PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

DETECCIÓN DE CAMBIOS TEMPORALES EN LOS PROCESOS DE NEGOCIO MEDIANTE EL USO DE TÉCNICAS DE SEGMENTACIÓN

DANIELA LORENA LUENGO MUNDACA

Tesis para optar al grado de
Magíster en Ciencias de la Ingeniería

Profesor Supervisor:
MARCOS ERNESTO SEPÚLVEDA FERNÁNDEZ

Santiago de Chile, Mayo de 2012

PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

DETECCIÓN DE CAMBIOS TEMPORALES EN LOS PROCESOS DE NEGOCIO MEDIANTE EL USO DE TÉCNICAS DE SEGMENTACIÓN

DANIELA LORENA LUENGO MUNDACA

Tesis presentada a la Comisión integrada por los profesores:

MARCOS SEPÚLVEDA FERNÁNDEZ

IGNACIO CASAS RAPOSO

BERNHARD HITPASS HEYL

GONZALO CORTÁZAR SANZ

Para completar las exigencias del grado de
Magíster en Ciencias de la Ingeniería

Santiago de Chile, Mayo de 2012

A mi familia y amigos por el apoyo que me dieron. Y a Esteban por su cariño y apoyo incondicional.

AGRADECIMIENTOS

Quiero agradecer a Marcos Sepúlveda, por su apoyo y excelente disposición durante el desarrollo de esta investigación. Además, agradezco a los profesores Josep Carmona y Bernhard Hitpass, por sus valiosos comentarios e ideas.

Quiero agradecer también a la Pontificia Universidad Católica de Chile, especialmente al grupo BPM del Departamento de Ciencias de la Computación de la Escuela de Ingeniería UC, por sus aportes y comentarios, y a CETIUC por el apoyo que me dieron para llevar adelante con éxito este desafío. Agradezco también a mis amigos por haberme comprendido y acompañado durante esta etapa de mi vida.

Finalmente quiero agradecer a mi Familia por el gran apoyo durante mi periodo estudiantil y por haberme dado las herramientas necesarias para la vida.

INDICE GENERAL

Pág.

INDICE DE TABLAS

Pág.

INDICE DE FIGURAS

Pág.

RESUMEN

Hoy en día, las organizaciones tienen la necesidad de estar constantemente cambiando para ajustarse a las necesidades del entorno. Estos cambios se reflejan en sus procesos de negocio, por ejemplo, un supermercado debido a cambios estacionales tendrá distinta demanda en distintos meses del año, por lo que sus procesos de abastecimiento o de reposición de productos podrían ser distintos en distintas épocas del año. Una forma de analizar con profundidad un proceso y entender cómo realmente se ejecuta en la práctica a través del tiempo, es en base al análisis de sus registros históricos almacenados en los sistemas de información, lo cual es conocido como minería de procesos. Sin embargo, en la actualidad la mayoría de las técnicas que existen para analizar y mejorar procesos consideran todos los registros de un proceso de manera estática, es decir, que el proceso no cambia a través del tiempo, lo cual en la práctica es poco realista dada las naturaleza dinámica de las organizaciones.

Nuestro trabajo propone una técnica de segmentación que encuentra las distintas versiones de un proceso a través del tiempo. Esta técnica se basa en una técnica existente de segmentación que solo considera características estructurales del proceso (flujo de actividades). Nuestra técnica incorpora de manera adicional la característica temporal de los procesos, de tal manera que los *clusters* que se generen al realizar la segmentación tengan una similitud estructural, pero también una cercanía temporal, de tal manera que representen distintas versiones del proceso.

Este documento presenta el detalle de la técnica propuesta y un conjunto de experimentos que reflejan que nuestra propuesta entrega mejores resultados que las técnicas existentes de segmentación.

Palabras Claves: Minería de Procesos, Concept Drift, Dimensión Temporal, Segmentación.

ABSTRACT

Nowadays, organizations need to be constantly evolving in order to adjust to the needs of their business environment. These changes are reflected in their business processes, for example: due to seasonal changes, a supermarket's demand will vary greatly during different months of the year, which means product supply and/or re-stocking needs will be different during different times of the year. One way to analyze a process in depth and understand how it is really executed in practice over time, is on the basis of an analysis of past event logs stored in information systems, known as process mining. However, currently most of the techniques that exist to analyze and improve processes assume that process logs are in a steady state, in other words, that the processes do not change over time, which in practice is quite unrealistic given the dynamic nature of organizations.

Our work proposes a clustering technique that finds the different versions of a process over time. This approach is based on an existing clustering technique which only considers the process's structural features (control-flow). Our approach also incorporates the process's temporal features, in such a way that the *clusters* which are generated have structural similarity, but also temporal proximity, representing the different versions of the process.

This document presents in detail the proposed technique and a set of experiments that reflect how our proposal delivers better results than existing clustering techniques.

Key words: Process Mining, Concept Drift, Temporal Dimension, Clustering.

1. INTRODUCCIÓN

Hoy en día, en un mundo globalizado e hiperconectado, las organizaciones tienen la necesidad de mantenerse en permanente cambio para adaptarse a las necesidades del entorno, lo que implica que sus procesos de negocio también deban estar cambiando constantemente. Para ilustrar esto, es posible considerar el caso de una tienda de juguetes, donde sus procesos de venta pueden variar de manera radical dependiendo de si son ejecutados en navidad o en período de vacaciones, debido principalmente a los cambios en los volúmenes de la demanda. En navidad se podría ejecutar un proceso que priorice la eficiencia y el volumen – *throughput* –, y en vacaciones se podría ejecutar un proceso con foco en la calidad de atención al cliente. En este ejemplo es fácil identificar los períodos en que la demanda cambia, por lo tanto, es posible que el responsable de la gestión del proceso tenga claridad respecto a los cambios que sufre el proceso de venta a través del tiempo. Sin embargo, si una organización tiene un proceso que se realiza en distintas oficinas autónomas, por ejemplo por que se encuentran en distintas ubicaciones geográficas, la evolución de los cambios en el proceso en cada oficina ya no es tan evidente para el responsable central del proceso; los cambios en cada oficina pueden ser distintos y estarse aplicando en distintos momentos en el tiempo. Entender los cambios que están ocurriendo en las distintas oficinas, podría ayudar a entender mejor cómo mejorar el diseño global del proceso. Poder entender estos cambios y modelar las distintas versiones del proceso, permiten al responsable de su gestión contar con

información más precisa y completa para tomar decisiones coherentes que redunden en una mejor atención o eficiencia.

Para lograr lo anterior, se han desarrollado diversos avances en la disciplina de gestión de procesos de negocio (BPM por sus siglas en inglés), disciplina que combina conocimiento sobre tecnologías de información y técnicas de gestión, las cuales son aplicadas a procesos de negocio operativos, con el objetivo de mejorar su eficiencia (van del Aalst, 2011).

Dentro de BPM, la minería de procesos se ha posicionado como una disciplina emergente, proveyendo un conjunto de herramientas que ayudan a analizar y mejorar los procesos de negocio (van del Aalst, 2011), en base al análisis de los registros de eventos que almacenan los sistemas de información durante la ejecución de un proceso. Sin embargo, a pesar de los avances desarrollados en este campo, aún existe un gran desafío, el cual consiste en incorporar el hecho que los procesos cambian a lo largo del tiempo, concepto que es conocido en la literatura como *Concept Drift* (Bose, van der Aalst, Zliobaite, & Pechenizkiy, 2011).

Dependiendo de la naturaleza del cambio, es posible encontrar diferentes tipos de *Concept Drift*, algunos de ellos son: *Sudden Drift* (cambio repentino y significativo a la definición del proceso), *Gradual Drift* (cambio gradual en la definición del proceso, permitiendo la existencia de dos definiciones de éste de manera simultánea) o *Incremental Drift* (la evolución del proceso se realiza a través de pequeños cambios

sucesivos a la definición del modelo). Pese a que existen todas estas variantes de *Drift*, en la actualidad las técnicas de minería de procesos existentes están limitadas a encontrar los puntos en el tiempo en que el proceso cambia, centrándose principalmente en cambios de tipo *Sudden Drift*. El problema de esta limitación es que en la práctica no es tan frecuente que procesos de negocio muestren un cambio repentino de su definición. Un ejemplo de un proceso que tiene un cambio repentino podría ser cuando las aerolíneas y los aeropuertos cambian sus procesos de seguridad debido a un nuevo reglamento. Por el contrario, es más común el *Gradual Drift*. Un ejemplo de esto es un proceso de abastecimiento de productos en una organización; cuando la organización introduce una nueva forma de ejecutar el proceso, la cual es aplicable sólo a las nuevas ordenes de abastecimiento que se generen desde ese momento. Las órdenes que ya se comenzaron a realizar deben terminar de procesarse siguiendo la forma antigua de ejecutar el proceso, por lo tanto, el cambio de una versión a otra del proceso es gradual, y en un intervalo de tiempo coexisten dos formas distintas de ejecutar el proceso.

Si aplicamos las técnicas de minería de procesos existentes en procesos que tengan cambios distintos a *Sudden Drift*, podríamos encontrarnos con resultados de poco sentido para el negocio.

En este documento proponemos un nuevo enfoque, el cual permite descubrir las versiones de un proceso cuando tiene distintos tipos de *Drift*, ayudando a entender cómo se comporta el proceso a través del tiempo. Para llevar a cabo esta tarea, se utiliza

técnicas existentes de segmentación en minería de procesos, pero que incorporan el tiempo como una variable adicional al control de flujo para generar los distintos *clusters*. Se utilizan técnicas de *Trace Clustering*, las cuales, a diferencia de otras técnicas basadas en métricas para medir la distancia entre secuencias completas (como la distancia de *Levenshtein* o *Generic Edit Distance* (Bose & van der Aalst, 2010)), tienen una complejidad lineal, permitiendo la entrega de resultados en menores tiempos (Bose & van der Aalst, 2010). El enfoque de nuestro trabajo contribuye al análisis del proceso, permitiendo al responsable de la gestión del proceso tener una visión más realista de cómo se comporta el proceso en distintos intervalos de tiempo. Con este enfoque es posible determinar las distintas versiones del proceso, las características de cada una de ellas, e identificar en qué momento ocurren estos cambios.

Este artículo está organizado de la siguiente manera. La sección 2 presenta el trabajo relacionado y en la sección 3 se describe el desafío de *Concept Drift* en minería de procesos. En la sección 4 se presenta el método de segmentación en el cual se basa nuestro trabajo, para luego presentar en la sección 5 el nuevo enfoque que extiende el método de segmentación ya discutido. En la sección 6 se presenta los experimentos realizados, resultados y análisis de estos, para finalmente, presentar en la sección 7 las conclusiones y el trabajo futuro.

2. TRABAJO RELACIONADO

La minería de procesos es una disciplina que ha concentrado gran interés en la actualidad, debido a la promesa de obtención de información adicional que permita conocer el comportamiento oculto de los procesos y así generar mejoras en su rendimiento. Esta disciplina asume que la información histórica almacenada en los sistemas de información sobre un proceso se encuentra en un registro, conocido como *log* de eventos (van der Aalst et al. 2003). Este registro contiene información histórica de las actividades que se llevan a cabo en cada ejecución del proceso, donde cada fila del registro está compuesta, al menos, por un identificador (id) asociado a cada ejecución individual del proceso, el nombre de la actividad ejecutada, su marca de tiempo (día y hora en que ocurre la actividad) y, opcionalmente, información adicional, como el ejecutor de la actividad u otros. Adicionalmente, en la literatura (Bose & van der Aalst, 2010) se define como traza de la ejecución de un proceso, a la lista ordenada de actividades invocadas por una ejecución en particular, lo que significa, por ejemplo, registrar todos los eventos asociados a la obtención de un crédito hipotecario de un cliente en particular, en el caso de un proceso de entrega de crédito hipotecario.

