
27       || businessObject instanceof DataOutput){
28       ShapeDecoratorUtil.createImageDQ(x.getChildren().get(0)
29         .getGraphicsAlgorithm(), getImageId(), 2, 20);
30     }
```

```
31      if(businessObject instanceof MessageFlow){
32        ShapeDecoratorUtil.createImageDQ(fp
33          .getPictogramElementForBusinessObject(businessObject)
34          .getGraphicsAlgorithm(), getImageId(), 10, 1);
35      }
36
37      if(businessObject instanceof BusinessRuleTask
38          || businessObject instanceof ManualTask
39          || businessObject instanceof UserTask
40          || businessObject instanceof ScriptTask
41          || businessObject instanceof ServiceTask
42          || businessObject instanceof SendTask
43          || businessObject instanceof ReceiveTask){
44        ShapeDecoratorUtil.createImageDQ(x.getChildren().get(0)
45          .getGraphicsAlgorithm(), getImageId(), 20, 2);
46      }else{
47        if(businessObject instanceof DataObject
48            || businessObject instanceof Task){
49          ShapeDecoratorUtil.createImageDQ(x.getChildren().get(0)
50            .getGraphicsAlgorithm(), getImageId(), 2, 2);
51          }
52        }
53        dqstate = true;
54      }
55    ...
56  }
```

Para desplegar la ventana de propiedades se ejecuta el método anteriormente descrito, el cual en su parte final realiza el despliegue de una ventana de propiedades normal y es en esta en donde se implementó una pestaña que permite describir los atributos asociados a la Calidad de Datos.

```
1  public void execute(ICustomContext context) {
2    ...
3
4    ObjectEditingDialog dialog = new ObjectEditingDialog(editor, businessObject);
5    if (dialog.open() == Window.OK)
6      changesDone = dialog.hasDoneChanges();
7    else
```

```
8       changesDone = false;
9   }
```

Para implementar la pestaña, se debe describir primero en el archivo `plugin.xml`, perteneciente al paquete `org.eclipse.bpmn2.modeler.ui`, dónde se describe el nombre de la pestaña y en los elementos en que está disponible, además, la clase que se encarga de manejar dicha pestaña.

```
1   <propertyTab
2     id="org.eclipse.bpmn2.modeler.dqbp.tab"
3     class="org.eclipse.bpmn2.modeler.ui.property.DQBPPropertySection"
4     type="org.eclipse.bpmn2.Message
5   ␣␣␣␣␣␣␣␣org.eclipse.bpmn2.DataStoreReference
6   ␣␣␣␣␣␣␣␣org.eclipse.bpmn2.DataOutput
7   ␣␣␣␣␣␣␣␣org.eclipse.bpmn2.DataInput
8   ␣␣␣␣␣␣␣␣org.eclipse.bpmn2.DataStore
9   ␣␣␣␣␣␣␣␣org.eclipse.bpmn2.DataObject
10  ␣␣␣␣␣␣␣␣org.eclipse.bpmn2.DataObjectReference
11  ␣␣␣␣␣␣␣␣org.eclipse.bpmn2.MessageFlow
12  ␣␣␣␣␣␣␣␣org.eclipse.bpmn2.Task"
13    label="dqBP␣Extension">
14  </propertyTab>
```

Esta descripción, permite que al ejecutar la ventana de propiedades se pueda visualizar una nueva pestaña, con el nombre de *dqBP Extension*. Este archivo, además, define la clase encargada de la pestaña, llamada `DQBPPropertySection`, perteneciente al paquete `org.eclipse.bpmn2.modeler.ui`. Esta clase se describe a continuación:

```
1   public class DQBPPropertySection extends DefaultPropertySection {
2     public DQBPPropertySection() {
3       super();
4     }
5     protected AbstractDetailComposite createSectionRoot() {
6       return new DQBPDetailComposite(this);
7     }
8     public AbstractDetailComposite createSectionRoot(Composite parent,
9       int style) {
10      return new DQBPDetailComposite(parent, style);
```

```
11    }
12    public class DQBPDetailComposite extends DefaultDetailComposite {
13      public DQBPDetailComposite(AbstractBpmn2PropertySection section) {
14        super(section);
15      }
16      public DQBPDetailComposite(Composite parent, int style) {
17        super(parent, style);
18      }
19
20      @Override
21      public AbstractPropertiesProvider getPropertiesProvider(EObject object) {
22        if (propertiesProvider == null) {
23          propertiesProvider = new AbstractPropertiesProvider(object) {
24            String[] properties = new String[] {
25                "name", //$NON-NLS-1$
26                "documentation", //$NON-NLS-1$$
27            };
28            @Override
29            public String[] getProperties() {
30              return properties;
31            }
32          };
33        }
34        return propertiesProvider;
35      }
36    }
37  }
```

El método `getPropertiesProvider()` entrega los elementos que se crean en la pestaña, en los cuales posteriormente, se definen los atributos de Calidad de Datos de los elementos.

