UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

DESARROLLO DE UN INTERFAZ GRÁFICO PARA EL MODELADO DE PROCESS CHAIN NETWORKS

Alberto López Moreno
Tutor: María Elena Gómez Martínez
Ponente: Juan De Lara Jaramillo

Julio 2020

DESARROLLO DE UN INTERFAZ GRÁFICO PARA EL MODELADO DE PROCESS CHAIN NETWORKS

AUTOR: Alberto López Moreno
TUTOR: María Elena Gómez Martínez

Dpto. Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio de 2020

Resumen

Este Trabajo Fin de Grado consiste en el desarrollo de un software integrado en el entorno Eclipse que permita generar e interpretar modelos de negocio de acuerdo a la notación Process Chain Networks (PCN).

El propósito de la herramienta es crear un plugin instalable en cualquier sistema que disponga de Eclipse, dando la posibilidad de crear diagramas desde cero, generando su modelo o generar un diagrama en base a un modelo previamente especificado.

La notación PCN utiliza una lógica basada en el flujo de procesos de negocio, en el que los agentes que interactúan con los procesos se denominan entidades, que pueden ser personas, empresas, clientes o incluso otros programas software. La característica más destacable de las entidades es la distribución de sus modelos de dominio, que forman grupos de procesos organizados según el nivel de interacción que tiene la entidad sobre ellos, de tal forma que existen procesos sobre los que la entidad tiene una interacción total y procesos marcados por su relación con otros procesos, más que por la propia interacción de su entidad.

Los procesos incluidos en el diagrama podrán estar conectados mediante el uso de relaciones, las cuales determinan el orden de flujo entre los procesos, desde el comienzo de la operación hasta la conclusión del proceso de negocio.

Con el fin de proporcionar más información sobre el flujo de procesos se utilizan etiquetas, que pueden simbolizar varios significados, como un coste monetario del proceso para la empresa o un posible retraso en el flujo, producido por el proceso especificado.

La herramienta proporciona la posibilidad de crear elemento a elemento un diagrama PCN al gusto del cliente mediante una barra de opciones organizada por categorías de elemento.

Palabras clave

Proceso, entidad, diagrama, relación, etiqueta, Eclipse, Sirius

Abstract

This Bachelor Thesis consists on the development of a software integrated on the Eclipse development environment, that allows users to generate and interpret business models using the Process Chain Networks notation (PCN).

The main purpose of the tool is to create an installable plugin in any system that has Eclipse installed, giving the user the possibility to create diagrams from the beginning generating a model or generate a diagram from a previously specified model.

PCN notation uses a logic based on the business processes flow, where the agents that interact with the processes are called entities, who can be people, companies, clients, or other software programs. The most valuable characteristic of the entities is the distribution of their domain models, that forms processes groups organised according to the level of interaction that the entity has above them, so there will be processes where the entity has a total interaction on them and processes defined by the relations that the process has with other processes.

The processes included on the diagram could be connected using relations, which determine the order of flow between the processes, from the beginning of the operation until the conclusion of the business process.

With the purpose of providing more information about the process flow, tags are used. These tags can symbolize different meanings, like a monetary cost of the process for the company or a possible delay on the flow produced by the specified process.

The tool supports the possibility of the creation a PCN diagram element by element as the client deserve with the use of a toolbar organised by element categories.

Keywords

Process, entity, diagram, relation, tag, Eclipse, Sirius

Agradecimientos

A mi tutora Elena por su paciencia y disposición conmigo como alumno, además de facilitar una comunicación casi inmediata.

A mi madre por estar siempre ahí.

A mi novia por creer en mí durante este último año y darme ánimos cuando más hacía falta.

Y por último a mis amigos por darme la posibilidad de evadirme durante el duro recorrido que ha supuesto este grado.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

1 Introducción

En esta sección se va a presentar una introducción que muestre un marco general del proyecto, indicando el propósito del trabajo y los objetivos del mismo.

Un modelo de negocio es una herramienta utilizada previamente a la toma de decisiones necesarias en un negocio, que permite definir qué se va a ofrecer al mercado, de qué forma se va a hacer, hacia qué clientes va dirigido el producto, cómo se va a vender y de qué forma se van a generar ingresos. Cuando se habla de modelo de negocio se suele definir como la forma en la que una empresa genera beneficios [1].

El modelo de negocio es una herramienta indispensable a la hora de definir las maneras en las que una empresa crea, provee y obtiene beneficio, y es una pieza clave en la toma de decisiones de negocio a corto, medio y largo plazo.

La modelización de procesos de negocio es una actividad esencial para empresas competitivas, ya que permite documentar, analizar, mejorar o agilizar los procesos fundamentales del negocio.

Básicamente la modelización de procesos de negocio proporciona la posibilidad de representar gráficamente los servicios o productos que ofrece un negocio o empresa, especificando con detalle las pautas y pasos que se necesitan para conseguir cumplir sus metas.

Se proponen en el mercado algunas notaciones distintas para el modelado de procesos tales como BPMN (Business Process Model and Notation), UML (Unified Modeling Language) o PCN (Process Chain Networks), las cuales se explicarán en esta memoria. Estas notaciones son gráficas e intuitivas, útiles para la documentación, pero es necesaria una herramienta que soporte dichas notaciones y una semántica formal común para cada notación si el objetivo es el análisis de procesos. Sin embargo, para la notación PCN, no existen herramientas gráficas que la soporten, a pesar de ser una notación que aporta una notable eficiencia y expresa con detalle información acerca del proceso de negocio involucrado.

1.1 Motivación

La elección de este proyecto se debe a la versatilidad que ofrece Eclipse a la hora de desarrollar cualquier proyecto, y la preferencia por un trabajo de desarrollo gráfico ante un desarrollo típicamente computacional o de programación.

PCN tiene un punto de vista diferente respecto a la categorización de actividades en comparación con otras notaciones similares, el cual consigue enfoque más extenso a la información que se muestra en sus diagramas.

Además, PCN diferencia entre los tipos de procesos basados en su naturaleza de interacción con el resto de procesos para entender cómo el proveedor del servicio puede reconfigurar dichos procesos acorde al resto de elementos.

Siendo una notación utilizada ampliamente en negocios, no existen herramientas que soporten el análisis, verificación y validación de modelos PCN que aseguren una buena funcionalidad de los diagramas PCN diseñados. Por ello es importante el desarrollo de una herramienta que acerque esta notación al mercado y los posibles usuarios que quieran aprovechar la potencia de la notación PCN.

1.2 Objetivos

El principal objetivo de este proyecto es desarrollar una herramienta gráfica abierta basada en ingeniería de modelos (Model-Driven Engineering, MDE) para el modelado de procesos con la notación Process Chain Network (PCN), ya que, a pesar de proporcionar un mejor análisis de procesos en comparación con otras notaciones para el modelado de negocios, no existen herramientas gráficas abiertas que soporten modelados con PCN, por lo que su desarrollo supone una oportunidad de progreso en este ámbito.

Una vez desarrollada la herramienta, el siguiente objetivo será la creación de un plugin de Eclipse que contenga la funcionalidad implementada, es decir, una extensión instalable en el software de Eclipse, dando la posibilidad de agregar la utilidad de la herramienta a otros proyectos o sistemas compatibles. Dicho plugin podrá ser una extensión de otras herramientas de modelización de procesos de negocio o de algún sistema de análisis de modelados PCN.

1.3 Organización de la memoria

La memoria del presente Trabajo de Fin de Grado consta de los siguientes capítulos:

En el Capítulo 2, se definen las nociones básicas necesarias para comprender el proyecto y su funcionalidad, explicando cada uno de los conceptos involucrados en el proceso.

En el Capítulo 3, se exponen las tecnologías disponibles en el mercado que tienen más relación con el proyecto, así como como ejemplos de sus funcionalidades.

En el Capítulo 4, se define el diseño que sigue la notación utilizada en el proyecto, en este caso PCN, explicando cada elemento de la notación y su función con el resto de componentes del proyecto.

En el capítulo 5, se expone la implementación de la herramienta creada paso a paso y la explicación de las funcionalidades que se han creado.

En el capítulo 6, se explica el proceso de pruebas del software creado y los resultados finales obtenidos utilizando la herramienta a un caso de uso.

En el capítulo 7, se indican las conclusiones finales del trabajo y posibles opciones de cara a futuro de ampliar el desarrollo del mismo.

2 Conceptos básicos

En esta sección se expondrán los conceptos básicos necesarios para entender este proyecto y su memoria, así como las herramientas que se han utilizado en el proceso.

Para comprender la terminología de este documento es importante distinguir entre los conceptos de modelo y metamodelo.

Por un lado, el metamodelo es el análisis, construcción y desarrollo de los macros, reglas, restricciones, modelos y teorías aplicables y útiles para el modelado [2]. Es la agrupación de "conceptos" dentro de un dominio determinado.

Por otro lado, un modelo es una abstracción de la situación en el mundo real, siguiendo las reglas del metamodelo que le corresponda. El modelo se ajusta a su metamodelo en la forma en que un programa de ordenador se ajusta a la gramática del lenguaje correspondiente en el que está definido [2].

Por tanto, podemos saber que un metamodelo compone una estructura de modelaje determinada junto con unas reglas y restricciones mientras que un modelo es un ejemplo práctico que sigue las condiciones de su metamodelo correspondiente.

Para tener una visión general del proyecto es necesario comprender el tipo de desarrollo que se ha utilizado. Típicamente el desarrollo de software se implementa a bajo nivel, mediante lenguajes de programación que siguen una determinada codificación, sin embargo, existe otra forma de desarrollar aplicaciones o herramientas de software de más alto nivel basada en la utilización de modelos de dominio.

Un modelo de dominio o también llamado modelo conceptual, es una representación visual de clases conceptuales o de objetos reales en un determinado dominio [3]. Dichos modelos son utilizados en muchos sectores, sin embargo, en este proyecto nos enfocaremos en su utilización en los procesos de negocio, ya que, los modelos de dominio de este tipo son utilizados por los analistas de negocios para entender cómo se desenvolverá el flujo de negocio en situaciones determinadas.

Los modelos de dominio se definen mediante la utilización de la metodología MDE [4], que son las siglas de Model-Driven Engineering (Ingeniería dirigida por modelos). Es una metodología de desarrollo de software enfocada en la creación de modelos de dominio. Destaca las representaciones abstractas de conocimiento y las actividades que componen una aplicación frente a los conceptos computacionales de la misma.

A raíz de la ingeniería dirigida por modelos, surge el desarrollo orientado a modelos (MDD, de las siglas Model-Driven Development), que proporciona un desarrollo colaborativo de software rápido y eficaz, utilizando la metodología MDE. El desarrollo orientado a modelos utiliza técnicas de modelado gráfico para definir relaciones de datos, lógica de procesos, o crear interfaces de usuario de forma que el desarrollador pueda lanzar el software sin la necesidad de ver ni una línea de código, por lo que es un desarrollo mucho más rápido que la utilización de los lenguajes de programación tradicionales [4].

Siguiendo el desarrollo orientado a modelos, surge el modelado específico para procesos de negocio, el cual representa la actividad de procesos de una empresa, con el fin de analizar mejorar o automatizar cada proceso del flujo de negocio. Este modelado específico para negocios define los denominados modelos de procesos de negocio.

Un proceso de negocio es un flujo de trabajo, formado por una serie de pasos (o procesos), requeridos para completar una transacción de negocio. Estos flujos de procesos pueden requerir la invocación de aplicaciones o la intervención humana para su realización.

La notación más común a la hora de definir modelos de procesos de negocio es BPMN (Business Process Model and Notation, Modelo y Notación de Procesos de Negocio), una notación gráfica que permite el modelado de procesos de negocio, con una estructura de flujo de trabajo.

Esta notación describe los pasos de un proceso de negocio y ha sido diseñada en especial para coordinar el flujo de los procesos con los mensajes que transcurren entre los participantes del mismo. Es la notación más utilizada en modeladores de procesos de negocio, lo cual ha conseguido que sea un estándar para todos los analistas de negocio y desarrolladores de este sector [5].

