Inteligencia Empresarial combinando técnicas de Minería de Procesos y Minería de Datos.

Business Intelligence by mixing Process Mining and Data Mining techniques.

Maurizio Alejandro Rendon Mattogno.
Departamento de Ingeniería Informática
Escuela Superior de Ingeniería y Tecnología
Trabajo de Fin de Grado

La Laguna, 05 de septiembre de 2014

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.

D. **Vanesa Muñoz Cruz**, con N.I.F. 78698687-R profesora Ayudante Doctor adscrita al Departamento de Ingeniería Informática de la Universidad de La Laguna.

C E R T I F I C AN

Que la presente memoria titulada:

"Inteligencia Empresarial combinando técnicas de Minería de Procesos y Minería de Datos."

ha sido realizada bajo su dirección por D. Maurizio Alejandro Rendon Mattogno, con N.I.F. Y.016.0512-D.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 5 de septiembre de 2014

Agradecimientos

Gracias a mis tutores Pedro Toledo y Vanesa Muñoz por guiarme en este proceso. Agradezco a mis compañeros de universidad por los momentos compartidos y por último pero no menos importante, quiero darle las gracias a mi familia, que son mi fuerza de voluntad y motivación.

Resumen

En el proyecto se pretende desarrollar una herramienta que integre las técnicas, de dos campos de la ciencia de la computación que están involucrados en la inteligencia empresarial, que son: la minería de datos y la minería de procesos. Ambos campos están dedicados a la extracción de información, pero la primera suele actuar sobre las bases de datos y la segunda en archivos de registros mejor conocidos como logs.

Para realizar esta tarea se combinarán dos modelos de cada campo de forma que se complemente el uno al otro. El modelo General Sequential Patterns (en adelante GSP) de Weka que descubre las secuencias más frecuentes de una base de datos de secuencias. Y el modelo de Petrinet de ProM que genera un workflow del proceso que es deducido de un fichero log. Ambos modelos tendrán como fuentes de información el mismo fichero log, pero para que Weka pueda trabajar con él, se deberá realizar una conversión previa del log a una base de datos de secuencias con una herramienta que facilita Weka.

El resultado será un modelo compuesto por el grafo del proceso y una lista de caminos del grafo más frecuentes. Este modelo facilita la detección de cuellos de botella, el rediseño del modelo de trabajo para mejorar su eficiencia, o añadir información extra al flujo de proceso como los costes o las personas involucradas en las actividades más frecuentes.

Palabras clave

Minería de Datos, Minería de Procesos, Inteligencia de Negocios (BI), ProM, Weka, General Sequential Patterns (GSP) y Kanban eXtream Programming (KXP).

Abstract

The project aims to develop a tool that integrates techniques from two different fields of computer science who are involved in business intelligence. They are the data mining and mining processes. Both camps are dedicated to the extraction of information, but the first works on database and the second on logs.

Two models of each field will combined so they complement each other. The Weka General Model Sequential Patterns (GSP) discovers the most frequent sequences from a sequence database. And the PetriNet ProM model generates a workflow process from a log file. Both models works on the same log file, but for Weka model it should be performed prior to a conversion from log to sequence database with a tool that Weka facilitates.

The result is a model made by the process graph and a list of the most frequent paths graph. This model facilitates the detection of bottlenecks, the workflow redesign to improve efficiency, add extra information into the workflow, or showing the costs involved in the most frequent activities.

Keywords

Data Mining, Process Mining, Business Intelligence (BI), ProM, Weka, General Sequential Patterns (GSP), and Kanban eXtream Programming (KXP).

Índice general

Contenido

Índice de Figuras

Índice de Tablas

CAPÍTULO 1

1. INTRODUCCIÓN

Desde los inicios de la informática, se han desarrollado técnicas que faciliten el procesamiento de datos de forma automatizada, pero el tiempo, el internet, las redes sociales y los avances en los sistemas informáticos, especialmente en la capacidad y velocidad de las unidades de almacenamiento han generado nuevas oportunidades.

Las grandes cantidades de datos que se encuentran alojados en la actualidad en todo tipo de dispositivos electrónicos, hacen que los mismos pasen de ser un producto, a convertirse en una materia prima, que hay que explotar para obtener el verdadero activo final elaborado; el conocimiento. Hoy más que nunca la información es poder; por esta razón las empresas, los gobiernos y las instituciones invierten ingentes cantidades de dinero en las famosas tecnologías de la información, las cuales van de la mano de diversos campos cómo el de la inteligencia empresarial, la minería de datos o la minería de procesos. **[1, 2]**

La **minería de datos** es un proceso de estudio automático o semi-automático de grandes cantidades de datos. Utiliza el análisis matemático, la estadística y los algoritmos de búsquedas próximos a la inteligencia artificial para deducir los patrones y tendencias que se encuentran ocultos en los datos. Su objetivo final es extraer el conocimiento para facilitar la toma de decisiones. A continuación se mencionan algunos modelos de minería de datos y se citan algunos ejemplos concretos para los que son habitualmente utilizados **[3,4]**:

- **Pronóstico**: cálculo de las ventas y predicción de las cargas del servidor o del tiempo de inactividad del servidor.
- **Riesgo y probabilidad**: elección de los mejores clientes para la distribución de correo directo, determinación del punto de equilibrio probable para los escenarios de riesgo, y asignación de probabilidades a diagnósticos y otros resultados.
- **Recomendaciones**: determinación de los productos que se pueden vender juntos y generación de recomendaciones.
- **Búsqueda de secuencias**: análisis de los artículos que los clientes han introducido en el carrito de la compra y predicción de posibles eventos.
- **Agrupación**: distribución de clientes o eventos en grupos de elementos relacionados, y análisis y predicción de afinidades.

La **minería de procesos** es una disciplina de investigación relativamente joven que ha surgido a partir del modelado de datos en el entorno empresarial, con el objetivo de facilitar la labor de extracción de modelos. Hay dos razones principales para el creciente interés en minería de procesos. Por un lado, se registran más y más eventos, proporcionando información detallada acerca de la historia de los procesos. Por otro lado, hay una necesidad de mejorar y apoyar los procesos de negocio en ambientes competitivos y que cambian rápidamente. La minería de procesos incluye:

- El descubrimiento automático de procesos. Por ejemplo: extraer modelos de procesos a partir de un registro de eventos.
- La verificación de conformidad, como monitorear desviaciones al comparar el modelo y el registro de eventos.
- La minería de redes sociales/organizacionales.
- La construcción automática de modelos de simulación, la extensión de modelos y la reparación de modelos.
- La predicción de casos, y las recomendaciones basadas en historia.

Resumiendo, la minería de procesos es una técnica de administración de procesos que permite analizar los procesos de negocio de acuerdo con un registro de eventos o log, para descubrir, monitorear y mejorar los mismos. **[5-7]**

En la actualidad, existen diferentes herramientas libres y comerciales para la minería de datos, como: Weka, Rapid Miner, Powerhouse y SAS. Otras tantas para la minería de procesos, entre las cuales, podemos destacar: ProM, Process Mining, Disco, ARIS Process Performance Manager. Sin embargo, no hay ninguna herramienta que explote ambas técnicas sobre los mismos datos y que ofrezcan por ejemplo: flujos de trabajos enriquecidos con la estructura de la información que está detrás de los procesos que intervienen en los workflows, o la utilización de los registros de eventos para compararlos con modelos probabilísticos.

1.1. ANTECEDENTES

ProM es un framework extensible que soporta una variedad de técnicas de minería de procesos en forma de plugins. ProM 6 es distribuido en partes, pata ofrecer una mayor flexibilidad. Por una parte está el paquete ProM core con licencia GPL y por el otro están los paquetes ProM plugins que están distribuidos normalmente bajo licencia L-GPL. Actualmente están disponibles más de 120 paquetes que contienen más de 500 plugins. **[7]**

Breve Historia **[8]**:

- En 2002, existían una gran variedad de herramientas básicas de minería de procesos. Todas ellas incapaces de procesar datasets de gran tamaño y sólo ejecutaban un determinado

algoritmo. Evidentemente, no tiene mucho sentido desarrollar una herramienta, por algoritmo. Por este motivo, se crea el framework extensible ProM. El objetivo de la primera versión era proveer un soporte común de carga, y filtrado de los eventos de los logs y la visualización de resultados.

- En 2004, se lanza ProM 1.1 la primera versión completa y funcional, con un total de 29 plug-ins; 6 de minería, 7 de análisis, 9 de exportación, 4 de importación y 3 de conversión.
- En 2006, se libera la versión 4.0 con 142 plug-ins.
- En 2009, ProM 5.2 con 286 plugins, Claramente ProM se convierte en un estándar de la minería de procesos, con grupos de investigación repartidos por el mundo, contribuyendo a su desarrollo.
- En 2010, nace ProM 6; con una interfaz de usuario reimplementada para lidiar con varios plugins, logs y modelos al mismo tiempo. Además, incluye un administrador de paquetes para poder añadir, eliminar y actualizar los plug-ins. De esta forma se evita tener instaladas las funcionalidades que no necesites.

Weka (Waikato Environment for Knowledge Analysis - Entorno para Análisis del Conocimiento de la Universidad de Waikato). Es una plataforma de software libre distribuido bajo licencia GNU-GPL, para aprendizaje automático y minería de datos escrito en Java y desarrollado en la Universidad de Waikato. **[3, 9]**

Breve Historia:

- En 1993, la Universidad de Waikato de Nueva Zelanda inició el desarrollo de la versión original de Weka (en TCL/TK y C).
- En 1997, se decidió reescribir el código en Java incluyendo implementaciones de algoritmos.
- En 2005, Weka recibe de SIGKDD (Special Interest Group on Knowledge Discovery and Data Mining) el galardón "Data Mining and Knowledge Discovery Service".
- En 2006, Pentaho Corporation adquirió una licencia exclusiva para usar Weka para Inteligencia de negocio (Business Intelligence), dando lugar al componente de minería de datos y análisis predictivo del paquete de software Pentaho Business Intelligence.

