Detección de equivalencias entre procesos de negocios semánticos

MONOGRAFIA PRESENTADA PARA OPTAR AL TITULO DE INGENIERO DE SISTEMAS

David Camilo Corrales Muñoz

Jose Eduardo Gómez Daza

Director: Dr. Ing. Juan Carlos Corrales

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Departamento de Telemática - Grupo de Ingeniería Telemática

Línea de Investigación de Aplicaciones y Servicios en Internet

Popayán, Febrero de 2011

Tabla de Contenido

Índice de figuras

Índice de Tablas

Anexos

Capítulo I

1 INTRODUCCIÓN

1.1 CONTEXTO

En la actualidad existen empresas que soportan los objetivos de la organización en su infraestructura TI. Para garantizar la agilidad ante el mercado y la rápida recuperación de la inversión en las diferentes áreas de negocio, dichas empresas basan sus fortalezas en arquitecturas que resuelven los problemas de heterogeneidad e interoperabilidad que facilitan el aprovisionamiento de sus servicios (típicos de cualquier infraestructura TI). En éste sentido la arquitectura orientada a servicios (SOA) es una alternativa que garantiza la rápida salida de los servicios al mercado, como respuesta a la competitiva presión global. SOA es una evolución de la computación distribuida diseñada para permitir la interacción de componentes software, llamados servicios Web, a través de la red [1].

Un servicio Web es definido como una aplicación software que utiliza interfaces estándares que permiten a cualquier servicio interoperar con otras aplicaciones en la Web. Una de las principales ventajas de los servicios web es la capacidad de convertirse en servicios más complejos, a partir de funciones suministradas por diferentes componentes, los cuales están dinámicamente integrados para garantizar una tarea de negocio más robusta, estos servicios Web complejos reciben el nombre de procesos de negocio [2]. Teniendo en cuenta lo dicho anteriormente, las aplicaciones SOA son creadas como una composición de varios servicios Web o procesos de negocio compartidos entre múltiples aplicaciones [1].

La composición que tiene como resultado un proceso de negocio, se refiere a la integración de un conjunto de servicios Web que trabajan buscando un objetivo de negocio en común. Las principales formas de composición de procesos son la coreografía y la orquestación [3]. En la orquestación, un proceso coordina un grupo de diferentes operaciones, donde los servicios involucrados no conocen todo el proceso del que forman parte, sólo el coordinador central de la orquestación conoce el objetivo global; de esta forma, la orquestación es centralizada con definiciones explícitas de las operaciones y con el orden de invocación de los servicios Web. En la coreografía no existe un coordinador central, cada servicio Web involucrado en la coreografía conoce exactamente cuando se tiene que ejecutar, las operaciones que debe realizar y los socios con los cuales tiene que interactuar. En la coreografía todos los servicios Web necesitan conocer cuál es el siguiente servicio que se va a ejecutar, qué operación, que mensajes se intercambian y el tiempo de ejecución [4].

La composición de procesos de negocio se puede realizar de manera manual o automática. La composición manual se realiza mediante protocolos para la descripción de procesos de negocio de manera sintáctica y semántica. La composición automática busca a través del descubrimiento automático encontrar los servicios más adecuados, con el fin de integrarlos en un proceso y lograr un objetivo de negocio sin necesidad de intervención humana [3].

En este marco diversos autores indican que el descubrimiento de servicios se puede realizar de dos formas: automática o manual. En [3] afirman que la técnica más utilizada

actualmente es la localización manual, la cual consiste en buscar las descripciones de servicios Web en repositorios y registros para posteriormente invocarlos, pero la recuperación manual de servicios se torna algo compleja cuando el número de estos es considerable, por lo cual utilizar esta técnica es menos eficiente que descubrir servicios de manera automática. La recuperación automática reduce considerablemente el tiempo invertido en la selección de un servicio con el fin de ser reutilizado, lo cual implica una significativa reducción en los costos de desarrollo de nuevas aplicaciones o servicios de valor agregado.

Teniendo en cuenta las consideraciones mencionadas anteriormente, el presente trabajo de grado aborda el descubrimiento de servicios de manera automática debido a las ventajas mencionadas, haciendo uso del protocolo de descripción semántico BPEL4SWS.