Actualmente, uno de los problemas en la minería de procesos, es que los algoritmos desarrollados suponen la existencia de información relativa a una única versión de un proceso en el *log* de eventos. Sin embargo, esto muchas veces no se cumple, por lo que aplicar los algoritmos de minería de procesos a estos *logs* (que podrían contener varias

versiones de un mismo proceso de manera simultánea) lleva a resultados poco representativos y/o de gran complejidad, que aportan poco a la tarea de análisis y mejora de procesos.

2.1 Descubrimiento de procesos

Un ejemplo claro del problema mencionado en la minería de procesos, se da en las técnicas de descubrimiento de procesos, que consisten en extraer un modelo único y representativo que consolida todas las distintas trazas que existen en el *log* de eventos (ver Figura 2-1) (van der Aalst et al. 2004), donde uno de los grandes desafíos existentes en estas técnicas, consiste en lograr modelos más simples, comprensibles y representativos que los que se pueden conseguir actualmente (van der Aalst, 2011).

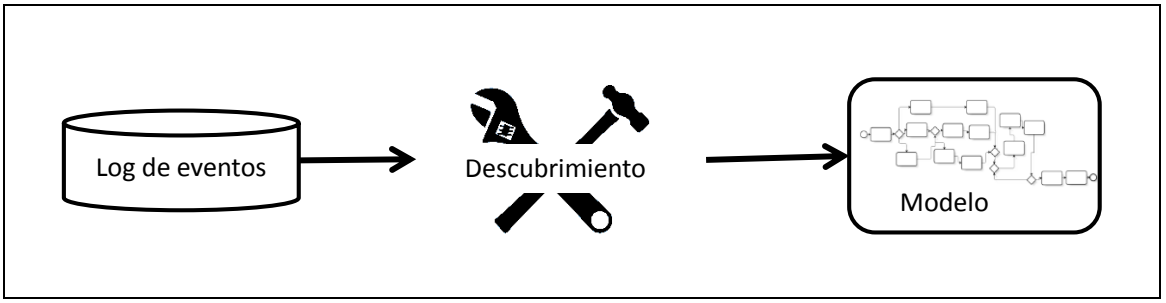


Figura 2-1: Descubrimiento de un único modelo del proceso a partir del *log* de eventos.

2.2 Segmentación del *log* de eventos

Para resolver el problema mencionado en la minería de procesos, se han propuesto técnicas de segmentación del *log* de eventos antes de aplicar técnicas de minería de procesos (Song, Günther, & van der Aalst, 2009), las que consisten en dividir el *log* de eventos en *clusters* homogéneos, para luego aplicar de manera independiente las técnicas de minería de procesos sobre cada uno de ellos y así obtener información o modelos más representativos. La

Figura **2-2** muestra la etapa de procesamiento del *log* y utiliza una técnica de descubrimiento como ejemplo de técnica de minería de procesos.

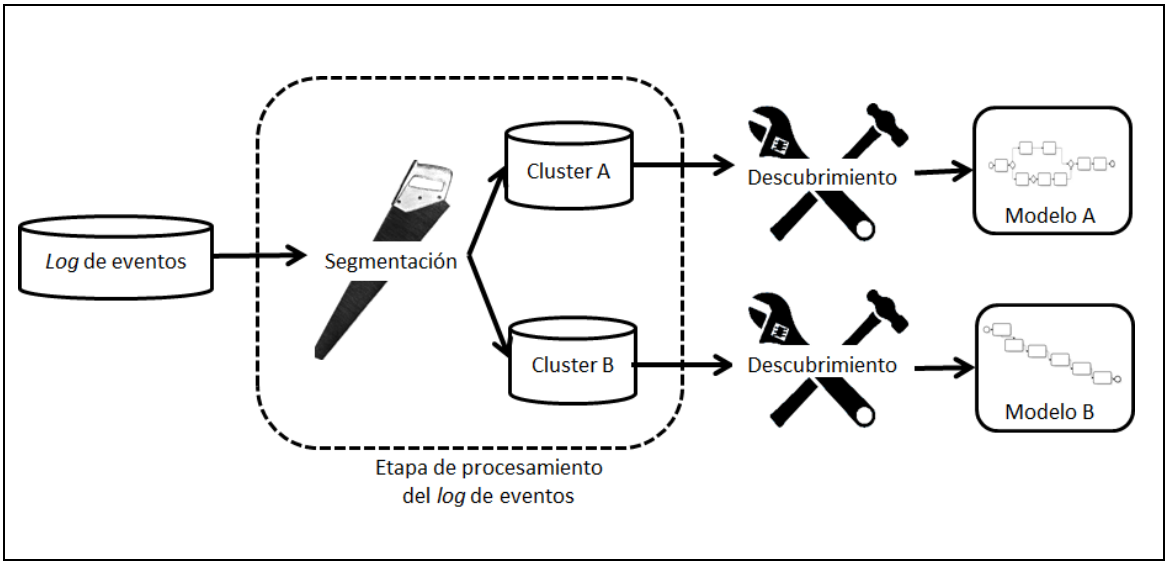


Figura 2-2: Etapa de procesamiento del *log* de eventos.

Para realizar esta segmentación es necesario definir una manera de representar a las trazas, de manera de poder agruparlas posteriormente de acuerdo a un criterio de similitud previamente definido. Actualmente existen varias técnicas de segmentación en la minería de procesos (Bose & van der Aalst, 2009) (Bose & van der Aalst, 2010), donde la mayoría de ellas considera principalmente información del flujo de las actividades. Estas técnicas pueden ser clasificadas en dos categorías:

1. Técnicas que transforman las trazas en un espacio vectorial, en donde cada traza se convierte en un vector, cuyas dimensiones corresponden a las características de control de flujo que se utilizan para medir la similitud entre trazas. La división del *log* puede ser utilizando una variedad de técnicas de segmentación en el espacio vectorial (Jain 1999), como por ejemplo: *Bag of activities, K-gram model* (Bose & van der Aalst, 2009) y *Trace clustering* (Song, Günther, & van der Aalst, 2009). Sin embargo, estas técnicas tienen el problema que carecen de información de contexto, lo que se ha intentado resolver con la técnica *Trace clustering based on conserved patterns* (Bose & van der Aalst, 2010).

2. Técnicas que operan con la traza completa. Estas técnicas utilizan métricas de distancia como *Levenshtein* y *Generic Edit Distance* (Bose & van der

Aalst, 2009), en conjunto con técnicas estándar de segmentación, asignando un costo a la diferencia entre trazas.

Sin embargo, las técnicas existentes en ambas categorías, a pesar de mejorar la segmentación a través de formar *clusters* de trazas estructuralmente similares, no consideran la dimensión temporal de la ejecución de los procesos, ni cómo el proceso va cambiando en el tiempo.

2.3 El Desafío de *Concept Drift*

Concept Drift, en BPM, se refiere a la situación en la cual un proceso ha sufrido cambios en su diseño dentro del periodo analizado (y no se conoce el momento en que se produjeron los cambios). Estos cambios se pueden deber a varios factores, pero principalmente se deben a la naturaleza dinámica de los procesos, que constantemente deben ajustarse a distintas necesidades del entorno (van der Aalst & Otros, 2011).

En el contexto de procesos de negocio, se han definido tres perspectivas para analizar un proceso (Figura 2-3), las cuales son:

1. Perspectiva de control de flujo, que está enfocada en el flujo del proceso, y su objetivo es encontrar una buena caracterización para todas las posibles ejecuciones del proceso.

2. Perspectiva organizacional, enfocada en información de los ejecutores de las actividades y cómo estos se relacionan.
3. Perspectiva de datos, que se enfoca en otras propiedades o datos de los procesos desde donde se podrían realizar análisis interesantes.

Los cambios en un proceso en los que se ha centrado el estudio de *Concept Drift* en el área de minería de procesos, tienen que ver con los cambios en la perspectiva de control de flujo, y pueden ser de dos tipos, cambios permanentes o cambios momentáneos, según la duración de los cambios. Cuando ocurren cambios en periodos cortos de tiempo y pocas instancias se ven afectadas, entonces se habla de cambios momentáneos. Estos cambios también son reconocidos en el lenguaje de procesos, como ruido o anomalías del proceso. Por otro lado, los cambios permanentes ocurren en períodos más prolongados de tiempo y/o hay una considerable cantidad de instancias afectadas por los cambios, lo cual hace referencia a un cambio en el diseño del proceso. Nuestro interés se centra en los cambios permanentes en la perspectiva de control de flujo, los cuales pueden dividirse en las siguientes 4 categorías (ver Figura 2-3. M1, M2 y M3 representan versiones distintas de un proceso):

• *Sudden Drift*: Se refiere a los cambios que ocurren de manera drástica, es decir, la forma de realizar el proceso cambia repentinamente de un momento a otro.

- *Recurring Drift*: Cuando los cambios que sufre el proceso ocurren de manera periódica, es decir, una forma de hacer el proceso se repite en otro periodo de tiempo posterior. Esto se puede deber a estacionalidades del proceso.
- *Gradual Drift*: Se refiere a cambios que no son drásticos, sino que en algún momento dos versiones del proceso se traslapan, ya que corresponde a la transición de una versión del proceso a la otra.
- *Incremental Drift:* Es cuando un proceso tiene pequeños cambios incrementales. Este tipo de cambios es más frecuente en organizaciones que adoptan metodologías ágiles de BPM.

Estos distintos cambios ocurren cuando existe inserción, eliminación, sustitución y/o reordenamientos de fragmentos del proceso, que pueden considerar una o varias actividades.

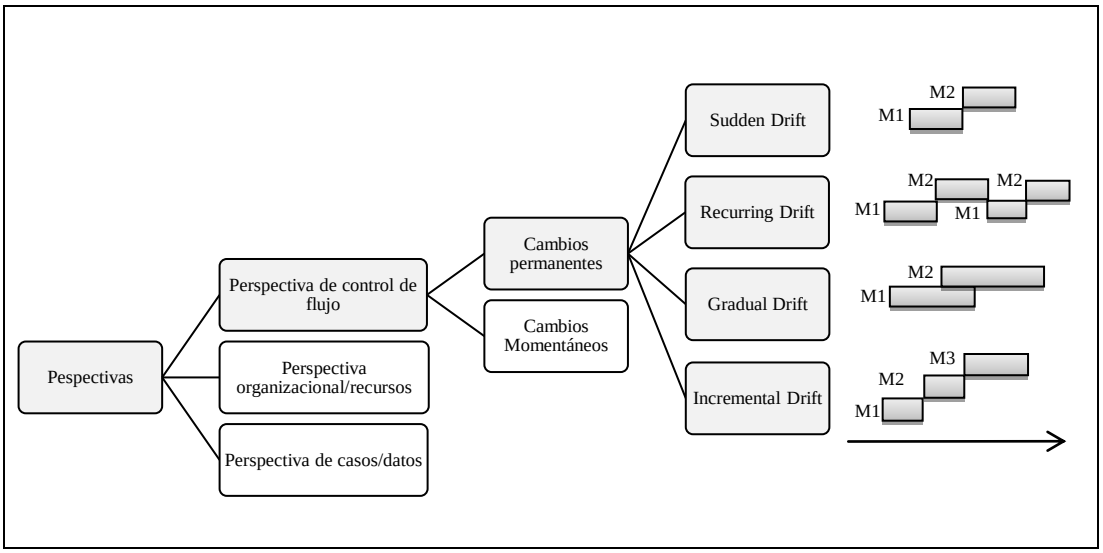


Figura 2-3: Tipos de cambios que pueden ocurrir en un proceso. Diagrama basado en (Bose R. P., van der Aalst, Zliobaite, & Pechenizkiy, 2011).

