Trabajo de Fin de Grado

Sistemas de Modelado Automático y Simulación de Organizaciones TIC

Automatic System Modeling and Simulation of ICT Organizations

José Miguel Hernández Martín

La Laguna, 8 de septiembre de 2015

D. **Vanesa Muñoz Cruz**, con N.I.F. 78.698.687-R profesora Ayudante Doctor adscrita al Departamento de Ingeniería Informática de la Universidad de La Laguna, como tutor

Sistemas de Modelado Automático y Simulación de Organizaciones TIC

D. **Pedro Antonio Toledo Delgado**, con N.I.F. 45.725.874-B profesor Ayudante adscrito al Departamento de Ingeniería Informática de la Universidad de La Laguna, como cotutor

C E R T I F I C A (N)

Que la presente memoria titulada:

"Sistemas de Modelado Automático y Simulación de organizaciones TIC"

ha sido realizada bajo su dirección por D. **José Miguel Hernández Martín**,

con N.I.F. 54.113.507-G.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 8 de septiembre de 2015

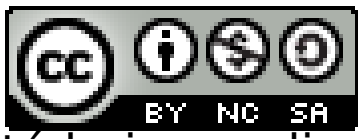
Agradecimientos

A mi familia porque siempre estuvieron apoyándome de principio a fin.

A todos mis profesores que han inculcado en mí, sus conocimientos y me han facilitado las bases para ser un buen profesional.

Por último y no menos importante a todas las personas que a lo largo de mi estancia en esta universidad han compartido momentos y vivencias conmigo , siempre me llevaré, y llevo, un trocito de cada uno.

Licencia



Resumen

Hoy en día, cada vez más organizaciones se preguntan cómo aprovechar mejor sus recursos y optimizar las actividades que realizan en su día a día. Estas organizaciones suelen disponer de un sistema de información, donde almacenan su información en ficheros de tipo log. Estos logs pueden posibilitar el estudio de la entidad acerca de como usa sus recursos, tiempo y actividades.

En este proyecto se usarán herramientas de minería de procesos construyendo, verificando y extendiendo a partir de logs, un modelo de proceso de negocio. Posteriormente a partir del descubrimiento de este modelo podemos realizar una simulación de eventos discretos con una herramienta para ello, permitiendo simular el proceso de negocio de la organización.

El trabajo conjunto de ambas herramientas vendría a satisfacer la necesidad de algunas empresas u organizaciones que deseen optimizar su modelo de proceso de negocio y por ende su propia entidad, pudiendo en algunos casos reducir costes, aumentar la productividad ...

Actualmente, no existe una herramienta que permita realizar el proceso descrito anteriormente de manera automática, teniendo que dedicar esfuerzos a modelar un proceso de negocio, lo que supone tiempo y costes adicionales para la empresa.

Este proyecto pretende desarrollar la integración de técnicas pertenecientes a dos campos presentes en las ciencias de la computación, la minería de procesos y la simulación de eventos discretos, con el objetivo de facilitar a las organizaciones simular o acceder a simulaciones de sus procesos de negocio para optimizar sus recursos.

Para ello, se ha hecho la selección de una herramienta de minería de procesos "ProM" y de simulación de eventos discretos "Psighos". Además, se ha definido un lenguaje "Process Tree" por el cual se especificará el modelo común entre ambos para realizar la integración. Por ello se ha creado un plugin ProM, que a partir de un modelo de procesos de negocio sea capaz de simular una organización de manera automática.

Palabras clave: Minería de procesos, Simulación de Eventos Discretos, ProM, Psighos, Patrones de Flujos de Trabajo.

Abstract

Nowadays, more and more organizations are wondering about how to make a better use of resources and optimize their activities each day. These organizations often have an information system, where the information are stored in files (log type). These logs let the study of the entity about as how the organization are using his resources, time and activities.

In this project will be use mining tools for building processes, verifying and extending logs from a business process model. Later of the discovery of this model we can make a discrete event simulation with a tool for this, allowing to simulate the business process of the organization.

The joint work of this tools would satisfy the need of some companies or organizations that want to optimize their business process model and hence it is own entity, allowing in some cases reduce costs, increase productivity...

Currently, there is not a tool that perform the above process automatically, having to spend efforts to create a business process model, increasing time and costs for the company.

This project aims to develop the integration of techniques from two fields present in the computer science, mining process and discrete event simulation, in order to facilitate to the organizations simulate or access to his simulations of their business processes to optimize their resources.

To do this, it has been done the selection of a process mining tool "Prom" and discrete event simulation "Psighos" tool. In adition has been defined a language "Process Tree" which will be the common model among both for be used in the integration. Therefore it has created a plugin ProM, which from a business process model will be able to simulate an organization automatically.

Keywords: Process Mining, Discrete Event Simulation, ProM, Psighos, Workflow Patterns.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1
Introducción

Hoy en día, el tratamiento de la información está adquiriendo cada vez más importancia, pues supone uno de los valores más fundamentales de una organización, potenciando cada año la apuesta de las entidades por invertir en investigación y desarrollo de las llamadas **Tecnologías de la Información**(IT), definida por la Asociación de la Tecnología de la Información de América(ITAA) como:

"El estudio, diseño, desarrollo, implementación, soporte o gestión de sistemas de información basado en ordenadores."

La información, su estructura y el manejo de la misma, de manera lógica crece a la par con la organizaciones. Este crecimiento da lugar a escenarios complejos dentro de la propia entidad, donde las interrelaciones entre la misma pueden ser tediosas, si el diseño y la planificación para mantener, manejar y utilizar la información no es el más correcto.

Entender como funcionan todos estos elementos y sus relaciones entre sí, es de gran importancia para conocer realmente como funciona una organización. El objetivo, no es otro que mejorar la eficiencia, efectividad, consistencia, productividad, ahorro y calidad de la organización, siempre que sea posible. Por ello, las empresas usan técnicas del campo de la ciencia de la computación de la simulación de eventos discretos.

Una **simulación de eventos discretos**[1,2] simula los cambios por parte de conjunto de variables de entorno en un intervalo discreto (no regular) de momentos en un sistema, manteniendo el estado del sistema entre eventos. Para realizar una simulación, es necesario un modelo del proceso de negocio de la organización, que simular. La figura 1.1 describe un pequeño modelo de proceso de negocio que representa el comportamiento de un banco al valorar la concesión de un préstamo a un cliente.

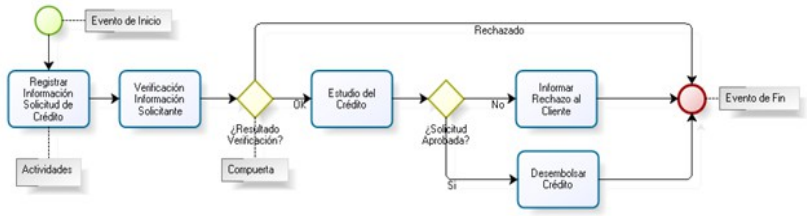


Figura 1.1 Ejemplo de modelo de proceso de negocio

Para definir modelo de proceso de negocio, primero debe definirse proceso de negocio.

“Un proceso de negocio es un conjunto estructurado y medible de actividades diseñadas para producir un producto específico, para un cliente o mercado concreto. Implica un fuerte énfasis en cómo se ejecuta el trabajo dentro de la organización, haciendo hincapié en el qué, característico de la focalización en el producto”[3].

A su vez, modelo de proceso de negocio puede ser definido como:

“La representación analítica o la ilustración de los procesos de negocio de una organización”.[4]

Para realizar una simulación de eventos discretos, se necesita un grupo de analistas, arquitectos, diseñadores y desarrolladores de la información, o un subconjunto de ellos, especializado en el estudio de modelos de procesos de negocio. Este personal especializado se encarga de estudiar la organización y obtener un modelo de proceso de negocios válido para ser simulado.

La necesidad de dedicar gente a la creación de modelos para poder simularlo, supone un aumento de los costes y de tiempo, que a veces las organizaciones no están dispuestas a costear.

Con el objetivo de facilitar a las organizaciones simular o acceder a las simulaciones de sus procesos de negocio para optimizar sus recursos, se hará uso de otra disciplina del campo de las ciencias de computación la minería de procesos, para elaborar una solución.

“La **minería de procesos**[5,6,7] es un campo de la ciencia de la computación, que extrae conocimientos de los registros disponibles en los sistemas de información de la organizaciones, con el objetivo de construir, verificar y extender a partir de logs, un modelo de proceso de negocio.” Véase en la figura 1.2 la descripción gráfica de minería de procesos.