1.2 PLANTEAMIENTO DEL PROBLEMA

En la actualidad existen lenguajes que permiten describir Procesos de Negocio (BP) de manera sintáctica y semántica. Los lenguajes sintácticos como BPEL, WSCL y WS-CDL, representan parámetros del servicio desde la perspectiva de la interfaz funcional que lo describe, mientras que los lenguajes semánticos como OWL-S [5] y WSMO [6], los cuales cuentan hoy en día con herramientas de modelado tales como OWL-S MODELLER [7] y WSMO studio [8] respectivamente, describen tanto el comportamiento y la forma de utilizar el servicio, como también el proveedor y el consumidor que interactúan con él.

En este marco, la recuperación de BP está clasificada en dos categorías: Recuperación de procesos a nivel sintáctico y semántico. El primero está basado en métodos de búsqueda que consideran las interfaces y el flujo de control del elemento que representan. El segundo permite que los sistemas de recuperación puedan inferir acerca de los conceptos sobre los cuales el cliente está realizando su búsqueda, utilizando criterios ontológicos que tienen en cuenta el significado y las relaciones entre conceptos.

En este sentido, diversos autores expresan que los algoritmos de recuperación de procesos de negocio que actúan sobre protocolos de descripción sintáctica como BPEL, no garantizan la correcta recuperación, ya que al no considerar la semántica de los procesos dichos algoritmos omiten componentes que son similares a los buscados por el cliente, u ofrecen al usuario resultados con baja medida de precisión denominados “falsos positivos” [9],[10].

Ahora bien, el descubrimiento de procesos de negocio semánticos es uno de los aspectos más avanzados para la implantación rápida y eficiente de nuevos servicios en una organización o la localización y modificación de los ya existentes. Descubrir procesos enriquecidos con anotaciones semánticas es complejo y requiere de tecnologías de punta como las ontologías, algoritmos de inferencia, algoritmos de comparación semántica, repositorios de procesos de negocio entre otras [3]. En la actualidad existen proyectos de investigación centrados en este enfoque, como es el caso de SUPER [11], cuyo objetivo es la unión de la gestión de procesos de negocio (BPM) con la tecnología semántica.

Recientes trabajos de investigación han enfocado sus esfuerzos en definir técnicas de recuperación de procesos a nivel semántico. Por ejemplo, en WSMX [12] los autores abordan el descubrimiento de servicios descritos con WSMO y en [13] se evalúa la similitud sobre modelos de procesos OWL-S. Sin embargo estos lenguajes comparados con los protocolos de descripción sintáctica (BPEL) no son ampliamente usados por la

industria del software, por lo tanto, los prototipos de recuperación de procesos implementados por los trabajos antes mencionados no han sido integrados a entornos de desarrollo orientados a servicios tales como OpenSOA de Netbeans o SOA Tools Platform (STP) de Eclipse.

Teniendo en cuenta todas las consideraciones descritas anteriormente se hace necesario definir un algoritmo que identifique equivalencias semánticas entre los servicios comparados con el fin de construir una clasificación de servicios basados en criterios de similitud. Por lo tanto, el presente proyecto plantea la siguiente pregunta de investigación: ¿Cómo facilitar la selección de procesos de negocio descritos con un lenguaje estándar como BPEL4SWS que utilice mecanismos de recuperación semántica para ser integrado a un entorno de desarrollo orientado a servicios?

1.3 ESCENARIO DE MOTIVACIÓN

Considere una organización que modele sus tareas mediante procesos de negocio compuestos por un conjunto de servicios Web. Suponga que al interior de la misma se desea modelar nuevos procesos de negocio que satisfagan las necesidades emergentes a partir de los procesos de negocio previamente modelados. En consecuencia, se considera necesario contar con un mecanismo automático de recuperación de procesos que involucre aspectos como la estructura del proceso, operaciones, entradas, y salidas de las actividades que componen los procesos, de esta manera los procesos recuperados puedan adaptarse ó reutilizarse. Este mecanismo le ahorra tiempo de análisis, gestión y reingeniería de procesos al desarrollador de servicios.

1.4 CONTRIBUCIONES

Las principales contribuciones de éste proyecto de grado son:

- **Adaptación del algoritmo Netmatch a un entorno SOA.** Adaptar el algoritmo Netmatch a un entorno de desarrollo orientado a servicios, ya que el área de desempeño de este algoritmo es la bioinformática, basado en el emparejamiento de redes biológicas.