1.2. OBJETIVO GENERAL

Desarrollar una herramienta que integre las técnicas de la minería de datos y la minería de procesos.

1.3. OBJETIVOS ESPECÍFICOS

- Construir un prototipo de plug-in Weka para la plataforma ProM.
- Analizar y seleccionar los modelos para la interpretación de la información.

- Diseñar e implementar la aproximación analizada dentro del prototipo.
- Realizar un proceso de validación con datos reales públicos o sintéticos.

1.4. ALCANCE

Se pretende crear un prototipo de plug-in de Weka para la plataforma ProM. En concreto se utilizará de la librería de asociaciones de Weka el Algoritmo Generalized Sequential Patterns (de ahora en adelante, GSP) y el workflow de ProM Petrinet; para mostrar las secuencias de procesos más frecuentes de un flujo de trabajo.

El plug-in debe generar una red de Petri con un listado de los patrones de secuencias. Opcionalmente se plantea la posibilidad de que al clicar sobre una secuencia, se marque el camino correspondiente en el petrinet.

1.5. PLANIFICACIÓN

Semana	Actividades	Puntos
Fase 1 - Investigación.		***2***
21/04 – 27/04	Familiarización con las herramientas ProM y Weka.	0.5
28/04 – 04/05	Investigación sobre el desarrollo de plug-ins para ProM.	0.5
05/05 – 11/05	Desarrollo de plugins de prueba.	1
Fase 2 - Análisis del problema de interpretación de la información.		***3***
12/05 – 18/05	Selección de los algoritmos y modelo de workflow a utilizar.	1
19/05 – 25/05	Estudio de funcionalidad e integración.	1
26/05 – 01/06	Diseño y estructura del plugin.	1
Fase 3 - Diseño y desarrollo de la aproximación analizada		***3***
02/06 – 15/06	Implementación de los modelos de datos y plug-ins.	1
16/06 – 29/06	Desarrollo de interfaz de configuración.	1
30/06 – 13/07	Muestra de la información y resultados (visualizadores).	1
Fase 4 - Validación con datos reales públicos y/o sintéticos.		***2***
14/07 – 20/07	Búsqueda de datos y realización de pruebas.	1
21/07 – 27/07	Depuración de errores.	1
Total		**10**

Tabla 1.1 *Planificación para el desarrollo del proyecto.*

CAPÍTULO 2

2. DISEÑO

2.1. METODOLOGÍA

El proyecto se rige por la metodología ágil Kanban combinado con algunas pinceladas de la metodología XP.

La palabra **kanban**, proveniente del japonés y su traducción aproximada es tarjeta visual. Este famoso método desarrollado por Toyota, fue adaptado al desarrollo software en 2004, por David Anderson de la Universidad de Negocios XIT de Microsoft. Su objetivo principal es la entrega a tiempo del producto, sin sobrecargar a los miembros del equipo. Consta tan sólo de tres reglas simples, lo cual demuestra, que es una de las metodologías adaptativas que tiene menos resistencia al cambio. Las reglas son **[10]**:

1. Mostrar el proceso.
2. Limitar el trabajo en curso.
3. Optimizar el flujo de trabajo.

La **programación extrema** o *eXtreme Programming* (XP) es una metodología de desarrollo ágil de la ingeniería de software formulada por Kent Beck. Se puede que es el resultado de combinar distintas características de otras metodologías, integrándolas de forma que se complementaran. Los valores que se extrajeron para la ejecución del proyecto son: el desarrollo iterativo e incremental, la realización de pruebas unitarias continuas, entregas pequeñas y simplicidad de diseño. **[11]**

Una vez analizadas las técnicas anteriores, se describe a continuación la metodología que denominaremos Kanban eXtreme Programming o **KXP**; creada para llevar a cabo la ejecución del proyecto.

1. Mediante la utilización de alguna herramienta que implemente la metodología kanban, se definen dos pizarras. La primera tendrá el nombre de Day y las segunda Week.
2. Ambas pizarras estarán compuestas por 3 columnas, denominadas: To Do, Doing y Done; como se muestra en la ***Figura 2.1***.
3. La pizarra Week, será rellenada al principio de cada semana con un número limitado de tareas. En mi caso, el límite se fijó en dos tareas por semana, ampliado a 3 si se finalizaban las tareas propuestas, antes de lo previsto.

4. Las tareas irán fluyendo a través de las columnas en función de su estado.
5. Cuando movemos una Week tarea a la columna Doing, se definirán las tareas por hacer en la columna To Do de Day. Limitando el número de tareas por día a 4, sólo ampliables si se finalizaran todas antes de acabar el día.
6. Cada día se irán asignando Day tareas en función de las Week tareas que se encuentren en la columna Doing.
7. Si la tarea a realizar es implementar una funcionalidad, se definirán sus pruebas unitarias y sólo, cambiará a la columna Done cuando pase las pruebas.
8. Las Week tareas pasan a la columna Done cuando no existan más Day tareas relacionadas, en las columnas To Do y Doing.
9. Al finalizar cada semana se evalúa el estado del proyecto.
10. Es importante recalcar que a la hora de programar se persigue la simplicidad, pero antes de empezar a programar algo se recomienda pensarlo brevemente y hacer un pequeño diagrama de clases de lo que queremos hacer. Esto se traducirá en ahorro de tiempo, aunque no lo parezca a primera vista, ya que, tendremos las cosas más claras.

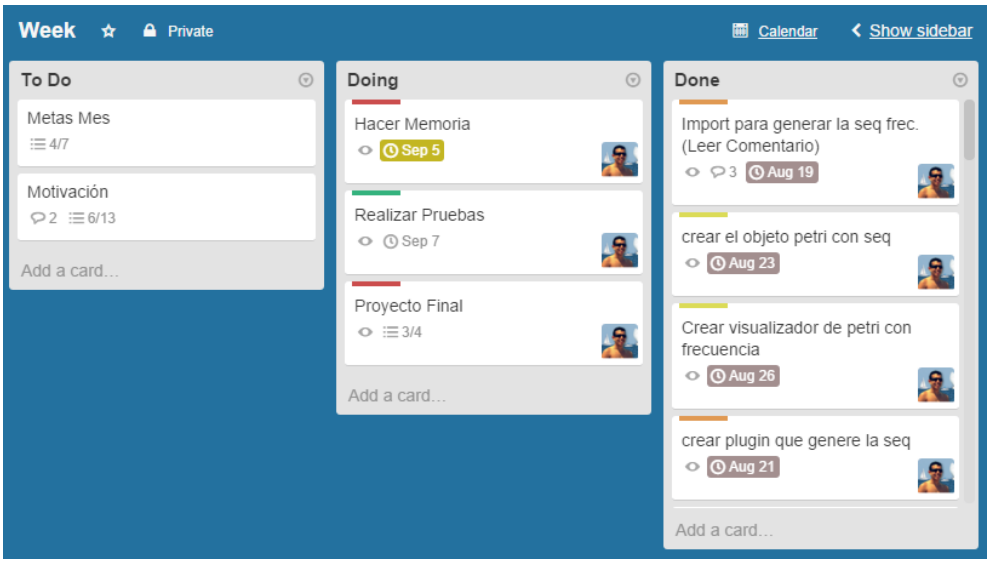

Figura 2.1 *Ejemplo de la pizarra Week con la herramienta Trello.*

2.2. DESCRIPCIÓN DEL ENTORNO DE TRABAJO

En este apartado se describen las herramientas utilizadas durante todo el proyecto. Es importante buscar que las versiones de las herramientas utilizadas sean compatibles, porque su integración y su buen funcionamiento, depende utilizar la combinación versiones adecuadas. **[12]**

- Prom 6
- Weka 3.6
- Java JDK 1.6.0_07
- Eclipse Gayaned v3.4.2
- Subclipse 1.1.0
- ObjectAid 1.1.6
- Trello
- Github

2.3. MODELO CONCEPTUAL

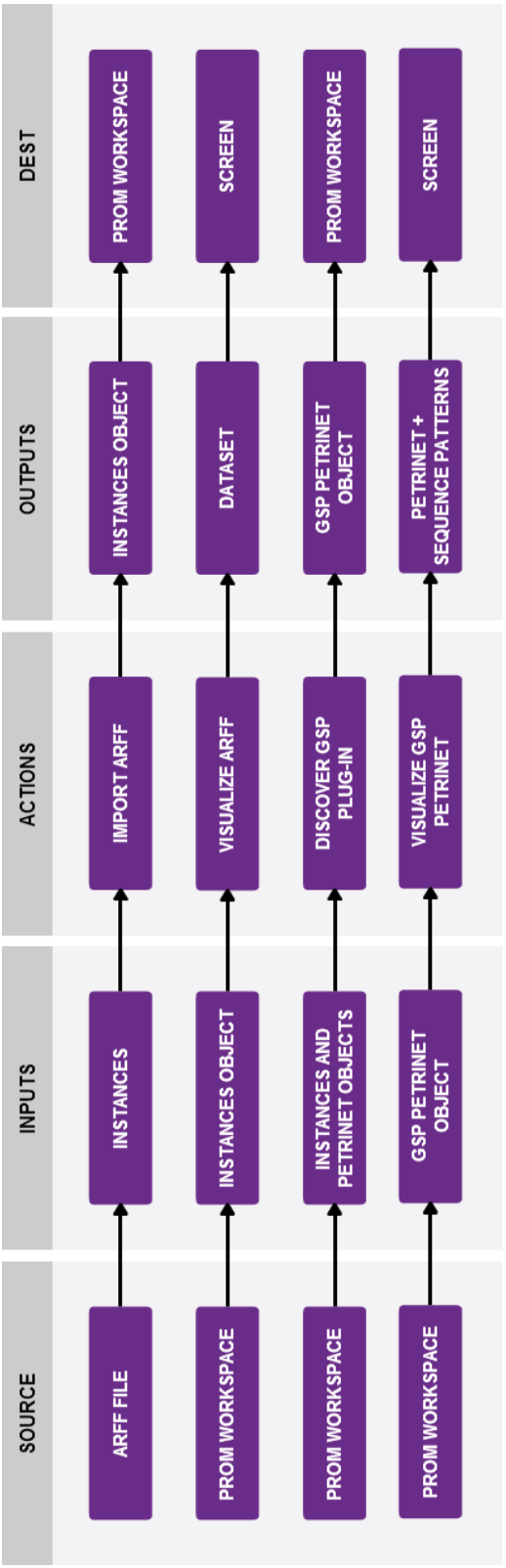

***Tabla 2.1** Diagrama Tabular desde el punto de vista del usuario.*

En la **Tabla 2.1** hay que destacar dos aspectos importantes. Lo primero que podemos apreciar son las distintas acciones que el usuario debe realizar para la utilización del plug-in. Lo segundo, es el flujo de la información que conlleva cada acción, sus requisitos y sus resultados.