Para resolver el problema de *Concept Drift,* han surgido nuevos enfoques que analizan el dinamismo de los procesos.

Bose (Bose, van der Aalst, Zliobaite, & Pechenizkiy, 2011) propone metodos para manejar el *Concept Drift*, mostrando que los cambios en el proceso están indirectamente reflejados en el *log* de eventos, y la detección de estos cambios es factible examinando la relación entre las actividades. Se basa en el supuesto que la relación entre la actividad que sigue o que precede a otra actividad es lo bastante rica como para revelar cambios en el control de flujo de un proceso. Se definen distintas métricas para medir la relación entre actividades. A partir de estas

métricas se propone un método estadístico, cuya idea base es considerar una serie sucesiva de valores e investigar si hay una diferencia significativa entre dos series. Si es que existe, ésta correspondería a un cambio en el proceso.

Stocker (Stocker, 2011), también propone un método para manejar *Concept Drift*, el cual considera las distancias entre pares de actividades de distintos trazos como una característica estructural para generar *clusters* cronológicamente subsecuentes. Aunque el enfoque que se le da al método es para encontrar puntos críticos de análisis y para identificar vulnerabilidades, la idea es que con este método se aprecien las diferentes formas en las cuales evoluciona el proceso a través del tiempo.

Los enfoques de Bose y Stocker se limitan a determinar el momento en el tiempo en que el proceso cambia, por lo que se centran en procesos con cambios repentinos, dejando fuera otro tipo de cambios.

Para resolver esto, en un artículo anterior (Luengo & Sepúlveda, 2011) propusimos un enfoque que utiliza técnicas de segmentación para descubrir los cambios que puede sufrir un proceso a través del tiempo, pero sin limitarse a un tipo de cambio en particular. En este enfoque, la similitud entre dos trazas está definida por información del flujo de las actividades y por información del momento en que se comienza a ejecutar cada traza.

En contraste con los enfoques anteriores, nuestra propuesta genera *clusters* que en la línea de tiempo pueden estar traslapados, ya que agrupamos en un mismo *cluster* instancias que tienen una similitud estructural y que se hayan comenzado a ejecutar en un periodo de tiempo cercano.

En este trabajo, presentamos una extensión al anterior artículo (Luengo & Sepúlveda, 2011), al incorporar una nueva forma de medir la distancia entre dos trazas y realizar una evaluación experimental más profunda.

3. EXTENDIENDO TÉCNICA DE SEGMENTACIÓN PARA INCORPORAR LA VARIABLE TEMPORAL

Como se explicó en la sección anterior, los enfoques existentes para tratar *Concept Drift* no son suficientemente efectivos para encontrar las versiones de un proceso cuando este tiene cambios de distintos tipos. Para resolver esta problemática recurrimos a la técnica *Trace Clustering* basado en conservación de patrones; propuesta por Bose (Bose & van der Aalst, 2010), que permite realizar segmentación del *log* de eventos considerando solo las secuencias de actividades de cada traza. Esta técnica utiliza patrones comunes entre trazas para realizar la segmentación y presenta mejor desempeño que otras técnicas que usan solo las secuencias de actividades para realizar la segmentación.

Nuestro trabajo se basa en esta técnica, y la extiende incorporando la variable temporal de manera adicional a las que ya utiliza para realizar la segmentación.

En esta sección presentaremos *Trace Clustering* basado en conservación de patrones, y la extensión a esta técnica propuesta por nosotros.

3.1 *Trace clustering* basado en conservación de patrones

La idea básica que se plantea en este artículo (Bose & van der Aalst, 2010) es considerar subsecuencias de actividades que se repiten en múltiples trazos como conjuntos de características para realizar la segmentación. A diferencia del enfoque *K-gram* que considera subsecuencias de tamaño fijo, en este enfoque las

subsecuencias pueden ser de distintos largos. Cuando dos instancias tienen en común un significativo número de subsecuencias, entonces se asume que tienen una similitud estructural y estas instancias son asignadas al mismo *cluster*.

Para entender el concepto de subsecuencia, es necesario comprender algunos términos:

- *Secuencia*: Una secuencia corresponde a una sucesión de elementos. En este contexto, corresponde a una sucesión de actividades.

- *Maximal Pair:* En una secuencia, un *Maximal Pair* es un par de subsecuencias idénticas, tal que los elementos inmediatamente a la derecha e inmediatamente a la izquierda sean distintos (Figura 3-1).

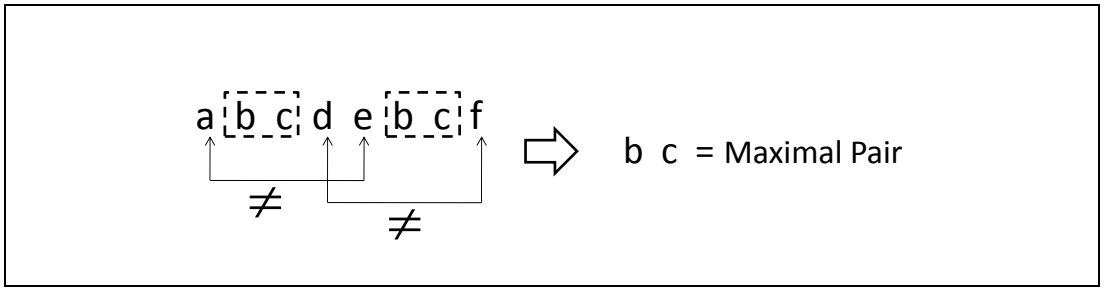


Figura 3-1: Representación gráfica de un *Maximal Pair*, "bc" en la secuencia "abcdebcf".

Hay seis tipos de subsecuencias, de las cuales las tres principales corresponden a: (a) *Maximal Repeat* (MR), (b) *Super Maximal Repeat* (SMR) y (c) *Near Super*

Maximal Repeat (NSMR), y tres secundarias que derivan de las anteriores. De acuerdo a los resultados presentados en el artículo donde éstas son definidas, con las subsecuencias SMR se obtienen modelos más simples, de acuerdo a las siguientes métricas:

- Promedio del número de actividades por *cluster*
- Número promedio de arcos
- Número promedio de arcos por nodo

Sin embargo, al analizar el comportamiento del proceso y utilizar otras métricas (que se detallan en la sección 4.3 de este documento), las subsecuencias que entregan mejores aproximaciones del comportamiento del proceso a través del tiempo son las MR.

Por lo tanto, haremos la definición formal solo de MR, ya que son las subsecuencias que utilizamos para desarrollar nuestro enfoque; el trabajo podría ampliarse y utilizar las otras subsecuencias.

- Maximal Repeat (MR): Un Maximal Repeat en una secuencia T, es definido como una subsecuencia que ocurre en un Maximal Pair en T. Intuitivamente, una MR corresponde a una subsecuencia de actividades que se repite más de una vez en el *log*. Representa un patrón común dentro del proceso.

En la Tabla 1 se puede ver un ejemplo donde se determina la MR existentes en una secuencia. Lo que hace esta técnica es construir una única secuencia a partir del

log de eventos, la cual es obtenida concatenando todas las trazas, pero incorporando un delimitador entre ellas. Luego, sobre esta única secuencia se aplica la definición de MR. El conjunto de todos las MR descubiertas en esta secuencia, con más de una actividad, es llamado Conjunto de Características (*Feature set*).

Tabla 1: Ejemplo de *Maximal Repeat* y Conjunto de Características.

Secuencia	***Maximal Repeat***	***Conjunto de Características***
bbbcd-bbbc-caa	{a, b, c, bb, bbbc}	{bb, bbbc}

A partir del Conjunto de Características, se crea una matriz que nos permite calcular la distancia entre las distintas trazas. Cada fila de la matriz corresponde a una traza y cada columna a una característica del Conjunto de Características. Los valores de la matriz corresponden al número de veces que se encuentra cada característica en las distintas trazas (Tabla 2). Esta matriz la llamaremos, matriz de características estructurales.

Tabla 2: Matriz de características estructurales.

Traza\Conjunto de Características	**bb**	**bbbc**
bbbcd	2	1
bbbc	2	1
caa	0	0

Este enfoque de segmentación basado en patrones, utiliza como técnica de segmentación el "*Agglomerative Hierarchical Clustering*" con criterio de mínima varianza (Ward, 1963), utilizando la distancia euclidiana para medir la distancia entre trazas, la cual se define de la siguiente manera.

$$dist\ (A,B) = \sqrt{\sum_{i=1}^{n} (T_{Ai} - T_{Bi)}{}^{2}}$$

Donde:

$dist\ (A,B) = distancia\ entre\ la\ traza\ A\ y\ la\ traza\ B$

$n = número\ de\ caracteríticas\ del\ Conjunto\ de\ Características$

$T_{Ai} = número\ de\ veces\ que\ está\ la\ característica\ i\ en\ la\ traza\ A$

3.2 Extendiendo *Trace clustering* para incorporar la variable temporal

Para identificar los distintos tipos de cambios que pueden ocurrir en los procesos de negocio, debemos buscar la manera de identificar las versiones de un proceso. Si solo miramos las características estructurales (control de flujo) dejamos fuera información respecto a su temporalidad. Ambas características, estructurales y temporales, son muy importantes, ya que la estructura nos indica cuán similar es una instancia a otra y la temporalidad nos indica qué tan cercanas en el tiempo están estas dos instancias. Nuestro enfoque busca identificar las distintas formas de ejecutar el proceso utilizando ambas características (estructurales y temporales) al mismo tiempo, tal como se ilustra en la Figura 3-2.