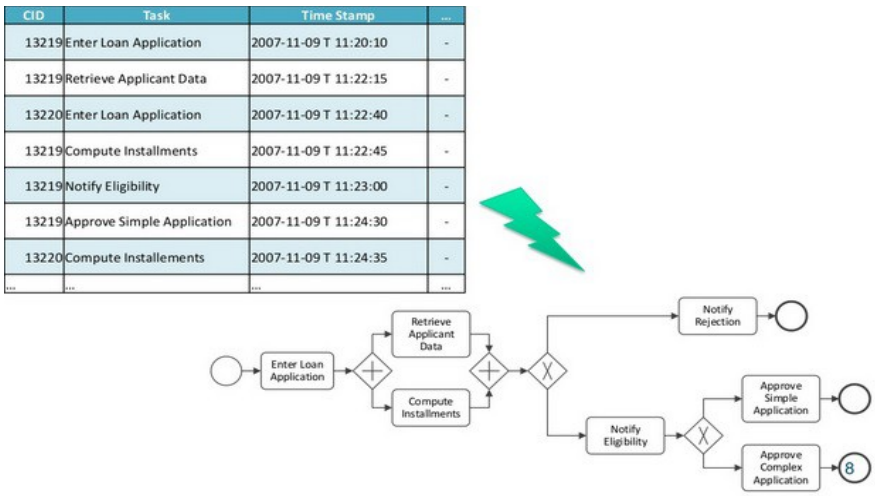


Figura 1.2 Ejemplo de una transformación hecha por una herramienta de minería de datos

Las herramientas de minería de procesos son capaces de extraer a partir de un log un modelo de procesos de negocio, de manera automática.

Actualmente no existe una herramienta de minería de procesos que cree automáticamente un modelo de proceso de negocios lo suficientemente completo como para ejecutar una simulación de eventos discretos. Por ello, la finalidad de este proyecto es el uso combinado de minería de procesos y simulación de eventos discretos, para crear una herramienta que sea capaz de desarrollar simulaciones a partir de la extracción semi-automática de los modelos utilizando técnicas en minería de procesos acompañada de la interacción del usuario con el sistema.

La simulación podría revelarnos o no, información para una potencial mejora en la empresa. En caso afirmativo, éstas podrían ser algunas de ellas:

- Una mejor definición de sus tareas.
- Una mejor configuración de sus recursos.
- Una mejora de su beneficio.
- Una mejora de los tiempos de ejecución.
- Una reducción de costes.
- Una satisfacción mayor por parte de las partes implicadas, no solo clientes sino también de trabajadores.
- Una mejora en la calidad de su producto, servicio y por ende de la organización.

1.1 Workflows

Como hemos comentado anteriormente se tratará de construir un modelo de procesos de negocio mediante una herramienta de minería de procesos para posteriormente poder llevar a cabo la simulación. La definición formal de estos modelos se explícita utilizando estructuras de flujos de trabajo o workflows.

Los **flujos de trabajo** (workflows, a partir de ahora), estudian aspectos relacionados con las tareas como, su estructura, como se realizan, como se relacionan, la información que manejan y como se llevan a cabo, permitiendo la interacción entre los participantes del flujo compartiendo información entre ellos, de forma ordenada y definida por un conjunto de reglas y una secuencia de actividades.

En este proyecto se trabajará con patrones de flujo de trabajo, porque es más fácil, resolver poco a poco las partes que componen un problema, como si se tratase, de una estrategia de divide y vencerás.

Los **patrones de flujo de trabajo** (**workflow patterns**, a partir de ahora), son aquellas estructuras identificables que componen un workflow y que algunos casos pueden repetirse de manera recurrente.

A pesar de que dos herramientas utilicen workflows como formalización de un modelo, esto no garantiza por sí mismo que cualquier workflow sea compatible con ambas herramientas, ni tan siquiera que exista un conjunto de workflows soportados por ambas. Sin embargo, como se ha descrito anteriormente cada workflow estará formado por un conjunto de patrones de workflow, conectados entre sí. En general, los modelos de workflow soportados por una herramienta quedan definidos mediante el conjunto de patrones de workflow de los que pueden estar compuestos los modelos que reconoce.

Como consecuencia de lo anterior, es importante analizar los patrones de workflow soportados en los modelos resultantes de la minería de procesos, así como cuáles son tradicionalmente empleados para las distintas simulaciones, avanzando con ello en el establecimiento de un lenguaje conjunto.

Se comienza por realizar un estudio de los patrones de workflow reseñados en la bibliografía. La creación de estos patrones nace de la colaboración entre el profesor Wil van der Aalst de la Universidad Tecnológica de Eindhoven y el profesor Arthur ter Hofstede de la

Universidad Tecnológica de Queensland desde 1999. Esta colaboración tenía como objetivo establecer una ley universal de conceptos de modelado de procesos de negocio.

En general debe tenerse en cuenta que hay diferentes perspectivas para la especificación de workflows:

- **Perspectiva de control de flujos**, que describe actividades, su ejecución y orden a través de diferentes estructuras de flujo de tareas.
- **Perspectiva de datos**, supone una capa de negocio y procesamiento de datos en la perspectiva de control.
- **Perspectiva de recursos**, que provee de una estructura que facilita la interacción con los elementos que intervienen en los workflows, como personas y los roles que estos desempeñan para ejecutar actividades.
- **Perspectiva operacional** describe acciones elementales ejecutadas por actividades.

La definición de algunos de estos patrones en detalle se realizará en el capítulo 2.2.2 desde el punto de vista de Psighos.

En (Kiepuszewski 2002) se exponen diferentes lenguajes de modelado de workflows, comentando la compatibilidad de los mismos con la perspectiva de control de flujos.

Generalizando y clasificando existen 4 modelos:

- Los **modelos de workflow estándar**.

Aparecen como una representación natural de los patrones, sin restricciones a la hora de usarlos. Esta libertad genera iteraciones arbitrarías y la posibilidad de usar múltiples instancias, pero existe la posibilidad de encontrar puntos muertos en la representación o iteraciones infinitas.

- Los **modelos de workflow seguros.**

Son modelos de workflow estándar que no permiten múltiples instancias, aún así, los puntos muertos puede ocurrir.

- Los **modelos de workflow estructurado.**

Los modelos workflow estructurado son aquellos que a cada construcción de división-or o and le sigue su homólogo para la unión-or o and. Sin ciclos arbitrarios y con la posibilidad de tener

múltiples instancias, evitando puntos muertos, pero derivando en una expresividad reducida.

- Los **modelos de workflow que usan sincronización.**

En los modelos que usan la sincronización, al igual que en los workflows estructurados, para cada división-or o and establece una unión-or o and. Usando los mecanismos de unión para manejar una sincronización entre las ramas usando tokens que pueden estar activos o inactivos según corresponda. Al propagar tokens a lo largo de la estructura de acuerdo a la definición formal de este tipo de modelos, debería de evitarse los puntos muertos, posibilitando la creación de múltiples instancias, sin descuidar la potencia expresiva.

1.2 Objetivos

1.2.1 Objetivos generales

- Realizar una herramienta que permita elaborar una comunicación entre los resultados de una herramienta de minería de procesos, para posteriormente ejecutar con esos resultados en una herramienta de simulación de eventos discretos una simulación.
- Facilitar el acceso a las organizaciones para simular o acceder a las simulaciones de sus procesos de negocio, con el objetivo de optimizar sus recursos.

1.2.2 Objetivos específicos

- Designar la herramienta de simulación de eventos discretos utilizada.
- Designar la de herramienta de minería de procesos utilizada.
- Establecer lenguaje que se usará para realizar la comunicación entre la simulación de eventos discretos y la minería de procesos.
- Elaborar un plug-in ProM conversor de manera que:
  - Su entrada sea, en parte, un workflow creado en ProM.
  - Su salida sea una llamada a Psighos, mostrando el resultado de la simulación.

- El workflow utilizado debe aprovechar la potencia de descripción de patrones de Psighos.

1.3 Alcance

El desarrollo abarcará la creación de un plugin ProM que a partir de workflows creados por otro plugin, transforme esos workflows en una entrada válida que puedan ser usados en la librería Psighos, para ejecutar una simulación. Se intentará realizar una conversión aprovechando la potencia expresiva de Psighos.

Si no se pudiera realizar una conversión, se intentarán buscar otras opciones de desarrollo, como la de por ejemplo, desarrollar una interfaz java portable compatible con ProM, que permita configurar el resto de elementos Psighos que son necesarios para ejecutar una simulación y que no tienen que ver con la definición de workflows.