La acción principal es el Discover GSP plug-in, que cómo podemos observar necesita como entradas una red de Petri y las Instancias del fichero Arff. Las demás acciones son las que nos permiten generar esas dos entradas.

2.4. DESCRIPCIÓN DE LA INTERFAZ

Al abrir el ProM 6, lo primero que se muestra es la pestaña del espacio de trabajo o Workspace, tal como se muestra en la **Figura 2.2**. Este espacio de trabajo, se encontrará vacío, si es la primera vez que lo utilizamos. En esta área se almacenan todos los objetos/datos con los que trabaja ProM, tanto las entradas, como las salidas.

Al cambiar a la pestaña de acción, se muestra una lista con todos los plugins que estén instalados. Para encontrar el plugin desarrollado, hay que escribir en el área de búsqueda, el nombre del plugin "Discover General Sequential Patterns". Posterior mente, al seleccionar el plugin, se muestran los datos requeridos como entrada para ejecutarlo como apreciamos en la **Figura 2.6.**

Conociendo cuales son los objetos de entrada necesarios, se debe proceder a su importación. Para la red de Petri, importar un fichero log y generar su el objeto Petrinet con uno de los plugins que facilita ProM, por ejemplo: "Mine for a Petri Net using Alpha-algorithm". Para importar el objeto instances, hay que transformar el mismo fichero log utilizado a un fichero arff (tipo de ficheros con los que trabaja Weka). Esto es recomendable realizarlo con la herramienta de transformación que facilita Weka, para realizar un filtrado de atributos y que todos los atributos sean nominales. Ahora sólo hay que importar el fichero generado pulsando el botón import de la **Figura 2.2**.

Por último, ejecutar el plugin presionando el botón Start de la venta de Acción que se encuentra en la **Figura 2.7.**

2.4.1. BOTÓN IMPORT

Este botón que se muestra en la **Figura 2.2**, se debe utilizar para importar los objetos descritos, anteriormente. Estos se obtienen con la importación de un fichero con extensión arff para el objeto instances y otro fichero normalmente con extensión xes para generar la red de Petri.

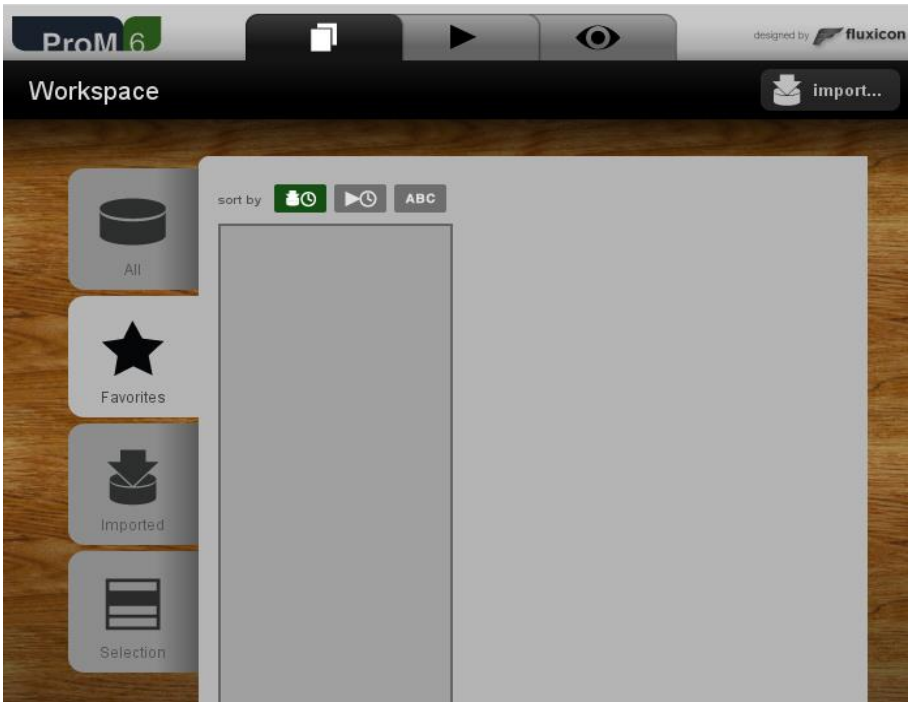

Figura 2.2 *Botón import ubicado en la esquina superior derecha del workspace de ProM 6.*

2.4.2. BOTÓN VIEW

Una vez importado el objeto podremos ver su contenido presionando el botón view como se muestra en la **Figura 2.3.**

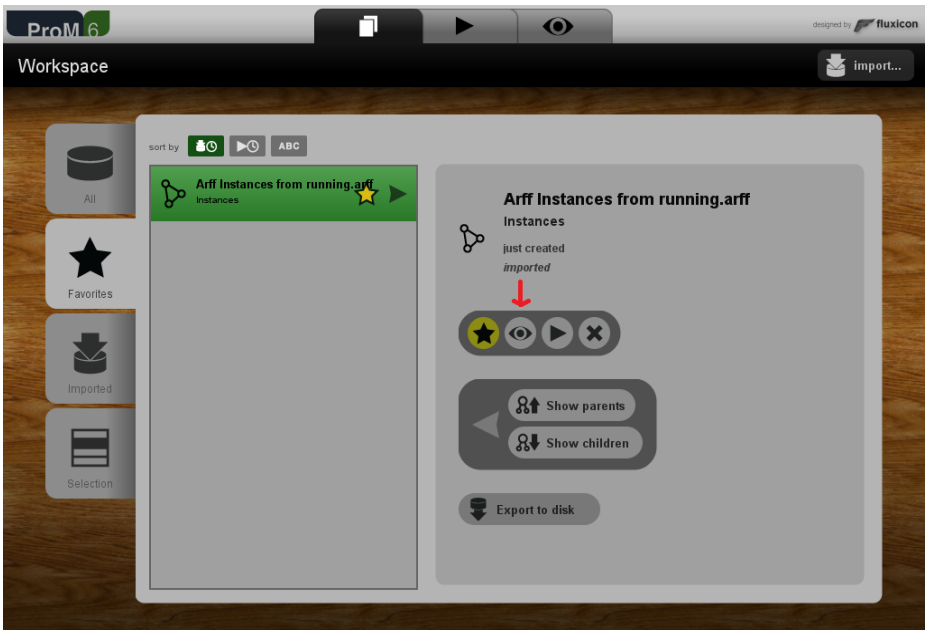

Figura 2.3 *Botón view. Para visualizar los objetos.*

Al presionar el botón view sobre un objeto, ProM cambiará a la pestaña de visualización y nos mostrará el contenido del objeto. En la **Figura 2.4** se aprecia el visualizer diseñado para los objetos instances.

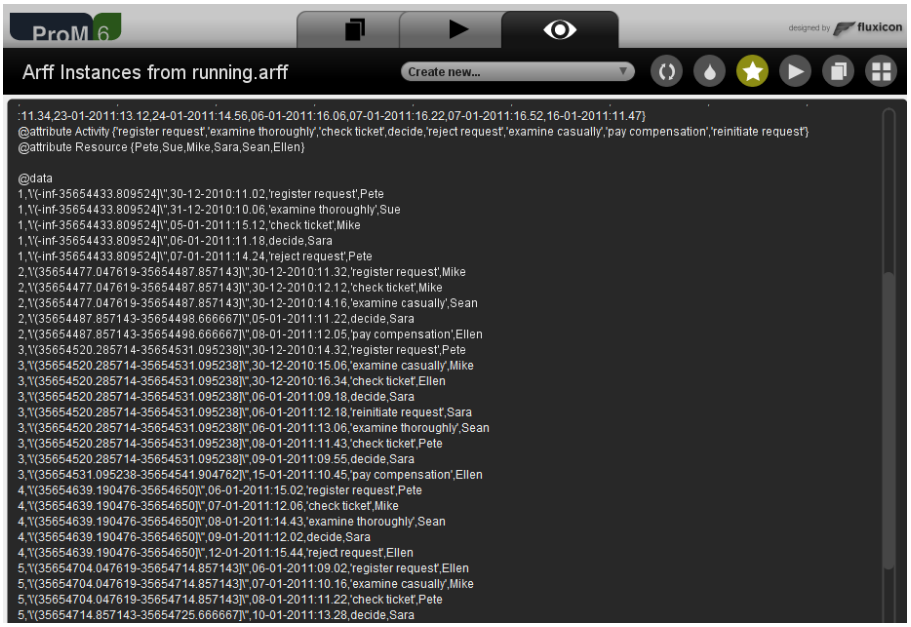

Figura 2.4 *Ventana de visualización de objetos.*

2.4.3. BOTÓN ACTION

Este botón cambia el ProM a modo de acción mostrando los plugins que se pueden ejecutar sobre el objeto seleccionado.