Este proyecto da una solución para el modelado de procesos de negocio siguiendo la notación PCN (Process Chain Network, red de cadena de procesos) y como su nombre indica, es una técnica de modelado de servicios que permite una visualización clara de la interacción entre clientes y proveedores de servicios mediante el uso de cadenas de procesos.

Al igual que BPMN, PCN es una técnica de modelado de negocios, utilizada frecuentemente para visualizar el proceso de distribución de servicios. Sin embargo, mientras que BPMN se centra en la comunicación entre las diferentes entidades de la organización (departamentos, roles y sistemas), PCN está centrado en la interacción con los servicios que recibe el usuario [6].

En su notación se incluyen entidades, que son los agentes que interaccionan con los procesos y determinan su flujo, y procesos, que son los propios pasos de la cadena o flujo. Además, PCN incluye *tags* (etiquetas) que pueden pertenecer a un proceso y determinan información adicional sobre el mismo, como el coste o beneficio que pudiera tener dicho proceso o si va a producir una demora en el flujo [6].

La forma en la que interactúan los procesos entre sí en PCN a través de las entidades y la utilización de las etiquetas para ampliar información sobre el flujo de procesos hacen que PCN sea una notación que proporcione un análisis para modelado de procesos de negocio superior al de BPMN.

El proyecto se ha desarrollado al completo en el entorno Eclipse, una plataforma de desarrollo que da la posibilidad de agregar muchos tipos de *plugin* distintos con el fin de extender su funcionalidad adaptada a las necesidades del usuario.

Aunque no está enfocado en un lenguaje específico, su popularidad reside principalmente entre la comunidad de desarrolladores de Java. Sus principales características son la eficiente gestión de proyectos que ofrece, su fácil e intuitivo depurador y su amplia gama de *plugins* o extensiones instalables que lo convierten en un entorno muy polivalente.

Para comprender el lenguaje utilizado para el desarrollo de modelos es necesario conocer el lenguaje XML, en el que está basado. XML son las siglas de Extensible Markup Language, que se traduce como Lenguaje de Marcas Extensibles. Los archivos XML se componen de etiquetas que aportan datos e información. Dichas etiquetas pueden definirse individualmente o de forma anidada, estableciendo el nombre que especifique el desarrollador a cada una [7].

La creación de modelos y otras utilidades del modelado se desarrollan utilizando XMI (XML Metadata Interchange, XML de intercambio de metadatos) y como su nombre indica es una extensión del lenguaje XML. Dicho lenguaje se ha definido para facilitar el intercambio de

modelos entre herramientas de modelado y repositorios de metadatos como en el caso de este proyecto, en el que los modelos PCN se definen utilizando XMI [8].

El lenguaje XMI es una especificación definida por la OMG (Object Management Group), un consorcio abierto sin ánimo de lucro dedicado a la definición de estándares relacionados con la tecnología.

A continuación, se muestra un ejemplo del código XMI utilizado en uno de los modelos del proyecto, en el que se puede comprobar la estructura típica basada en XML y la especificación de la OMG como fuente del lenguaje:

```
<?xml version="1.0" encoding="UTF-8"?>
<pcn:PCNDiagram xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:pcn="pcn">
  <diagramEntities name="restaurante"/>
</pcn:PCNDiagram>
```

En el fragmento de código podemos observar que se utiliza el metamodelo PCN para definir una clase llamada PCNDiagram, que a su vez contiene un elemento de tipo diagramEntities llamado “restaurante”.

Respecto a las funcionalidades necesarias de Eclipse, la versión básica del programa no contiene herramientas que permitan modelar diagramas, es por ello que se utiliza EMF (Eclipse Modeling Framework). EMF es un *framework* de modelado y generación de código desarrollado para Eclipse que permite crear herramientas o aplicaciones basadas en un modelo estructurado de datos. Desde un modelo definido en XMI, EMF proporciona herramientas para producir un conjunto de clases Java para dicho modelo.

Actualmente EMF es un estándar para modelos de datos sobre el que muchas otras tecnologías están basadas. Las herramientas de modelaje incluidas en EMF permiten la creación de herramientas que implementan los estándares de MDE definidos en OMG (Object Management Group). Incluido en EMF se utiliza Ecore, un metamodelo de EMF que da soporte al modelado que consiste en un lenguaje de definición de modelos [9].

Ampliando el enfoque de EMF hacia un desarrollo gráfico, se utiliza GMF (Graphical Modeling Framework). Es un *framework* de modelado gráfico para eclipse que permite desarrollar editores gráficos basados en EMF. La base de esta herramienta es GMF Runtime, un *framework* basado en Java que da la posibilidad de ejecutar editores gráficos para modelos EMF.

Existen principalmente dos proyectos de Eclipse para facilitar la creación y edición de editores basados en GMF, uno es GMF Tooling, el cual tiene un enfoque general y ha caído en desuso, y el otro es Eclipse Sirius, que es el más utilizado para crear editores basados en GMF debido a su enfoque interpretativo. Sirius es un proyecto de Eclipse que permite crear fácilmente un entorno de modelado gráfico utilizando las tecnologías de modelado de Eclipse, incluidas EMF y GMF.

Un proyecto de modelaje desarrollado en Sirius está compuesto por una serie de editores de Eclipse (diagramas, tablas, etc.) lo que permite al usuario crear editar y visualizar modelos EMF.

Sirius es una tecnología que permite ser extendida para necesidades específicas, agregando nuevos tipos de representaciones, nuevos lenguajes de consulta o dando la posibilidad de ejecutar código Java para interactuar con Eclipse o con cualquier otro sistema [10].

3 Estado del arte

En esta sección se explica el estado actual de PCN y las tecnologías relacionadas a dicha notación con el fin de dar un contexto general acerca de la tecnología involucrada en el proyecto. Se analizarán algunas de las herramientas más comunes de BPMN, el estado actual de las herramientas destinadas a PCN y otras funcionalidades desarrolladas que dan soporte al modelado de procesos de negocio con PCN

3.1 Herramientas para BPMN

Las notaciones disponibles a la hora de modelar diagramas de procesos de negocio son escasas. La notación gráfica más utilizada para este tipo de diagramas es BPMN, la cual puede ser modelada con numerosas herramientas. Algunas de estas herramientas se exponen a continuación:

- **Bonita BPM.** Es una solución que permite el diseño de procesos y contiene un motor potente para la ejecución de los mismos. Además, presenta una interfaz intuitiva y fácil de utilizar, aportando utilidades innovadoras para modelizar, desarrollar, ejecutar y llevar el control de los procesos de negocio [11].
- **Heflo.** Es un editor de diagramas BPMN online que da la posibilidad de descargar el diagrama en formato de imagen además de importar y exportar diagramas BPMN. Cuenta con un sencillo sistema de control de versiones [12].
- **Lucidchart.** Es un software disponible en versión de aplicación o en línea que da la posibilidad de modelar diagramas BPMN, pero también soporta otras notaciones gráficas como UML, ERD, o diagramas de secuencia.

  Su punto fuerte es la posibilidad de enviar o compartir los diagramas de forma sencilla, ya que cuenta con un sistema de trabajo en grupo en el que los usuarios pueden tener acceso de edición, de comentarios o de lectura. Además, cuenta con plantillas iniciales de cada tipo de diagrama, lo que proporciona una mayor accesibilidad inicial. [13].
- **Intalio BPM.** Mientras que la mayoría de softwares que soportan BPMN en un solo tipo de usuario (usuarios técnicos o usuarios del mundo empresarial, inclinado hacia modelar flujos de trabajo) el software de Intalio es apto para ambos enfoques.

  Está basado en un conjunto de *frameworks* y arquitecturas conocidas en la industria y tiene una comunidad de usuarios que aportan soporte, mejoras, y detección y corrección de bugs [14].
- **Genmymodel.** Comenzó como una herramienta de UML exclusivamente, pero tras una serie de actualizaciones incluyó en su funcionalidad el modelado de procesos de negocio con BPMN.

  Este software ofrece un repositorio centralizado para el modelado colaborativo de varios usuarios simultáneos y es una herramienta de modelado pura, por lo que cuenta con algunas características únicas como el soporte para exportar modelos en formato XMI y la generación de código a varios lenguajes [15].
- **Draw.io.** Esta herramienta ofrece la posibilidad de crear diagramas de distintas notaciones (BPMN entre ellos) de una forma muy rápida e intuitiva.

Es una herramienta online con la cual es posible comenzar a crear diagramas sin ni siquiera darse de alta, además de ser fácilmente compatible con Google Drive, Dropbox o OneDrive para guardar los modelos.

Al poder diseñar diagramas tan libremente no tiene ningún control de validación, es decir, no entiende la lógica de los diseños, sino que simplemente coloca los elementos donde se solicite, por lo que es una herramienta peligrosa si no se posee suficiente experiencia [15].

- **BPMN.io.** Es la mejor herramienta específica para BPMN hoy en día, utiliza una librería Javascript de código abierto llamada BPMN-js para crear e incrustar diagramas en el navegador.

  Tiene una interfaz intuitiva que permite modelar rápidamente y sus diagramas pueden guardarse tanto en formato XML como en imágenes. Mediante una herramienta creada por la misma empresa que BPMN.io llamada Cawemo, se añade un mecanismo de colaboración entre usuarios [15].

Como se ha comentado anteriormente, al ser la notación más utilizada, BPMN cuenta con numerosas opciones para modelar diagramas, sin embargo, en la actualidad existen muy pocas opciones para la notación PCN (Process Chain Network).

3.2 Estado de PCN

La notación PCN es una técnica de modelado de procesos similar a BPMN la cual está compuesta por entidades y procesos. Las entidades son los agentes que participan en un proceso, y sus decisiones afectan al desarrollo del mismo. Los procesos son el elemento principal del diagrama, establecen un flujo de pasos o acciones mediante el uso de relaciones entre ellos, formando cadenas de procesos. Estos pasos son llevados a cabo por las entidades.

Actualmente no existen herramientas comerciales o de software abierto que soporten el modelado de PCN. Sin embargo, en la Universidad Rey Juan Carlos de Madrid un grupo de investigación desarrolló INNoVaServ, un software de modelado basado en EMF/GMF enfocado en el diseño de modelos de negocio y operaciones de servicio [16]. Este software solo soportaba las notaciones Canvas, e3Value y Service Blueprint, pero tras una actualización añadieron el modelado de PCN, convirtiéndose en la primera solución de software que admite esta notación [16].

La extensión del prototipo para el soporte de PCN se ha desarrollado mediante un plugin EMF/GMF siguiendo los siguientes pasos fundamentales:

- Especificar el modelo correspondiente cumpliendo con el metamodelo Ecore de Eclipse.
- Crear el diagramador de GMF.
- Desarrollo de modificaciones de código implementando funcionalidades complejas.

Para el desarrollo de esta extensión solo se utiliza la funcionalidad proporcionada por GMF, pese a que en un principio se contempló la idea de utilizar Eclipse Sirius y otras herramientas similares. Esta decisión se debe a la libertad que proporciona GMF para el desarrollador, permitiendo la representación de ciertas figuras gráficas que requieren de un control de desarrollo del que no disponen el resto de herramientas contempladas.

3.3 Otras herramientas

Aparte de las notaciones de modelado, para la gestión y desarrollo de diagramas de procesos de negocio existen otras herramientas o lenguajes involucrados, entre los que destacan los siguientes:

- **BPEL.** Son las siglas de Business Process Execution Language (Lenguaje de Ejecución de procesos de Negocio). Es un lenguaje diseñado por OASIS, organización que se encarga de definir estándares a nivel mundial en el ámbito del comercio electrónico y los servicios web. Está definido en XML y es el encargado de consumir varios servicios en un orden especificado y realizar una función concreta automáticamente.

  Con BPEL es posible crear un solo servicio que reciba todos los parámetros necesarios para realizar las distintas operaciones que lleve a cabo el flujo de procesos. Es una herramienta especializada para la creación de servicios compuestos, aunque si el servicio es grande, complejo y no se realiza el diseño gráfico correctamente, el mantenimiento puede convertirse en algo complicado [17].

  BPEL puede utilizarse internamente en una empresa o entre empresas distintas y ha sido diseñado específicamente como un lenguaje para la definición de procesos de negocio. Internamente el papel de BPEL es estandarizar la integración de aplicaciones y ampliar la integración de sistemas ya aislados [17].