Capítulo 2
Estado del arte

2.1 Minería de procesos

Existen varias herramientas de Minería de procesos como ProM, Process Mining, Disco, Aris Process Perfomance, Futura Reflect, BPMone, Celonis Discovery y Orchesta o QPR ProcessAnalyzer, entre otros.

Se ha elegido utilizar una de las herramientas de minería más utilizadas actualmente a nivel académico, ProM, desarrollada por la Universidad Tecnológica de Eindhoven.

ProM[8] abreviatura de en inglés “Process Mining Framework” o en español Marco/Framework de Minería de Procesos, es una herramienta de código abierto, que almacena algoritmos de minería de procesos. Permitiendo tanto a usuarios de algoritmos como desarrolladores usar o extender su funcionalidad por medio de plug-ins. Como esta escrito en java es independiente de la plataforma.

En este momento muchos de los plugins están agrupados por paquetes, el paquete ProM core esta distribuido bajo una licencia GPL y los otros paquetes de ProM normalmente están distribuidos bajo una licencia L-GPL pudiendo usar todo el entorno de ProM de manera gratuita y sin restricciones, pero teniendo en cuenta que si distribuyes tu propio plugin debes hacerlo bajo tu propia licencia y siempre bajo la licencia L-GPL.

Dentro de ProM se usan lenguajes como:

- Redes de petri[9] son un lenguaje que provee una herramienta de modelado muy efectiva para la representación y el análisis de procesos concurrentes.(C.A. Petri, 1962)
- Process Tree[10] es una representación abstracta de un bloque estructurado de workflows.

- YAWL(Yet Another Workflow Language) es un lenguaje basado en workflow patterns que sigue un sistema de transiciones etiquetado, siguiendo un modelo de workflows patterns estructurado a la hora de ser definido dentro de el plug-in.
- BPML (Business Process Modeling Language) son meta lenguajes que se integran con la gestión y el sistema de información de la organización basándose en XML.

Algunos plugins de minería de procesos que ProM utiliza, son:

- **ILP** genera una red de petri marcada a partir de un log dado. Hace uso de Programación Lineal Entera (IPL) para encontrar lugares de acuerdo a la teoría de las regiones.
- **FM** (Fuzzy Miner o minero borroso/confuso) explora interactivamente los procesos de registros de eventos. Adecuado para la minería de procesos que muestra una gran cantidad de comportamientos estructurados y conflictivos.
- **HM** (Heuristic Miner- Minero heurístico) realiza una minería de procesos, abstrayéndose de los comportamientos excepcionales y el ruido presentes en algunos logs.
- **IM** (Inductive Miner- Minero inductivo) genera una red de petri marcada o un árbol de procesos a partir de un log dado. Hace uso de una estrategia de divide y vencerás.
- **IMi** (Inductive Miner Infrequently- Minero inductivo infrecuente) es una extensión de Inductive Miner que introduce un filtro para comportamientos infrecuentes en todos los pasos de IM.

2.2 Simulación de eventos discretos

Desde el punto de vista de la Simulación de eventos discretos hay varias herramientas como, Sighos/Psighos[2], DesmoJ y ProModel.

Está vez la librería/herramienta utilizada será Psighos, debido a que su diseño esta orientado a usar patrones definidos en workflowpatterns[11] por el Profesor Wil van der Aalst, facilitando así el desarrollo de este proyecto. Esta librería ha sido desarrollada por la Universidad de La Laguna.

Psighos es una librería de simulación de eventos discretos que se creó a partir de añadirle a la librería Sighos (Universidad de La Laguna) la funcionalidad de usar varios hilos, para ejecutar de manera mas eficiente los patrones que realizan múltiples instancias de manera concurrente en sus simulaciones.

Actualmente funciona con una serie de elementos como tipos de recursos, recursos, grupos de trabajo, actividades, flujos de trabajo, entre otros.

Con anterioridad, se disponía de una librería anexa a Psighos, que permitía declarar los elementos de una simulación en un xml (XMLSIGHOS), pero actualmente esta funcionalidad no está disponible, en la última versión de la librería.

En el siguiente apartado, analizaremos con un poco más de detalle estos elementos y otros, que nos servirán de utilidad para realizar el desarrollo.

2.2.1 Breve descripción de los componentes de Psighos.

- **Simulación o Simulation**

El elemento simulación es el contenedor del resto de componentes de modelado, es el componente que ejecuta la simulación en sí misma. Concretamente contiene un tiempo de finalización , un conjunto de recursos , de tipos de recursos , de actividades , de workflows y de generadores de elementos.

Ejemplo: Simulación del proceso de pago en un supermercado.

- **Tipos de recurso o Resource Type**

Los tipos de recurso agrupa los recursos por su funcionalidad o rol, permitiendo valernos de un recurso según la función que aporta indistintamente de cual sea.

Ejemplo: Una caja registradora.

- **Recursos o Resources**

Representa cualquier cosa ya sea material o humana para llevar a cabo con éxito una tarea. Pudiendo incluso asignar roles según un periodo de tiempo determinado, a cada uno de ellos. Pudiendo tener un recurso varios roles en diferentes intervalos de tiempo o varios en el mismo intervalo.

Ejemplo: La persona encargada de la caja.

- **Actividades o Activities**

Representa una pieza de trabajo atómica o un paso lógico del proceso, que generalmente necesita recursos de algún tipo para desarrollarse.

En el caso de Psighos, es una tarea que tendrá que realizar un elemento en algún momento de su ciclo de vida.

Ej: Pasar los productos por el lector

- **Grupos de Trabajo o Workgroup**

Establece pares de un recurso determinado, con un rol determinado, además de, indicar cuanta cantidad de recursos hay de ese tipo.

Ej: Una persona en un supermercado puede en un determinado horario ejercer de cajera y en otro de gerente.

- **Elementos o Elements**

Son los actores principales de la simulación, una entidad que interactúa con el sistema resolviendo actividades a su paso por la simulación

Ej: El cliente que realiza la compra.

- **Generador de elementos o Element Generator**

Este componente se encarga de ir generando los elementos siguiendo un patrón temporal o respondiendo a un evento de simulación.

Ej: Simular la llegada de clientes a una caja generándolos de una manera determinada.

- **Workflows**

Comprende todos aquellos aspectos que guardan relación con las actividades (dentro del modelo de negocio), como están estructuradas, que hace cada una de ellas, cuando se realizan, que relación guardan entre sí y como se mueve la información para conseguir un determinado fin.

Ej: Todo el proceso compuesto por subprocesos y actividades que empieza desde que un cliente llega a la caja y sale con su compra pagada.

2.2.2 Tratamiento de workflows.

Conociendo los elementos de Psighos nos centraremos en el estudio de actividades y más concretamente identificación de workflows para elaborar el plugin ProM que conecte ambas herramientas.

Psighos usa tokens siguiendo la estrategia introducida por los modelos de sincronización de flujos de trabajo o Synchronising Workflow Models anteriormente nombrados en el capítulo 1.1. Esta librería de simulación usa una serie de estrategias para relajar los límites impuestos entre múltiples instancias y iteraciones arbitrarías que este modelo impone. Este hecho proporciona a Psighos una gran potencia de generación de workflows.

Psighos dentro del extenso mundo de los patrones, modela patrones de flujos de control.

Thus, Rusell et al.(2006) propuso 43 patrones de flujos de control (WCP1-WCP43)[11]

] que sirven para modelar diferentes escenarios definidos en esta perspectiva.

Los patrones están ordenados según su comportamiento o tipo en, patrones de control básico, de ramificación avanzada y sincronización, de múltiples instancias, basados en estados, patrones de cancelación, de iteración, terminación o disparadores.

No todos ellos están soportados por Psighos. En el Apéndice A, podemos ver el conjunto de patrones de flujo representados por Psighos. Ahora analizaremos algunos de ellos, utilizando cajas para representar actividades o tareas y bolitas o tokens que representan como se comportan esos elementos en el patrón.

Flujo de control básico

- **WCP1: Secuencia**

  A partir de ahora WCP1 Sequence.

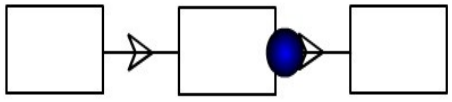


Figura 2.1 WCP1 Sequence

El patrón Sequence activa una tarea, cuando se ha completado la

tarea que la precede, realizando el mismo proceso. Es decir, en el caso de la figura 2.1, la segunda tarea no se ejecutará hasta que la primera se haya completado.