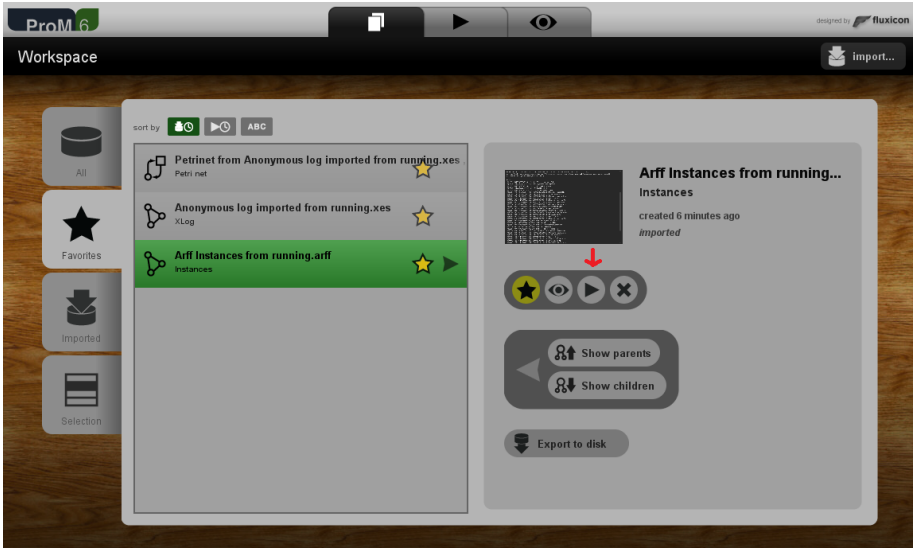

Figura 2.5 *Botón action de un objeto.*

La siguiente imagen, muestra la ventana Actions, que es donde se encuentran listados todos los plugins instalados, es decir, todas las acciones que podemos ejecutar. A través de la barra de búsqueda podemos filtrar las acciones, para encontrar fácilmente las operaciones con las que queremos trabajar. Al hacer clic sobre un plugin, se muestra los objetos requeridos y de salida.

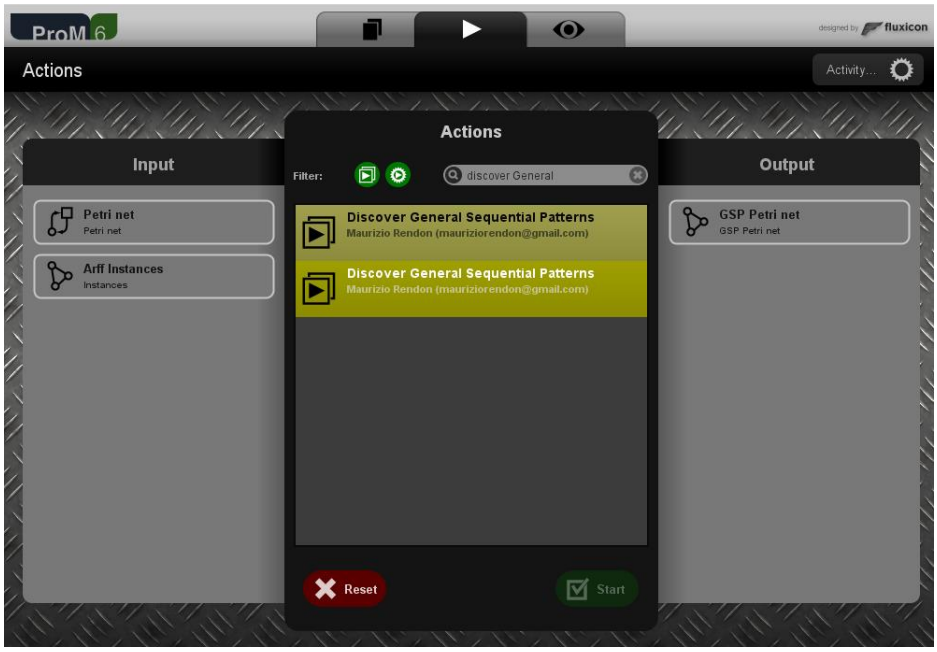

Figura 2.6 *Ventana de Acciones filtradas por "Discover General" sin objetos de entrada.*

Una vez seleccionados los objetos de entrada se procede a ejecutar el plugin seleccionado, pulsando el botón Start.

Figura 2.7 *Ventana Actions con los objetos seleccionados para el plugin Discover GSP.*

2.4.4. VENTANA DE CONFIGURACIÓN DEL ALGORITMO

Al presionar el botón de Start, se abrirá una o varias ventanas de configuración, si el plugin así lo requiere. En el caso del plugin Discover GSP se corresponde a la siguiente ventana única:

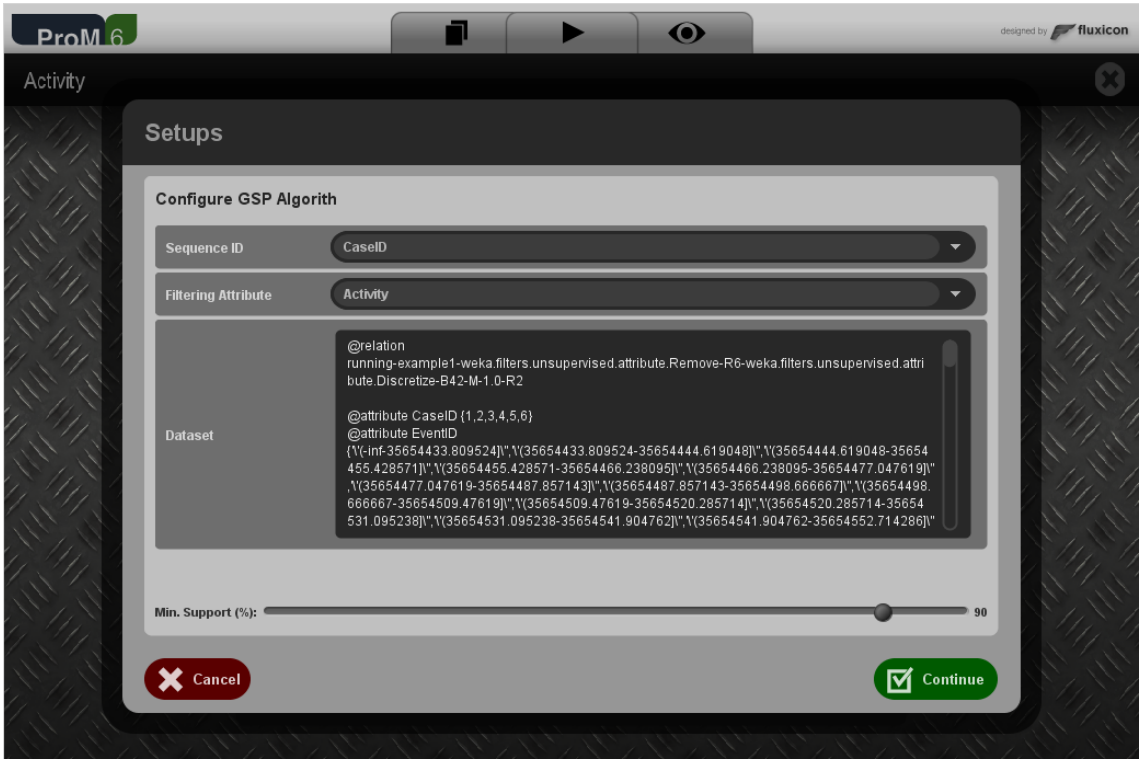

Figura 2.8 *Ventana Setups. Dónde se configuran los parámetros del Algoritmo GSP.*

En esta ventana podemos configurar los parámetros que recibe el algoritmo que son:

- Sequence ID: Atributo identificativo de las secuencias. Normalmente es el primer atributo de la base de datos.
- Filttering Attribute: Todas las secuencias generadas deberán contener un ítem que se corresponda al atributo seleccionado de la base de datos.
- Min. Support: Establece el porcentaje de soporte que debe tener cada secuencia para ser considerada como frecuente.

2.4.5. VISUALIZACIÓN DEL RESULTADO

Una vez configurados los parámetros, se ejecutará el algoritmo, generará el objeto resultante y automáticamente abrirá el visualizador del mismo.

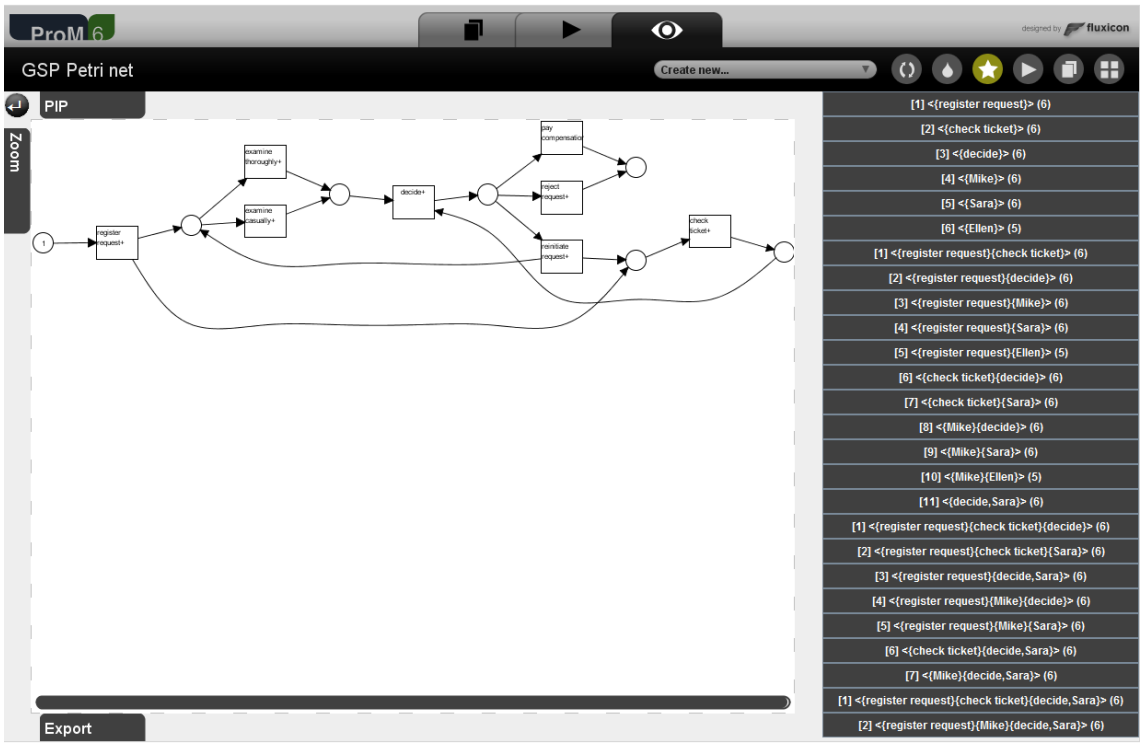

Figura 2.9 *Ventana de Visualización del resultado final del Discover GSP plug-in.*

En la imagen superior se observa la red de Petri del lado izquierdo y las secuencias más frecuentes, es decir los caminos más transitados del grafo, al lado derecho. Visualizar el camino frecuente facilita la comprensión de los resultados, su análisis y el descubrimiento de posibles mejoras, por ejemplo mejorar la eficiencia reduciendo los cuellos de botella.