  Soporta dos tipos de procesos de negocio:

  - Procesos de negocio ejecutables: especifican los detalles exactos de los procesos de negocio. En la mayor parte de casos BPEL se utiliza para procesos de este tipo [17].
  - Procesos abstractos de negocio: Solo especifican el intercambio de mensajes entre las partes implicadas en el proceso, sin incluir detalles del flujo. No son ejecutables ni se suelen utilizar [17].

En la Figura 1 podemos ver un ejemplo sencillo de un proceso BPEL que llama a las operaciones diHola y diAdios en función del valor de un booleano devolviendo el saludo correspondiente.

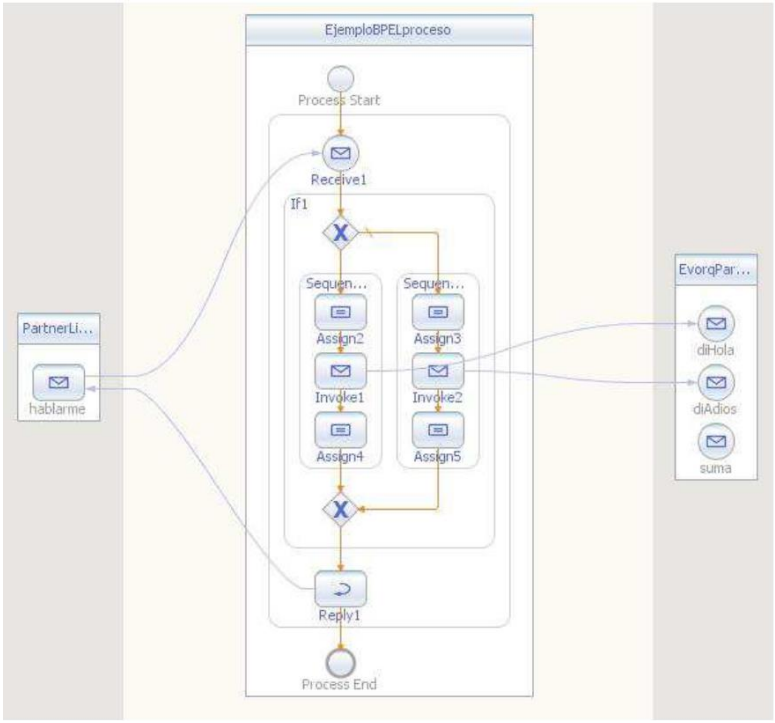

Figura 1 Ejemplo de proceso BPEL obtenido de [22]

- **XPDL.** XPDL es el lenguaje de definición de proceso de XML (de sus siglas XML Process Definition Language, un formato estandarizado para intercambiar definiciones de modelado de procesos. Este lenguaje define un esquema XML para especificar la parte declarativa del proceso. Intenta ofrecer una manera estándar de representación de procesos de forma que sean importados o exportados por cualquier editor que soporte dicho estándar [18].

  XPDL permite la representación gráfica de los procesos incluyendo coordenadas X e Y para cada nodo correspondiente, representando los procesos mediante la etiqueta <Activity>, especificando dentro el ID y nombre del proceso, las transiciones entre procesos mediante la etiqueta <Transition>, especificando Id y nombre de la transición junto con los procesos fuente y destino (From, To), y las coordenadas mediante la etiqueta <Coordinates> incluyendo las coordenadas X e Y [18]. Por lo tanto, un ejemplo de definición de un proceso de ejemplo con XPDL sería:

```
<Activities>
  <Activity Id="123456" Name="Proceso ejemplo">
</Activities>
```

  Mientras que la definición de una transición sería:

```
<Transitions>
  <Transition Id="123123" Name="Transición ejemplo"
              From="123456" To="987654">
    ...
</Transitions>
```

La definición de una coordenada sería:

```
<Coordinates XCoordinate="120.0" YCoordinate="90.0"/>
```

Estos son los componentes fundamentales de XPDL, aunque dispone de más etiquetas para representar elementos de otros tipos.

- **YAWL. YAWL (Yet Another Workflow Language)** es un lenguaje basado en patrones de flujos de trabajo, gratuito y de código abierto. Es soportado por un sistema de software que incluye un motor de ejecución, un editor gráfico y un gestor de tareas. El objetivo del YAWL es definir un lenguaje que soporte todos los patrones de flujos de trabajo, extendiendo la formalización mediante tres constructores principales, nombrado or-join, grupos de cancelación y actividades de varias instancias [19].

  Como podemos comprobar en la Figura 2, YAWL es similar a XML. Dicha figura muestra la implementación en YAWL del comienzo de una aplicación, en la que se define el metadata de un programa entre las etiquetas `<metadata>` y `</metadata>` y una variable de ejemplo entre las etiquetas `<localVariable>` y `</localVariable>`. Además, se puede ver también la especificación de YAWL en la parte inicial y algunas otras opciones de configuración.

```
<?xml version="1.0" encoding="UTF-8"?>
<specificationSet xmlns="http://www.yawlfoundation.org/yawlschema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="
2.1" xsi:schemaLocation="http://www.yawlfoundation.org/yawlschema http://www.yawlfoundation.org/yawlschema/YAWL_Schema2.1.xsd">
  <specification uri="CreditAppProcess2.0">
    <metaData>
      <title>Credit application process</title>
      <creator>tut</creator>
      <description>Credit application process</description>
      <version>0.1</version>
      <persistent>false</persistent>
      <identifier>UID_a725098a-7fca-4ead-bb3d-485b721aee29</identifier>
    </metaData>
    <schema xmlns="http://www.w3.org/2001/XMLSchema" />
    <decomposition id="CreditApplication" isRootNet="true" xsi:type="NetFactsType">
      <localVariable>
        <index>0</index>
        <name>completeApp</name>
        <type>boolean</type>
        <namespace>http://www.w3.org/2001/XMLSchema</namespace>
        <initialValue>false</initialValue>
      </localVariable>
```

Figura 2 Ejemplo de código en el lenguaje YAWL obtenido de [21]

4 Diseño

El diseño del proyecto está limitado por la notación gráfica PCN, la cual se explica a continuación.

PCN está compuesto básicamente por dos abstracciones, procesos y entidades, que interactúan entre sí formando un diagrama del cual se puede obtener la información necesaria del flujo de negocio.

Todas las figuras mostradas en este capítulo se han creado utilizando la herramienta desarrollada, excepto la Figura 10.

4.1 Procesos

Los procesos son pasos que son llevados a cabo en secuencia y siguiendo un determinado orden, formando un flujo o cadena de procesos que representa el transcurso del proceso de negocio correspondiente.

Los procesos son llevados a cabo por las entidades y actúan sobre los recursos disponibles en el negocio. El flujo de procesos se realiza mediante relaciones entre dichos procesos y se representa mediante flechas. Dichas relaciones aparte de seguir el flujo de información puede representar la utilización de recursos materiales.

Existen dos tipos de relaciones definidas entre procesos, las relaciones estándar, que representan una relación directa y sin restricciones entre dos procesos y se definen mediante una flecha de línea continua tal y como muestra la Figura 3, y las relaciones con retraso, las cuales representan la posibilidad de que entre los procesos que involucra exista un retardo de tiempo. Las relaciones con retardo se definen mediante una flecha con línea discontinua como muestra la Figura 4.

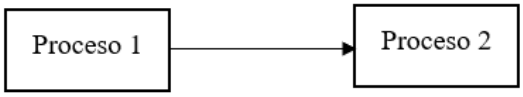

Figura 3 Relación Estándar

Figura 4 Relación con retardo

Existen diferentes tipos de procesos dependiendo de cada situación, por lo tanto, es necesario reflejarlo en la notación. Para ello, la notación PCN cuenta con varias representaciones de procesos que definen características propias e independientes.

Los diferentes tipos de procesos se explican a continuación.

4.1.1 PCNStandardStep

Es el proceso básico de PCN, define una acción de negocio estándar y puede contener etiquetas que aporten información extra, que se explicarán más adelante (Etiquetas4.3).

Se representan mediante un rectángulo en cuyo interior se define la acción que realiza este determinado proceso. En las Figura 3 y Figura 4 podemos observar la representación de este tipo de procesos.

4.1.2 PCNWaitStep

Es un proceso de espera, es decir, indica una pausa en el flujo de procesos, cuya longitud puede ser determinada mediante una etiqueta que índice el tiempo de espera tiempo o determinada por la finalización de una acción especificada.

La especificación de una acción que determine hasta cuando espera el proceso se incluye definido dentro del propio proceso de PCNWaitStep, aunque también puede representar una espera indefinida, desbloqueada cuando ocurre otro proceso relacionado.

La representación de los procesos PCNWaitStep se realiza con un triángulo invertido, como se muestra en la Figura 5.

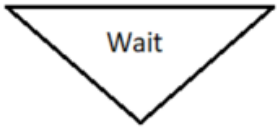

Figura 5 Proceso PCNWaitStep

4.1.3 PCNDoAndWaitStep

Los procesos PCNDoAndWaitStep determinan que la entidad realiza la acción del proceso y después queda en espera, pasando desde ese momento a obtener la funcionalidad de un PCNWaitStep.

Se representa mediante un rectángulo con dos de sus lados extremos redondeados, como muestra la Figura 6.

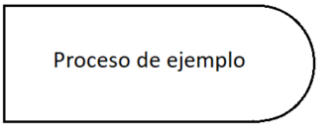

Figura 6 Proceso PCNDoAndWaitStep

4.1.4 PCNDecisionStep

PCNDecisionStep representa un proceso que implica una toma de decisiones por parte de la entidad, por lo que puede modificar el flujo de procesos.

En todos los casos tiene dos relaciones a procesos destino, uno al que se dirigirá en caso de que la decisión se resuelva negativa y otro en caso de que sea positiva.

Este proceso se representa mediante un rombo, tal y como se muestra a continuación en la Figura 7. En la figura se pueden apreciar también las relaciones de decisión involucradas en el proceso.

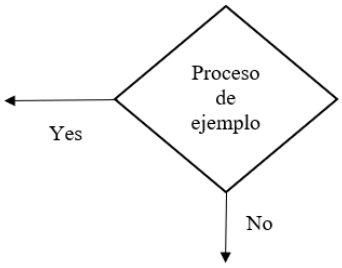

Figura 7 Proceso PCNDecisionStep

4.1.5 PCNInnovationStep

Este proceso representa un paso que supone una innovación en el modelo de negocio, pero sus características son las mismas que las de un PCNStandardStep.

Se representa de la misma forma que un PCNStandardStep, pero con una doble línea como borde en lugar de una simple, tal y como se muestra en la Figura 8.

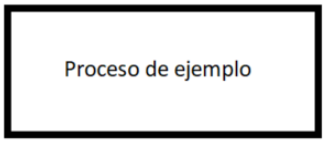

Figura 8 Proceso PCNInnovationStep

4.1.6 PCNOutsourcedStep

PCNOutsourcedStep representa un proceso desarrollado por una entidad externa a la empresa, es decir una subcontratada, aunque a efectos prácticos tiene la misma funcionalidad que un PCNStandardStep.

Se representa mediante un rectángulo con todos sus bordes redondeados, como se muestra en la Figura 9.

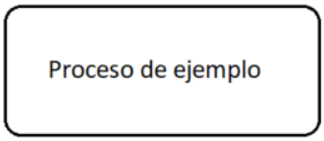

Figura 9 Proceso PCNOutsourcedStep

4.2 Entidades

Las entidades son los agentes que participan y toman decisiones sobre los pasos involucrados en el flujo de procesos.

Puede haber varias entidades en un flujo de procesos y se representan mediante un triángulo en el que se especifica el nombre de la entidad, aunque la disposición del flujo de procesos se define debajo de dicho triángulo, en el dominio de la entidad. El dominio contiene el

conjunto de procesos o pasos a los que las decisiones de la entidad afectan y aquellos pasos que la entidad puede iniciar.