- **WCP2: División Paralela**

  A partir de ahora WCP2, Parallel Split o And-split.

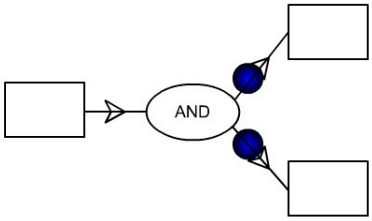


Figura 2.2 WCP2 ParallelSplit

El patrón Parallel Split divide una rama en varias, ejecutando ambas ramas de manera concurrente.

- **WCP3: Sincronización**

  A partir de ahora WCP3, Synchronization o And-join.

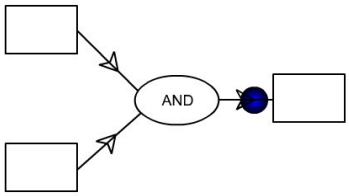


Figura 2.3 WCP3 Synchronization

El patrón Synchronization es la unión de varias ramas (and-split) en una, este evento se ejecuta cuando todas las ramas concurrentes son activadas, es decir, todas ha alcanzado el nodo And.

- **WCP4: Selección exclusiva**

  A partir de ahora WCP4, Exclusive Choice o Xor-split.

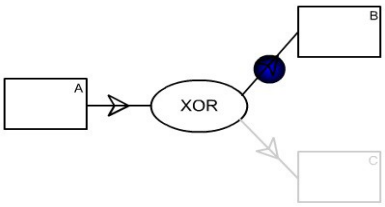


Figura 2.4 WPC4 ExclusiveChoice

En el patrón Exclusive Choice, la rama se divide en varias, donde se evalúa una condición, en cada rama, la primera que sea evaluada como verdadera continuará el flujo a través del xor (únicamente 1). Hay una variante de este patrón en Psighos, que usa porcentajes para decidir que nodo debe continuar el flujo, denominado Probability Selection Flow – Flujo de Selección por Probabilidad).

- **WCP5: Unión simple**

  A partir de ahora WCP5, Simple Merge o Xor-join.

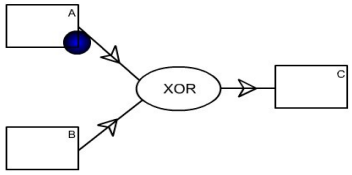


Figura 2.5 WCP5 SimpleMerge

El patrón simple merge, corresponde a un xor-split, donde el token que llegue primero continua el flujo.

Ramificación avanzada y sincronización

- **WCP6: Multi-Elección**

  A partir de ahora WCP6, Multi-Choice o Or-split.

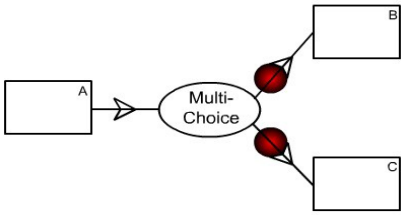


Figura 2.6 WCP6 Multi-Choice

El patrón de Multi-Choice, corresponde a un or-split, todos las ramas que reúnan la condición podrán continuar el flujo. Se parece al xor-split, con la diferencia de que puede continuarse también el flujo por las dos ramas, con un token en cada una.

- **WCP7: Unión Sincronizada Estructurada**

  A partir de ahora WCP7 o Structured Synchronizing Merge.

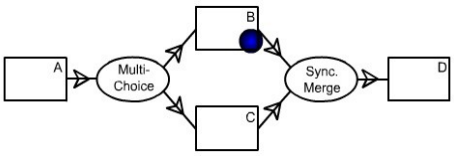


Figura 2.7 WCP7 StructuredSynchronizing Merge

El patrón Structured Synchronizing Merge, corresponde a usar un patrón Multi-Choice para dividir las ramas, conjuntamente con un patrón Synchronization para unirlas.

- **WCP8: Discriminador estructurado**

  A partir de ahora WCP8 o Structured Discriminator.

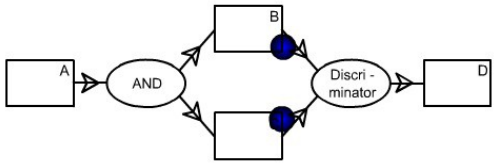


Figura 2.8 WCP8 Structured Discriminator

El patrón Structured Discriminator, es una variante del And-join, pero solo dejando entrar al primer flujo que cruce el nodo discriminador, en caso de entrar a la vez se elegirá uno de ellos.

Modelos basados en estado

- **WCP40: Enrutamiento intercalado**

  A partir de ahora WCP40 o Interleaved Routing.

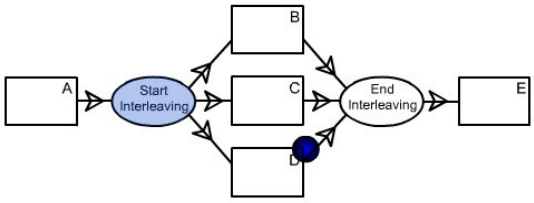


Figura 2.9 WCP40 InterleavedRouting

El patrón Interleaved Routing, corresponde a usar un patrón Parallel-Flow para dividir las ramas, conjuntamente con un patrón Synchronization para unirlas.

Patrones de iteración

- **WCP21: Iteración estructurada**

  A partir de ahora WCP21 o Structured Loop.

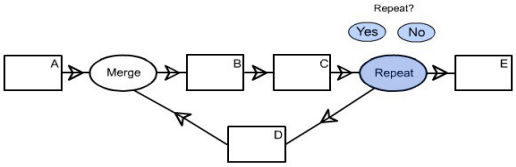


Figura 2.10 WCP21 StructuredLoop

El patrón Structured Loop ejecuta una serie de tareas o sub-procesos repetidamente, hasta que se cumpla una condición que interrumpa el bucle. La comprobación de esta condición, puede hacerse antes de ejecutar las actividades que componen el loop, o después de hacerlo.

Capítulo 3
Diseño de la solución

A continuación, explicaremos de manera general el flujo de elementos partícipes en la elaboración de los resultados de una simulación de eventos discretos.

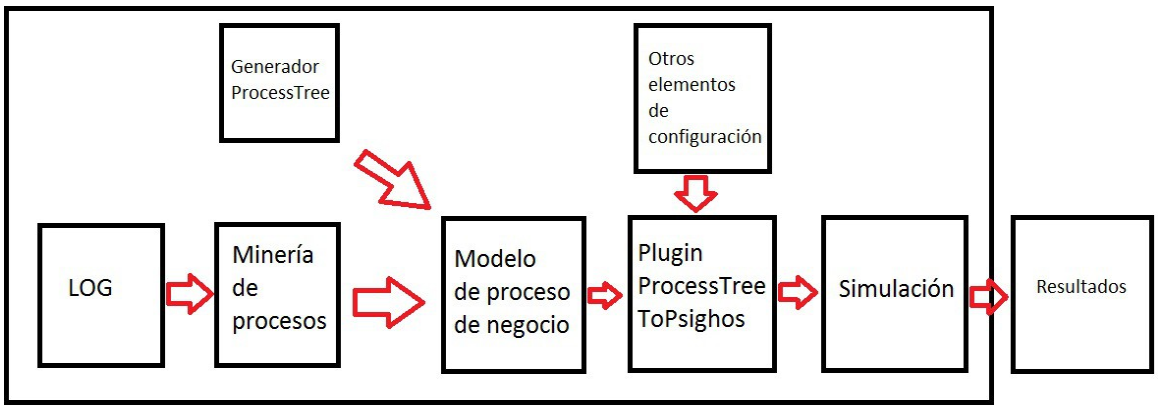


Figura 3.1 Elementos que generan resultados de simulación

Como vemos en el modelo 3.1, el plug-in ProcessTreeToPsighos que supone el puente de unión entre la minería de procesos y la simulación de eventos discretos, necesita un modelo de proceso de negocio como entrada. Para obtener nuestro modelo, tal y como muestra el sistema, tenemos dos opciones:

- Utilizar una herramienta de minería de procesos, para construir a partir de un log, un modelo.
- Utilizar un generador de modelos, para generar un modelo aleatorio, que posteriormente se podría editar para adaptarse a nuestro problema.

Para saber que algoritmo de minería de procesos o que generador de modelos podemos usar, antes debemos elegir que lenguaje se

utilizará para modelar nuestra solución.

Posteriormente teniendo el modelo elegido, lo transformaremos mediante nuestro plugin a workflows válidos para la entrada de la simulación. Como este proceso de conversión es semi-automático, necesitaremos de algunos elementos de configuración extra, ajenos a los workflows para realizar nuestra simulación, y así, obtener el resultado de la misma.