CAPÍTULO 3

3. DESARROLLO

Para el desarrollo del prototipo de plug-in de Weka para la plataforma ProM se seleccionó de la librería de asociaciones de Weka el Algoritmo Generalized Sequential Patterns (de ahora en adelante, GSP) y el modelo de trabajo Petrinet de ProM. En los siguientes apartados se describen los modelos elegidos, cómo se integra la información de ambos modelos en unos sólo y los beneficios que se obtienen del modelo fusionado.

3.1. PETRINETS

La red de Petri es uno de los más antiguos lenguajes de modelado de procesos, son una representación generalizada de la teoría de autómatas. Un petrinet es un grafo compuesto por lugares, transiciones, arcos dirigidos y marcas o fichas que ocupan posiciones dentro de los lugares. Básicamente se utilizan para representar gráficamente sistemas de eventos discretos concurrentes de forma automatizada. Para ello, se utiliza el siguiente conjunto de reglas:

- Los arcos conectan un lugar a una transición así como una transición a un lugar.
- No puede haber arcos entre lugares ni entre transiciones.
- Los lugares contienen un número finito o infinito contable de marcas.
- Las transiciones se disparan, es decir consumen marcas de una posición de inicio y producen marcas en una posición de llegada.
- Una transición está habilitada si tiene marcas en todas sus posiciones de entrada.

Las áreas de aplicación de las redes de Petri son diversas, pero las principales son: el análisis de datos, el diseño de software, la evaluación de fiabilidad, elaboración de flujos de trabajo (workflow) y la programación concurrente. En el proyecto se utilizarán para generar el workflow que describe el funcionamiento de una aplicación o empresa a partir de registros logs. **[8]**

3.2. ALGORITMO GSP

Es una extensión del algoritmo de asociación Apriori. Lo que persigue es hallar las secuencias más frecuentes que se encuentran en una base de datos de secuencias. Una secuencia es una sucesión de acciones o tareas compuesta por lo que el algoritmo denomina itemsets. A continuación se describe brevemente el funcionamiento del algoritmo, para entender mejor su funcionamiento y los resultados que genera.

- Inicialmente todos los ítems de la BD es un candidato de longitud 1.
- Por cada nivel se van incrementando la longitud de las secuencias. Es decir, si estamos en el nivel 3, las secuencias tendrán longitud 3. (nivel-k = longitud-k).
- En cada nivel se escanea la base de datos para contar el soporte que tiene cada secuencia candidata.
- Se generan los candidatos del nivel siguiente con longitud k+1 y se seleccionan las secuencias frecuentes. Esta fase es la principal diferencia con algoritmo A Priori, porque en este, las combinaciones de las semillas se realizan sin tomar en cuenta el orden; mientras que en el GSP el orden sí importa.
- Repetir el proceso hasta que no se pueda encontrar más candidatos o generar más patrones.
- Nota: En el nivel 1 sólo se cuenta el soporte de los candidatos y descartamos a los que no cumplen las expectativas **[13, 14]**.

Ejemplo:

Apoyo Mínimo Establecido = 2

Candidatos Nivel 1:
<a>, <b>, <c>, <d>, <e>, <f>, <g>, <h>

Seq. ID	Sequence
10	<(bd)cb(ac)>
20	<(bf)(ce)b(fg)>
30	<(ah)(bf)abf>
40	<(be)(ce)d>
50	<a(bd)bcb(ade)>

Base de Datos Secuencial

Se cuenta el soporte y se realiza la selección de los candidatos:

Cand	Sup
<a>	3
<b>	5
<c>	4
<d>	3
<e>	3
<f>	2
~~<g>~~	1
~~<h>~~	1

Seq. ID	Sequence
10	<(bd)cb(ac)>
20	<(bf)(ce)b(fg)>
30	<(ah)(bf)abf>
40	<(be)(ce)d>
50	<a(bd)bcb(ade)>

Base de Datos Secuencial

1. A la hora de contar, el orden importa.
2. El contador se incrementa si la secuencia candidata es subsecuencia de una secuencia de los registros.

Candidatos Nivel 2:

	<a>	<b>	<c>	<d>	<e>	<f>
<a>	<aa>	<ab>	<ac>	<ad>	<ae>	<af>
<b>	<ba>	<bb>	<bc>	<bd>	<be>	<bf>
<c>	<ca>	<cb>	<cc>	<cd>	<ce>	<cf>
<d>	<da>	<db>	<dc>	<dd>	<de>	<df>
<e>	<ea>	<eb>	<ec>	<ed>	<ee>	<ef>
<f>	<fa>	<fb>	<fc>	<fd>	<fe>	<ff>

Apoyo y selección de los candidatos:

Cand	Sup
<aa>	2
<ab>	2
<ac>	2
<ad>	1
<ae>	1
<af>	1
<ba>	2
<bb>	4
<bc>	4
<bd>	3
<be>	3
<bf>	2
<ca>	2
<cb>	3
<cc>	1
<cd>	2
<ce>	3
<cf>	1

Cand	Sup
<da>	>=2
<db>	>=2
<dc>	>=2
<dd>	1
<de>	1
<df>	0
<ea>	0
<eb>	1
<ec>	0
<ed>	1
<ee>	1
<ef>	0
<fa>	1
<fb>	>=2
<fc>	1
<fd>	0
<fe>	1
<ff>	>=2

Seq. ID	Sequence
10	<(bd)cb(ac)>
20	<(bf)(ce)b(fg)>
30	<(ah)(bf)abf>
40	<(be)(ce)d>
50	<a(bd)bcb(ade)>

Base de Datos Secuencial

3.3. INTEGRACIÓN DE LA INFORMACIÓN

Los modelos antes explicados generan los siguientes resultados: el GSP descubre las secuencias de tareas frecuentes y la red de Petri el diagrama flujo de esas tareas. Ambos resultados se obtienen de un mismo fichero log. La diferencia está en que la red de Petri se crea directamente del fichero

de registro con extensión XES o CSV, mientras que para el GSP debemos transformar dicho fichero a un fichero Arff, para ello se utilizará una herramienta conversor facilitada por Weka.

Una vez creados la Petrinet y las instancias a partir de los ficheros anteriores el plugin recibe estos objetos y se encarga de ejecutar el algoritmo utilizando las instancias como su dataset y la red de Petri como su taxonomía. El prototipo genera como salida un objeto GSP Petrinet que es el mismo petrinet pero con el conjunto de secuencias.

Para mostrar el resultado se llama al mecanismo de visualización que facilita ProM para las redes de Petri y se le anexa un nuevo componente gráfico formado por las secuencias frecuentes que detectó el algoritmo como se muestra en la **Figura 2.9.**

3.4. BENEFICIOS DEL MODELO RESULTANTE

Al poder observar la red de Petri y secuencias más frecuentes (caminos más transitados del grafo) se facilita la comprensión de los resultados, su análisis y el descubrimiento de posibles mejoras. Los beneficios más destacados son:

- Mejora de la eficiencia, detectando y reduciendo los cuellos de botella.
- Detectar el coste o tiempo de las acciones más frecuentes que se realizan en un proceso.
- Conocer los empleados que realizan las actividades más importantes.
- Prevenir caídas del sistema.
- Rediseño del flujo de trabajo.
- Entre otros.

CAPÍTULO 4

4. PRUEBAS Y RESULTADOS

4.1. PRUEBA 1

Esta es una prueba de comprobación, con un pequeño conjunto de datos, para asegurar el buen funcionamiento del algoritmo y para explicar el significa de los resultados que arroja.

```
 - 3-sequences

[1] <{A,q}{w}> (5)
[2] <{A,q}{e}> (5)
[3] <{q}{w}{e}> (5)

 - 4-sequences

 [1] <{A,q}{w}{e}> (5)
```

@DATA
1,A,q
1,C,w
1,F,e
2,A,z
2,C,x
3,A,q
3,B,w

```
      Leyenda:
[1]   Secuencia número 1.
{A,q} Son dos ítems o atributos del mismo evento (misma fila).
{w}   es un ítem de la siguiente evento pero de la misma transacción.
(5)   Significa que el patrón tiene un soporte igual a 5.
```

El resultado anterior fue contrastado con la ejecución del algoritmo hecho a mano y es correcto.