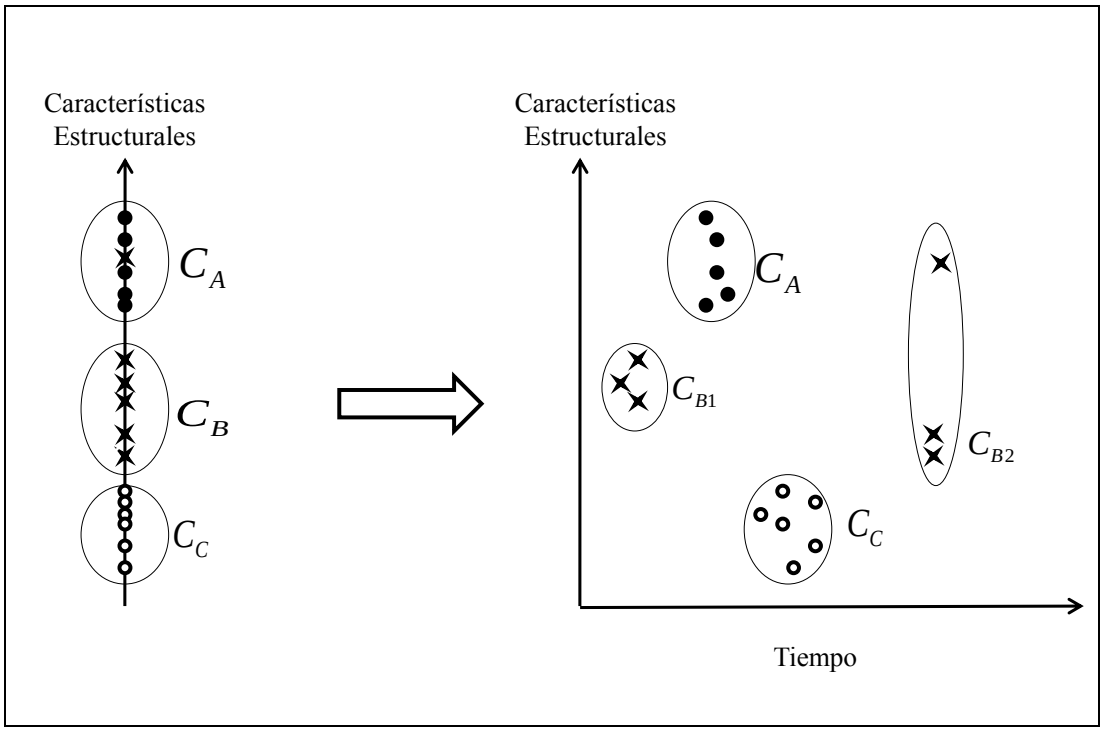


Figura 3-2: Ejemplo de la relevancia de considerar el tiempo en el análisis.

Al agregar la característica temporal como una dimensión adicional al espacio vectorial de características estructurales, los *clusters* generados entregan mayor información del proceso y permiten construir modelos que reflejen mejor la realidad.

Consideramos que la variable temporal relevante de analizar es el tiempo en que se comienza a ejecutar cada instancia del proceso. Otras variables, como por ejemplo la duración total que demora ejecutar una instancia completa del proceso, es una variable cuyo valor depende en fuerte medida de factores que son difíciles de

controlar, por ejemplo, la destreza de los ejecutores en realizar las actividades, o, el momento temporal en que se lleva a cabo, como días feriados o celebraciones especiales. Para mitigar el efecto de factores externos que son difíciles de controlar, utilizaremos como variable temporal solo el inicio de cada ejecución, que nos indica en qué momento se comienza a ejecutar una nueva instancia del proceso.

Para cada traza, almacenamos en la dimensión tiempo, el número de días que han transcurrido desde una marca de tiempo de referencia, por ejemplo, días transcurridos desde el 1 de enero de 1970 hasta la marca de tiempo en la cual se comienza a ejecutar la primera actividad de la traza (Tabla 3).

Tabla 3: Matriz de características estructurales más la dimensión temporal

Traza\Conjunto de Características	**bb**	**bbbc**	**Tiempo**
bbbcd	2	1	tiempo 1
bbbc	2	1	tiempo 2
caa	0	0	tiempo 3

En este nuevo enfoque también se utiliza "*Agglomerative Hierarchical Clustering*" con criterio de mínima varianza (Ward, 1963) como técnica de segmentación.

Para calcular la distancia entre dos trazas utilizamos la distancia euclidiana, pero modificada, de tal manera que considere al mismo tiempo las características estructurales y la característica temporal.

Primero definimos T_{Ji}, como la característica i de la traza J. Si la característica i no se encuentra en la traza J, su valor será 0, de lo contrario, su valor será el número de veces que la característica i se encuentra en la traza J.

$T_{J\,(n+1)}$ corresponde a la característica temporal de la traza J y su valor es el número de días (podrían ser horas, minutos o segundos, dependiendo del proceso) que han transcurrido desde una marca de tiempo de referencia. Se le da el índice (n+1) para indicar que se agrega a las características estructurales.

Definimos L como el conjunto de todas las trazas del *log*, luego la expresión $\max_{J \in L}(T_{Ji})$ representa la mayor cantidad de veces que está la característica i en alguna traza del *log* de eventos. De la misma forma, $\min_{J \in L}(T_{Ji})$ corresponde a la menor cantidad de veces que está la característica i en alguna traza.

$\min_{J \in L}(T_{J\,(n+1)})$ y $\max_{J \in L}(T_{J\,(n+1)})$ corresponden al mayor y menor, respectivamente, instante de tiempo en que se comenzó a ejecutar una traza del *log* de eventos.

También, definimos $D_E(A,B)$ y $D_T(A,B)$, como la distancia estructural y la distancia temporal entre la traza A y la traza B respectivamente.

$$D_E(A,B) = \sqrt{\sum_{i=1}^{n}\left(\frac{T_{Ai} - T_{Bi}}{\max\limits_{J \in L}(T_{Ji}) \; - \; \min\limits_{J \in L}(T_{Ji})}\right)^2}$$

$$D_T(A,B) = \sqrt{\left(\frac{T_{A(n+1)} - T_{B(n+1)}}{\max\limits_{J \in L}(T_{J\,(n+1)}) \; - \; \min\limits_{J \in L}(T_{J\,(n+1)})}\right)^2}$$

Donde:

$n = número\ de\ características\ del\ Conjunto\ de\ Características\ Estructurales$

Ambas distancias, D_E y D_T, están normalizadas, sin embargo el dominio de D_E es mayor al de D_T. Es por ello que también definimos Min_E, Max_E, Min_T and Max_T:

$$Min_E = \min_{A,B \in L} \sqrt{D_E(A,B)} \quad , \quad A \neq B$$

$$Max_E = \max_{A,B \in L} \sqrt{D_E(A,B)} \quad , \quad A \neq B$$

$$Min_T = \min_{A,B \in L} \sqrt{D_T(A,B)} \quad , \quad A \neq B$$

$$Max_T = \max_{A,B \in L} \sqrt{D_T(A,B)} \quad , \quad A \neq B$$

Min_E y Max_E corresponden a la distancia mínima y máxima (normalizada) entre todas las trazas, solo considerando las características estructurales.

Min_T y Max_T corresponden a la distancia mínima y máxima (normalizada) entre todas las trazas, solo considerando las características temporales.

La nueva forma para medir la distancia entre dos trazas, $dist\ (A, B)$, incorpora el parámetro μ , al que llamaremos ponderador de la dimensión temporal, que sirve para ponderar las características estructurales y temporales. Adicionalmente, esta nueva forma para medir la distancia ajusta D_E y D_T , de tal manera que el peso de ambas distancia, D_E y D_T, sea equivalente.

$$dist(A,B) = (1-\mu)\ \frac{D_E(A,B) - Min_E}{Max_E - Min_E} + \mu\ \frac{D_T(A,B) - Min_T}{Max_T - Min_T}$$

El ponderador de la dimensión temporal, μ, puede tener valores entre 0 y 1, según la relevancia que se le de a la característica temporal.

Notar que Max_T es igual a 1, cuando A y B cumplen lo siguiente:

$T_{A(n+1)} = \min_{J \in L}(T_{J\ (n+1)})$ y $T_{B(n+1)} = \min_{J \in L}(T_{J\ (n+1)})$, es decir, cuando la traza A es la primera instancia en comenzar a ejecutarse y la traza B es la última en comenzar.

4. EVALUACIÓN

Analizamos la técnica propuesta usando seis *log* de eventos obtenidos de distintos procesos sintéticos. Para medir su desempeño utilizamos el *plug-in Guide Tree Miner* (Bose & van der Aalst, 2010) disponible en ProM 6.1 [1] y también una versión modificada de este *plug-in* que incorpora los cambios propuestos.

La evaluación se llevó a cabo usando distintas métricas para medir la efectividad de clasificación del nuevo enfoque versus el enfoque base, aquel que realiza la segmentación del *log* utilizando solo las características estructurales. Como nuestro enfoque tiene como parámetro el ponderador de la dimensión temporal, μ, realizamos varias pruebas para ver cómo clasificaba el algoritmo al asignarle distintos valores a este ponderador. Adicionalmente, otro parámetro que se requiere como dato de entrada del *plug-in* es el número de *clusters* en que se desea segmentar el *log*. Para los distintos *logs* sintéticos se usó un número de *clusters* igual a la cantidad de modelos originales. De acuerdo a la literatura (Damer & Otros, 2012), no existe una técnica que permita definir el número óptimo de *clusters* a partir del *log* de eventos, y definir un técnica para esto está fuera del alcance de este trabajo.

En este capítulo se detalla la evaluación realizada.

---

[1] ProM es un *framework* extensible que soporta una variedad de técnicas de minería de procesos en forma de *plug-ins*. Se puede conseguir en www.processmining.org.

4.1 Construcción del *log* de eventos

Para poder probar la efectividad de la propuesta, se construyeron 6 *logs* de eventos sintéticos con CPN Tools (Ratzer & Otros, 2003) (Alves De Medeiros & Günther, 2005). Cada *log* está compuesto por instancias asociadas a distintos modelos, los cuales están distribuidos de distintas maneras a lo largo de un año. En las figuras 4-1 a 4-6 se presenta el detalle de cada log sintético. En la parte superior de cada diagrama se presenta el comportamiento del proceso en 12 meses, cada barra horizontal corresponde a las trazas de una versión del proceso (de acuerdo al momento en que se comienzan a ejecutar las trazas).

M_i corresponde al modelo i que representa a la versión i del proceso. Cada modelo está diagramado en notación BPMN (*Business Process Modeling Notation*), adicionalmente se indica la cantidad de trazas simuladas de cada modelo para el *log* correspondiente.

Por ejemplo, el Log (a) (Figura 4-1), tiene dos modelos o versiones del proceso, un modelo se ejecutó entre los meses de enero y agosto, mientras que el otro modelo se ejecutó entre los meses de junio y diciembre. Se puede observar que entre los meses de junio y agosto, existe un traslape entre los modelos, es decir, existen al mismo tiempo dos formas distintas de ejecutar el proceso.

El conjunto de *logs* trata de representar los distintos cambios que puede sufrir un proceso, pero principalmente el *Gradual Drift*, que es el cambio que ha sido poco estudiado en minería de procesos.