Dentro de cada entidad se diferencian tres sectores distintos que contienen los procesos, representados mediante columnas y distinguidos entre sí mediante la especificación del tipo de sector con etiquetas situadas en la parte superior de la columna, en el borde con el triángulo de la entidad. Las columnas de dichos sectores están separadas mediante línea discontinua y se pueden diferenciar en:

- **Interacción directa.** Se encuentran en los extremos de la entidad que tengan interacción con otra entidad, por lo que puede haber un solo sector de interacción directa que contenga procesos o dos en el caso de que la entidad se relacione con dos entidades. Como su nombre indica, expresa interacciones directas entre las entidades, como, por ejemplo, pedirle la cuenta a un camarero, siendo el camarero y el cliente entidades de un modelo. Se define mediante la etiqueta "Dir.".
- **Proceso independiente.** Se encuentra en el centro de la entidad, y en todos los casos existe únicamente una columna de este tipo, ya que no guarda relación con otras entidades. En esta sección se reflejan los procesos que no necesitan de otras entidades ni de sus recursos, representa una acción interna e independiente de la entidad, como podría ser un cliente desplazándose a un restaurante, siendo el cliente y el restaurante las entidades. Esta sección se reconoce mediante la etiqueta "ind.".
- **Interacción subrogada.** Se encuentra entre la columna de procesos independientes y las de interacción directa. Al igual que en las de interacción directa puede haber una o dos columnas que alberguen procesos de este tipo, dependiendo del número de entidades con las que tenga relación. Los procesos contenidos en esta columna implican alguna interacción con los recursos de otra entidad, como, por ejemplo, volviendo al caso del restaurante, una interacción subrogada sería pagar la cuenta en la bandeja que el camarero ha dejado sobre la mesa.

En la Figura 10 puede apreciarse la definición de los diferentes sectores del diagrama explicados:

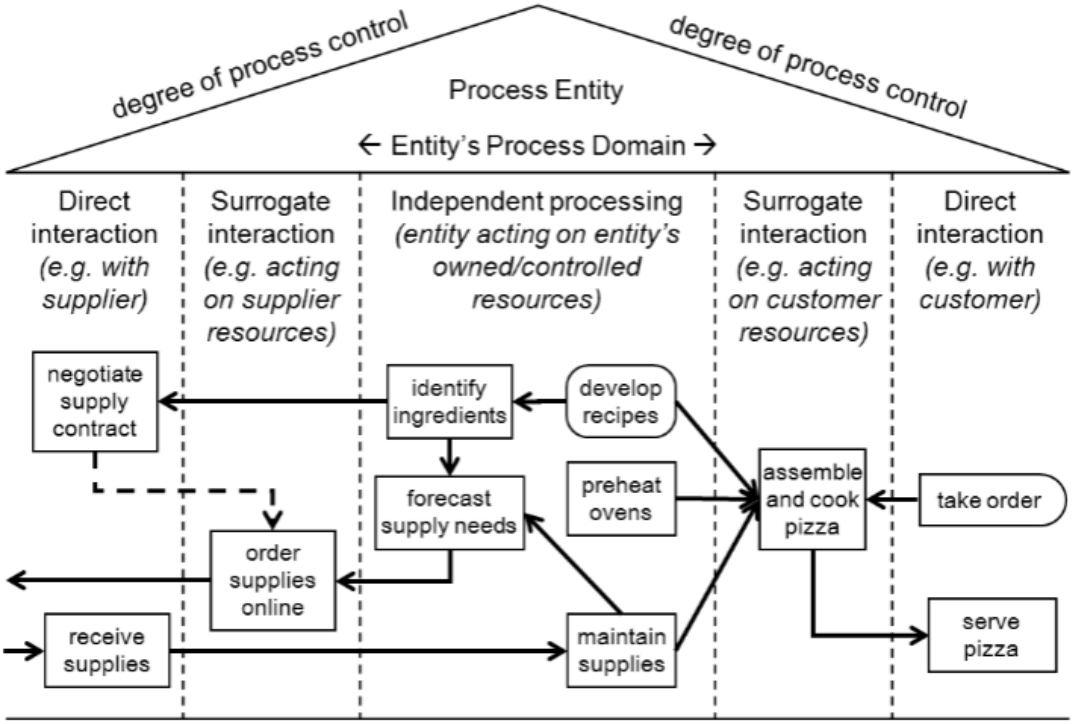

Figura 10 Representación de una entidad de ejemplo. Recuperado de [20]

La Figura 10 representa el ejemplo de una entidad en PCN, incluyendo procesos en sus dominios y relaciones entre dichos procesos. Como sugiere dicha figura, la representación en triángulo de cada entidad no es un diseño al azar, sino que representa el grado de control de procesos de cada entidad. Este factor señala que cuanta mayor es la interacción directa entre personas, menos control existe, y se representa mediante la altura del triángulo en cada sitio.

A mayor altura del triángulo, menos control por parte de la entidad, lo que explica que la entidad tiene prácticamente todo el control sobre los procesos independientes.

4.3 Etiquetas

Habiendo definido las ideas principales de la notación PCN se explicará el uso de las etiquetas de procesos.

Las etiquetas son pequeños iconos o textos cortos que amplían la información del proceso al que estén adjuntas. Se sitúan en el borde de la representación del proceso y pueden ser de varios tipos:

- **Etiquetas monetarias.** Expresan si el proceso al que pertenezcan supone un coste o un beneficio económico para el proveedor del servicio. En caso de beneficio se representa como "$" y en caso de coste como "-$".

  En la notación de PCN toman el nombre de PCNMonetaryBenefit (beneficio) y PCNMonetaryCost (coste).

- **Etiquetas no monetarias.** Señalan si el proceso proporciona un coste o beneficio no monetario al cliente. Un coste no monetario podría ser la denegación de una petición o una demora del servicio y un beneficio podría ser la entrega correcta de un

producto. Se representan mediante el icono de una cara sonriente en caso de beneficio (☺) o el icono de una cara triste en caso de coste (☹).

En la notación de PCN toman el nombre de PCNNonMonetaryBenefit (beneficio) y PCNNonMonetaryCost (coste).

- **Etiquetas textuales.** Proporcionan información adicional mediante texto, como, por ejemplo, el precio que supondrá un determinado proceso para el cliente.

  Los procesos de tipo PCNWaitStep pueden tener una etiqueta textual independiente que indica el tiempo de espera. En la notación de PCN toman el nombre de PCNTextualTag.

- **Etiquetas de probabilidad.** Solo se encuentran en los procesos de decisión (PCNDecisionStep), y definen las probabilidades de obtener una decisión afirmativa o negativa en dicho proceso mediante porcentajes.

  Se representan mediante "Y=x%" para la probabilidad de decisión positiva y "N=x%" para la probabilidad de decisión negativa, siendo x el porcentaje correspondiente en cada caso. En la notación de PCN toman el nombre de PCNProbabilitiesTag.

- **Etiqueta de sincronización.** Expresa la posibilidad de una demora en el proceso al que corresponde y se representa mediante el icono de un reloj (⏱). En la notación de PCN toman el nombre de PCNSyncTag.

Conociendo todos los elementos y características de PCN se pasará a definir la etapa de desarrollo, en la que se crean todos los elementos explicados en el diseño para alcanzar una funcionalidad correcta ante cualquier modelo.

5 Desarrollo

En la ventana en tiempo de ejecución se ha creado un proyecto de modelado con un modelo de ejemplo sobre el que empezar a diseñar. Este modelo debe tener configurado el Ecore de PCN y su temática es un modelo de negocio de car2go en el que interactúan el cliente, car2go como servicio y un punto de validación que comprueba la licencia de conducir.

Sirius permite añadir al diagrama cuatro tipos de elemento: nodos, contenedores, relaciones y elementos decorativos. A grandes rasgos los procesos están representados por nodos mientras que las entidades y sus dominios se representan con contenedores.

En el desarrollo inicial se han implementado únicamente los nodos sin tener en cuenta contenedores para comprobar su correcto funcionamiento y así añadir la implementación de los contenedores sobre seguro más adelante.

A continuación, se explica con detalle cómo se ha implementado cada elemento explicado en el diseño y como se relacionan estos elementos con el modelo de PCN.

5.1 Nodos

Los nodos se utilizarán fundamentalmente para representar cada proceso del flujo de procesos. Todos ellos cuentan con un diseño específico que corresponde con su notación en PCN, implementado mediante la opción de añadir estilo al nodo de Sirius. Este estilo podrá ser una figura geométrica de las que propone Sirius (rectángulo, rombo, círculo o elipse), una nota, un gráfico, o una imagen especificada en un directorio de iconos contenido dentro del directorio de diseño.

Todos los nodos se han configurado mediante los mismos parámetros, entre los que se incluye una clase de dominio, que será el parámetro que relacione cada nodo con su representación de PCN y una id, para identificar el nodo por su nombre, que será determinada en función de la clase de dominio por razones de claridad y simpleza. Dichos parámetros pueden configurarse en la pestaña “General” de las propiedades.

Además de los parámetros, cada nodo contiene las posibles etiquetas que pueda utilizar, añadidas como nodos adjuntos al borde del nodo principal. El estilo de dichas etiquetas se ha definido con una imagen especificada del directorio de iconos, excepto la etiqueta textual, que se ha definido simplemente como una nota.

Al añadir un nuevo estilo que determine el diseño del nodo, Sirius proporciona la posibilidad de modificar propiedades de dicho estilo. Para todos los nodos principales se define el parámetro del estilo “Label Expression” con la codificación “feature:action”, en la pestaña Label de propiedades del estilo. Esta propiedad indica el texto que contendrá dicho nodo, que en caso de PCN será la acción que realice cada entidad en el proceso que representa el nodo.

Además del parámetro “Label Expression”, para todos los nodos se ha definido el color blanco de fondo y el negro para el borde del nodo, ya que así lo establece PCN. Dichos parámetros relacionados con el color (Label Color para la etiqueta y Border Color para el borde) se encuentran en la pestaña “Color” de las propiedades del estilo.

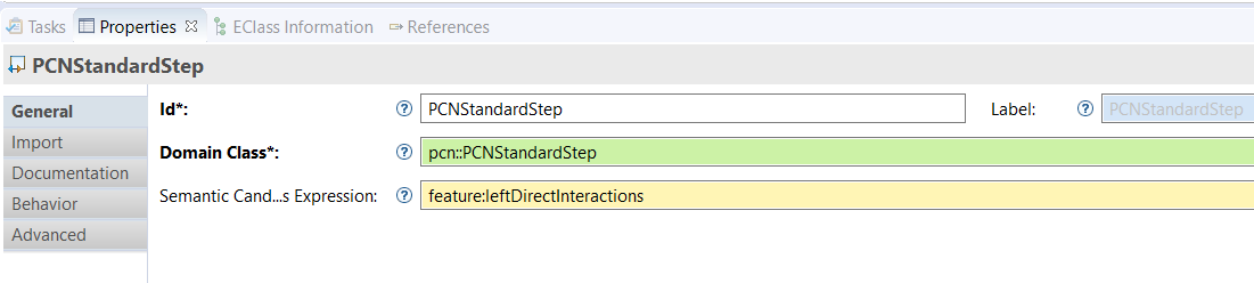

Figura 11 Parámetros de implementación de PCNStandardStep

En la Figura 11 se puede observar un ejemplo de cómo se distribuyen los parámetros en Sirius (del uno de los nodos en este caso) sobre los que se hablará a lo largo de toda esta sección, aunque estos variarán dependiendo del elemento.

5.1.1 PCNStandardStep

En los parámetros de este nodo se ha definido como clase dominio "pcn::PCNDoAndWaitStep", relacionándolo con su correspondencia en PCN. El parámetro id se ha definido mediante el nombre del nodo en PCN, al igual que en el resto de nodos del proyecto, por lo que, en este caso en particular, el id del nodo es PCNStandardStep.

Este nodo se representa como un rectángulo, por lo que su diseño está definido por el rectángulo que proporciona Sirius al crear un nuevo estilo.

Este nodo puede contener como nodos adjuntos al borde las etiquetas PCNTextualTag, PCNMonetaryBenefit, PCNMonetaryCost, PCNSyncTag, PCNNonMonetaryBenefit y PCNNonMonetaryCost.