- **Adición de nuevas características al algoritmo Netmatch.** El algoritmo Netmatch detecta grafos y sub grafos de una consulta determinada. Como principal aporte se adiciono, la detección de subestructuras que estén contenidas en el grafo de consulta según los grafos publicados (caso inverso).

- **Prototipo para la detección de equivalencias entre procesos de negocio semánticos.** El presente proyecto de grado, creo un mecanismo que recupera procesos de negocio de forma sintáctica y semántica, para procesos BPEL4SWS. El prototipo desarrollado consta de 5 fases. En la primera fase se recuperan procesos de negocio según el flujo de control del proceso de consulta, además si el usuario desea, el algoritmo realiza un filtro que recupera procesos que además de tener la misma estructura, contenga los mismos tipos de actividades que el proceso de consulta. En la segunda fase se evalúan los procesos recuperados en la fase 1 por el atributo operación de las actividades básicas que componen cada proceso. La tercera y cuarta fase nuevamente evalúa los procesos recuperados en la fase 1 por los atributos entrada y salida respectivamente. Finalmente la última

fase realiza una evaluación total de los procesos analizados en las anteriores etapas.

- El prototipo realizado en el presente trabajo de grado frente a otros trabajos como [3, 14], tiene un optimo desempeño en tiempo de respuesta, al permitir el emparejamiento de un proceso de consulta contra 60 procesos publicados, mientras que los trabajos mencionados anteriormente solo permiten el emparejamiento 1:1 y no 1:N, además los tiempos de ejecución de [3, 14] son considerablemente altos.

- **Uso de una ontología para la evaluación de procesos.** Utilizar una ontología para el descubrimiento de procesos al inferir sobre los conceptos de un dominio, enriquece las búsquedas recuperando un conjunto de procesos útil para el usuario.

- **Creación de un repositorio de procesos BPEL4SWS.** La generación de 60 procesos de negocio BPEL4SWS a partir de procesos BPMO, se considera como un aporte para el tema actual de investigación ya que BPEL4SWS carece de procesos de negocio publicados en la web y el proyecto que desarrollo este lenguaje [11] restringe el acceso a repositorios de gran relevancia, ya que son considerados confidenciales.

- Motivar el desarrollo y la investigación sobre la detección de equivalencias entre procesos de negocio semánticos en la comunidad académica de la Universidad del Cauca, implementando un prototipo de recuperación de procesos que cree un precedente alrededor de éste tema, y despertar así el interés por estas tecnologías en la comunidad investigativa del país, debido al positivo panorama social y económico factible de obtener con su difusión.

- El presente trabajo de grado continuo con la investigación realizada por el Dr. Juan Carlos Corrales M, en su proyecto doctoral denominado “Behavioral Matchmaking for Service Retrieval”, tomando como punto de partida el algoritmo NetMatch.

1.5 ALCANCE

En el presente proyecto se analizaron mecanismos para la determinación del grado de correspondencia y similitud semántica los cuales fueron adaptados hacia el descubrimiento y recuperación de procesos de negocio, para poder ser utilizados en el presente trabajo de grado. Este proyecto solo considera el descubrimiento de procesos de negocio abstractos, y solo tiene en cuenta las actividades básicas y estructuradas, más la actividad interactionActivity innatas de los procesos modelados con el lenguaje BPEL4SWS. Por otra parte las fases de evaluación de entradas y salidas de las actividades del proceso, solo funcionan para la ontología SSID, asimismo las entradas y salidas de las actividades básicas que pertenecen a los procesos de negocio del repositorio deben estar obligatoriamente enriquecidas semánticamente con dicha ontología. Para la evaluación de calidad del prototipo se usaron solo 5 procesos de consulta contra 60 procesos publicados, pertenecientes al área de las telecomunicaciones.

1.6 CONTENIDO DE LA MONOGRAFÍA

Organizada en siete capítulos presentados a continuación:

- **Capítulo II. Estado del Arte**

  Presenta una visión general sobre los trabajos relacionados y los conceptos que giran en torno a la detección de equivalencias entre procesos de negocios semánticos.