Estas son las principales modificaciones realizadas a la herramienta para la creación de un prototipo de la extensión *dqBP*, de esta forma se logra generar el primer plug-in que permite describir requisitos de Calidad de Datos en un modelo de Procesos de Negocio.

Capítulo 6

Pruebas

En este capítulo se relatarán las pruebas a los elementos, las cuales fueron realizadas para comprobar el correcto funcionamiento del prototipo, mostrando paso a paso la descripción de requisitos de Calidad de Datos en el Modelo de Procesos de Negocio.

Para las pruebas se utilizará un modelo básico (Figura 6.1), diseñado con la herramienta que BPMN2 Modeler de Eclipse, pero no consta de la extensión *dqBP*, al cual se le integrará la marca de Calidad de Datos y la descripción de ésta.

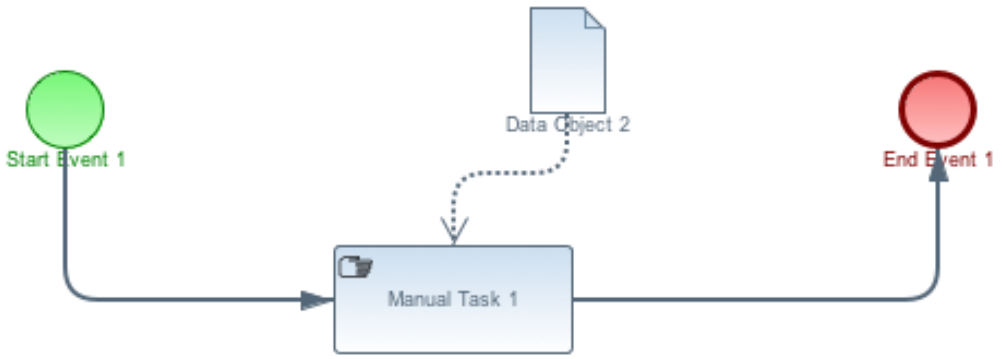


Figura 6.1: Modelo básico de un Proceso de Negocio.

6.1. Menú con la extensión

Luego de crear el modelo de procesos de negocio básico, se comprueba que el menú que se visualiza al superponerse sobre los elementos se visualice y muestre la marca cuando corresponda, y de igual manera, suceda con el menú que se visualiza al realizar click derecho sobre los elementos.

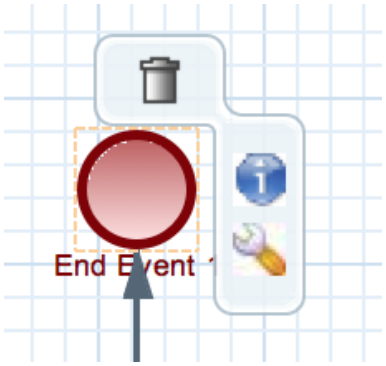


Figura 6.2: Menú Contextual de un End Event.

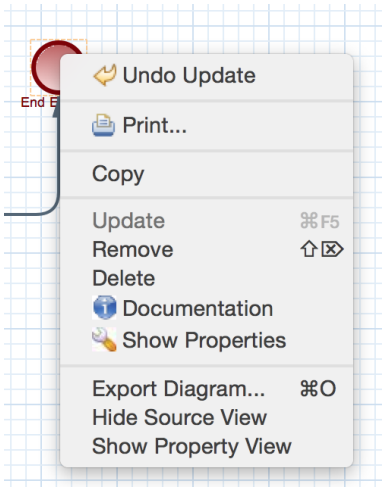


Figura 6.3: Menú de opciones de un End Event.

En las Figuras 6.2 y 6.3, podemos ver que con un elemento que no pertenece a los descritos en la Tabla 2.1, la opción de utilizar la extensión *dqBP* no está presente.