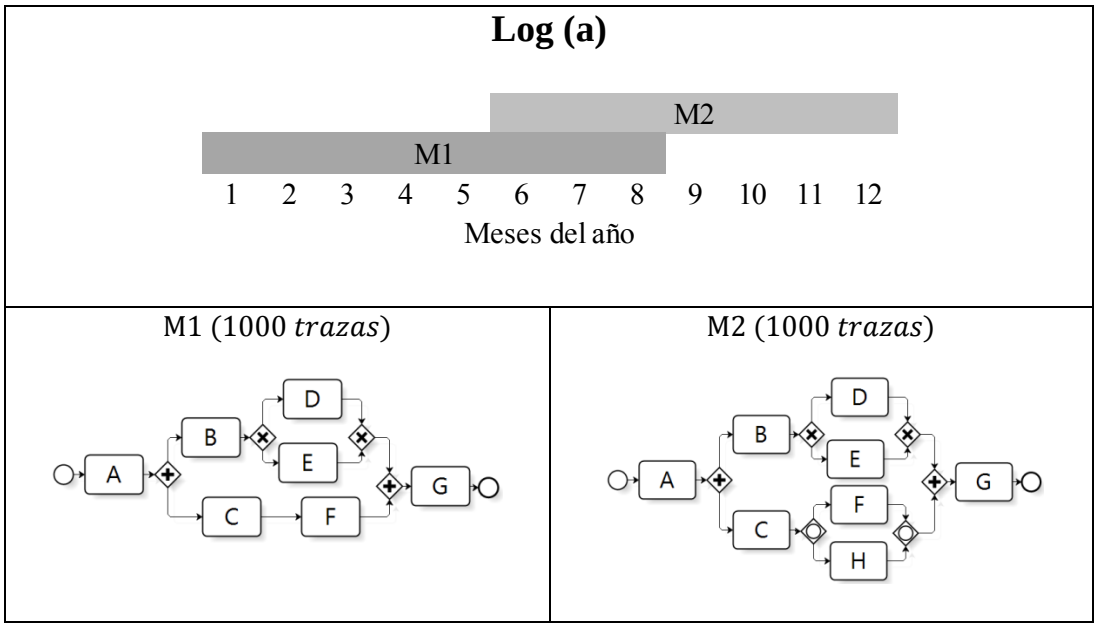


Figura 4-1: Log sintético (a).

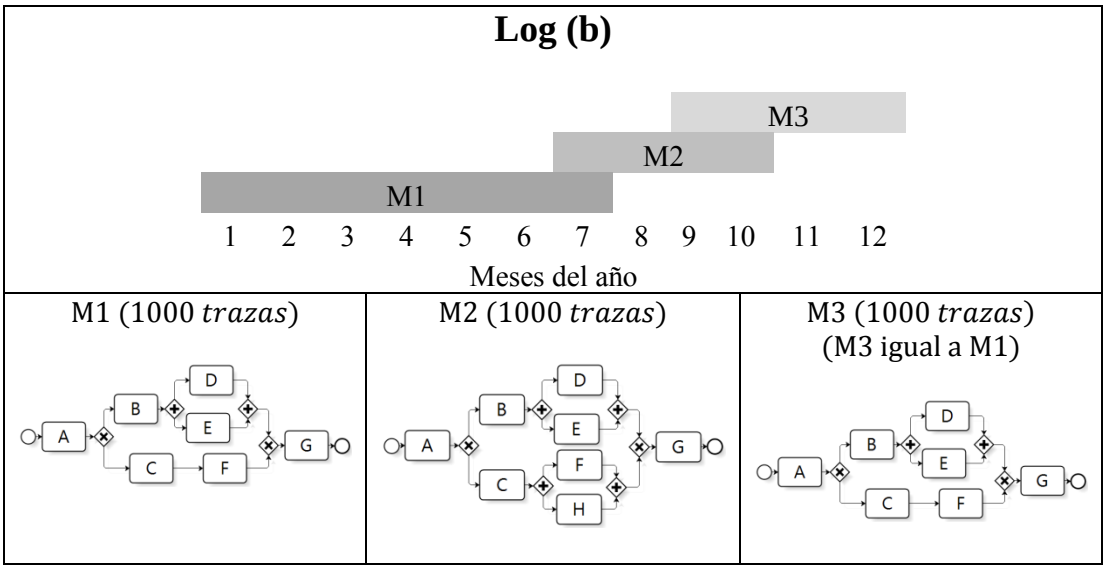


Figura 4-2: Log sintético (b).

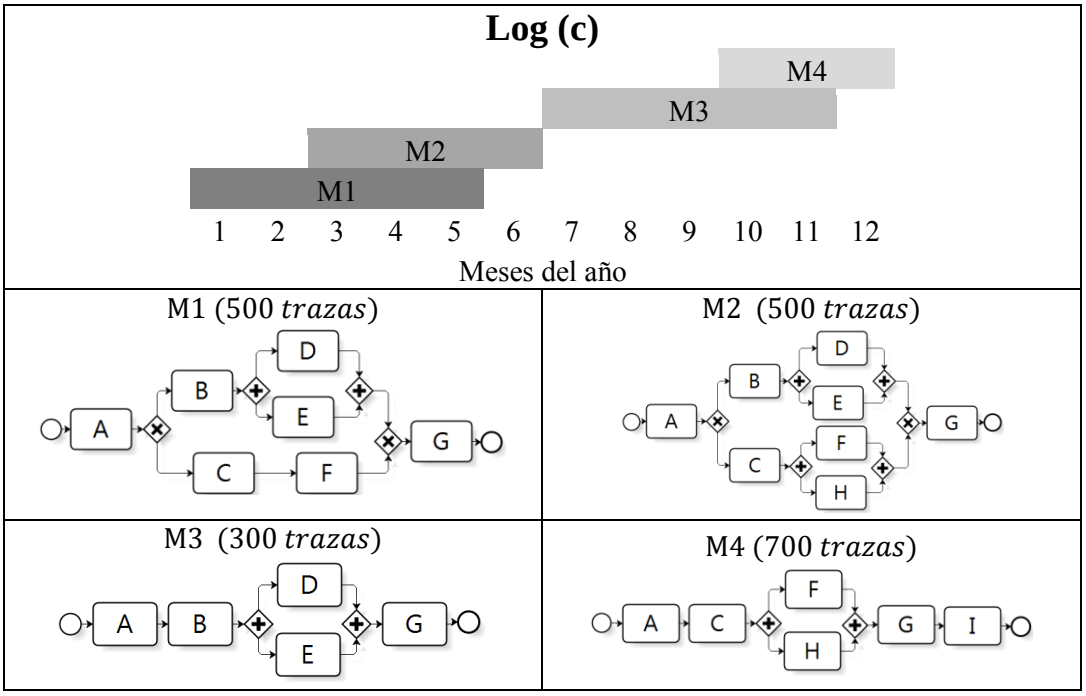


Figura 4-3: Log sintético (c).

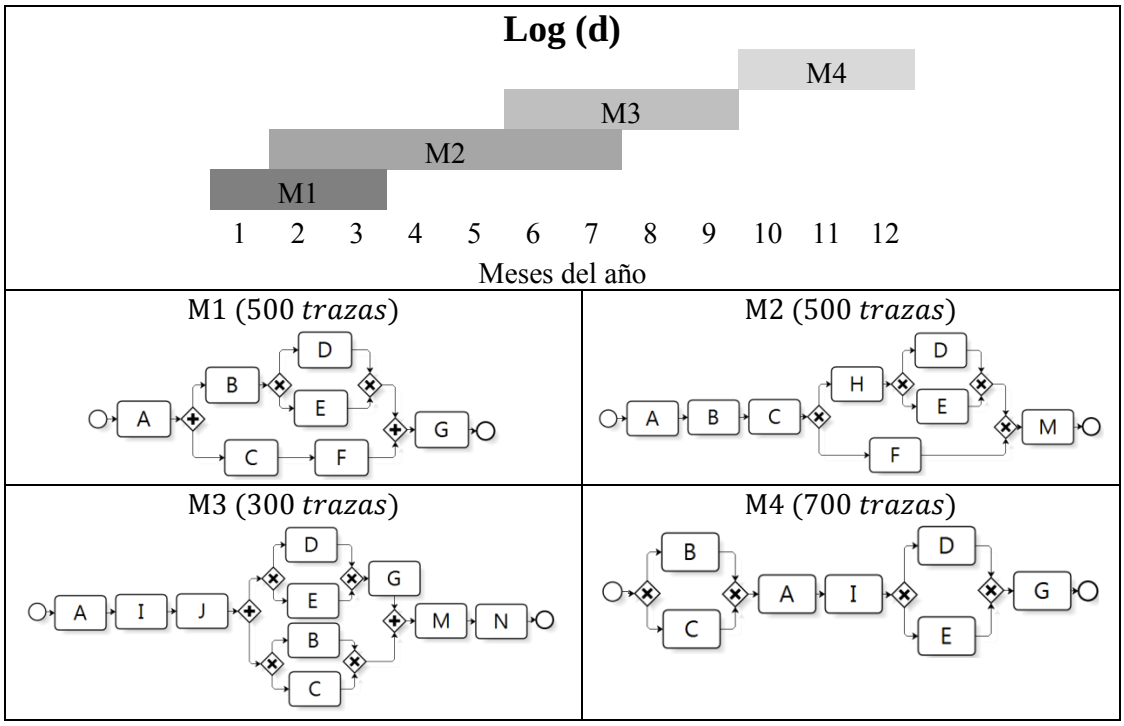


Figura 4-4: Log sintético (d).

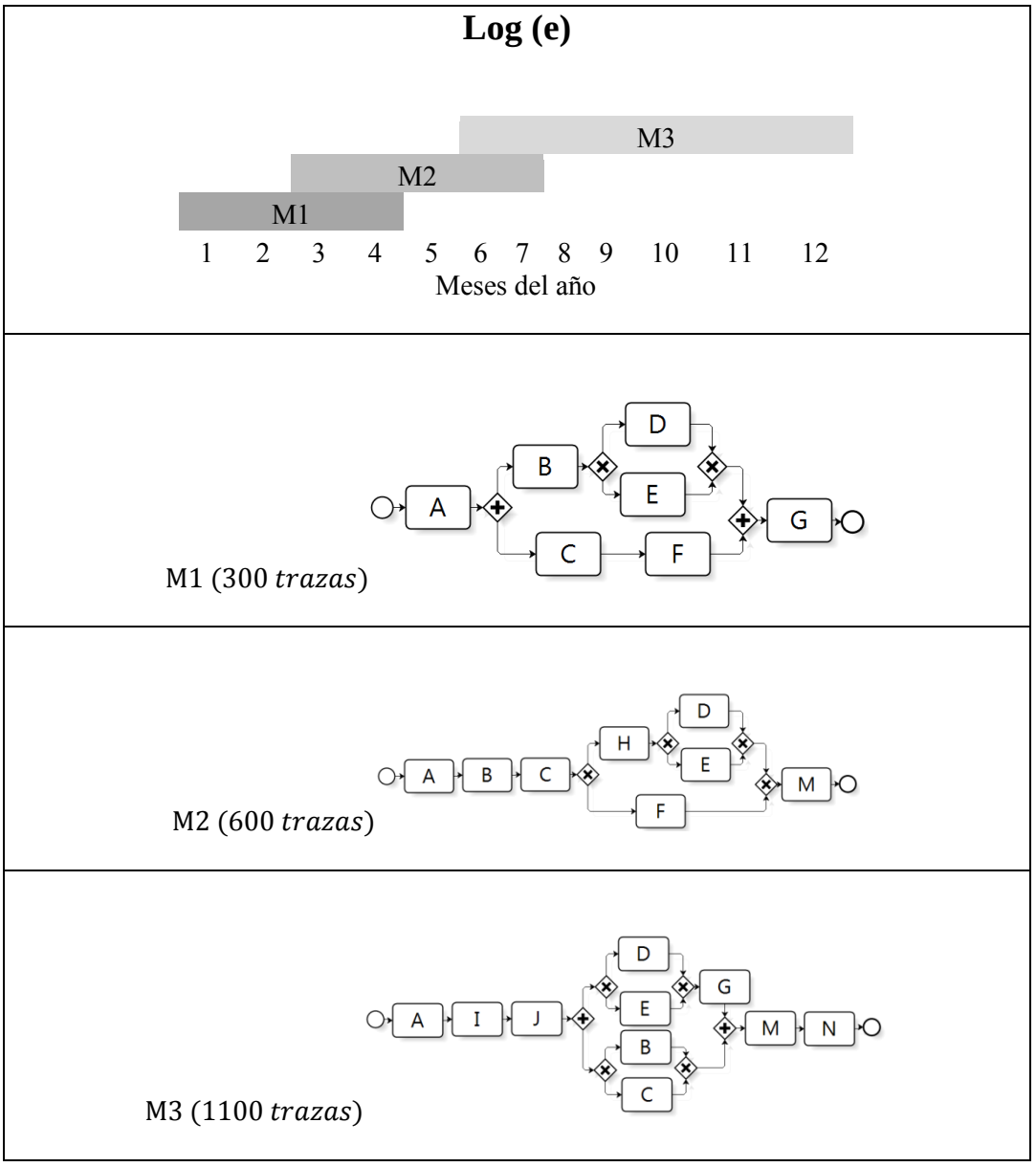


Figura 4-5: Log sintético (e).