La implementación final del nodo se mostraría como se puede observar en la Figura 12:

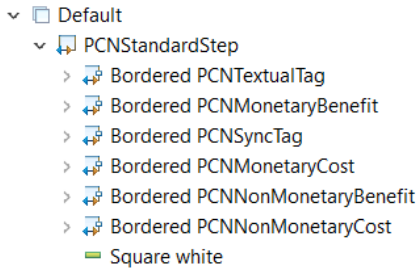

Figura 12 Implementación del nodo PCNStandardStep

5.1.2 PCNDoAndWaitStep

Este nodo se representa como un rectángulo con uno de sus lados redondeados, figura de la cual no disponen los estilos propuestos por Sirius, por lo que su diseño se ha definido mediante una imagen de la forma deseada, almacenada en el directorio de iconos.

Respecto a los parámetros del nodo, el dominio de clase es "pcn:PCNDoAndWaitStep" y su id es "PCNDoAndWaitStep".

En la parte de diseño, se incluye la imagen como nuevo estilo del nodo, estableciendo la ruta de la imagen en el parámetro "Image Path", en la pestaña General. En este caso la ruta de la imagen será "pcn.design/icons/DoAndWait.png", siendo pcn.design el proyecto de diseño, icons el directorio de imágenes y DoAndWait.png la imagen de la figura en formato PNG.

Las etiquetas que puede contener este nodo son PCNTextualTag, PCNMonetaryBenefit, PCNMonetaryCost, PCNSyncTag, PCNNonMonetaryBenefit y PCNNonMonetaryCost, que coinciden con PCNStandardStep, ya que la funcionalidad de PCNDoAndWaitStep es realmente la de un proceso normal que al terminar realiza un wait.

5.1.3 PCNDecisionStep

Este nodo se representa con un rombo, por lo que es posible utilizar el estilo de diamante propuesto por Sirius.

El dominio de clase es "pcn:PCNDecisionStep", por lo que la id que se le ha otorgado es PCNDecisionStep.

Las etiquetas que puede contener son PCNTextualTag, PCNMonetaryBenefit, PCNMonetaryCost, PCNSyncTag, PCNNonMonetaryBenefit y PCNNonMonetaryCost, que coincide también con PCNStandardStep.

5.1.4 PCNWaitStep

Este nodo se representa con un triángulo invertido, figura de la que no dispone Sirius en los estilos que propone, por lo que se ha definido mediante una imagen incluida en el directorio de iconos. El nombre de dicha imagen es InverseTriangleIcon, por lo que la ruta determinada en las propiedades del estilo será "pcn.design/icons/InverseTriangleIcon.png".

Respecto a los parámetros generales del nodo, el dominio de clase es "pcn::PCNWaitStep" y por lo tanto su id es "PCNWaitStep".

Este nodo solo puede contener la etiqueta PCNDelayTimeTag, que determina el tiempo de espera correspondiente.

5.1.5 PCNInnovationStep

Este nodo se representa de la misma forma que un PCNStandardStep, aunque el borde del nodo es doble, es decir, se representa mediante un rectángulo con el borde más ancho, por lo que es posible utilizar el rectángulo que proporciona Sirius al añadir un nuevo estilo y cambiar el grosor del borde en la pestaña Border de las propiedades del estilo. Para cambiar dicho grosor se ha establecido el parámetro "Border Size Computation Expression" con el valor 4.

Respecto a los parámetros del nodo, el dominio de clase es "pcn::PCNInnovationStep" por lo que su id será "PCNInnovationStep".

Las etiquetas que puede contener son PCNTextualTag, PCNMonetaryBenefit, PCNMonetaryCost, PCNSyncTag, PCNNonMonetaryBenefit y PCNNonMonetaryCost, que coincide también con PCNStandardStep ya que representa lo mismo que un proceso estándar con el añadido de ser un proceso de innovación en el modelo de negocio.

5.1.6 PCNOutSourcedStep

Este nodo se representa con un rectángulo con las puntas redondeadas, figura de la que no dispone Sirius en los estilos que propone, por lo que se ha definido mediante una imagen incluida en el directorio de iconos. El nombre de dicha imagen es OutSourcedStep.png, por lo que la ruta determinada en las propiedades del estilo será "pcn.design/icons/ OutSourcedStep.png".

Puede contener las etiquetas PCNTextualTag, PCNMonetaryBenefit, PCNMonetaryCost, PCNSyncTag, PCNNonMonetaryBenefit y PCNNonMonetaryCost, que coinciden con la mayoría de los procesos ya que representa lo mismo que un proceso estándar que se desarrolla en una empresa externa o subcontratada.

5.2 Relaciones

Las relaciones entre procesos se representan mediante flechas y pueden ser principalmente de dos tipos, como se ha explicado en el apartado de diseño.

Existe una particularidad respecto a las relaciones cuyo origen es un proceso PCNDecisionStep, ya que dependiendo de si la decisión es afirmativa o negativa la relación sería PCNPositiveDecisionRelation o PCNNegativeDecisionRelation respectivamente. Estas relaciones son similares a PCNStandardRelation, sin embargo, se caracterizan por contener la etiqueta "Yes" o "No" dependiendo de si la decisión ha sido afirmativa o negativa.

Todas las relaciones se han creado añadiendo un elemento nuevo al diagrama de tipo "Relation Based Edge", en el cual es necesario definir únicamente los parámetros de la pestaña General. Estos parámetros son el id, los elementos que podrán ser origen de la relación, los elementos que podrán ser destino de la relación y la conexión entre la relación en el diagrama con su correspondencia en PCN (parámetro Target Finder Expression de la pestaña General.

Al crear una nueva relación, se incluye automáticamente su estilo, es decir, no es necesario agregar un estilo nuevo como en el caso de los nodos. En este estilo se ha determinado la forma que tendrá cada extremo de la flecha, el estilo de línea y el estilo del recorrido que realiza la línea de un punto a otro. La forma que tendrá cada extremo de la flecha se determina en la pestaña Decorators, en la que se define la forma de destino y la de origen.

En el caso de PCN, la forma de destino deberá ser la requerida en el origen (parámetro Source Arrow) y la forma de origen será la requerida en el destino (parámetro Target Arrow), ya que en Sirius está definido de esa forma.

El estilo de línea se configura en la pestaña General, modificando el parámetro "Line Style", mientras que el estilo de recorrido que sigue la flecha se define en "Routing Style". Las posibilidades del estilo de recorrido se componen por "Straight", que muestra la línea recta entre un nodo y el relacionado, "Manhattan", que se compone únicamente de líneas horizontales y verticales estableciendo giros en ángulo recto y "Tree", que define el diagrama en forma de árbol, estableciendo la entrada de la flecha al destino siempre por la parte superior del nodo. Debido a su claridad y su aspecto se ha seleccionado la opción Manhattan para todas las relaciones implementadas.

A continuación, se detallan las características individuales de cada tipo de relación.

5.2.1 PCNStandardRelation

Su id es PCNStandardRelation, y sus nodos de destino son todos los procesos del proyecto, ya que hacia cualquiera de ellos puede haber una relación estándar. Sus nodos de origen son todos los procesos excepto PCNDecisionStep.

Como estilo de línea se ha seleccionado "solid", que se representa como una línea continua.

Para relacionar el elemento con su representación de PCN se ha establecido en el parámetro "Target Finder Expression" la codificación "feature:PCNStandardDependencySource", que representa la dependencia de origen estándar.

5.2.2 PCNDelayedRelation

Su id es PCNDelayedRelation y sus nodos posibles de origen y destino son los mismos que para PCNStandardRelation, es decir, todos los procesos pueden recibir una PCNDelayedRelation y todos excepto PCNDecisionStep pueden ser origen de este tipo de relación.

El estilo de línea en este caso será “dot”, por lo que se mostrará una línea de puntos discontinua.

El parámetro de relación con la notación PCN se codifica como “feature:PCNDelayedDependencySource”, por lo que representará a las dependencias de relación con retardo del modelo.

5.2.3 PCNNegativeDecisionRelation y PCNPositiveDecisionRelation

Las id de estas relaciones son PCNNegativeDecisionRelation y PCNPositiveDecisionRelation, y el parámetro de relación con la notación PCN se codifica como “feature:PCNNegativeDecisionSource” y “feature:PCNPositiveDecisionSource” respectivamente.

El nodo de destino de estas relaciones puede ser cualquiera de los desarrollados en el proyecto, al igual que en las otras relaciones. Sin embargo, el único nodo de origen desde el que parten estas relaciones es PCNDecisionStep.

El estilo de línea es de tipo “solid” al igual que las relaciones PCNStandardStep, es decir, la flecha será de línea continua.

Para establecer las etiquetas “Yes” y “No” en la línea según la decisión se ha modificado el parámetro “Label Expression”, que se encuentra en la pestaña Label dentro del estilo de línea.

En la Figura 13 se puede observar la implementación final de las relaciones disponibles para la notación PCN:

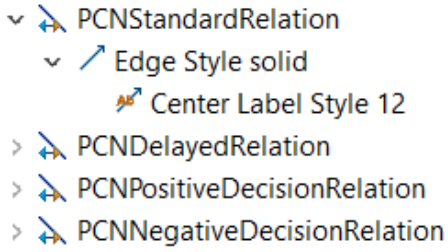

Figura 13 Implementación de relaciones

5.3 Contenedores

Los contenedores se comportan como nodos que pueden agrupar otros nodos o contenedores en su interior. Sus parámetros generales son similares a los de los nodos, componiéndose por un id, la clase de dominio que represente, una expresión que permite filtrar los elementos a los que da representación el contenedor en un parámetro llamado “Semantic Candidate Expresion” y un parámetro para determinar la organización de los nodos que existan dentro del contenedor, pudiendo ser de forma libre, en forma de lista, alineados horizontalmente y alineados verticalmente.

Los estilos que se pueden crear para los contenedores pueden ser una imagen del directorio de iconos, un paralelogramo, o un rectángulo al que se puede añadir un degradado de colores vertical u horizontal.

En este proyecto se han utilizado contenedores para la implementación de las entidades. Se ha creado un contenedor principal con id PCNProcessEntity para representar cada entidad. La clase dominio de dicho contenedor es “pcn::PCNProcessEntity” y la organización que se ha elegido para representar los elementos internos al contenedor es “Horizontal Stack”, es decir, se representarán en una fila horizontal. Dichos elementos serán los distintos tipos de dominio de las entidades (interacciones directas y subrogadas y procesos independientes).

Para conseguir el diseño que propone PCN de las entidades ha sido necesario utilizar una imagen previamente diseñada, compuesta por el característico triángulo de entidad y un espacio vacío debajo en el que se encuentran los dominios de la entidad. Dicha imagen se encuentra en el directorio de iconos determinada como EntityFormIcon por lo que su ruta será /pcn.design/icons/EntityFormIcon.png.

Para establecer la etiqueta interior de la entidad hay que reflejar el orden de entidad y el nombre, por lo que el parámetro “Label Expression” de la pestaña Label se ha designado como “aql:' ' + self.order + '\n ' + self.name”. Esta expresión utiliza el lenguaje aql y establece un espacio seguido del orden, un salto de línea y el nombre de la entidad, mostrándose como refleja la Figura 14:

Figura 14 Etiqueta de entidad

5.3.1 Dominios de entidad

Los dominios de las entidades se definen cada uno como un contenedor distinto, dentro del contenedor principal de entidad. Dichos contenedores tendrán las id PCNLeftDirectInteractions, para las interacciones directas a la izquierda, PCNLeftSurrogateInteractions, para las interacciones subrogadas a la izquierda, PCNCenterIndependentInteractionsActivities para los procesos independientes de la entidad, PCNRightSurrogateInteractions para las interacciones subrogadas a la derecha y PCNRightDirectInteractions para las interacciones directas a la derecha.

La clase de dominio de estos contenedores será “pcn::PCNProcessEntity”, ya que están contenidos en la propia entidad y no representan ningún elemento distinto de PCN.

El parámetro “Semantic Candidates Expression” de los contenedores de dominio debe ser “var:self”. De esta forma filtrará mostrando solo los elementos de la entidad correspondiente, ya que si este parámetro no estuviera definido aparecerían todas las entidades representadas en el contenedor de dominio.