- **Capítulo III. Algoritmo para la detección de subestructuras**

  Este capítulo corresponde a la contextualización en torno a la temática de detección de sub estructuras de grafos, comenzando por definir los conceptos más relevantes de la teoría de grafos, para posteriormente realizar una definición formal de la técnica de detección de grafos isomorfos y los distintos algoritmos que implementan esta técnica. Seguidamente se verifica los estudios de rendimiento en tiempo de ejecución de los algoritmos de detección de isomorfismo, para la posterior selección de uno de ellos (VF2).

- **Capítulo IV. Mecanismo para la detección de equivalencias entre procesos de negocio semánticos**

  Esta sección explica el mecanismo manual para transformar procesos de negocio BPEL4SWS a grafos. Por otra parte se explican las etapas que conducen a la adaptación del algoritmo seleccionado en el capítulo III. Dichas fases son: i) Recuperación de procesos de negocio por flujo de control, ii) Evaluación de los procesos de negocio según los nombres de operación, iii) Evaluación de los procesos de negocio según las entradas, iv) Evaluación de los procesos de negocio según las salidas, y v) Evaluación total de los procesos recuperados.

- **Capítulo V. Arquitectura y Metodología de desarrollo del prototipo**

  Este capítulo aborda la descripción de los principales artefactos de la metodología de desarrollo utilizada para la solución del prototipo y la arquitectura del mismo.

- **Capítulo VI. Experimentación y Evaluación**

  Presenta la especificación del plan pruebas realizadas sobre el prototipo y los resultados de las evaluaciones de desempeño obtenidos a partir de la ejecución de dichas pruebas.

- **Capítulo VII. Conclusiones y Trabajos Futuros**

  Finalmente, se analizan los resultados del trabajo realizado, se detallan las principales contribuciones obtenidas en la ejecución del proyecto y se expone un conjunto de recomendaciones importantes para el desarrollo de trabajos futuros.

Capítulo II

2 ESTADO DEL ARTE

En este capítulo se presentan las bases teóricas y estándares necesarios para comprender la temática del presente trabajo de grado, el cual propone la detección de equivalencias entre procesos de negocios semánticos. Posteriormente se expone el estado actual de los trabajos relacionados respecto a los mecanismos de recuperación de procesos de negocio. Finalmente, se realiza un resumen que presenta los principales aportes de este capítulo.

2.1 CONCEPTOS GENERALES

2.1.1 SERVICIOS WEB

Los Servicios Web son definidos como aplicaciones software modulares e independientes, las cuales pueden ser descritas, publicadas, localizadas e invocadas a través de una red, basadas en estándares abiertos de internet, cuyo fin es la interoperabilidad entre distintas aplicaciones [15].

Según lo dicho anteriormente y tomando en cuenta la definición presentada en [16] se puede decir que un servicio Web pertenece a un conjunto de estándares y protocolos que intercambian datos entre aplicaciones software, las cuales pueden estar desarrolladas en diferentes lenguajes de programación. Bajo este esquema, los Servicios son descritos utilizando interfaces estándares WSDL (Web Services Description Language) y son invocados a través del protocolo SOAP (Simple Object Access Protocol). La implementación de servicios Web proporciona ventajas como la interoperabilidad entre aplicaciones software independientemente de sus propiedades o de las plataformas sobre los cuales estén instalados.

2.1.2 PROCESOS DE NEGOCIO

Un proceso de negocio, puede definirse, como un servicio Web complejo con funciones suministradas por diferentes servicios Web, los cuales ya existen en la Web y están dinámicamente integrados para garantizar una tarea de negocio más compleja [2], dicho en otras palabras un proceso de negocio es la combinación de un conjunto de actividades dentro de una empresa con una estructura que describe su orden lógico y dependencia, cuyo objetivo es producir un resultado deseado [17].

Los procesos de negocio, son la base para comprender, la forma en que opera un negocio o empresa en sus diferentes áreas, por lo cual su estructura suele ser amplia y compleja, requiriendo ser modelada para facilitar su entendimiento y ejecución. Por lo tanto, se hace necesario modelar los procesos de negocio, para organizar y documentar la información relativa a las actividades que estos ejecutan [18].