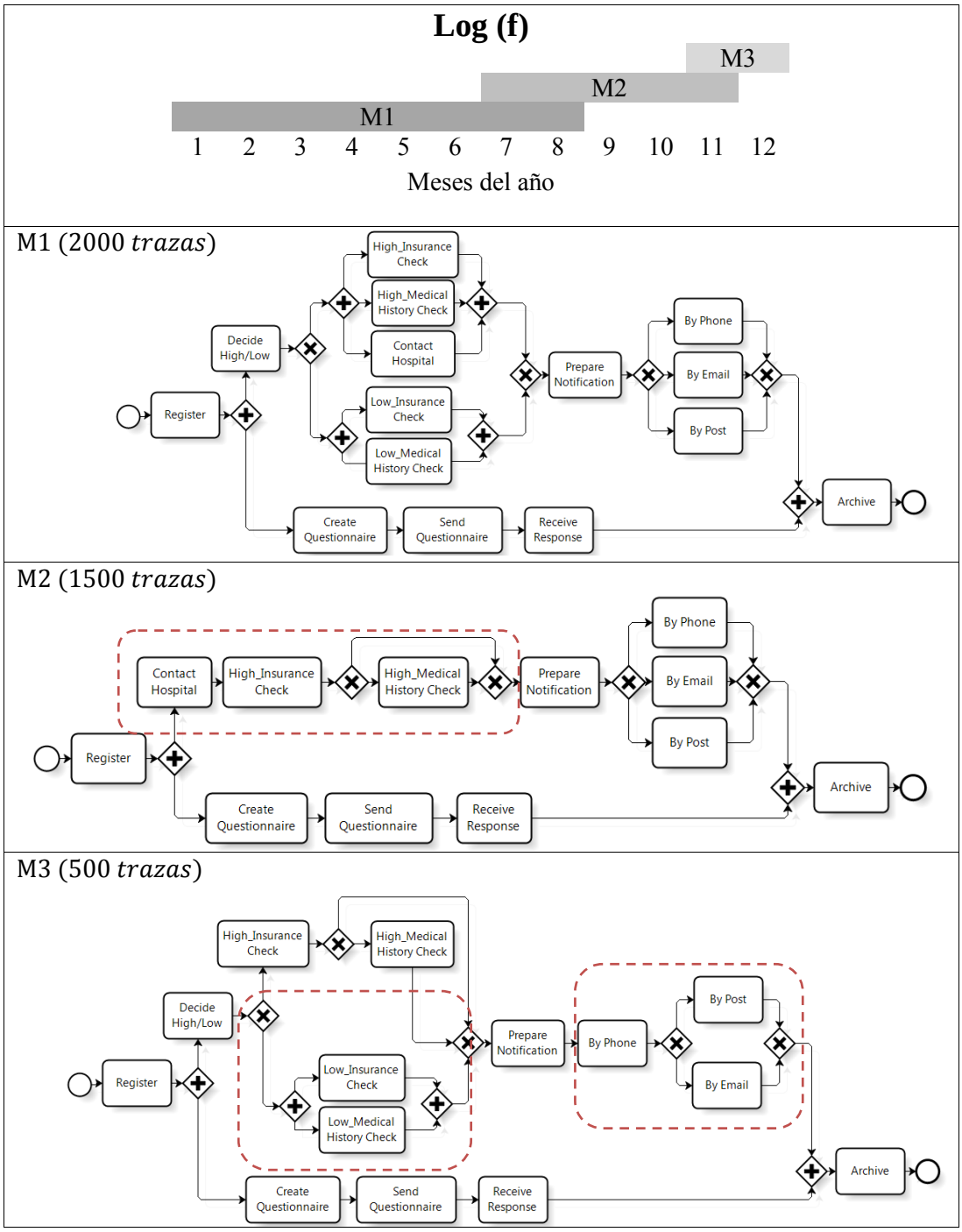


Figura 4-6: Log sintético (f).

De los seis *logs* sintéticos, el Log (f) (Figura 4-6) es especial, ya que se construyó pensando en un proceso real de emisión de un seguro médico. Este proceso comienza con el registro de una solicitud. Luego, por una parte se debe clasificar la solicitud como prioritaria o poco prioritaria, y notificar al solicitante; y por otra parte, se debe crear y enviar un cuestionario al solicitante para que indique detalles de la solicitud. Una vez clasificada la solicitud se realizan varias actividades en paralelo (de acuerdo al tipo de solicitud), para finalmente preparar una notificación con el resultado de la solicitud. Esta notificación se le debe comunicar al solicitante por alguno de los tres medios disponibles, teléfono, correo o correo electrónico. Una vez realizadas todas las actividades mencionadas y habiendo recibido respuesta del solicitante, se finaliza el proceso archivando la solicitud.

Los rectángulos con línea segmentada representan las variaciones que ocurren en un modelo con respecto al modelo anterior. Por ejemplo, en M1, se debe clasificar la solicitud como alta o baja, pero en M2 ya no está dicha posibilidad, ya que siempre se clasifican las solicitudes como altas. Adicionalmente, se cambia el orden en que se ejecutan las actividades, restringiendo la flexibilidad de hacerlas en paralelo. Luego, en M3, siempre se debe notificar al solicitante del resultado de la solicitud por teléfono, y adicionalmente escoger otro medio para notificar, el cual puede ser correo o correo electrónico. Mientras que en M1 y M2 solo se debe escoger uno de los tres medios.

Dado estos *logs* de eventos, nuestro primer objetivo es evaluar si el algoritmo presentado en este documento es capaz de detectar las distintas versiones de los procesos.

4.2 Experimentos

Se realizaron varios experimentos con los 6 *logs* sintéticos. Para ilustrar estos experimentos se puede observar la Figura 4-7, donde se muestra la secuencia de pasos que se realizaron con uno de los *logs*, el Log (a) (Figura 4-1).

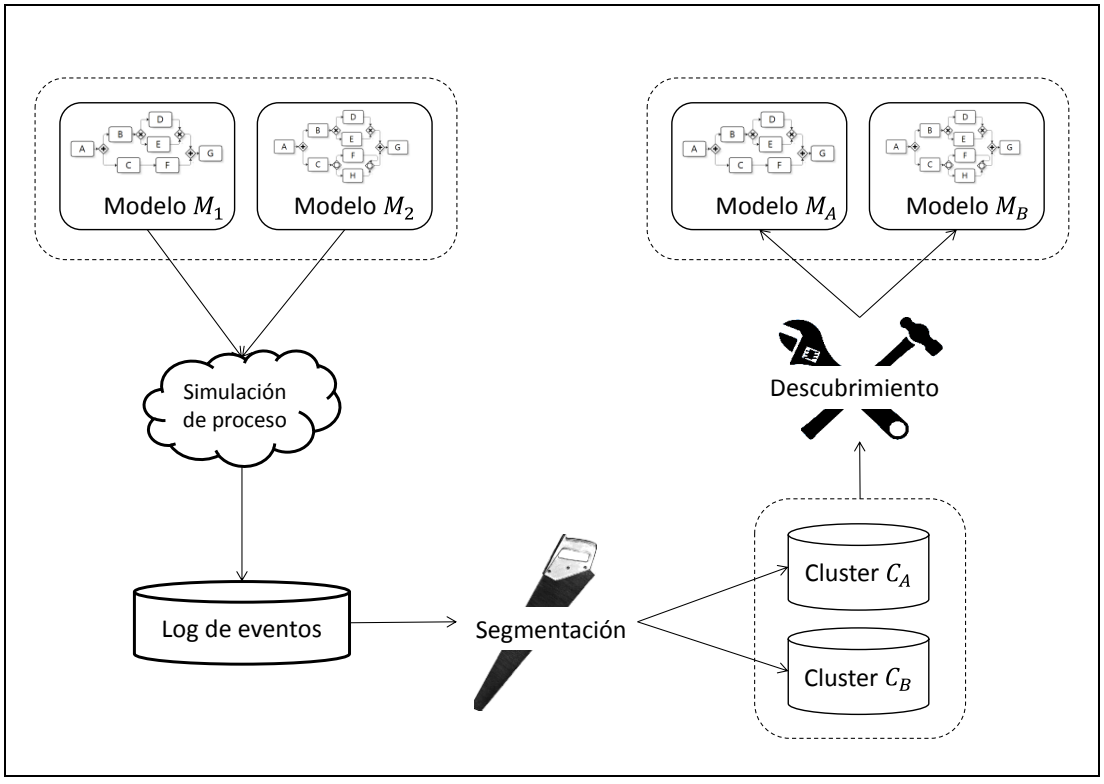


Figura 4-7: Pasos para realizar las pruebas experimentales.

Para crear el *log* sintético, se usó simulación a partir de dos modelos, M1 y M2, utilizando CPN Tools. El método se inicia con la técnica de segmentación que recibe el *log* sintético y genera los *clusters*, en este caso dos *clusters* (C_A y C_B), ya que se utiliza el conocimiento que el *log* se generó a partir de dos modelos (M_1 y M_2). Se aplican dos técnicas de segmentación:

- *Trace clustering* basada en conservación de patrones.
- *Trace clustering* extendida, incorporando la variable temporal.
    - En esta última, se utilizó el ponderador de la dimensión temporal, μ, con distintos valores, que variaban entre 0 y 1.

Luego, para cada uno de estos *clusters* se realiza descubrimiento de proceso, generando dos nuevos modelos (M_A y M_B). Esta misma secuencia de pasos se realizó para los 6 *logs* sintéticos.

4.3 Definición de métricas

Para evaluar el enfoque propuesto debemos definir métricas de desempeño, que nos indiquen si el enfoque realiza o no una buena segmentación.

El desempeño del enfoque puede ser medido en dos momentos (Figura 4-8):

- Conformidad 1: Las métricas de Conformidad 1 se miden entre un modelo original y un *cluster* generado, para lo cual se debe asignar un modelo a cada *cluster*. Esto se hace considerando el modelo con mayor probabilidad de haber generado el *cluster*. Esta probabilidad se calcula de acuerdo a la

métrica *accuracy*. Se calcula el *accuracy* entre todos los modelos y todos los *clusters*, y posteriormente a cada *cluster* se le asigna el modelo con el que obtuvo mayor *accuracy*. Cabe recordar que en este caso se seleccionó un número de *clusters* igual a la cantidad de versiones del proceso (lo cual no se sabría en una situación real), por lo que habrá un modelo para cada *cluster*, y viceversa.