4.2. PRUEBA 2

Las pruebas presentadas a continuación, han sido realizadas sobre conjuntos de datos sintéticos, los cuales se adjuntan en el apéndice 2.

```
GeneralizedSequentialPatterns
=============================
Number of cycles performed: 4
Total number of frequent sequences: 15

- 1-sequences

[1] <{registerrequest}> (6)
[2] <{checkticket}> (6)
[3] <{decide}> (6)
[4] <{paycompensation}> (6)
```

```
- 2-sequences

[1] <{registerrequest}{checkticket}> (6)
[2] <{registerrequest}{decide}> (6)
[3] <{registerrequest}{paycompensation}> (6)
[4] <{checkticket}{decide}> (6)
[5] <{checkticket}{paycompensation}> (6)
[6] <{decide}{paycompensation}> (6)

- 3-sequences

[1] <{registerrequest}{checkticket}{decide}> (6)
[2] <{registerrequest}{checkticket}{paycompensation}> (6)
[3] <{registerrequest}{decide}{paycompensation}> (6)
[4] <{checkticket}{decide}{paycompensation}> (6)

- 4-sequences

 [1] <{registerrequest}{checkticket}{decide}{paycompensation}> (6)
```

Este es el resultado que podemos obtener utilizando sólo la herramienta Weka con el algoritmo GSP, a partir de una base de datos secuencial. Las secuencias nos dan una idea de cuál es el flujo de trabajo y cuáles son las tareas más repetidas; sin embargo la información que podemos extraer no es completa y los resultados son difíciles de analizar.

4.3. PRUEBA 3

Si ejecutamos el plugin en ProM podemos observar los patrones con el flujo de trabajo o red de Petri. De esta forma es más fácil comprobar cuáles son los caminos más transitados y detectar posibles cuellos de botella o tendencias.

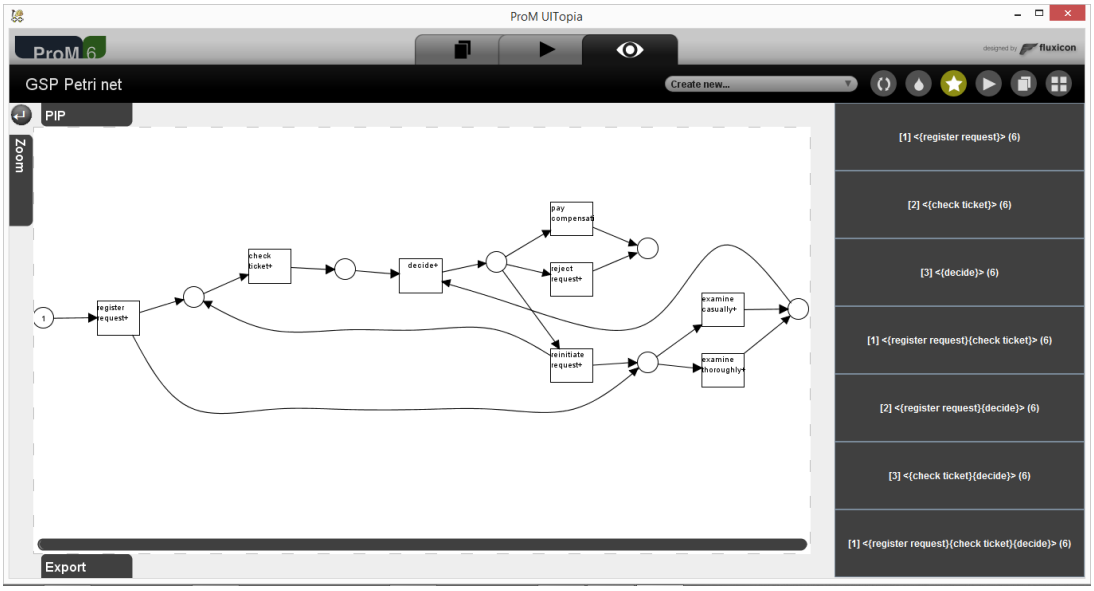

Figura 4.2 *Resultado Final del Discovery GSP Plug-in sólo de actividades.*

4.4. PRUEBA 4

En esta prueba se utiliza el mismo conjunto de datos, pero con ítems adicionales como la persona que realiza la acción. Al utilizar más ítems (atributos) es importante que filtremos el resultado por el ítem que más nos interese analizar; normalmente se filtra por el atributo de tareas o actividades, ya que se corresponden con el flujo de trabajo de la red de Petri.

En esta prueba hay que tener en cuenta que una actividad puede ser realizada por diferentes personas. Los resultados ahora nos muestran las secuencias de tareas más frecuentes con las personas que realizan esas tareas con más frecuencias.

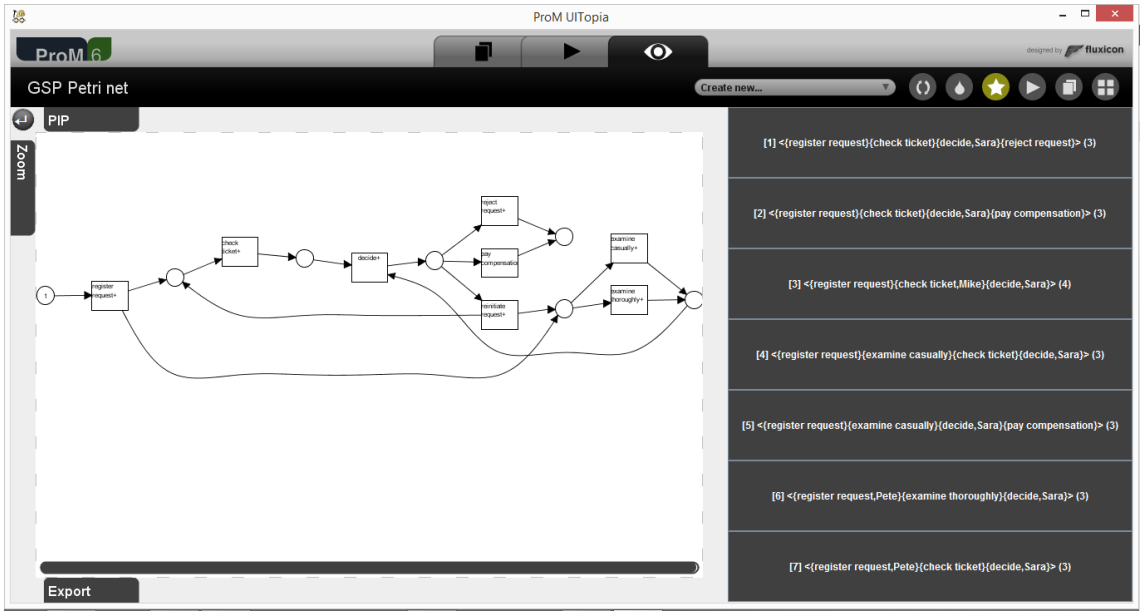

Figura 4.3 *Resultado Final del Discovery GSP Plug-in filtrado por el atributo actividad.*

Este último ejemplo es sólo una muestra del potencial del plugin; podríamos analizar por ejemplo, los costos que implican esas tareas que más se repiten en lugar de las personas que las realizan, o incluso una combinación de las anteriores.

Es muy importante tener claro qué es lo que queremos analizar, puesto que en ocasiones puede ser necesario filtrar o eliminar los atributos que no nos interesan del fichero arff.

CAPÍTULO 5

5. CONCLUSIONES Y TRABAJOS FUTUROS

Tanto la minería de datos como la de procesos utilizan el término minería, porque ambas escarban en enormes minas de datos para extraer la preciada información que se encuentra oculta. La minería de datos escarba en bases de datos y la minería de procesos en logs, pero hemos demostrado con la construcción del prototipo, que es posible utilizar las técnicas de minería de datos en logs y combinarlas con técnicas propias de minería de procesos. Como un trabajo a futuro se podría estudiar el proceso inverso.

En las pruebas se puede observar que el prototipo de plug-in desarrollado es capaz de combinar las secuencias frecuentes con el flujo de trabajo y además permite sobrecargar las secuencias con información extra. Los resultados obtenidos además de ser buenos, son una muestra de los beneficios que implica la combinación de las técnicas.

Resulta obvio, que aún queda mucho camino por recorrer. Se puede explotar aún más el potencial de las técnicas conocidas. Estas se pueden mejorar, combinar o incluso crear nuevas; sólo hay que continuar explorando el laberinto de túneles de las minas de datos.

CAPÍTULO 6

6. SUMMARY AND CONCLUSIONS

Data mining an processes mining use the term mining, because they both dig in massive data mines to extract the valuable hidden information. Data mining digs into databases and processes mining into logs, but we have shown that it is possible to use data mining techniques on logs and combine them with their own process mining techniques. As future work could study the reverse process.

In tests it can be seen that the prototype developed plug-in is able to combine the frequent sequences with the workflow and allows overloading sequences with extra information. The results obtained, are an example of the benefits that involves combining techniques.

Obviously, there is still a long way on the go, and we can further exploit the potential of the techniques that we know, we can improve them, or create new ones; you just have to continue exploring the maze of tunnels of data mines.

CAPÍTULO 7

7. PRESUPUESTO

El proyecto se ha desarrollado utilizando única y exclusivamente herramientas de software libre. Por lo que los costes del proyecto sólo se corresponden con los gastos de mano de obra. A continuación se muestra una tabla con el presupuesto fijado:

Referencia	Cantidad	Coste
Software	8	0 €
PC	1	0 €
Personal	1	5000€
Total	10	5000€

Tabla 7.1 Presupuesto del Proyecto.

JUSTIFICACIÓN DEL PRESUPUESTO

El proyecto se ha realizado en un total de 40 días, de los cuales se han dedicado una media de 5 horas/día. Suponiendo un pago de 25€ la hora, muy por debajo de lo que debería cobrar un ingeniero informático. El coste de mano de obra es de unos 5000€. Para ahorrar costes, se utilizó exclusivamente software libre y el personal contratado debe disponer de su propio ordenador y su sitio de trabajo.