2.1.3 SERVICE-ORIENTED ARCHITECTURE (SOA)

La arquitectura orientada a servicios (SOA por sus siglas en ingles) es una evolución de la computación distribuida diseñada para permitir la interacción de componentes software, llamados servicios, a través de la red. Las aplicaciones SOA son creadas como una composición de varios servicios y estos servicios pueden ser compartidos entre múltiples aplicaciones [1]. SOA no es como tal una tecnología, es más bien un concepto o marco de diseño para el desarrollo de sistemas que integren aplicaciones independientemente del lenguaje en que fueron implementados y/o sistema operativo en el que estén implantados, de tal forma que pueda acceder a sus funcionalidades por medio de la red [19].

Una solución SOA consiste de tres principales componentes lógicos: Consumidores (Consumers), Infraestructura SOA (SOA Infrastrucutre) y Productores (Producers). A su vez, la infraestructura SOA se compone de: Aplicaciones (Application), Servicio de Soporte (Service Support) y Servicios (Services).

- **Consumidores:** Es una entidad que hace uso de los servicios ofrecidos por el productor.
- **Aplicaciones:** Provee una interfaz grafica para que los consumidores ejecuten tareas.
- **Servicios:** Es una entidad que ejecuta una tarea específica cuando es invocada.
- **Servicios de Soporte (también llamado ESB- Enterprise Service BUS):** Es una entidad que provee un conjunto de funcionalidades a SOA, como enrutamiento, seguridad, entre otras.
- **Productores:** Es una entidad que ofrece un servicio o funcionalidad especifica.

La orientación a servicios permite que las aplicaciones desarrolladas bajo SOA tengan un bajo acoplamiento, sean más fáciles de reutilizar, ampliamente escalables y faciliten la integración entre sistemas heterogéneos.

Finalmente vale la pena resaltar que algunos autores como [20], añaden un nuevo rol a SOA, que recibe el nombre de Agregador de Servicio, el cual actúa como un proveedor ofreciendo los servicios compuestos, y como un Cliente al consumir los servicios existentes.

Como ya se mencionó anteriormente, en el enfoque de SOA las aplicaciones pueden ser creadas como una composición de varios Servicios compartidos entre múltiples aplicaciones, por lo tanto a continuación son descritos brevemente los dos principales modelos de composición de Servicios: la coreografía y la orquestación [3].

- Se considera ***orquestación de servicios***, cuando un proceso coordina un grupo de diferentes operaciones, donde los Servicios involucrados no conocen todo el proceso del que forman parte, sólo el coordinador central de la orquestación conoce el objetivo global. Para la centralización de los servicios compuestos, la orquestación toma como base las definiciones explícitas de las operaciones y el orden de invocación de los servicios Web participantes en la cooperación.

- Por su parte, en la ***coreografía de servicios*** no existe un coordinador central, cada Servicio Web involucrado en la coreografía conoce exactamente cuando tiene que ejecutar, las operaciones que debe realizar y los socios con los cuales

tiene que interactuar. En la coreografía todos los Servicios Web necesitan conocer cuál es el siguiente Servicio que se va a ejecutar, qué operación, que mensajes se intercambian y el tiempo de ejecución [4].

Por otro lado, cabe mencionar que la composición de Servicios Web a través de los modelos antes descritos involucra una fase previa denominada “recuperación”, la cual se encarga de seleccionar los servicios más similares que puedan ser fácilmente adaptados al flujo de ejecución del nuevo servicio compuesto.

En este marco, diversos autores indican que la recuperación de servicios es realizada de dos formas: automática o manual. En [3] los autores afirman que la técnica más utilizada actualmente es la localización manual, la cual consiste en buscar las descripciones de servicios Web en repositorios y registros para posteriormente invocarlos en la composición. Si bien es cierto que esta técnica recuperan servicios lo más aproximados a las necesidades del usuario, su utilización en términos de tiempo de recuperación, es menos eficiente que descubrir servicios de manera automática ya que los registros o repositorios donde se encuentran los Servicios Web pueden contener cientos de ellos.

Por su parte, la recuperación automática proporciona una selección mucho más flexible de servicios, ya que éstos pueden ser descubiertos automáticamente considerando diversos elementos que los describen tal como su estructura, funcionalidades, mensajes intercambiados, semántica del servicios, propiedades no funcionales, fiabilidad, seguridad, calidad de servicio, entre otras [21].

Concluyendo, dado que las aplicaciones SOA son creadas como una composición de varios servicios (Procesos de Negocio) los cuales se encuentran en registros o repositorios, y considerando que éstos pueden contener cientos de ellos, el presente trabajo de grado abordará la temática de detección de equivalencias entre procesos de negocios que facilite la recuperación de éstos.