- Conformidad 2: En Conformidad 2, las métricas se miden entre un modelo original y un modelo generado, donde el modelo generado corresponde al modelo descubierto a partir del *cluster* que se seleccionó en Conformidad 1, como coincidente con el modelo original.

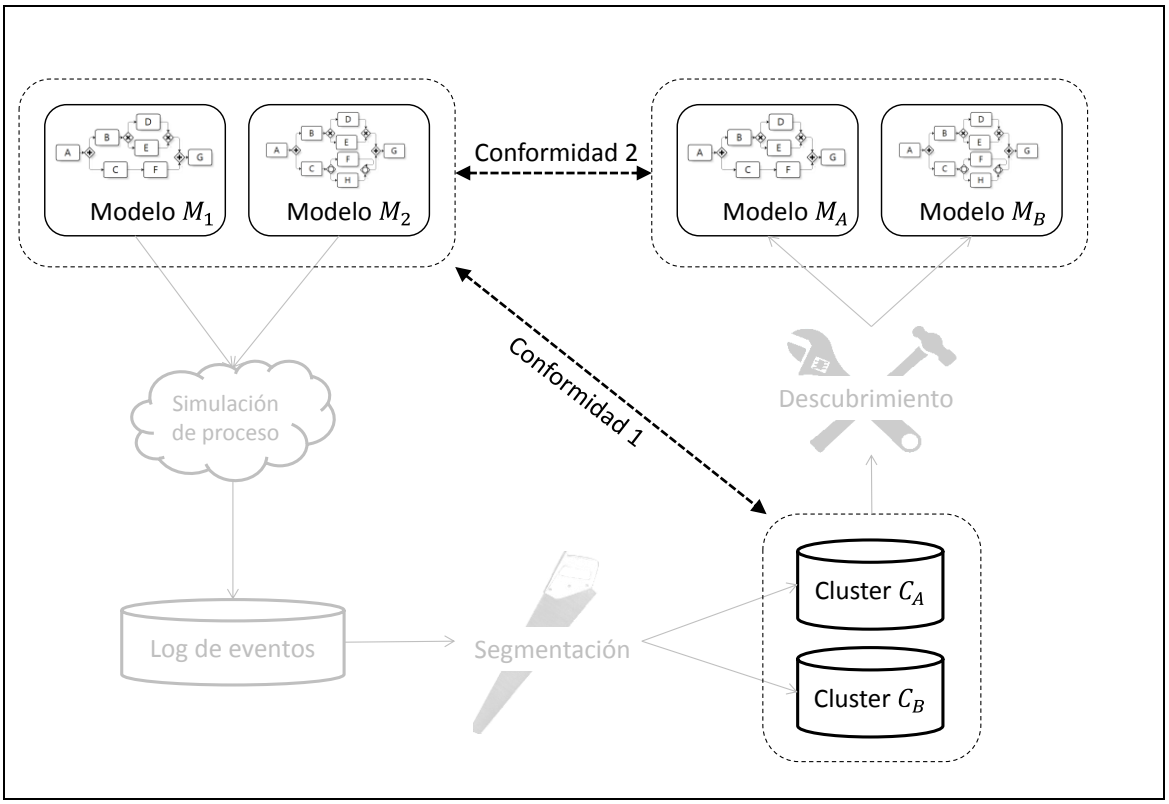


Figura 4-8: Conformidad para medir el desempeño de la segmentación.

Las métricas que se utilizan para medir Conformidad 1 son las siguientes:

- *Accuracy*: Indica el número de instancias correctamente clasificadas en cada cluster, de acuerdo a lo que se conoce del *log* de eventos original. Sus valores están entre 0% y 100%, donde 100% corresponde a cuando la segmentación se hizo de manera exacta.
- *Fitness*: Indica cuánto del comportamiento observado en un *log* de eventos (por ejemplo, cluster C_A) es capturado por el modelo original del proceso

(por ejemplo, modelo M_2) (Rozinat & van der Aalst, 2006). Sus valores están entre 0 y 1, donde 1 corresponde a que el modelo es capaz de representar todas las trazas del *log*. En la práctica, para el cálculo de *fitness*, se utilizó el *plug-in Conformance Checker* disponible en ProM 5.2.

- Precisión: Se refiere a la generalidad del modelo, donde se prefiere modelos con un mínimo de comportamiento para representar lo mejor posible el registro del *log* de eventos. Sus valores están entre 0 y 1, donde 1 significa que el modelo no tiene comportamiento adicional a lo que indican los trazos. En la práctica, se utilizó el *plug-in ETConformance* (Muñoz-Gama & Carmona, 2010) disponible en ProM 6.1, para el cálculo de la precisión.

Las métricas que se utilizan para medir Conformidad 2 son las siguientes (Rozinat, Alves De Medeiros, Günther, Weijters, & van der Aalst, 2007):

- *Behavioral Precision* (B_P): Cuantifica la precisión del modelo descubierto con respecto al modelo original que fue usado para generar este *log*. Cuando el modelo descubierto permite comportamiento adicional respecto al modelo original dado y al *log*, B_P tendrá un valor menor a 1.
- Structural Precison (S_P): Cuantifica cuántas conexiones tienen en común el modelo descubierto y el modelo original. Cuando el modelo descubierto

tiene conexiones que no aparecen en el modelo original, S_P tendrá un valor menor a 1.

- Behavioral Recall (B_R): Cuantifica la generalización del modelo descubierto con respecto al modelo original que fue usado para generar este *log*. Cuando el modelo original permite comportamiento adicional respecto al modelo descubierto, B_R tendrá un valor menor a 1.
- Structural Recall (S_R): Cuantifica cuántas conecciones tienen en común el modelo descubierto y el modelo original. Cuando el modelo original tiene conexiones que no aparecen en el modelo descubierto, S_R tendrá un valor menor a 1.

Los valores de estas 4 métricas están entre 0 y 1, donde 1 es el mejor valor esperado. Estas 4 métricas se calcularon utilizando los *plug-ins* "*Behavioral Precision and Recall Analysis*" y "*Structural Precision and Recall Analysis*", disponibles en ProM 5.2.

En el estado del arte no se presenta ninguna métrica que resuelva el problema de medir si la segmentación de un *log* es buena o mala. En este trabajo propusimos un set de métricas que analizan los resultados en dos instancias, directamente sobre los *clusters* generados y cuando se genera un modelo a partir de cada *cluster*. Sin embargo, estas métricas no se pueden analizar de manera

independiente, es necesario complementarlas para tener una mejor visión de los resultados.

Las métricas propuestas tienen el problema que sólo se puede utilizar cuando conocemos información a priori del proceso. Por lo que ante un caso real, estas métricas no se podrían aplicar.

4.4 Resultados

Es importante destacar que la comparación que realizamos es cuando el *log* es segmentado en un número de *clusters* igual al número de modelos originales con los que se creó el *log* de eventos, lo cual no se podría saber a priori en un caso real.

La Tabla 4 resume los resultados de aplicar el enfoque base (solo características estructurales) y el nuevo enfoque (variando el valor del parámetro μ), a los distintos *logs* de eventos sintéticos presentados entre la Figura 4-1 a la Figura 4-6. En esta tabla se muestra la métrica de a*ccuracy*, la cual nos indica el porcentaje de trazas correctamente clasificadas.

Para cada *log* utilizado, el porcentaje de *accuracy* más alto se alcanza con nuestro enfoque, pero con distintos valores de μ (varía entre 0,2 y 0,9). La razón de que se comporte distinto con distintos valores de μ, se debe a que en cada *log* la relevancia que tiene la distribución temporal de las trazas versus la estructura del proceso no es la misma.

Tabla 4: Métrica de *accuracy* calculada para los 6 *logs* sintéticos de prueba.

Enfoque	***μ***	***Log (a)***	***Log (b)***	***Log (c)***	***Log (d)***	***Log (e)***	***Log (f)***
Base	-	57%	38%	55%	86%	99%	53%
Nuevo	0,0	59%	52%	49%	82%	59%	51%
	0,1	65%	52%	49%	82%	59%	68%
	0,2	87%	52%	63%	**100%**	59%	64%
	0,3	87%	52%	63%	**100%**	59%	62%
	0,4	95%	52%	63%	**100%**	59%	63%
	0,5	**100%**	52%	63%	**100%**	59%	**95%**
	0,6	**100%**	52%	73%	**100%**	**100%**	94%
	0,7	96%	**89%**	73%	**100%**	**100%**	67%
	0,8	78%	88%	73%	74%	84%	55%
	0,9	79%	77%	**77%**	68%	77%	78%
	1,0	81%	78%	68%	73%	72%	55%

Es interesante hacer un análisis más detallado de los *logs* que obtuvieron mejor desempeño con valores extremos de μ, es decir, los Logs (c) y (d). El Log (c) (Figura 4-3), posee cuatro versiones del proceso que son muy similares, ya que todos los cambios que se realizan, corresponden a eliminación de actividades (salvo M4, al cual se le agrega una actividad nueva), de tal manera que la nueva versión de ejecución del proceso, es uno de los caminos de la versión anterior (para cada una de las versiones). Al ser tan parecidas las versiones del proceso, el valor de μ debe ser alto, para darle mayor importancia a la característica temporal

(que es en la que más se diferencian) y así poder realizar una segmentación más precisa.

Por otro lado, al analizar el Log (d), se puede decir que cada una de las 4 versiones que posee son muy distintas entre sí; se agregan, permutan y eliminan actividades al pasar de una versión a otra. Por esta razón, con valores bajos de μ (0,2), es posible clasificar con 100% de exactitud las trazas del Log, ya que la estructura es decisiva para agruparlas; por tal motivo, se le da mayor peso a la componente estructural.

Utilizamos el Log (f) para hacer un análisis más profundo de los resultados, midiendo para este Log, todas las métricas definidas tanto en Conformidad 1 como en Conformidad 2 (Tabla 5).

Tabla 5: Detalle de métricas al analizar el Log (f) de la Figura 4-6.