El parámetro que define la distribución de los elementos dentro del contenedor (Children Representation) será en todos los dominios “Free Form”, ya que al representarlo como lista

no es posible definir un estilo para cada tipo de nodo, y las opciones Horizontal Stack y Vertical Stack no sirven en este caso.

Dentro de cada contenedor de dominio se define el estilo, que se representa con un rectángulo blanco utilizando el gradiente de Sirius. Además, se establece una etiqueta para cada contenedor, con lo que se define el tipo de dominio. Estas etiquetas se establecen en el parámetro "Label Expression" de la pestaña Label, y se representan como "Dir." para las interacciones directas, "Sur." para las interacciones subrogadas y "ind." para los procesos independientes de la entidad, obteniendo una representación como la que se muestra en la Figura 15:

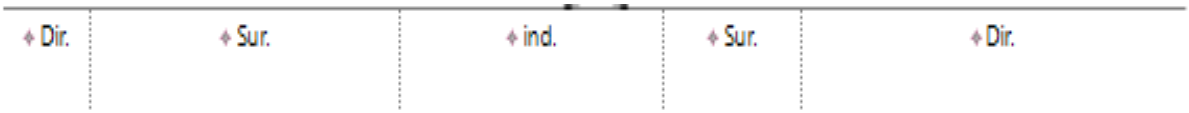

Figura 15 Etiquetas de dominios

Dentro de cada contenedor de dominio se encuentra el diseño de cada uno de los tipos de nodo, de tal forma que sea posible representar todos los posibles procesos en todos los dominios disponibles.

Para diferenciar el dominio en el que se encuentra el proceso, se establece el parámetro "Semantic Candidate Expression" en cada tipo de nodo, por lo que dentro de cada nodo de los contenedores de dominio habrá una codificación que filtre los nodos obteniendo solo los de dicho dominio. Este parámetro se codificará como "feature:leftDirectInteractions" en el contenedor de interacciones directas de la izquierda, "feature:leftSurrogateInteractions" en el contenedor de interacciones subrogadas de la izquierda, "feature:centerIndependentInteractionsActivities" en el contenedor de procesos independientes, "feature:rightSurrogateInteractions" en el contenedor de interacciones subrogadas de la derecha y feature:rightDirectInteractions en el contenedor de interacciones directas de la derecha.

5.4 Etiquetas

Las etiquetas aportan información adicional sobre el proceso, por lo que son un elemento importante de la notación PCN.

En el proyecto las etiquetas se definen como "Bordered Nodes" de los procesos, es decir, nodos adjuntados al borde de los nodos de proceso.

Para implementar las etiquetas se configuran los tres parámetros habituales de los nodos, teniendo un id, una clase de dominio y el filtro que relacione específicamente cada etiqueta con su correspondiente en PCN (Semantic Candidate Expression).

Respecto al estilo, el parámetro "Label Position" se ha definido en todas las etiquetas como "border", estableciendo la posición de la etiqueta en el borde del nodo. Las imágenes utilizadas para el estilo de las etiquetas se han creado con un editor de imagen.

A continuación, se detalla la implementación de cada etiqueta en particular.

5.4.1 PCNTextualTag

Su clase de dominio es "pcn::PCNTextualTag", por lo que su id se ha definido como PCNTextualTag. A la hora de relacionar la etiqueta con el elemento correspondiente de PCN se define el parámetro de filtro como "feature:textualTags".

El estilo de esta etiqueta será simplemente una nota cuya “Label Expression” sea “feature:tag”, mostrando el tag correspondiente como texto.

5.4.2 PCNMonetaryBenefit

Su clase de dominio es “pcn::PCNMonetaryBenefit”, por lo que su id será PCNMonetaryBenefit. El parámetro de expresión filtro se ha definido como “feature:monetaryBenefits”, de forma que tenga relación con su representación según PCN.

El estilo de esta etiqueta se define con una imagen cuyo nombre es “icon_MonetaryBenefit”, contenida en el directorio de iconos, por lo que se establece su ruta como “/pcn.design/icons/icon_MonetaryBenefit.png”. Esta imagen representa el símbolo de un dólar, expresando un beneficio económico para la empresa.

5.4.3 PCNMonetaryCost

Su clase de dominio es “pcn::PCNMonetaryCost” por lo que su id será PCNMonetaryCost y el parámetro de expresión de filtro se define como “feature:monetaryCosts”.

El estilo de esta etiqueta se define también con una imagen cuyo nombre es “icon_MonetaryCost.png”, por lo que su ruta a través del directorio será “/pcn.design/icons/icon_MonetaryCost.png”. Esta imagen representa lo contrario a PCNMonetaryBenefit, mostrando un dólar con el signo negativo (-$), que expresa un coste económico negativo para la empresa.

5.4.4 PCNNonMonetaryBenefit

Su clase dominio es “pcn::PCNNonMonetaryBenefit” por lo que su id será PCNNonMonetaryBenefit y el parámetro de expresión de filtro se define como “feature:nonMonetaryBenefits”.

El estilo de esta etiqueta se define también con una imagen cuyo nombre es “icon_NMBenefit.png”, por lo que su ruta a través del directorio será “/pcn.design/icons/icon_NMBenefit.png”. Esta imagen representa una cara sonriente, que expresa un beneficio no monetario para el cliente.

5.4.5 PCNNonMonetaryCost

Su clase dominio es “pcn::PCNNonMonetaryCost” por lo que su id será PCNNonMonetaryCost y el parámetro de expresión de filtro se define como “feature:nonMonetaryCosts”.

El estilo de esta etiqueta se define con una imagen cuyo nombre es “icon_NMCost.png”, por lo que su ruta a través del directorio será “/pcn.design/icons/icon_NMCost.png”. Esta imagen representa una cara triste, la cual expresa un coste no monetario para el cliente, como un retraso o un rechazo de algún servicio.

5.4.6 PCNSyncTag

Su clase dominio es “pcn::PCNSyncTag” por lo que su id será PCNSyncTag y el parámetro de expresión de filtro se define como “feature:syncTags”.

El estilo de esta etiqueta se define con una imagen cuyo nombre es “icon_SyncTag.png”, por lo que su ruta a través del directorio será “/pcn.design/icons/icon_SyncTag.png”. Esta imagen representa un reloj, que expresa un retardo en la involucración del proceso correspondiente.

5.4.7 PCNDelayTimeTag

Su clase dominio es “pcn::PCNDelayTimeTag” por lo que su id será PCNDelayTimeTag y el parámetro de expresión de filtro se define como “feature:delayTimeTags”.

El estilo de esta etiqueta será igual que una etiqueta textual normal, tratándose de una nota cuya “Label Expression” sea “feature: delayTime”, representando el tiempo correspondiente del PCNWaitStep al que esté asociado esta etiqueta.

5.5 Herramientas

El desarollo explicado hasta el momento contempla la funcionalidad para representar un modelo PCN en la herramienta Sirius, mostrando todos los elementos. Para añadir o eliminar elementos al modelo, es necesario el uso de secciones de herramientas, en esta subsección se explicará en profundidad la implementación de dichas secciones de herramientas.

Las secciones de herramientas, se muestran en una barra de opciones del diagrama representado. Dichas secciones se han dividido según el tipo de herramienta o los elementos a los que afecta.

5.5.1 Sección Relations tools

Esta sección de herramientas contiene las utilidades de las relaciones. Permite crear cada tipo de relación del diagrama.

Para la creación de una relación se define un elemento nuevo de tipo “Edge Creation” en la sección. Este elemento requiere la configuración de dos parámetros, la id del elemento y la relación que creará el elemento.

Se ha creado un elemento de creación para cada tipo de relación del proyecto, por lo que el elemento con id “createPCNDelayedRelation” creará las relaciones “PCNDelayedRelation”, el elemento “createPCNStandardRelation” creará las relaciones “PCNStandardRelation”, el elemento “createPCNPositiveDecisionRelation” creará las relaciones “PCNPositiveDecisionRelation” y, por último, el elemento “createPCNNegativeDecisionRelation” creará las relaciones de tipo “PCNNegativeDecisionRelation”.

Una vez creado el elemento en la sección, se ha aplicado un cambio de contexto en cuya expresión se ha añadido la codificación “var:target”, definiendo el elemento de destino.

Tras añadir el cambio de contexto se ha añadido la operación Set a dicho cambio aplicando en el parámetro de expresión de valor la codificación “var:source”, definiendo así el elemento de origen. Además, para cada tipo de relación hay que establecer el tipo de elemento que se define como origen mediante el parámetro “Feature Name”. En el caso de “PCNDelayedRelation” el origen se establecerá como “PCNDelayedDependencySource”, para “PCNStandardRelation” se establecerá como “PCNStandardDependencySource”, para “PCNPositiveDecisionRelation” se establecerá como “PCNPositiveDecisionSource” y para “PCNNegativeDecisionRelation” se establecerá como “PCNNegativeDecisionSource”.

Al eliminar un nodo en Sirius se eliminan automáticamente las relaciones que tenga con el resto de nodos y la información que tengan dichos nodos vinculados acerca del nodo eliminado, por lo que no es necesario realizar una herramienta específica de eliminación de relaciones.

5.5.2 Sección Entity Tools

En esta subsección se detallan las dos herramientas desarrolladas para crear y eliminar entidades del diagrama.

Se ha desarrollado la creación de un elemento de tipo contenedor cuya id será “Create PCNProcessEntity” y se vincula al elemento “PCNProcessEntity”. Para ello es necesario definir dicho elemento vinculado en el parámetro “Container Mappings” de la pestaña General. Una vez creado, se define un cambio de contexto sobre el contenedor, de tal forma que mediante la codificación “var:container” en el parámetro “Browse Expression” se expresa que la funcionalidad creada se aplicará sobre el contenedor especificado. Dentro del cambio de contexto se ha definido una operación de creación de instancia, en cuyo nombre de referencia se ha establecido “diagramEntities”, que recoge las entidades del modelo y en el parámetro del nombre de tipo se ha indicado “pcn::PCNProcessEntity”, posibilitando así la creación de nuevas entidades.

La herramienta para eliminar entidades del modelo no se encuentra en la barra de herramientas, ya que funciona en conjunto con la opción de eliminar elementos de Sirius, situada en la pestaña superior de opciones del diagrama. Sin embargo, para que al utilizar dicha función de Sirius se elimine la entidad y todos sus elementos contenidos correctamente, es necesario definir una herramienta que recorra el contenedor de entidad y elimine su contenido. Para ello, se ha creado en la sección “Entity Tools” una herramienta de edición de elemento de tipo “Delete element”. Dentro de dicha herramienta se definen los parámetros de id, que se ha establecido como “Delete PCNProcessEntity” y el parámetro que mapea el elemento a eliminar, que en este caso será PCNProcessEntity.

Una vez creada la herramienta se define en ella un cambio de contexto en cuyo parámetro de expresión se define “var:self” indicando que la operación se realizará sobre el elemento especificado. Dentro de dicho cambio de contexto se recorrerán los cinco contenedores que tiene la entidad, eliminando los posibles nodos que tuvieran. Para ello, se definen cinco operaciones “For” distintas, una por cada contenedor, en las que se implementa una operación de tipo “Remove”. En cada operación “For” se definen dos parámetros, la expresión sobre la que itera la operación y el nombre de dicho iterador.

Para la operación “For” que recorre las interacciones directas de la izquierda se define en el parámetro de expresión “feature:leftDirectInteractions”, para la operación sobre interacciones subrogadas de la izquierda se define “feature:leftSurrogateInteractions”, para las interacciones independientes se define “feature:centerIndependentInteractionsActivities”, para las interacciones subrogadas de la derecha se define “feature:rightSurrogateInteractions” y para las interacciones directas de la derecha se define “feature:rightDirectInteractions”.

Finalmente se define una operación de “Remove” en el cambio de contexto, que eliminará la estructura de entidad.

5.5.3 Secciones de creación de procesos.

Para la implementación de herramientas de creación de procesos se han definido cinco secciones de herramientas distintas, organizando los tipos de interacciones por separado en función del dominio de entidad al que correspondan dichas interacciones. De esta forma para crear por ejemplo un proceso de tipo PCNStandardStep en la columna de interacciones directas de la izquierda se deberá utilizar la sección correspondiente (Section Create LeftDirectInteractions). Además, con esta implementación se restringe la posibilidad de crear interacciones de un tipo de dominio en la columna de otro dominio.