2.1.4 PROTOCOLOS PARA LA DESCRIPCIÓN DE SERVICIOS

En la actualidad existen numerosos protocolos para la descripción de procesos de negocio, los cuales se dividen en dos clases: protocolos sintácticos y protocolos semánticos.

Los protocolos sintácticos proveen una interfaz y un lenguaje estándar para poder obtener la integración y automatización de procesos, ejemplos de estos protocolos son: BPEL4WS y BPMN, mientras que los protocolos semánticos son aquellos que pueden inferir en los conceptos en los cuales el cliente está realizando una solicitud o proporcionando un servicio, para esto se utilizan criterios ontológicos que tiene en cuenta el significado y las relaciones entre conceptos [3]. WSML, BPMO, OWLS y BPEL4SWS son algunos de los protocolos semánticos más utilizados en la actualidad.

A continuación se describirán los protocolos antes mencionados.

PROTOCOLOS SINTÁCTICOS

WS-BPEL (Web Services – Business Process Execution Language)

WS-BPEL (Web Services Business Process Execution Language) es un lenguaje estandarizado por OASIS [22], basado en XML que permite la especificación, ejecución, y la descripción de orquestación de servicios Web basados en procesos de negocio, los cuales exportan e importan funcionalidades mediante el uso de interfaces WSDL de los servicios Web [23]. BPEL soporta el modelado de dos tipos de procesos: ejecutables y abstractos. Un proceso Abstracto es un protocolo de negocio, el cual especifica el intercambio de mensajes entre diferentes partes desde la perspectiva de una sola organización sin revelar el comportamiento interno [24]. Un proceso ejecutable, especifica el orden de ejecución entre un número de actividades que lo constituyen, las partes que están involucradas, los intercambios de mensajes entre estas partes y los mecanismos de manejo de fallos y excepciones [25]. Ahora bien, la estructura de un proceso WS-BPEL se divide en cuatro secciones: definición de relaciones con los socios externos, que es el cliente que utiliza el proceso de negocio y los servicios Web a los que llama el proceso; definición de las variables que emplea el proceso; definición de los distintos tipos de manejadores que puede utilizar el proceso (manejadores de fallos y de eventos), y descripción del comportamiento del proceso de negocio [26].

Por otra parte, WS-BPEL ofrece un conjunto de actividades para el desarrollo de los procesos de negocio, actividades que se clasifican en: básicas y estructuradas.

- **Actividades básicas:** Describen pasos elementales del comportamiento del proceso, estas son:
    - <**receive>:** Espera por la llegada de un mensaje externo de petición para una operación de un Servicio Web proporcionado por el propio proceso.
    - **<invoke>:** Invocación de una operación de un Servicio Web proporcionado por una entidad externa al proceso.
    - **<reply>:** Permite al proceso de negocio enviar una respuesta a una petición.
    - **<assing>:** Asigna un valor a una o más variables.
    - **<wait>:** Permite a un proceso de negocio especificar un tiempo de espera, antes de realizar una acción.
    - **<terminate>:** Termina la ejecución de un proceso de negocio.
    - **<empty>:** Permite a un proceso de negocio especificar un tiempo durante el cual no se hace nada.

- **Actividades estructuradas:** son usadas para definir el flujo de control y la lógica del negocio [14], además de determinar el orden en que se ejecuta un conjunto de actividades. Estas actividades ayudan a estructurar el proceso BPEL ayudando a mantener el flujo del proceso. Las principales actividades estructuradas son:

    - **<sequence>:** Reúne un conjunto de actividades para ser ejecutadas secuencialmente.
    - **<flow>:** Especifica un conjunto de actividades que deben ser ejecutadas concurrentemente.
    - **<pick>:** Esta actividad espera que ocurra uno de varios eventos y ejecuta la actividad asociada a dicho evento.
    - **<while>:** Ejecuta iterativamente una o varias actividades, mientras se cumpla una condición.

    - **<switch>:** Permite ejecutar actividades siempre y cuando se cumplan ciertas condiciones.