Como objetivo final, se habrá realizado de manera semi-automática a partir de un log o un generador de modelos, una simulación de eventos discretos.

En los siguientes capítulos explicaremos que lenguaje se ha elegido para modelar el proceso de negocio, derivando por ende en el algoritmo de minería de procesos y generador de modelos elegido, así como, la implementación del plug-in y los resultados que se han obtenido.

Capítulo 4
Modelo de procesos de negocio

4.1 Elección de un lenguaje para el modelo de procesos de negocio

Dentro de ProM podemos distinguir entre patrones que un lenguaje (process tree, petri nets, bpml, yawl,...) puede representar, y patrones que un algoritmo de descubrimiento de minería de procesos implementado en ProM, es capaz de descubrir.

Nuestro plugin ProM de conversión es en esencia un algoritmo de descubrimiento de workflows. Este plugin a partir de un modelo de proceso de negocio, extraerá los workflows que descubra del modelo para adaptarlos a workflows que use Psighos.

La manera de representar workflows por parte de Psighos se rige por un lenguaje con una serie de reglas poco estrictas, siguiendo una metodología de declaración de patrones modular y no estructurada.

Para encontrar un lenguaje que pueda aprovechar la potencia del lenguaje de Psighos, según Discovering Block-Structured Process Models From Event Logs Containing Infrequent Behaviour[12], la mejor solución sería el uso de **petri nets** o **redes de petri**, pero el proceso de descubrimiento de patrones en las redes de petri no es algo trivial.

En lugar de ello, teniendo en cuenta como Psighos representa sus patrones se ha optado por utilizar **Process Tree**, un lenguaje cuya definición se acerca a la filosofía que usa la librería de simulación de eventos discretos y que ofrece opciones interesantes de

modelado a través de un generador/editor aleatorio de código.

También podemos utilizar un algoritmo de minería de datos para este este lenguaje, llamado IMi(**Inductive Miner Infrequent**-Minero Inductivo Infrecuente), que teniendo en cuenta el anterior documento, donde lo analiza y lo compara con otros algoritmos de su mismo ámbito, en base a la precisión, robustez, generalización y su simplicidad, nos provee de un buen modelo a partir de un log.

En conclusión, se puede realizar una conversión a workflows Psighos usando una estrategia de traducción con patrones de tipo estructurado (capítulo 1.1). Los modelos que el lenguaje (ProcessTree) puede representar, son un subconjunto de los lenguajes que podría interpretar Psighos para ejecutar sus simulaciones.

4.1.1 Process Tree

Es un modelo de procesos de bloques-estructurado muy utilizado en ProM, que tiene numerosos conversores a todo tipo de lenguajes, así como, algoritmos de descubrimiento que son capaces de crearlo, entre el que destaca como ya nombramos anteriormente IMi. Además de ello existe un generador/editor para este lenguaje que crea modelos aleatorios y editables.

Está compuesto por tres partes:

- **Perspectiva de control de flujo**

Describe un una estructura de bloques usando grafos dirigidos sin ciclos. Usando nodos y vértices, donde estos últimos representan las relaciones jerárquicas de los anteriores.

Los nodos pueden ser bloques o tareas.

  - Las tareas pueden ser automáticas (sin originador) o manuales (con originador).
  - Los bloques describen relaciones causales entre sus hijos. Todos los bloques deben tener al menos 1 hijo, salvo los bloques que representan iteradores que tienen exactamente 3. Conoceremos la forma de estos bloques y su significado, más adelante en el apartado 5.1

- **Perspectiva de datos**

En esta perspectiva se definen las variables y las expresiones. Las variables puede ser leídas o escritas por los nodos y son usadas en expresiones.

“Una expresión es una formula lógica que usa variables descrita como un string”. Están presentes en los vértices salientes de un bloque Xor, Or y en el segundo y tercer vértice del bloque LoopXor.

- **Perspectiva de recursos**

Hay de 3 tipos recursos, roles y grupos. Además existen posibilidades de configuración, que te permite modificar el comportamiento de cada una de las perspectivas, ocultando o bloqueando vértices, seleccionando que variables o expresiones pueden ser leídas o no, así como, decidir que originadores son válidos para ejecutar una tarea manual en concreto.

Todas estas posibilidades nos serán de utilidad más adelante para forzar lógicamente, el descubrimiento de patrones, en el caso de que la traducción no sea directa.

En el Apéndice B[13] se puede ver la representación gráfica de estas perspectivas, con los elementos que los componen.

Capítulo 5
Implementación del Plug-in

5.1 Adaptando Nodos Process Tree a Workflows Psighos

Para elaborar esta adaptación[14,15,16,17] necesitamos dos materias primas principales. Una de ellas es el propio Process Tree, véase figura 5.2, que queremos traducir a flujos de trabajo usados por Psighos y la otra (figura 5.1) supone un SingleFlow fuente o source de donde empezar a construir la traducción y un SingleFlow final o sink para tener una referencia a la hora de cerrar la conversión adecuadamente, siguiendo un modelo de workflows estructurada. Estos nodos de ayuda source y sink irán cambiando de situación según se vayan recorriendo los nodos dependiendo de la llamada recursiva.

A partir de ahora tomaremos como base estas dos materias prima para explicar la conversión de cada casos adaptados a partir de nodos ProcessTree, los cuáles se pueden transformar en workflows válidos en lenguaje Psighos.

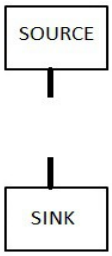


Figura 5.1 SingleFlow Source y Sink

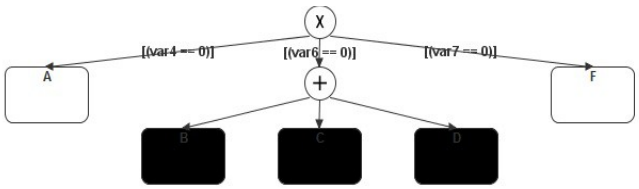


Figura 5.2 Ejemplo de ProcessTree sencillo

5.1.1 Convertir Tarea automática

Figura 5.3 Nodo tarea automática

Un nodo tarea automática es un nodo hoja del árbol de procesos, que contiene información sobre una tarea en concreto. En nuestro desarrollo, para nosotros este nodo hoja es un SingleFlow, ya que estos patrones Psighos envuelven una actividad. Al ser una actividad automática la consideraremos como una actividad no presencial añadiéndole un modificador a la actividad correspondiente.

Por lo tanto, para transformar una tarea automática debemos:

1. En primera instancia reconocer que el nodo en el que nos encontramos en el ProcessTree, es un nodo de tarea automática.
2. Creamos una actividad no presencial psighos.
3. Creamos un SingleFlow envolviendo esa tarea.
4. Por último conectamos el SingleFlow source con nuestro nuevo SingleFlow generado, y este último se conecta con el SingleFlow sink.

Este proceso estaría descrito gráficamente en la figura 5.4, donde la caja SingleFlow generado TA correspondería una actividad no presencial , transformada de un nodo tarea automática de un árbol de procesos.

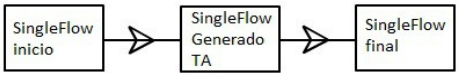


Figura 5.4 Resultado de la transformación de una tarea automática

5.1.2 Convertir Tarea manual

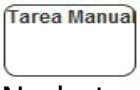


Figura 5.5 Nodo tarea manual

Un nodo tarea manual al igual que la tarea automática es un nodo

hoja del árbol de procesos, que contiene información sobre una tarea en concreto. En nuestro desarrollo, para nosotros este nodo hoja es un SingleFlow, ya que estos patrones Psighos envuelven una actividad. Al ser una actividad manual la consideraremos como una actividad presencial dirigida en el tiempo.

Por lo tanto, para transformar una tarea manual debemos repetir el mismo proceso que en la tarea automática, únicamente cambiando el paso 2 por crear una actividad presencial dirigida en el tiempo Psighos, en vez de una no presencial.

De igual manera el resultado sería el mismo patrón de flujo descrito en la figura 5.6, donde la caja SingleFlow generado TM, correspondería con una actividad presencial dirigida en el tiempo, transformada a partir de un nodo tarea manual del árbol de procesos.

Figura 5.6 Resultado de la transformación Tarea Manual

5.1.3 Convertir Xor

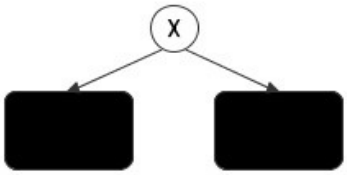


Figura 5.7 Nodo Xor

Un nodo xor, véase figura 5.7, es una operación de los árboles de proceso que siguen un patrón WP4 o Exclusive Choice/Elección exclusiva. Homólogamente, en Psighos dicho patrón está soportado, por ello podemos hacer una traducción casi directa entre ambas representaciones.