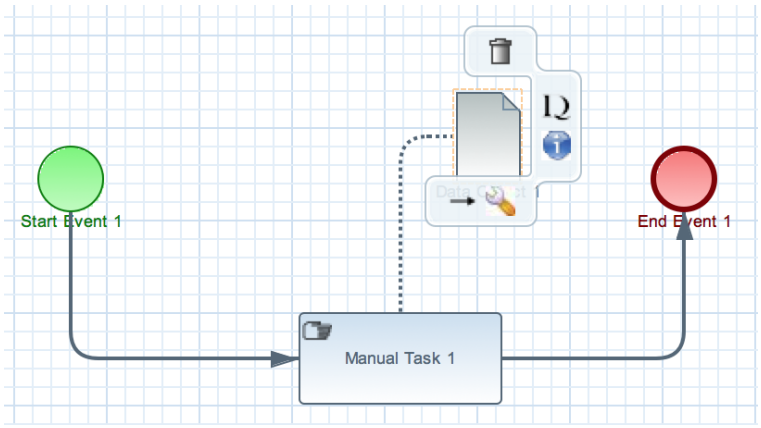


Figura 6.4: Menú Contextual de un Data Object.

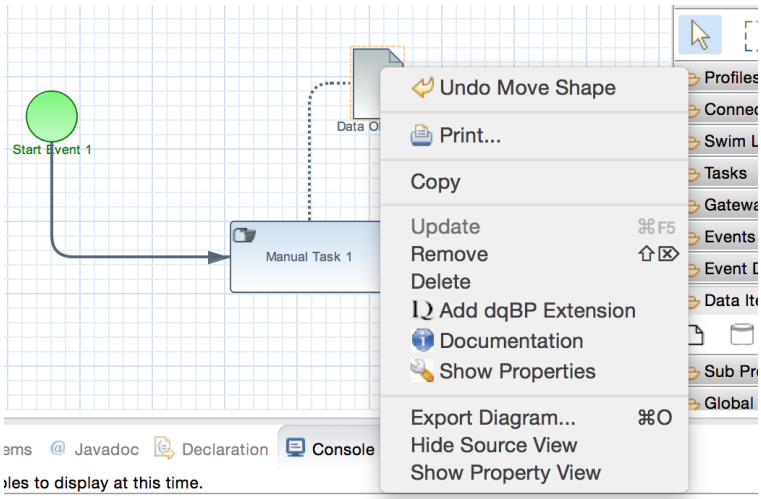


Figura 6.5: Menú de opciones de un Data Object.

Distinto es lo que sucede en las Figuras 6.4 y 6.5, en donde sí se puede visualizar la opción de la extensión para ser incorporada al elemento correspondiente en el modelo.

Por lo tanto, el plug-in discrimina a los elementos en que se puede incorporar *dqBP*, además muestra de manera gráfica la opción de integración de la extensión, lo cual genera cercanía con el usuario y aumenta la usabilidad de la herramienta.

6.2. Descripción de requisitos de Calidad en el Modelo

Al agregar la extensión de Calidad de Datos en uno de los elementos permitidos, se debe desplegar una ventana con la opción de integración de dichos atributos. En este primer prototipo solo de integrará la opción de cambio de nombre y descripción de la extensión. Pero en este último recuadro de texto será posible agregar la Fuente y Soporte del elemento.

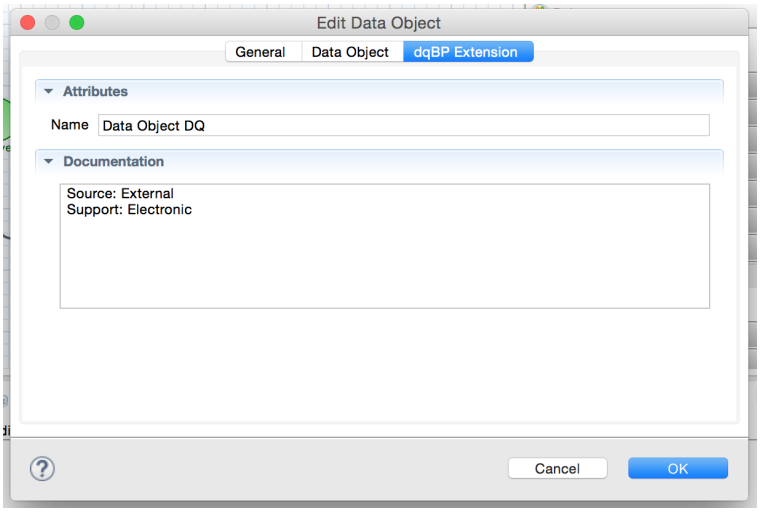


Figura 6.6: Ventana Propiedades Data Object: *Extension dqBP*.