BIBLIOGRAFÍA

[1] José Hernández Orallo, M. José Ramírez Quintana, César Ferri Ramírez. Introducción a la Minería de Datos. Editorial Pearson, 2004. ISBN: 978-84-205-4091-7. Págs. 3-9, 18, 22, 65-94

[2] Sinnexus Business Intelligence, Data Mining. http://www.sinnexus.com/business_intelligence/datamining.aspx

[3] Ian H. Witten, Eibe Frank, Mark A. Hall. Data Mining Practical Machine Learning Tools and Techniques, 3rd ed. Editorial Elsevier, 2011. ISBN: 978-0-12-374856-0. Págs. 8, 52, 314, 403, 407, 416, 487.

[4] Microsoft Developer Network. Data Mining. http://msdn.microsoft.com/es-es/library/ms174949.aspx

[5] IEEE Task Force on Process Mining. Manifiesto sobre Minería de Procesos. Págs. 1, 2, 4, 13, 20, 21 http://www.win.tue.nl/ieeetfpm/lib/exe/fetch.php?media=shared:pmm-spanish-v1.pdf

[6] Wikipedia. Minería de Procesos. http://es.wikipedia.org/wiki/Miner%C3%ADa_de_procesos

[7] Prom6 Official Page, Process Mining Group, Eindhoven Technical University. © 2010: http://www.promtools.org/doku.php

[8] Wil M. P. van der Aalst. Process Mining Discovery, Conformance and Enhancement of Business Processes. Editorial Springer, 2011. ISBN 978-3-642-19344-6. Págs. 11, 29-38, 95-107, 265-267, 331, 337.

[9] Weka Official Page, Machine Learning Group at the University of Waikato: http://www.cs.waikato.ac.nz/ml/weka/index.html

[10] Desarrollo Ágil con Kanban. http://www.desarrolloweb.com/articulos/desarrollo-agil-kanban.html

[11] Universidad de Valencia. Programación extrema: http://www.uv.mx/universo/486/infgral/infgral_15.html

[12] How to become a ProM developer: https://svn.win.tue.nl/trac/prom/wiki/setup/HowToBecomeAProMDeveloper

[13] Ramakrishnan Srikant and Rakesh Agrawal. Mining Sequential Patterns: Generalizations and Performance Improvements. IBM Almaden Research Center. 650 Harry Road, San Jose, CA 95120. Págs. 1-9.

[14] Data Mining: Mining sequence patterns in transactional databases: http://www.cs.nyu.edu/courses/spring08/G22.3033-003/8timeseries.ppt

[15] H.M.W. (Eric) Verbeek, R. P. Jagadeesh Chandra Bose, ProM6 Tutorial. August 2010. Págs. 1-25.

[16] ProM6 plugin development, Michael Westergaard: https://westergaard.eu/2012/11/prom-6-plug-in-development-part-1-basics/

[17] Github. Repositorio del Plug-in: https://github.com/alu0100611724/WPP

APÉNDICE A

DETALLES DE LA IMPLEMENTACIÓN APÉNDICE 1

El desarrollo del plug-in se realizó en un repositorio remoto Github. Se crearon dos ramas, una para el proceso development, donde se guardan las versiones beta y luego cuando se termina una funcionalidad y se prueba que funciona correctamente, se acopla a la rama master.

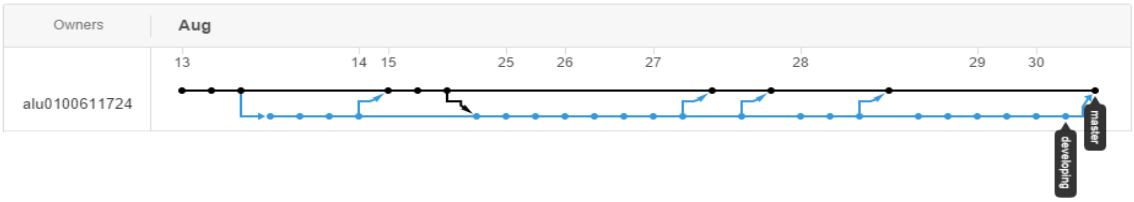

Figura A.1 *Grafo de los commits y las ramas creadas en Github.*

OBJETOS

Los objetos son simples tipos de datos en ProM. Se crean como un objeto normal en Java, salvo por pequeñas diferencias. La primera, es que los objetos no deben implementar la interfaz Serializable, porque ProM lo hace de forma automática. La segunda, es que un objeto nunca debe ser cambiado una vez creado; aunque no tienen por qué ser estrictamente inmutables, es una buena práctica. A continuación se muestran los objetos creados en el proyecto con sus métodos públicos y sus respectivas relaciones. **[15, 16]**

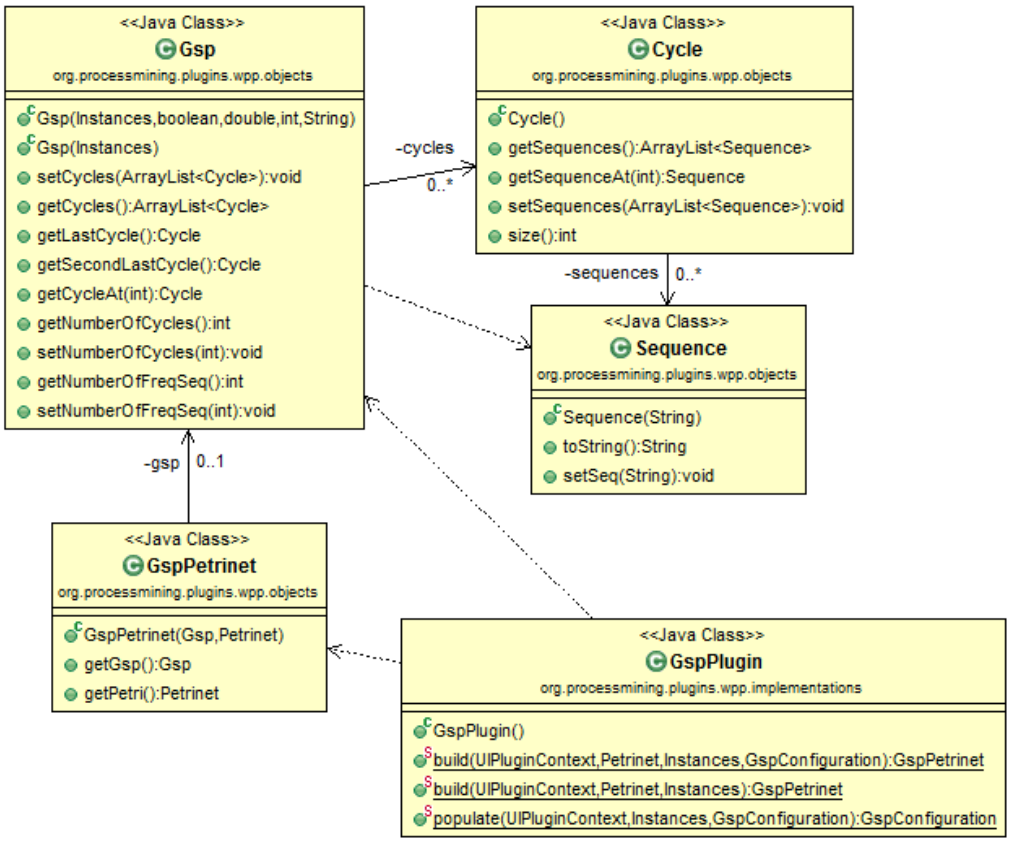

Figura A.2 *Diagrama de Clases de los objetos creados para guardar la información del plug-in, con sus métodos públicos y sus relaciones.*

En la Figura 3.2 observamos la estructura creada para almacenar los resultados del algoritmo GSP que arroja la clase GspPlugin. La clase o el tipo (si utilizamos la terminología de ProM) GSP está compuesto por uno o más ciclos y estos a su vez contienen una o más secuencias. El tipo GspPetrinet contiene una red de Petri y un objeto GSP.

INTERFAZ DE CONFIGURACIÓN Y PLUG-INS

Los Plug-ins son básicamente métodos, son mecanismos de ProM cuya idea principal es transformar un objeto en otro. Es importante recordar que los objetos originales, nunca deben ser cambiados y los nuevos creados, no deben ser alterados una vez el plug-in finaliza.

ProM usa una particular interfaz de estilo, basado en SlickerBox y es extendido por ProM a través del paquete Widgets. Todas las configuraciones se deberán realizar utilizando el ProMPropertiesPanel, que provee de una vista genérica de propiedades; esto nos evita tener que usar un JPanel y fijar manualmente los colores y el estilo. **[15, 16]**

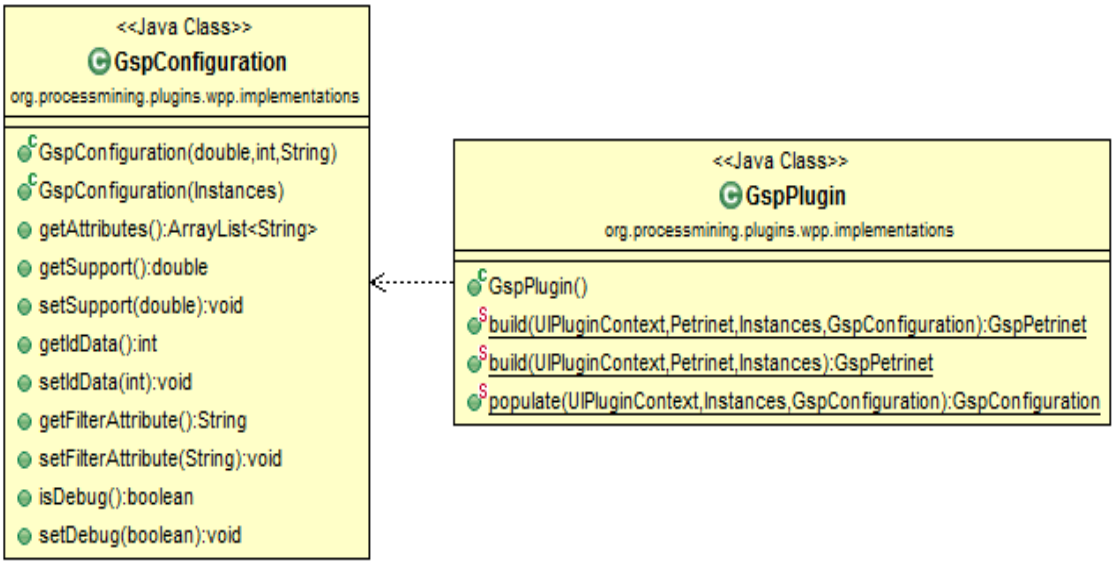

Figura A.3 *Diagrama de las clases plugin y su configuración.*

El GspPlugin utiliza la clase configuración para extraer los parámetros introducidos por el usuario a través de la venta de Setup del ProM.