Durante la transformación se hará uso de dos patrones, tales como WP4 exclusive choice o xor-split y WP5 simple merge o xor-join, representados respectivamente por las figuras 5.8 y 5.9.

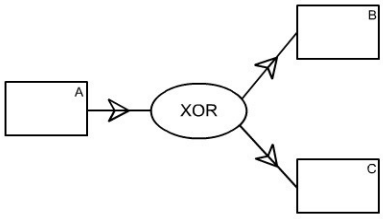


Figura 5.8 WCP4 ExclusiveChoice

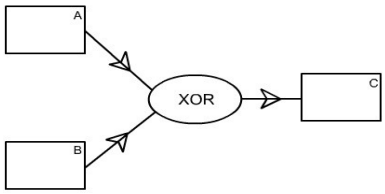


Figura 5.9 WCP5 SimpleMerge

Para realizar la transformación se siguen los siguientes pasos:

1. Se crea un patrón WP4 y WP5.
2. Se une el SingleFlow “source” de la figura 5.1 con nuestro patrón WP4 y el patrón WP5 con el SingleFlow “sink” de la misma figura.
3. Se recorre los nodos hijos del Nodo xor, empezando un nuevo problema de conversión según el tipo nodo.

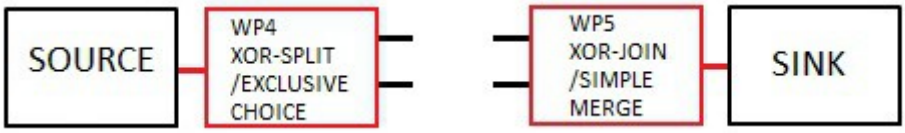


Figura 5.10 Resultado de la transformación de un nodo Xor

El resultado será un flujo de trabajo estructurado, como el de la figura 5.10, donde posteriormente serán embebidas otras estructuras cualquiera según se vayan analizando los nodos hijos.

Caso particular del Xor

Cada vértice que alcanza un nodo xor provee información sobre la probabilidad de 0->1 de alcanzar ese nodo, es decir, que el patrón Probability Selection Flow, también es posible de implementar a partir de este nodo, mediante lógica con variables.

5.1.4 Convertir Seq

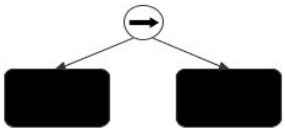


Figura 5.11 Nodo Secuencial

El nodo Seq es el más fácil de transformar ya que simplemente nos indica que una tarea debe ir después de otra, como el patrón

WP1 sequence o secuencia. Todas nuestras transformaciones unen el nodo SingleFlow source con el nodo descubierto y a su vez éste se une con el SingleFlow sink. Por lo tanto, se debe seguir iterando el árbol en busca de posibles nodos y los resultados de sus hijos se ejecutarán secuencialmente.

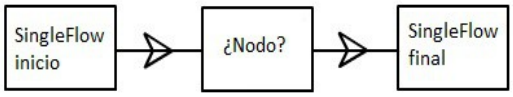


Figura 5.12 Resultado de la conversión de un nodo Seq

5.1.5 Convertir And

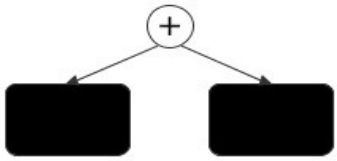


Figura 5.13 Nodo And

Un nodo And, véase figura 5.13, es una operación de los árboles de proceso que siguen un patrón WP2 o Parallel Split. Homólogamente, en Psighos dicho patrón está soportado, por ello podemos hacer una traducción casi directa entre ambas representaciones.

Para transformar este operador ProcessTree, usaremos el ya citado WP2 Parallel Split y patrón WP3 Synchronization, que corresponden respectivamente a un and-split (véase figura 5.14) y un and-join (véase figura 5.15).

La manera estructurada en la que para el patrón Parallel Split le corresponde un patrón WP3, puede interpretarse también como un patrón WP17 InterleavedRoutingFlow.

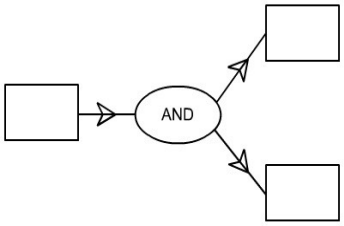


Figura 5.14 WCP2 ParallelSplit

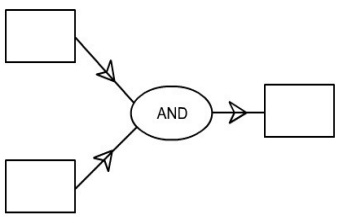


Figura 5.15: WCP3 Synchronization

Para realizar la transformación se sigue el mismo esquema que el nodo Xor citado anteriormente:

1. Se crea un patrón WP2 y WP3.

1. Se une el SingleFlow “source” de la figura 5.1 con nuestro patrón WP2 y el patrón WP3 con el SingleFlow “sink” de la misma figura.

2. Se recorre los nodos hijos del Nodo And, empezando un nuevo problema de conversión según el tipo de nodo.

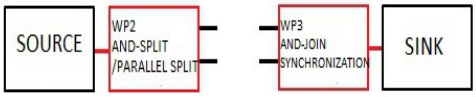


Figura 5.16: Resultado de la transformación de un nodo And

El resultado de nuevo como en Xor será un flujo de trabajo estructurado, como el de la figura 5.16, donde posteriormente serán embebidas otras estructuras cualquiera dentro de las mismas según se vayan analizando los nodos hijos.

Caso particular del And

Los vértices de los nodos puede bloquearse o ocultarse, esto permite la posibilidad de implementar patrones de bloqueo. Añadiendo la posibilidad de controlando esta variable proporcionada por los árboles de proceso añadir un caso particular del patrón Synchronization, el patrón WP9 Discriminator Blocking pudiendo a alguno de los predecesores del And-join (Synchronization) la posibilidad de bloquear uno los nodos que llegan a él, aumentando la potencia de la traducción.

5.1.6 Convertir Or

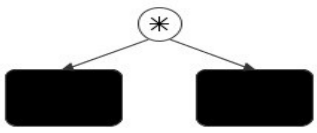


Figura 5.17 Nodo OR

Un nodo Or, véase figura 5.17, es una operación de los árboles de proceso que siguen un patrón WP6 o Multi Choice. Homólogamente, en Psighos dicho patrón está soportado, por ello podemos hacer una traducción casi directa entre ambas representaciones.

La conversión sigue fiel al mismo esquema de transformación descrito en Xor y And, es este caso se usarán los patrones WP6 Multi-Choice y el patrón WP8 Multi-merge representados por las figuras 5.18 y 5.19 respectivamente.

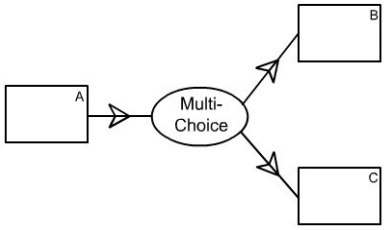


Figura 5.18: WCP6 Multi-Choice

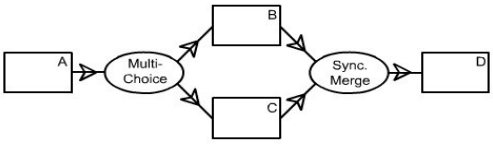


Figura 5.19: WCP8 Multi-Merge

Para realizar la transformación seguimos el esquema de los operadores Xor y And, de la siguiente forma:

1. Se crea un patrón WP6 y WP8.
2. Se une el SingleFlow “source” de la figura 5.1 con nuestro patrón WP6 y el patrón WP8 con el SingleFlow “sink” de la misma figura.
3. Se recorre los nodos hijos del Nodo Or, empezando un nuevo problema de conversión según el tipo de nodo.

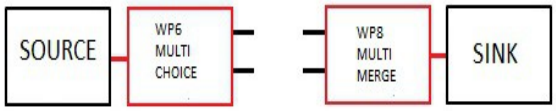


Figura 5.20: Resultado de la conversión de un nodo Or

El resultado una vez más será un flujo de trabajo estructurado, como el de la figura 5.20, donde posteriormente serán embebidas

otras estructuras cualquiera dentro de las mismas según se vayan analizando los nodos hijos.