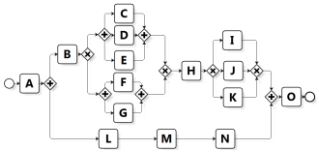
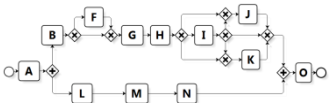
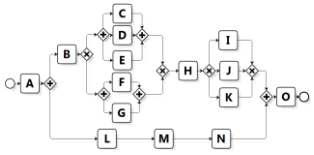
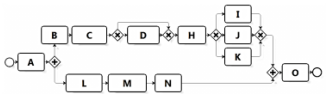
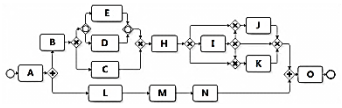
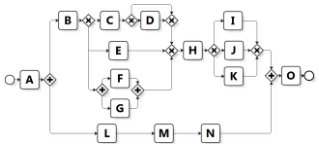
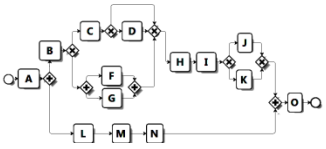
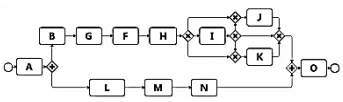
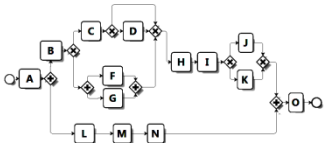
Enfoque	*μ*	*Accuracy*	*Fitness*	*Precisión*	*Promedio* B_P *y* S_P	*Promedio* B_R *y* S_R	*Promedio*
Base	-	53%	0,93	0,78	0,77	0,81	0,76
Nuevo	0,0	51%	0,93	0,73	0,73	0,70	0,72
	0,1	68%	0,93	0,81	0,86	0,86	0,83
	0,2	64%	0,92	0,80	0,83	0,82	0,80
	0,3	62%	0,92	0,80	0,80	0,78	0,78
	0,4	63%	0,92	0,79	0,83	0,86	0,81
	0,5	**95%**	**0, 95**	0,85	**0,97**	**0,96**	**0, 94**
	0,6	94%	**0,95**	**0,86**	0,95	0,94	0,93
	0,7	67%	0,92	0,84	0,84	0,89	0,83
	0,8	55%	0,94	0,87	0,88	0,94	0,84
	0,9	78%	0,94	0,81	0,89	0,95	0,87
	1,0	55%	0,93	0,87	0,88	0,95	0,84

Todas las métricas calculadas para el Log (f) tienen buen desempeño cuando el parámetro μ vale 0,5 o 0,6. Al ser promediadas las cinco métricas, el promedio más alto se alcanza con μ igual a 0,5. Para este caso particular, aplicamos el *plug-in Heuristic Miner* de ProM 6.1 como técnica de descubrimiento para obtener los modelos asociados a cada *cluster*.

Tabla 6 muestra los modelos originales del Log (f), los modelos generados con el enfoque base y los modelos generados con el nuevo enfoque con µ igual a 0,5, todos en notación BPMN. Cada fila corresponde a una versión del proceso. Se puede observar que los modelos generados con el enfoque base, cuyo *accuracy* es de 53%, difieren bastante de los modelos originales. Por otro lado, los modelos generados con el nuevo enfoque tienen un 95% de *accuracy*, lo cual se ve reflejado en los modelos, ya que los modelos 1 y 3 coinciden con los originales, mientras el modelo 2 solo se diferencia en una parte del diagrama.

Aún si el porcentaje de *accuracy* alcanzara el 100%, no se puede asegurar que los modelos originales y los generados vayan a ser iguales, ya que esto dependerá de la técnica de descubrimiento que utilicemos. Por tal razón, las métricas que se miden en Conformidad 2 tienen un poco de sesgo, ya que son calculadas a partir de los modelos generados con el *plug-in Heuristic Miner*.

Tabla 6: Comparación de los modelos generados con el enfoque base y el nuevo enfoque, y los modelos originales del Log (f) de la Figura 4-6.

	Modelo Original (Diferentes versiones)	Modelo Generado (Enfoque base)	Modelo Generado (Nuevo enfoque, μ = 0,5)
Modelo 1	z 		
Modelo 2			
Modelo 3			

5. CONCLUSIONES Y TRABAJO FUTURO

En este documento se presentan las limitaciones de las actuales técnicas de segmentación en minería de procesos, las cuales se centran en agrupar ejecuciones similares (estructuralmente) de un proceso, de tal manera de formar grupos homogéneos de ejecuciones, para que el análisis sobre cada grupo sea de mayor simplicidad que si se analiza el conjunto de datos completo. Al centrarse solo en la estructura de las ejecuciones, dejan de lado el comportamiento del proceso a través del tiempo (*Concept Drift*). Para ello surgen nuevas técnicas, pero que también presentan limitaciones, ya que se centran en encontrar los puntos en que cambia el proceso, limitándose a un tipo de cambio, el *Sudden Drift*. Dada esta situación, presentamos un enfoque que utiliza la lógica que usan las técnicas de segmentación, de manera de encontrar las distintas versiones de un proceso cuando éste presente distintos tipos de cambios, permitiendo entender las variaciones que ocurren en el proceso y cómo realmente se está ejecutando en la práctica a través del tiempo.

Nuestro trabajo se centra en la identificación de los modelos asociados a cada versión del proceso, pero no identifica qué tipo de cambios (Drift) presenta el proceso a través del tiempo. La técnica que proponemos, es una herramienta que ayuda a las personas involucradas en el negocio a tomar decisiones, por ejemplo, se puede determinar si los cambios que se producen en la ejecución del proceso, son realmente los esperados, y en base a esto, tomar mediadas si se descubren

comportamientos anormales. También, al conocer y comparar las distintas versiones de un proceso, se puede identificar buenas y malas prácticas, las cuales son de alta utilidad al momento de querer mejorar o estandarizar los procesos.

En este documento presentamos un conjunto de métricas para medir el desempeño del enfoque. El desempeño se considera bueno, cuando el enfoque es capaz de segmentar los datos de la misma forma en que fueron creados. Por lo tanto, estas métricas requieren información a priori del proceso, lo cual no es posible en casos reales.

Cada métrica aquí presentada mide distintos aspectos que son difíciles de analizar por sí solos, sin embargo, al utilizarlos en conjunto, permiten tener una visión de distintas perspectivas del problema, haciendo más completo el análisis.

Un aspecto clave de nuestro enfoque de segmentación, es el valor que se le da al ponderador de la dimensión temporal, μ, el cual está estrechamente relacionado con la naturaleza del proceso. Valores altos de μ le dan mayor importancia al tiempo para realizar la segmentación, mientras que valores bajos de μ, le dan más importancia a las características estructurales del proceso.

Realizamos diferentes experimentos con *logs* sintéticos de procesos, de tal manera que representaran los distintos cambios que puede sufrir un proceso. Los resultados muestran que el enfoque propuesto en este documento tiene un mejor desempeño en la segmentación del *log* y que existe al menos un valor del parámetro μ que permite

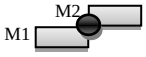
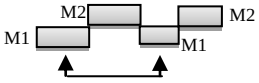
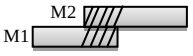
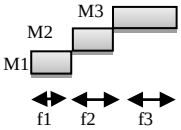
entregar mejores resultados en comparación a utilizar solo la técnica de segmentación estructural (*Trace clustering* basado en patrones). Esto se logra, ya que nuestro enfoque es capaz de agrupar las trazas del *log* de tal manera de identificar similitud estructural y cercanía temporal al mismo tiempo.

Un de las métricas utilizadas es el *accuracy*. En algunos experimentos, esta métrica, alcanza el 100%, es decir, que todas las trazas son clasificadas correctamente. Cuando no se alcanza el 100% de *accuracy,* se debe a que hay procesos que tienen versiones que pese a ser distintas, son similares entre sí, pudiendo incluso tener trazas ejecutables en las dos versiones del proceso, lo cual hace que las clasificación no sea exactamente igual a lo que se esperaba. A pesar de que puede existir esta diferencia en algunos casos, la segmentación que entrega nuestro enfoque tiene mejor significancia para el negocio, ya que describe el dinamismo del proceso a través del tiempo.

Nuestro trabajo futuro en esta línea de investigación es probar este nuevo enfoque con procesos reales. También queremos trabajar en desarrollar los algoritmos existentes para que sean capaces de determinar automáticamente el número óptimo de *clusters*; para ello es necesario definir nuevas métricas que nos permitan calcular el número óptimo de *clusters* sin saber a priori información de las versiones del proceso.

Resulta interesante extender este trabajo para que la técnica propuesta sea capaz de clasificar los cambios que sufren los procesos y dar información automática respecto a ellos (Tabla 7).

Tabla 7: Información que se puede extraer automáticamente al detectar el tipo de cambio en el proceso.

Tipos de cambios	**Información que se puede obtener**
	Punto de cambio al pasar de una versión a otra
	Periodicidad con que se repite una misma versión del proceso
	Periodo de traslape en que conviven dos versiones del proceso
	Frecuencia con que se van realizando cambios en el proceso

BIBLIOGRAFÍA

Alves De Medeiros, A. K., & Günther, C. (2005). Process Mining : Using CPN Tools to Create Test Logs for Mining Algorithms. *Proceedings of the Sixth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, (págs. 177–190).

Bose, R. P., & van der Aalst, W. (2010). Trace Clustering Based on Conserved Pastterns : Towards Achieving Better Process Models. *BUSINESS PROCESS MANAGEMENT WORKSHOPS*, (págs. 170-181).

Bose, R. P., van der Aalst, W., Zliobaite, I., & Pechenizkiy, M. (2011). Handling Concept Drift in Process Mining. *23rd International Conference on Advanced Information Systems Engineering.* Londres.

Bose, R., & van der Aalst, W. (2009). Context Aware Trace Clustering : Towards Improving Process Mining Results. *SIAM*, (págs. 401-412).

Damer, N., Jans, M., Depaire, B., & Vanhoof, K. (2012). Making Compliance Measures Actionable: A New Compliance Analysis Approach. *Lecture Notes in Business Information Processing*, 159-164.

Luengo, D., & Sepúlveda, M. (2011). Applying Clustering in Process Mining to find different versions of a business process that changes over time. *Lecture Notes in Business Information Processing*, 153-158.

Muñoz-Gama, J., & Carmona, J. (2010). A fresh look at precision in process conformance. *Proceeding BPM'10 Proceedings of the 8th international conference on Business process management*, (págs. 211-226).

Ratzer , A. V., Wells , L., Lassen , H. M., Laursen , M., Frank , J., Stissing , M. S., y otros. (2003). CPN Tools for editing, simulating, and analysing coloured Petri nets. *Proceedings of the 24th international conference on Applications and theory of Petri nets* (págs. 450-462). Eindhoven: Springer-Verlag.

Rozinat, A., & van der Aalst, W. (2006). Conformance testing: Measuring the fit and appropriateness of event logs and process models. *Business Process Management Workshops*, (págs. 163-176).

Rozinat, A., Alves De Medeiros, A. K., Günther, C., Weijters, A., & van der Aalst, W. (2007). Towards an Evaluation Framework for Process Mining Algorithms. *Genetics.*

Song, M., Günther, C., & van der Aalst, W. (2009). Trace Clustering in Process Mining. *4th Workshop on Business Process Intelligence (BPI 08),* (págs. 109-120). Milano.

Stocker, T. (2011). Time-based Trace Clustering for Evolution-aware Security Audits. *Proceedings of the BPM Workshop on Workflow Security Audit and Certification*, (págs. 471-476). Clermont-Ferrand.

van der Aalst, W. (2011). *Process Mining, Discovery, Conformance and Enhancement of Business Processes.*

van der Aalst, W., Adriansyah, A., Alves de Medeiros, A. K., Arcieri, F., Baier, T., Blickle, T., y otros. (2011). Process Mining Manifesto.

van der Aalst, W., van Dongen, B., Herbst, J., Maruster, L., Schimm, G., & Weijters, A. (2003). *Data & Knowledge Engineering*, 237-267.

van der Aalst, W., Weijters, T., & Maruster, L. (2004). Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 1128 - 1142 .

Ward, J. (1963). Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, 236-244.