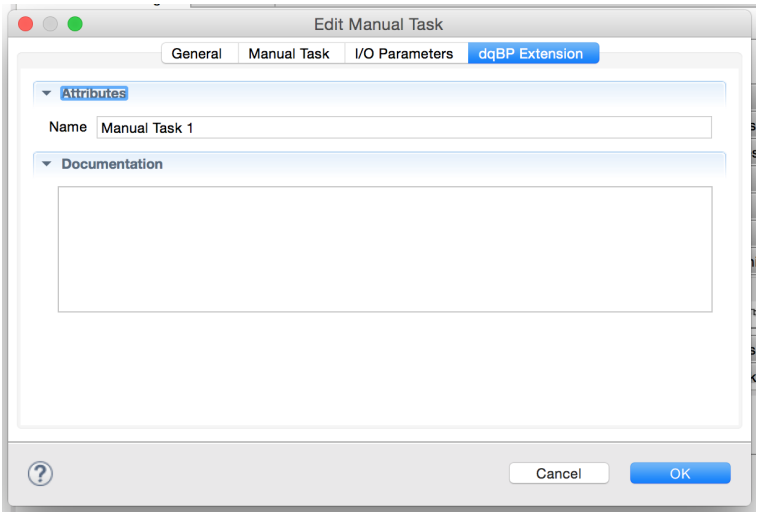


Figura 6.7: Ventana Propiedades Manual Task: *Extension dqBP*.

En las Figuras 6.6 y 6.7 se puede ver la nueva pestaña en las propiedades de los elementos

que permiten la marca *dqBP*. Cabe destacar que está pestaña esta presente sólo en los elementos de BPMN en que se permite el *dqBP*, lo cual se puede visualizar en la Figura 6.8.

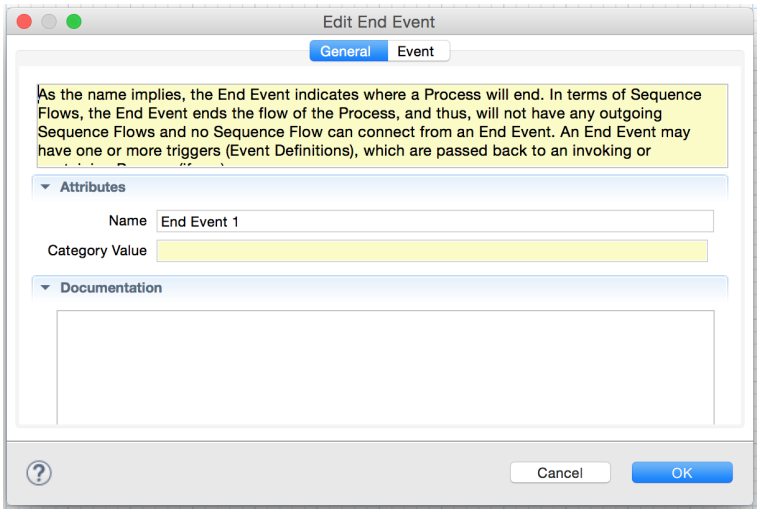


Figura 6.8: Ventana Propiedades End Event: *Extension dqBP*.

6.3. Inserción de la marca en los Elementos permitidos

Luego de registrar la descripción de un elemento de Calidad de Datos, se debe apreciar la marca de calidad de datos en los elementos pertenecientes al modelo y que incorporan dicha marca. Esta inserción puede apreciarce en la Figura 6.9, donde aparece la marca incorporada en los elementos.

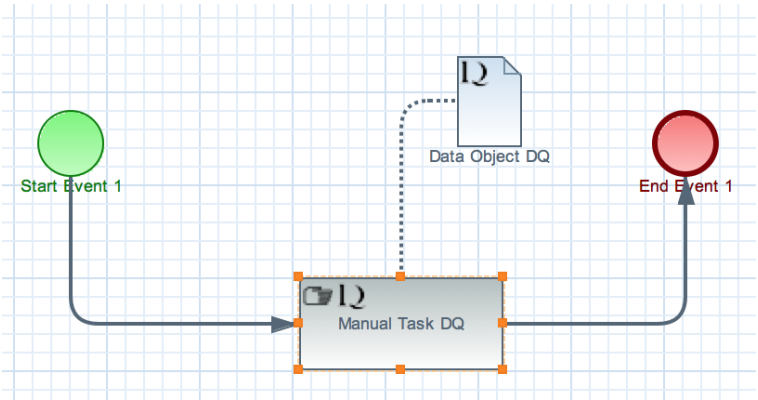


Figura 6.9: Elementos con la marca *dqBP*.