Caso particular del Or

Como realizamos una traducción estructurada y en el caso del Or su homólogo es un patrón Multi-Merge para unir a todos los hijos correspondientes, existe la posibilidad de que mediante lógica siempre y cuando se use el editor de árboles de proceso o una posible interfaz, de cambiar el patrón de unión Multi-Merge por un patrón Synchronization y así crear un nuevo patrón WCP7 StructuredSynchroMergeFlow, infiriendo una vez más por lógica la apareción de un patrón.

5.1.7 Convertir XorLoop

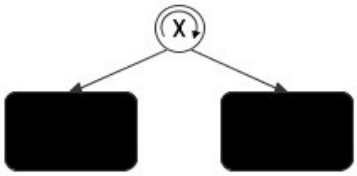


Figura 5.21 Nodo XorLoop

Un nodo XorLoop, véase figura 5.21, es la operación más compleja de los árboles de proceso que sigue un patrón WP21 o Structured Loop. Homólogamente, en Psighos dicho patrón está soportado, en tres variantes, figuras 5.22, 5.23 y 5.24.

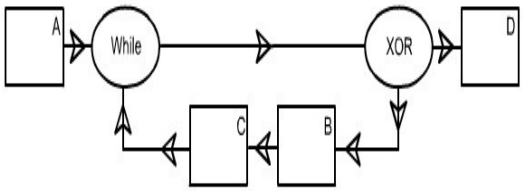


Figura 5.23 WCP21
StructuredLoop PreTest

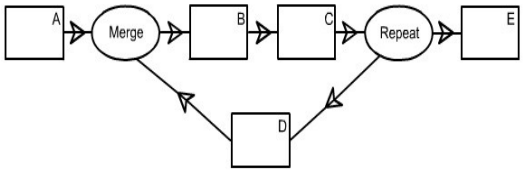


Figura 5.22 WCP21
StructuredLoop PostTest

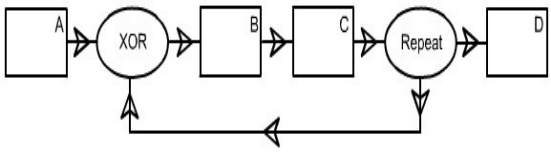


Figura 5.24 WCP21
StructuredLoop N-iterations

En un árbol de procesos un nodo XorLoop siempre tiene 3 nodos hijos que representan:

- Do(Cuerpo), Redo(Iteración) y Exit(Salida).

Para transformar este nodo elaboraremos la siguiente estrategia, primero de todo debemos unir el source con un patrón de unión.

Basándonos en las anteriores transformaciones, donde conectábamos de alguna manera los nodos con el SingleFlow source y el SingleFlow sink, esta vez, haremos lo siguiente:

- Debemos empezar la transformación con el primer hijo, asegurándonos de llamar recursivamente a ese hijo usando (do):
  - Source: Patrón Simple Merge
  - Sink: Patrón Structured Loop (Post-test)

- En segundo hijo (redo):
  - Source deberá ser: Patrón Structured Loop (Post-test) para embeber la estructura iterativa dentro de él.
  - Sink: debe ser de nuevo el patrón de unión.
- El tercer hijo (exit):
  - Source: Patrón Structured Loop (Post-test) exit, donde
  - Sink: El sink no debe cambiarse, se usa el valor que tiene cuando se descubre el nodo XoorLoop.

Jugando con la lógica podríamos establecer el patrón en diferentes lugares extendiendo el significado de la transformación a la variante del mismo patrón Structured Loop (Post-test) o Loop (número de veces determinado), usando el editor de árbol de procesos o alguna interfaz gráfica que se desarrolle posteriormente, para cambiar la lógica de la transformación.

5.2 Diagrama de clases de la implementación.

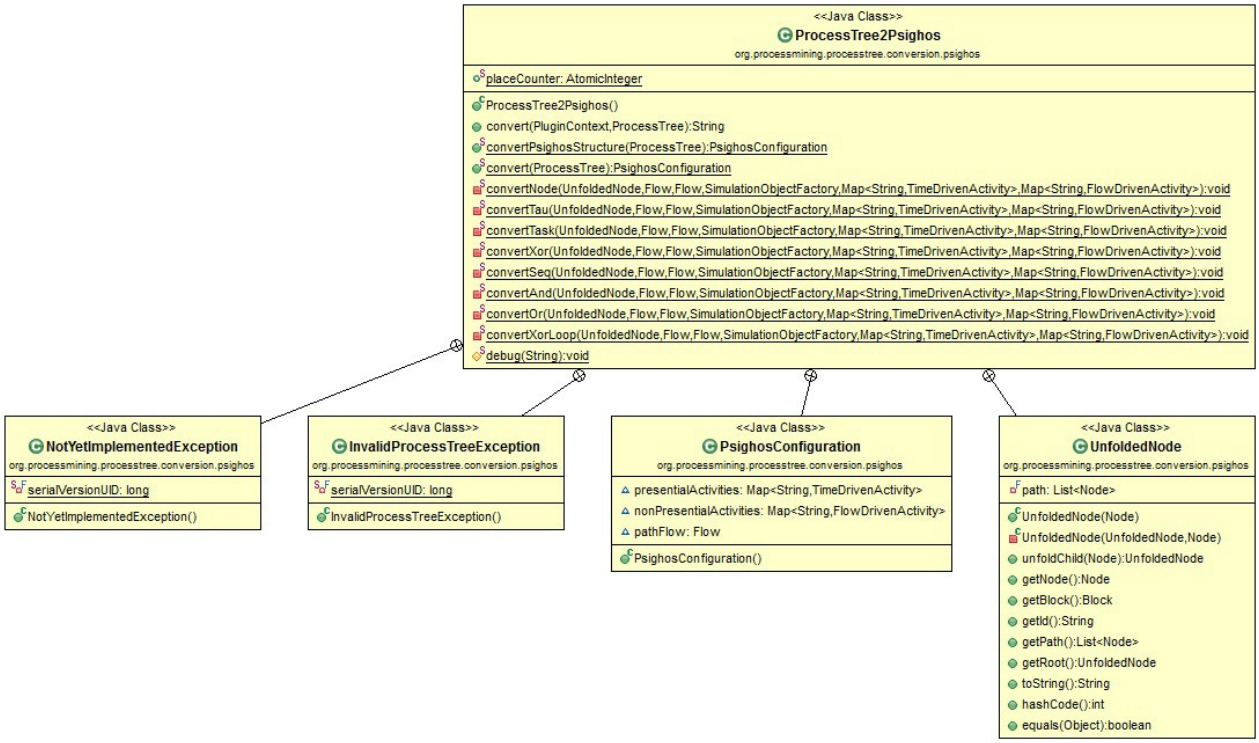


Figura 5.25 Diagrama de clases de la implementación

La figura 5.25 representa el diagrama de clases del conversor. Se ha usado una clase UnfoldedNode que es la encargada de ir leyendo el camino del Process Tree de manera recursiva siguiendo una búsqueda en profundidad. Esta búsqueda corresponde con un algoritmo basado en la estructura LIFO (last in first out) que busca recorrer los nodos de un grafo por medio de back tracking, es decir, va localizando los recorridos posibles y en el caso de no poder continuar, vuelve al punto donde existen nuevos recorridos posibles, con el fin de visitar todos los nodos.

Además siempre guardamos un Flow inicio y final, para tener la referencia de donde empieza y acaba un patrón (perspectiva estructurada).

Habiendo recorrido todos los nodos obtenemos un workflow en clases Psighos que es insertado junto con otros elementos en una simulación.

Capítulo 6
Conclusiones y líneas futuras

En este proyecto se ha elegido una herramienta de minería de procesos (ProM), así como, una herramienta/librería de simulación de eventos discretos (Psighos), con el objetivo de elaborar un desarrollo que facilite el acceso a las organizaciones, para simular o acceder a las simulaciones de sus procesos de negocio.

Para ello, se ha elegido un lenguaje común a las herramientas anteriormente citadas, (Process Tree), para crear un plug-in de conversión hecho en ProM que a partir de la extracción semi-automática de los modelos utilizando técnicas de minería de procesos, es capaz de hacer un uso combinado de minería de procesos y simulación de eventos discretos para realizar una simulación.

Comprobando posteriormente, que el lenguaje usado puede representar un subconjunto de los lenguajes que podría interpretar Psighos para ejecutar sus simulaciones.