VISUALIZADORES

Los visual visualizadores son solo un tipo especial de plug-ins, que siempre retorna un parámetro JComponet, que contiene la representación gráfica de un objeto. **[15, 16]**

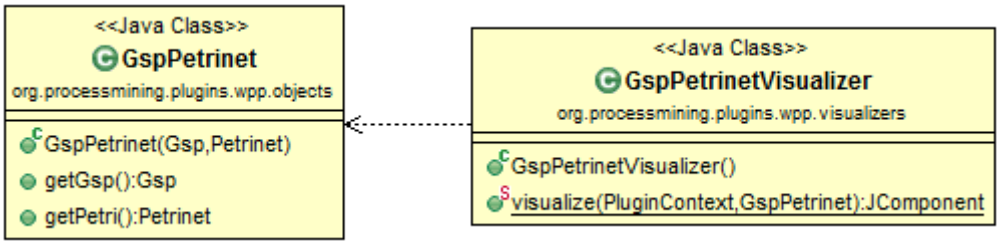

Figura A.4 *Diagrama de Clases del visualizador del objeto GspPetrinet.*

El objeto GspPetrinet a su vez contiene como vimos anteriormente un objeto Petrinet y un objeto GSP.

Para obtener acceso al código, consulta el repositorio Github. **[17]**

APÉNDICE B

DATASETS DE PRUEBAS APÉNDICE 2

@attribute CaseID {1,2,3,4,5,6}
@attribute EventID {'\'(-inf-35654433.809524]\"','\'(35654433.809524-35654444.619048]\"','\'(35654444.619048-35654455.428571]\"','\'(35654455.428571-35654466.238095]\"','\'(35654466.238095-35654477.047619]\"','\'(35654477.047619-35654487.857143]\"','\'(35654487.857143-35654498.666667]\"','\'(35654498.666667-35654509.47619]\"','\'(35654509.47619-35654520.285714]\"','\'(35654520.285714-35654531.095238]\"','\'(35654531.095238-35654541.904762]\"','\'(35654541.904762-35654552.714286]\"','\'(35654552.714286-35654563.52381]\"','\'(35654563.52381-35654574.333333]\"','\'(35654574.333333-35654585.142857]\"','\'(35654585.142857-35654595.952381]\"','\'(35654595.952381-35654606.761905]\"','\'(35654606.761905-35654617.571429]\"','\'(35654617.571429-35654628.380952]\"','\'(35654628.380952-35654639.190476]\"','\'(35654639.190476-35654650]\"','\'(35654650-35654660.809524]\"','\'(35654660.809524-35654671.619048]\"','\'(35654671.619048-35654682.428571]\"','\'(35654682.428571-35654693.238095]\"','\'(35654693.238095-35654704.047619]\"','\'(35654704.047619-35654714.857143]\"','\'(35654714.857143-35654725.666667]\"','\'(35654725.666667-35654736.47619]\"','\'(35654736.47619-35654747.285714]\"','\'(35654747.285714-35654758.095238]\"','\'(35654758.095238-35654768.904762]\"','\'(35654768.904762-35654779.714286]\"','\'(35654779.714286-35654790.52381]\"','\'(35654790.52381-35654801.333333]\"','\'(35654801.333333-35654812.142857]\"','\'(35654812.142857-35654822.952381]\"','\'(35654822.952381-35654833.761905]\"','\'(35654833.761905-35654844.571429]\"','\'(35654844.571429-35654855.380952]\"','\'(35654855.380952-35654866.190476]\"','\'(35654866.190476-inf)\"}
@attribute dd-MM-yyyy:HH.mm {30-12-2010:11.02,31-12-2010:10.06,05-01-2011:15.12,06-01-2011:11.18,07-01-2011:14.24,30-12-2010:11.32,30-12-2010:12.12,30-12-2010:14.16,05-01-2011:11.22,08-01-2011:12.05,30-12-2010:14.32,30-12-2010:15.06,30-12-2010:16.34,06-01-2011:09.18,06-01-2011:12.18,06-01-2011:13.06,08-01-2011:11.43,09-01-2011:09.55,15-01-2011:10.45,06-01-2011:15.02,07-01-2011:12.06,08-01-2011:14.43,09-01-2011:12.02,12-01-2011:15.44,06-01-2011:09.02,07-01-2011:10.16,08-01-2011:11.22,10-01-2011:13.28,11-01-2011:16.18,14-01-2011:14.33,16-01-2011:15.50,19-01-2011:11.18,20-01-2011:12.48,21-01-2011:09.06,21-01-2011:11.34,23-01-2011:13.12,24-01-2011:14.56,06-01-2011:16.06,07-01-2011:16.22,07-01-2011:16.52,16-01-2011:11.47}
@attribute Activity {'register request','examine thoroughly','check ticket',decide,'reject request','examine casually','pay compensation','reinitiate request'}
@attribute Resource {Pete,Sue,Mike,Sara,Sean,Ellen}

@data
1,'\'(-inf-35654433.809524]\",30-12-2010:11.02,'register request',Pete
1,'\'(-inf-35654433.809524]\",31-12-2010:10.06,'examine thoroughly',Sue
1,'\'(-inf-35654433.809524]\",05-01-2011:15.12,'check ticket',Mike
1,'\'(-inf-35654433.809524]\",06-01-2011:11.18,decide,Sara
1,'\'(-inf-35654433.809524]\",07-01-2011:14.24,'reject request',Pete
2,'\'(35654477.047619-35654487.857143]\",30-12-2010:11.32,'register request',Mike
2,'\'(35654477.047619-35654487.857143]\",30-12-2010:12.12,'check ticket',Mike

2,\'(35654477.047619-35654487.857143]\",30-12-2010:14.16,'examine casually',Sean
2,\'(35654487.857143-35654498.666667]\",05-01-2011:11.22,decide,Sara
2,\'(35654487.857143-35654498.666667]\",08-01-2011:12.05,'pay compensation',Ellen
3,\'(35654520.285714-35654531.095238]\",30-12-2010:14.32,'register request',Pete
3,\'(35654520.285714-35654531.095238]\",30-12-2010:15.06,'examine casually',Mike
3,\'(35654520.285714-35654531.095238]\",30-12-2010:16.34,'check ticket',Ellen
3,\'(35654520.285714-35654531.095238]\",06-01-2011:09.18,decide,Sara
3,\'(35654520.285714-35654531.095238]\",06-01-2011:12.18,'reinitiate request',Sara
3,\'(35654520.285714-35654531.095238]\",06-01-2011:13.06,'examine thoroughly',Sean
3,\'(35654520.285714-35654531.095238]\",08-01-2011:11.43,'check ticket',Pete
3,\'(35654520.285714-35654531.095238]\",09-01-2011:09.55,decide,Sara
3,\'(35654531.095238-35654541.904762]\",15-01-2011:10.45,'pay compensation',Ellen
4,\'(35654639.190476-35654650]\",06-01-2011:15.02,'register request',Pete
4,\'(35654639.190476-35654650]\",07-01-2011:12.06,'check ticket',Mike
4,\'(35654639.190476-35654650]\",08-01-2011:14.43,'examine thoroughly',Sean
4,\'(35654639.190476-35654650]\",09-01-2011:12.02,decide,Sara
4,\'(35654639.190476-35654650]\",12-01-2011:15.44,'reject request',Ellen
5,\'(35654704.047619-35654714.857143]\",06-01-2011:09.02,'register request',Ellen
5,\'(35654704.047619-35654714.857143]\",07-01-2011:10.16,'examine casually',Mike
5,\'(35654704.047619-35654714.857143]\",08-01-2011:11.22,'check ticket',Pete
5,\'(35654714.857143-35654725.666667]\",10-01-2011:13.28,decide,Sara
5,\'(35654714.857143-35654725.666667]\",11-01-2011:16.18,'reinitiate request',Sara
5,\'(35654714.857143-35654725.666667]\",14-01-2011:14.33,'check ticket',Ellen
5,\'(35654714.857143-35654725.666667]\",16-01-2011:15.50,'examine casually',Mike
5,\'(35654714.857143-35654725.666667]\",19-01-2011:11.18,decide,Sara
5,\'(35654714.857143-35654725.666667]\",20-01-2011:12.48,'reinitiate request',Sara
5,\'(35654714.857143-35654725.666667]\",21-01-2011:09.06,'examine casually',Sue
5,\'(35654714.857143-35654725.666667]\",21-01-2011:11.34,'check ticket',Pete
5,\'(35654714.857143-35654725.666667]\",23-01-2011:13.12,decide,Sara
5,\'(35654725.666667-35654736.47619]\",24-01-2011:14.56,'reject request',Mike
6,\'(35654866.190476-inf)\",06-01-2011:15.02,'register request',Mike
6,\'(35654866.190476-inf)\",06-01-2011:16.06,'examine casually',Ellen
6,\'(35654866.190476-inf)\",07-01-2011:16.22,'check ticket',Mike
6,\'(35654866.190476-inf)\",07-01-2011:16.52,decide,Sara
6,\'(35654866.190476-inf)\",16-01-2011:11.47,'pay compensation',Mike