Dentro de cada sección se han implementado las herramientas de creación de todos los nodos del proyecto (PCNStandardStep, PCNWaitStep, PCNDoAndWaitStep, PCNDecisionStep, PCNInnovationStep y PCNOutsourcedStep). Para ello, por cada tipo de proceso, se crea una herramienta de creación de nodo, en la que se especifica el id de la herramienta y el nodo que creará dicha herramienta. El id de la herramienta se ha definido añadiendo "create" al nombre del nodo correspondiente, por lo que la herramienta para crear un nodo PCNStandardStep será por ejemplo createPCNStandardStep, definiendo de la misma forma el resto de ids de las herramientas de creación de nodos.

Dentro de cada herramienta de creación se define un cambio de contexto en cuyo parámetro de expresión se codifica "var:container", indicando así que el nodo se creará en el contenedor en el que se encuentra.

Finalmente, para crear el nodo se define dentro del cambio de contexto la creación de una nueva instancia, en cuyos parámetros se indica el nombre de referencia en el que será guardada la nueva instancia y el tipo de elemento que se desea crear. El nombre de referencia se determinará según el dominio en el que se quiera crear el nodo, por lo que corresponderá con la sección en la que está definida la herramienta. Dicho nombre de referencia será "leftDirectInteractions" para las interacciones directas de la izquierda, "leftSurrogateInteractions" para las interacciones subrogadas de la izquierda, "centerIndependentInteractionsActivities" para los procesos independientes de entidad, "rightSurrogateInteractions" para las interacciones subrogadas de la derecha y "rightDirectInteractions" para las interacciones directas de la derecha. Respecto al tipo de elemento que se desea crear, se definirá con el mismo nombre del dominio de clase de cada nodo, siendo por ejemplo "pcn::PCNStandardStep" para los PCNStandardStep, "pcn::PCNWaitStep" para los PCNWaitStep, etc.

Dentro de cada creación de instancia se define una última operación de tipo "Set", para definir el nombre del nodo que se mostrará en el interior. Por defecto este nombre será "Set action on properties" en todos los nodos excepto en PCNWaitStep que será "Wait", los cuales se pueden modificar en las propiedades del nodo una vez creado. Para implementar esta funcionalidad se ha definido el parámetro "Feature Name" de todos los nodos como "action", indicando que el valor que se define será el parámetro "action" del nodo, y el valor por defecto que se ha dado se indica en el parámetro "Value Expression".

5.5.4 Sección Create Tags

En esta sección se definen las herramientas necesarias para incorporar nuevas etiquetas a los nodos disponibles.

Para implementar dichas herramientas no es necesario crear una sección específica para cada dominio de entidad, ya que las etiquetas tendrán siempre el mismo significado independientemente del dominio al que pertenezca el proceso involucrado.

En primer lugar, se ha creado una herramienta de creación de nodo, ya que las etiquetas no son más que subnodos situados al borde de los nodos de procesos. Los parámetros de dicha herramienta serán el id, que se ha definido como "create" junto con el nombre de la etiqueta siendo por ejemplo "createPCNMonetaryBenefit" el id de la herramienta de creación de etiquetas de beneficio monetario, y los nodos que creará la herramienta (en el parámetro "Node Mappings"), por lo que se seleccionan para este parámetro todas las etiquetas del tipo que se quiere crear, independientemente del dominio del proceso sobre el que se quiera adjuntar la etiqueta.

En la herramienta de creación se define un cambio de contexto, que tendrá la codificación “var:container” indicando que dicho el subnodo se creará en la clase contenedora de la etiqueta, que será el proceso seleccionado.

Dentro del cambio de contexto se define una operación para crear una nueva instancia, en este caso una etiqueta. Los parámetros de la instancia a definir son el tipo de elemento que se crea y el nombre de referencia en el que la nueva instancia se guardará. El tipo de elemento se especifica como “pcn::” junto con el tipo de etiqueta correspondiente, siendo “pcn::PCNMonetaryBenefit” el tipo para las etiquetas PCNMonetaryBenefit, y el nombre de referencia será “monetaryBenefits” para “PCNMonetaryBenefit”, “monetaryCosts” para “PCNMonetaryCost”, “textualTags” para “PCNTextualTag”, “syncTags” para “PCNSyncTag”, “nonMonetaryBenefits” para “PCNNonMonetaryBenefit”, “nonMonetaryCosts” para “PCNNonMonetaryCost” y “delayTimeTags” para “PCNDelayTimeTag”.

5.6 Creación del plugin

Tras finalizar el desarrollo de la herramienta se ha procedido a crear un plugin que proporcione la posibilidad de instalar el software de modelado de PCN en cualquier programa Eclipse.

Para comenzar, es necesario agregar el directorio de iconos a la exportación de la herramienta. Para ello se debe seleccionar el fichero MANIFEST.MF que se encuentra dentro del directorio META-INF de la carpeta que incluye el diseño del proyecto. Una vez seleccionado se accede a la pestaña “Build” y se añade el directorio “icons” del apartado “Binary Build”. De esta forma el plugin reconocerá y representará las imágenes utilizadas en el proyecto.

A continuación, se añadirá el “Feature project” en tiempo de ejecución, seleccionando el metamodelo de PCN. Después se añadirá también en tiempo de ejecución el “Update Site Project”, sobre el que se añade el “Feature project” creado. A continuación, se ha compilado el “Update Site Project” obteniendo una serie de fichero en el directorio de Update que se deben exportar como un archivo comprimido (“Archive File” en el directorio General).

Para utilizar el plugin creado se debe instalar en el *workspace* principal, como cualquier plugin de Eclipse. Se seleccionará el archivo comprimido zip para la instalación y se reiniciará Sirius para completar el proceso.

6 Integración, pruebas y resultados

En esta sección se detallarán las pruebas realizadas sobre la herramienta con el fin de localizar posibles errores y los resultados obtenidos de las mismas.

Para probar la funcionalidad de la herramienta es necesario que reconozca modelos ya creados incluidos en XMI y que exista la posibilidad de añadir nuevos elementos a dicho modelo. También será necesario probar a crear una nueva representación de un diagrama y generar correctamente el fichero XMI, siendo este el proceso inverso al mencionado previamente.

A continuación, se detallará como se han realizado las pruebas mencionadas.

6.1 Generar diagrama desde modelo XMI

Para probar la correcta representación del diagrama a raíz de un modelo dado se ha utilizado un modelo propuesto por la tutora. Dicho modelo recoge un sistema de procesos de negocio basado en la funcionalidad de car2go, una empresa de alquiler de vehículos, en cuyo modelo intervienen como entidad el cliente, un punto de validación que comprueba algunos parámetros del cliente y el propio servicio de car2go.

Tras generar la representación de dicho modelo, se ha comprobado que todos los elementos se representan correctamente. Dicha representación se muestra en la Figura 16.

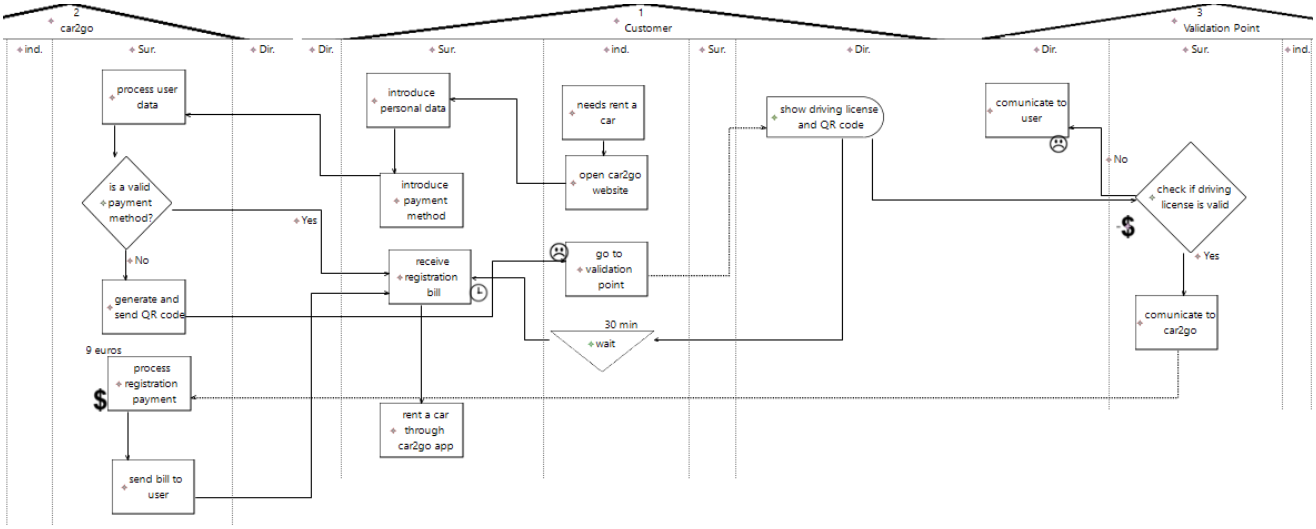

Figura 16 Representación del modelo de car2go

Debido a que la representación del modelo de car2go no contiene todos los tipos de nodos o etiquetas de la notación PCN, se ha creado una nueva entidad en la que se han incluido dichos elementos, comprobando así que todos los componentes se representan correctamente. La entidad que recoge los nodos y etiquetas restantes se muestra en la Figura 17:

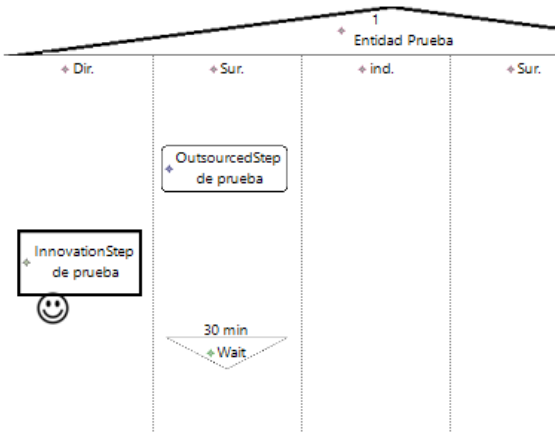

Figura 17 Representación de elementos adicionales

Como se puede observar, los nodos que faltaban por probar en la representación del modelo son PCNInnovationStep y PCNOutSourcedStep, además de la etiqueta de PCNNonMonetaryBenefit. Para probar la etiqueta PCNDelayTimeTag se ha creado un nodo de tipo PCNWaitStep, ya que es el único nodo en el que es posible su representación.

Además de las pruebas mencionadas, se ha intentado provocar el error añadiendo los nodos de un dominio sobre otros dominios o fuera de las entidades, creando relaciones incompatibles entre nodos (relación de decisión fuera de un PCNDecisionStep por ejemplo) y etiquetas sobre nodos que no fueran compatibles (como por ejemplo PCNDelayTimeTag sobre un nodo que no fuera PCNWaitStep). Todas las comprobaciones intentadas han funcionado correctamente.