6.4. Descripción de los Requisitos de Calidad en el código XML del Modelo

Las proyecciones de este prototipo, es que el modelo de proceso de negocio se pueda analizar, mejorar y, finalmente, describir Casos de Uso, que contemplen la Calidad de Datos, como se describió en el capítulo 2. Para realizar lo anterior es esencial dejar registrada la información del modelo, el IDE Eclipse con su herramienta de modelado de Procesos de Negocio, lo realiza por medio de la escritura de un fichero XML con extensión ".bpmn2". Debido a esto, la información captada por el modelo con Calidad de Datos debe verse reflejada en el código XML.

A continuación se muestra parte del código XML resultante luego de la incorporación de Calidad de Datos:

```
1  <bpmn2:definitions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2      xmlns:bpmn2="http://www.omg.org/spec/BPMN/20100524/MODEL"
3      xmlns:bpmndi="http://www.omg.org/spec/BPMN/20100524/DI"
4      xmlns:dc="http://www.omg.org/spec/DD/20100524/DC"
5      xmlns:di="http://www.omg.org/spec/DD/20100524/DI"
6      xmlns:ext="http://org.eclipse.bpmn2/ext"
7      xmlns:xs="http://www.w3.org/2001/XMLSchema" id="Definitions_1"
8      exporter="org.eclipse.bpmn2.modeler.core"
9      exporterVersion="1.2.0.201506231116"
10     targetNamespace="http://org.eclipse.bpmn2/default/process">
11   <bpmn2:process id="process_1" name="Default Process" isExecutable="false">
12     <bpmn2:startEvent id="StartEvent_1" name="Start Event 1">
13       <bpmn2:outgoing>SequenceFlow_4</bpmn2:outgoing>
14     </bpmn2:startEvent>
15     <bpmn2:endEvent id="EndEvent_1" name="End Event 1">
16       <bpmn2:incoming>SequenceFlow_5</bpmn2:incoming>
17     </bpmn2:endEvent>
18     <bpmn2:manualTask id="ManualTask_1_DQ"  name="Manual Task DQ">
19       <bpmn2:documentation id="Documentation_12">Source: Internal
20 Support: No-Electronic</bpmn2:documentation>
21       <bpmn2:incoming>SequenceFlow_4</bpmn2:incoming>
22       <bpmn2:outgoing>SequenceFlow_5</bpmn2:outgoing>
23     </bpmn2:manualTask>
24     <bpmn2:sequenceFlow id="SequenceFlow_4" sourceRef="StartEvent_1"
```

```
25          targetRef="ManualTask_1_DQ" />
26      <bpmn2:sequenceFlow id="SequenceFlow_5" sourceRef="ManualTask_1_DQ"
27          targetRef="EndEvent_1" />
28      <bpmn2:dataObject id="DataObject_2_DQ" name="Data Object DQ">
29        <bpmn2:documentation id="Documentation_10">Source: External
30 Support: Electronic</bpmn2:documentation>
31      </bpmn2:dataObject>
32      <bpmn2:association id="Association_1" sourceRef="DataObject_2_DQ"
33          targetRef="ManualTask_1_DQ" />
34    </bpmn2:process>
35    ...
36 </bpmn2:definitions>
```

En las lineas 18, 19 y 20 del código XML anterior, se puede ver reflejado el ingreso de información de calidad de datos para el elemento *Manual Task* y en las líneas 28, 29 y 30 se puede ver la misma adquisición de Calidad de Datos para el elemento *Data Object*.

En base a todo lo anterior, se puede concluir que el prototipo del plug-in desarrollado, cumple con los requerimientos establecidos en el capítulo 5, pudiendo especificar Calidad de Datos en algunos de los elementos de BPMN, de esta manera se concreta el primer paso del Método BPiDQ*.

Capítulo 7

Conclusiones

Con la implementación de este prototipo, se da el primer paso para lograr la especificación de Calidad de Datos en los elementos de la notación BPMN, para ser usados en una herramienta de modelado de procesos de negocio.

La especificación de Calidad de Datos va más allá de colocar una marca gráfica en los elementos de BPMN 2.0, ya que, aparte de eso, el método BPiDQ* y en particular, este prototipo que implementa parte del método, logra definir atributos de Calidad de Datos que intervienen en los elementos del modelo de Proceso de Negocio.