En líneas futuras se pretende formalizar con mayor detalle los workflow patterns inferidos mediante lógica programada. Por otra parte se hace necesario en futuros proyectos o desarrollos, la creación de una interfaz gráfica que permita definir todos los elementos en tiempo de ejecución, que no depende de workflows, pero que son parte de los elementos necesarios para ejecutar una simulación.

Capítulo 7
Summary and Conclusions

In this project has been chosen a process mining tool (ProM) and a tool/library of discrete event simulation (Psighos), with the aim of elaborate a development that facilitates the access of organizations, to simulate or access to their simulations of business processes.

To do this, it has been chosen a common language to the aforementioned tools (Process Tree), to create a plug-in conversion done in ProM as of the semi-automatic extraction of the models using process mining techniques, it is able to use a combination between mining processes and simulation of discrete event simulation to perform simulation.

Concluding later that the language used can represent a subset of the languages that could be interpreted Psighos to run their simulations.

In future lines the intention is to formalize in greater detail the workflow patterns inferred by programmed logic. Moreover it is necessary in future projects and developments, the creation of a graphic interface to define all the elements at runtime, which does not depend on workflows, but are part of the elements necessary to run a simulation.

Capítulo 8
Presupuesto

El proyecto se ha desarrollado a coste de software cero ya que en todo momento hemos usado herramientas cuya licencia es de código abierto, por lo tanto el único gasto es el de personal y equipos.

Sofware usado:

- ProM 6.5.1
- Psighos
- JDK 1.7.0_79
- Bitbucket, repositorio privado (máximo 5 personas)
- Redmine
- Eclipse Luna 4.4.2 (eGit, ObjectAid)

Items	Cantidad	Coste
Software	6	0,00 €
PC	1	39,58 €
Personal	1	1.434,00 €
Total	10	1.473,58 €

Tabla 8.1 Resumen del presupuesto

8.1 Justificación del presupuesto

El proyecto se ha realizado en un total de 60 días, con una media aproximada de unas 3.5h/día. Tomando como ejemplo un contrato a 40 horas semanales con un sueldo neto de 1150 euros/mes. El coste de la hora sería 7.18€ por debajo del sueldo medio por hora que ganaría un programador junior java. Para calcular el precio de amortización de los equipos, se tiene en cuenta que un equipo informático dura aproximadamente 4 años. Considerando el precio del equipo usado 950€, la amortización a razón de 2 meses de trabajo sería, (950*2) / 48, lo que supone 39.58€ en amortización de equipos. Por tanto, como muestra la tabla 8.1 el precio total del desarrollo son 1473,58€ (impuestos no incluidos).

Apéndice A: Workflows soportados por Psighos y el plugin creado en ProM

Familia de patrones	Patrón	Psighos	Plugin ProcessTreeTo Psighos
Flujo de Control Básico	WCP1 Sequence	Sí	Sí
	WCP2 Parallel Split	Sí	Sí
	WCP3 Synchronisation	Sí	Sí
	WCP4 Exclusive Choice	Sí	Sí
	WCP5 Simple Merge	Sí	Sí
Sincronización y Ramificación Avanzada	WCP6 Multi-Choice	Sí	Sí
	WCP7 Structured Synchronising Merge	Sí	Sí
	WCP8 Multi-Merge	Sí	Sí
	WCP9 Structured Discriminator	Sí	Sí
	WCP28 Blocking Discriminator	Sí	Sí*
	WCP29 Cancelling Discriminator	No	No
	WCP30 Structured Partial Join	Sí	Sí*

	WCP31 Blocking Partial Join	Sí	Sí*
	WCP32 Cancelling Partial Join	No	No
	WCP33 Generalised AND-Join	Sí	Sí*
	WCP37 Local Synchronising Merge	Sí	No
	WCP38 General Synchronising Merge	No	No
	WCP41 Thread Merge	Sí	No
	WCP42 Thread Split	Sí	No
Multiple-Instancia	WCP12 Multiple Instances without Synchronization	Sí	No
	WCP13 Multiple Instances with a Priori Design-Time Knowledge	Sí	No
	WCP14 Multiple Instances with a Priori Run-Time Knowledge	No	No
	WCP15 Multiple Instances without a Priori Run-Time Knowledge	No	No
	WCP34 Static Partial Join for Multiple Instances	Sí	Sí*
	WCP35 Cancelling Partial Join for Multiple Instances	No	No
	WCP36 Dynamic Partial Join for Multiple Instances	No	No
Basado-Estados	WCP16 Deferred Choice	No	Sí
	WCP17 Interleaved	Sí	Sí*

	Parallel Routing		
	WCP18 Milestone	No	No
	WCP39 Critical Section	No	No
	WCP40 Interleaved Routing	Sí	Sí*
Cancelación y forzar el completado	WCP19 Cancel Task	No	No
	WCP20 Cancel Case	No	No
	WCP25 Cancel region	No	No
	WCP26 Cancel Multiple Instance Activity	No	No
	WCP27 Complete Multiple Instance Activity	No	No
Iteración	WCP10 Arbitrary cycles	Sí	Sí*
	WCP21 Structure Loop	Sí	Sí
	WCP22 Recursion	No	No
Terminación	WCP11 Implicit Termination	Sí	No
	WCP43 Explicit Termination	No	No
Disparador	WCP23 Trasient Trigger	No	No
	WCP24 Persistent Trigger	No	No

* Se puede inferir el patrón mediante lógica de programación

Tabla 1 Workflow Patterns soportados por Psighos/ProcessTreeToPsighos.

Apéndice B: Esquema de representación de un Process Tree

Figura 1 Meta-Modelo ProcessTree. Rojo perspectiva de control de flujos. Azul perspectiva de datos. Verde perspectiva de recursos.

Bibliografía

[1] Wikipedia. *Discrete event Simulation.* Disponible en:

https://en.wikipedia.org/wiki/Discrete_event_simulation

[2] Iván Castilla 2010/11 *Explotación de los sistemas multi-núcleo para la simulación paralela de eventos discretos con Java.*

[3] IEEE Task Force on Process Mining - *Manifiesto sobre Minería de procesos*. Disponible en:

http://www.win.tue.nl/ieeetfpm/lib/exe/fetch.php?media=shared:pmm-spanish-v1.pdf

[4]Kawtar Benghazi, José Luis Garrido Bullejos, Manuel Noguera García. *Introducción al Modelado de Procesos de Negocio.* Disponible en:

http://www.ugr.es/~mnoguera/collaborative_systems-business_processes_10-11.pdf

[5] Wil van der Aalst 2011, *Process Mining: Discovery, Conformance and Enhancement of Business Processes*

[6] Wikipedia. *Minería de Procesos.* Disponible en:

https://es.wikipedia.org/wiki/Miner%C3%ADa_de_procesos

[7] Marlon Dumas. *Minería de procesos y de reglas de negocio.* Disponible en:

http://es.slideshare.net/MarlonDumas/minera-de-procesos-y-de-reglas-de-negocio

[8] Prom6 Official Page, Process Mining Group, Eindhoven Technical University. © 2010. Disponible en:

http://www.promtools.org/doku.php

[9] Mercedes Granda. *Redes de Petri – Definición, formalización y ejecución.* Disponible en:

http://www.ctr.unican.es/asignaturas/mc_procon/Doc/PETRI_1.pdf

[10] S.J.J. Leemans, D. Fahland, y W.M.P. van der Aalst Discovering *Block-Structured Process Models From Event Logs - A Constructive Approach.* Disponible en:

http://wwwis.win.tue.nl/~wvdaalst/publications/p719.pdf

[11] Wil van der Aalst, *Control-Flow Patterns.* Disponible en :

http://www.workflowpatterns.com/patterns/control/

[12] Sander J.J. Leemans, Dirk Fahland, y Wil M.P. van der Aalst *Discovering Block-Structured Process Models From Event Logs Containing Infrequent Behaviour*. Disponible en:

http://www.win.tue.nl/~dfahland/publications/LeemansFA_2013_bpi.pdf

[13] ProM. *Documentation Process Tree 2014.* Disponible en:

https://svn.win.tue.nl/trac/prom/browser/Documentation/ProcessTree/DocumentationProcessTree2014.pdf

[14] ProM. *Wiki for ProM developer.* Disponible en:

https://svn.win.tue.nl/trac/prom/wiki

[15] ProM. *Plugins documentation.* Disponible en*:*

https://svn.win.tue.nl/trac/prom/browser/Documentation

[16] ProM. *Repository of all ProM plugins.* Disponible en:

https://svn.win.tue.nl/repos/prom/Packages/

[17] Michael Westergaard. *ProM6 plugin development.* Disponible en:

https://westergaard.eu/category/tutorials/prom-tutorial/