6.2 Generar XMI mediante representación de diagrama.

Para generar el modelo XMI se ha creado un diagrama PCN y se han ido añadiendo todos elementos disponibles por la herramienta (incluidas relaciones y etiquetas), generando el siguiente diagrama:

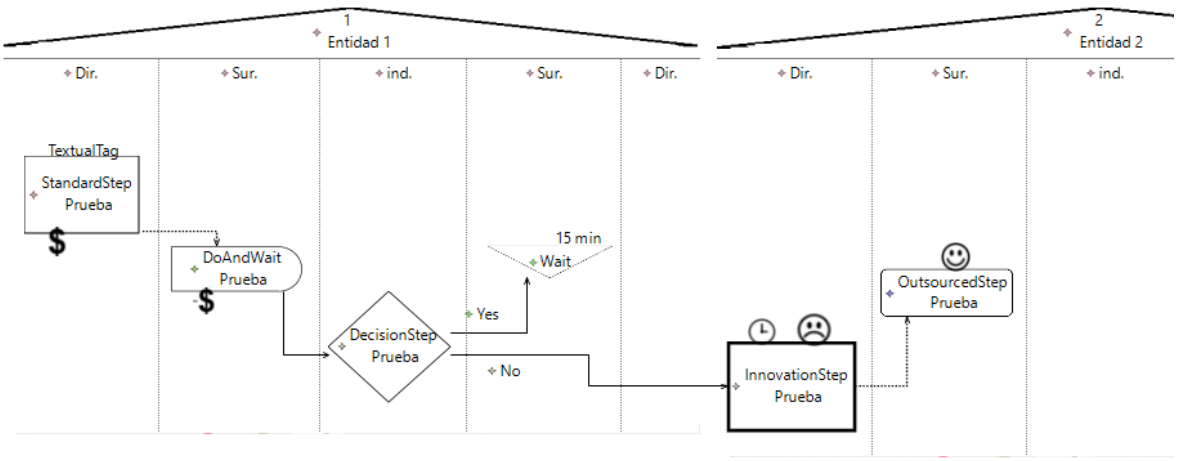

Figura 18 Diagrama de prueba

Solo faltaría comprobar que se ha creado el correspondiente modelo en XMI, actualizándose cuando se realizan cambios en el diagrama.

Observando el código XMI podemos comprobar que los elementos se han generado correctamente. Por ejemplo, para el nodo generado a la izquierda del diagrama

(StandardStep Prueba), podemos comprobar que se ha definido correctamente en el código y que incluye las dos etiquetas que se han creado en el nodo, además del dominio en el que se encuentra (leftDirectInteractios) y la relación que le conecta con un nodo del dominio leftSurrogateInteractions:

```
    <leftDirectInteractions xsi:type="pcn:PCNStandardStep" action="StandardStep
Prueba"PCNDelayedDependency="//@diagramEntities.0/@leftSurrogateInteractions.0"
>
      <monetaryBenefits/>
      <textualTags tag="TextualTag"/>
    </leftDirectInteractions>
```

De esta forma comprobamos que funciona correctamente la utilidad de la herramienta. Para comprobar el funcionamiento del plugin se han seguido las mismas pruebas con éxito.

7 Conclusiones y trabajo futuro

En esta sección se concluirá el trabajo realizado, basado en la experiencia y el aprendizaje obtenido durante el proyecto y se darán algunas propuestas de proyectos o desarrollos futuros compatibles con el software realizado.

7.1 Conclusiones

A lo largo del proyecto se ha investigado acerca de desarrollo sobre Eclipse, especialmente a través de la herramienta Sirius, pudiendo conocer la versatilidad que tiene dicho entorno de desarrollo. Además, el uso de notaciones para el modelado de procesos de negocio proporciona una visión abstracta y organizada de este ámbito de la ingeniería del software.

La idea principal del proyecto era definir una herramienta que fuera capaz de modelar procesos de negocio mediante la notación PCN, la cual se ha desarrollado con éxito, dando la posibilidad además de instalar el software generado sobre cualquier sistema con el programa Eclipse instalado, es decir, desarrollar la herramienta como una aplicación *stand-alone*.

Durante el desarrollo se ha procurado optimizar la herramienta al máximo de forma que sea lo más intuitiva y sencilla posible para un cliente que conozca el funcionamiento de la notación PCN, utilizando nombres de fácil relación como es el caso de los id de los elementos o generando los nodos y entidades con un tamaño estándar, de forma que los procesos sean legibles al crearlos y no requieran de mucha modificación, facilitando al usuario crear diagramas con rapidez.

7.2 Trabajo futuro

La principal funcionalidad de la herramienta es el modelado de diagramas mediante PCN, sin embargo, una posible ampliación del software creado podría ser el soporte de la herramienta para otras notaciones.

Pese a que existen numerosas herramientas disponibles para BPMN, incluir dicha notación en el proyecto ampliaría las posibilidades de mercado del software, además de que el desarrollo sería bastante similar al utilizado para PCN.

Otra posible extensión de la herramienta sería diseñarla como un programa externo a Eclipse, de forma que se pudiera instalar y ejecutar en el sistema sin necesidad de tener Eclipse instalado.

Como última idea a desarrollar se podría implementar la herramienta para dispositivos móviles, ya que existen paquetes de Eclipse dirigidos a desarrolladores de Android. Mediante esta funcionalidad se podrían generar diagramas PCN directamente en móviles y tablets, proporcionando una gran versatilidad de cara a las empresas que quieran exponer sus procesos de negocio de forma rápida y sencilla.

8 Referencias

[1] «emprendedores.es,» [En línea]. Available: https://www.emprendedores.es/estrategia/que-significa-modelo-de-negocio/.

[2] I. Garcia-Magariño, J. Pavon y J. Gomez-Sanz, «Representación de las Relaciones en los Metamodelos con el Lenguaje Ecore,» Departamento de Ingeniería del Software e Inteligencia Artificial, Madrid.

[3] C. Larman, « UML y Patrones.,» eMadrid , Madrid, 2003.

[4] J. B. &. A. R. Quintero, «MDA y el papel de los modelos en el proceso de desarrollo de software.,» 2007.

[5] «bpmn.org,» [En línea]. Available: http://www.bpmn.org/.

[6] E. Gomez-Martinez, F. Perez-Blanco, J. de Lara, J. M. Vara y E. Marcos, «Formal verification of Process Chain Networks using Model-driven Engineering and Petri nets,» de *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing (SAC '19)*, New York, NY, USA, 2019.

[7] T. Bray y J. Paoli, «Extensible markup language (XML) 1.0,» 2000.

[8] J. B. Quintero, «MDA Y EL PAPEL DE LOS MODELOS EN EL PROCESO DE DESARROLLO DE SOFTWARE,» Medellín (Colombia), 2007.

[9] D. Musat Salvador y J. Perez Benedí, «Tutorial de introducción a EMF y GMF. Definición de lenguajes específicos de dominio,» Madrid.

[10] «www.eclipse.org,» [En línea]. Available: https://www.eclipse.org/sirius/overview.html.

[11] «softwareseleccion.com,» [En línea]. Available: https://www.softwareseleccion.com/bonita+bpm-p-1268.

[12] «heflo,» [En línea]. Available: https://www.heflo.com/es/conozca-heflo-bpm/.

[13] «portalprogramas,» [En línea]. Available: https://www.portalprogramas.com/intalio-bpm/.

[14] «lucidchart,» [En línea]. Available: https://www.lucidchart.com/pages/es/ejemplos/herramientas-bpm.

[15] «ingenieriadesoftware.es,» [En línea]. Available: https://ingenieriadesoftware.es/herramientas-modelar-diagramas-uml-online-navegador/.

[16] «kybele,» [En línea]. Available: http://www.kybele.etsii.urjc.es/innovaserv/.

[17] X. Fu, T. Bultan y J. Su, «Analysis of interacting BPEL web services.,» New York, NY, USA, 2004.

[18] W. M. van der Aalst, «Patterns and xpdl: A critical evaluation of the xml process definition language.,» 2003.

[19] «yawlfoundation.github.io/,» [En línea]. Available: https://yawlfoundation.github.io/.

[20] «docplayer.net,» [En línea]. Available: https://docplayer.net/20243525-Introduction-to-pcn-analysis-1.html.

[21] «yawlfoundation.org,» [En línea]. Available: http://www.yawlfoundation.org/pages/resources/examples.html.

[22] M. A. Lopez-Gordo, P. Garcia-Sanchez, P. Castillo Valdivieso, J. Gonzalez Peñalver y M. I. Garcia Arenas, «Web 2.0: Arquitectura orientada a servicios en Java. Enseñanza y Aprendizaje de Ingeniería de Computadores,» 2011.

9 Glosario

PCN	Process Chain Networks
UML	Unified Modeling Language
MDE	Model-Driven Engineering
MDD	Model-Drivel Development
BPMN	Business Process Model and Notation
XML	Extensible Markup Language
XMI	XML Metadata Interchange
EMF	Eclipse Modeling Framework
GMF	Graphical Modeling Framework
OMG	Object Management Group
ERD	Entity Relationship Diagram
BPEL	Business Process Execution Language
XPDL	XML Process Definition Language
YAWL	Yet Another Workflow Language

10 Anexos

A Manual de instalación

Antes de instalar la herramienta desarrollada se requiere la instalación de software adicional.

En primer lugar, se deberá instalar el entorno de desarrollo Eclipse en su versión Eclipse IDE 2020-03 (o superior), tras ello deberán añadirse al entorno los paquetes disponibles en el Marketplace de Eclipse relacionados con EMF (EMF Client Platform, EMF Compare, EMF Forms y EMFStore model repository) y el paquete Sirius en su versión 6.2 o superior, disponible también en el Marketplace de Eclipse.

Para comenzar a utilizar la herramienta sin el plugin generado se deberán seguir los siguientes pasos:

1. Descomprimir los ficheros del proyecto.
2. Crear un nuevo proyecto en Eclipse, especificando el *workspace* en el que se vaya a trabajar.
3. Introducir en el *workspace* que se vaya a utilizar los directorios Pcn.ecore, Pcn.ecore.edit y Pcn.ecore.editor.
4. Ejecutar el programa para abrir la ventana en tiempo de ejecución.
5. Incluir en el proyecto en tiempo de ejecución el directorio Pcn.design.

Tras realizar estos pasos el usuario podrá incluir modelos al proyecto y desarrollar diagramas PCN.

Para utilizar la herramienta mediante el plugin se deberán seguir los siguientes pasos:

1. Descomprimir los ficheros del proyecto.
2. Seleccionar la opción Install New Software de la pestaña “Help” en la parte superior de la ventana de Eclipse.
3. Seleccionar Add, luego Local y seleccionar el fichero PCN_finalPlugin.zip.

Tras reiniciar Sirius la herramienta estará disponible para usarse en Eclipse mediante el plugin.

B Manual del programador

Para modificar o ampliar la funcionalidad de la herramienta será necesario instalar sus componentes tal y como se explica en el manual de instalación, sin utilizar el plugin generado.

Una vez agregados los directorios de ficheros correspondientes al *workspace* seleccionado, el desarrollo se podrá llevar a cabo modificando el archivo pcn.odesign. Dicho archivo se encuentra en tiempo de ejecución en el directorio pcn.design, dentro de la carpeta description. Tras haber accedido a dicho fichero se podrá modificar mediante la variación de los parámetros de cada elemento del diseño o añadiendo nuevos elementos y secciones.

Para crear nuevos elementos de diagrama, ya sean relaciones nodos o contenedores, se debe hacer click derecho sobre el nodo "Default" y seleccionar la opción "New diagram element", especificando el tipo de elemento que se desea crear en el menú desplegado. Una vez creado un elemento nuevo se podrán modificar sus parámetros y crear subnodos sobre el elemento de la misma forma que se ha creado dicho elemento. En caso de que el elemento creado sea un contenedor, se podrán añadir nuevos contenedores en su interior, además de subnodos.

Para modificar el estilo de un elemento se hará click derecho sobre el elemento, se seleccionará "New style" y se especificará el tipo de estilo deseado. En caso de que se requiera de alguna imagen o icono para el estilado de un elemento del diseño, se deberá almacenar dicha imagen en el directorio "icons", que está situado dentro del directorio de pcn.design.

Para agregar una nueva sección de herramientas se deberá hacer click derecho sobre el nodo "Default", seleccionar la opción "New tool" y seleccionar la opción "Section".

Una vez creada la sección se podrán crear en su interior herramientas de creación de elemento mediante la opción "New element creation", en la que se podrá elegir entre nodo, contenedor o relación, o herramientas de edición de elemento en la opción "New element edition" en el caso de que se quiera crear una herramienta de eliminar elemento o alguna otra utilidad de edición del elemento.

Los cambios de contexto se realizarán sobre el elemento "Begin", generado automáticamente en el interior de la nueva herramienta que se haya generado en la sección. Las operaciones que se realizan sobre los cambios de contexto se generarán haciendo click derecho sobre el cambio de contexto y seleccionando la opción "New operation", donde se mostrarán las operaciones disponibles para seleccionar la operación requerida.