Además, luego de someter a un análisis y posterior evaluación a las herramientas Open Source modeladoras de Procesos de Negocio, con criterios previamente definidos, se identifica la herramienta apta para la construcción del prototipo de esta extensión e incluso, para otras extensiones definidas o por definir, que utilicen BPMN 2.0 como notación para el modelado de Procesos de Negocio. Debido a que la herramienta seleccionada permite el desarrollo de un plug-in con la incorporación de marcas gráficas en los elementos y además, la especificación de característica asociadas a la marca.

Me es relevante mencionar que en este trabajo la modificación de la herramienta no presenta mayores dificultades. Los inconvenientes, aparecen en una etapa previa a la modificación, cuando se deben identificar qué clases o componentes se modificarán o si se deben crear nuevas clases, para que el funcionamiento de la herramienta no se vea interferido o entorpecido. La mayor parte de este trabajo se centró en la identificación de los componentes que se debían modificar o extender para implementar el funcionamiento de la extensión.

Para finalizar, queda mencionar que como trabajo futuro está el conseguir un prototipo que implemente el método BPiDQ* completamente. De esta manera se probará la efectividad del método y se brindará a los analistas de negocios, la primera herramienta que incorpore la Calidad de Datos en el modelado de procesos de negocio.

Bibliografía

BPMN Offensive Berlin. Bpmn 2.0 poster, 2013. URL http://bpmb.de/poster. [Web; accedido el 21-04-2015].

Bonitasoft. Bonita bpm: Open source bpm, 2009. URL http://www.bonitasoft.com. [Web; accedido el 23-04-2015].

Angélica Caro, Alejandra Fuentes, and M. Antonieta Soto. Desarrollando sistemas de información centrados en la calidad de datos. *Ingeniare. Revista chilena de ingeniería*, 21:54 – 69, 04 2013. ISSN 0718-3305. URL http://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0718-33052013000100006&nrm=iso.

Michele Chinosi and Alberto Trombetta. Bpmn: An introduction to the standard. *Computer Standards & Interfaces*, 34(1):124 – 134, 2012. ISSN 0920-5489. doi: http://dx.doi.org/10.1016/j.csi.2011.06.002. URL http://www.sciencedirect.com/science/article/pii/S0920548911000766.

Codehoop. Bpmn2 visual editor for eclipse, 2012. URL https://github.com/imeikas/BPMN2-Editor-for-Eclipse. [Web; accedido el 24-04-2015].

Eclipse Foundation. Bpmn 2.0 modeler project, 2013. URL https://www.eclipse.org/bpmn2-modeler/. [Web; accedido el 24-04-2015].

Camunda Services GmbH. Camunda modeler, 2013. URL http://camunda.org. [Web; accedido el 24-04-2015].

Yaoqiang Inc. Yaoqiang bpmn editor, 2010. URL http://sourceforge.net/projects/bpmn/. [Web; accedido el 22-04-2015].

ISO/IEC-25012. ISO/IEC 25012: Software Engineering - Software Quality Requirements and Evaluation (SQuaRE) - Data Quality Model. *ISO/IEC-25012*, 2008. URL http://www.iso.org/iso/catalogue_detail.htm?csnumber=35736.

Modeliosoft. Modelio entorno de modelado, 2011. URL https://www.modelio.org. [Web; accedido el 22-04-2015].

MyBPMN. Mybpmn, 2009. URL http://mybpmn.sourceforge.net. [Web; accedido el 22-04-2015].

Luis Ortega, Angélica Caro, and Alfonso Rodríguez. Identificacioón de herramientas para el modelado de procesos de negocio desde la perspectiva de su extensibilidad. In *III Workshop on Business Process Management. Talca, Chile.*, 2014.

Alfonso Rodríguez and Angélica Caro. Obteniendo Casos de Uso centrados en la Calidad de los Datos desde Procesos de Negocio descritos con BPMN. *RISTI - Revista Ibérica de Sistemas e Tecnologias de Informacão*, pages 65 – 80, 12 2012. ISSN 1646-9895. URL http://www.scielo.mec.pt/scielo.php?script=sci_arttext&pid=S1646-98952012000200006&nrm=iso.

Signavio. Signavio core componentes, 2010. URL https://code.google.com/p/signavio-core-components/. [Web; accedido el 22-04-2015].

Luis Stroppi. Bpmnx, 2010. URL https://code.google.com/p/bpmnx/. [Web; accedido el 23-04-2015].

Stephen A. White and Derek Miers. *Guia de Referencia y Modelado BPMN, Comprendiendo y utilizando BPMN*. Future Strategies Inc., 2009.