Además de las actividades básicas y estructuras, WS-BPEL brinda un conjunto de elementos que cumplen otras funciones:

- **PartnerLinks:** En WS-BPEL los Servicios Web siempre son modelados como partnerLink, además este estándar permite describir la relación entre los socios del proceso (procesos que interactúan con otros). Los PartnerLinks son usados para describir Servicios socios.
- **Variables:** Estas se asocian con tipos de mensajes que pueden ser especificados como entradas o salidas para actividades como: receive, invoke y reply, por medio de las variables se mantienen los mensajes intercambiados entre socios del negocio, como también mensajes usados solamente dentro del proceso.
- **Compensation:** Proceso por el cual se provee una alternativa para restaurar los datos que una actividad realizo, en caso de que ocurra algún problema.
- **Correlation Sets:** Permite la asociación de un mensaje entrante con una instancia del proceso, además permite especificar grupos de operaciones correlacionadas en una instancia de un Servicio Web.
- **Event Handlers:** Estos solo son ejecutados cuando ocurre un evento, si no ocurre un evento el proceso lo abandona y continua la ejecución. Hay dos clases de eventos:
    - **<onEvent>:** Evento que espera un mensaje entrante.
    - **<onAlarm>:** Utilizado cuando se necesita esperar cierto lapso de tiempo antes de que algo ocurra.
- **Partner Link Types:** Representa el rol de cada socio del Servicio en la comunicación.
- **Port Types:** Concepto ligado a WSDL, que muestra las operaciones soportadas y ofrecidas por un Servicio Web, que referencia al documento WSDL.

BPMN (BUSINESS PROCESS MODELING NOTATION)

BPMN es un estándar creado inicialmente por BPMI (Business Process Management Initiative) y actualmente mantenida por OMG (Object Management Group) que se encarga de modelar flujos de proceso de negocio y servicios web. Esta notación ha sido especialmente diseñada para coordinar la secuencia de los procesos y los mensajes que fluyen entre los participantes de las diferentes actividades [27]. El principal objetivo de BPMN es proporcionar una notación que sea entendible por todos los usuarios que utilicen procesos de negocio, desde los analistas de negocio que crean los borradores iníciales, hasta los desarrolladores responsables de la aplicación que llevarán a cabo los procesos y finalmente los usuarios del negocio que estarán a cargo de supervisar estos procesos. Un segundo objetivo es asegurarse que los lenguajes diseñados en XML para la ejecución de procesos de negocio, tales como BPEL4WS (Business Process Execution Language for Web Services) y BPML (Business Process Modeling Language) puedan ser expresados visualmente mediante una notación estándar [27]. BPMN define un Diagrama de Procesos de Negocio (BPD), basado en la

técnica denominada “Flow Chart”, la cual es utilizada para representar gráficamente la secuencia de todas las actividades que ocurren durante un proceso [27].

Según [28, 29] un BPD se compone por un conjunto de elementos gráficos, que permiten desarrollar fácilmente un diagrama, estos elementos se encuentran clasificados en cuatro categorías:

- **Objetos de flujo:** Son los encargados de definir el comportamiento de los procesos, esto a su vez se dividen en:
  - **Eventos:** Se pueden definir como sucesos que ocurren durante el curso de un proceso de negocio, estos afectan el flujo del proceso y usualmente tienen una causa y un resultado. Los eventos están clasificados en: Inicio, Intermedios, Fin.
  - **Actividades:** Estas representan el trabajo que es ejecutado dentro de un proceso de negocio, pueden ser atómicas o compuestas. Las actividades atómicas se conocen como tareas compuestas y las actividades compuestas se conocen como Subprocesos. Las tareas son usadas cuando el trabajo en el proceso no es descompuesto en más detalle y los subprocesos cuando un conjunto de actividades necesitan ser incluidos dentro de un proceso.
  - **Compuertas:** Son elementos del modelado que se utilizan para controlar la divergencia y la convergencia del flujo, estas se dividen en cinco tipos: compuerta exclusiva, basada en eventos, paralela, inclusiva y compleja.
- **Objetos de Conexión:** Son elementos usados para conectar dos objetos del flujo dentro de un proceso, y se clasifican en 3 tipos: líneas de secuencia, asociaciones y líneas de mensaje.
- **Canales:** son elementos utilizados para organizar las actividades del flujo en diferentes categorías visuales que representan áreas funcionales, roles o responsabilidades, existen dos tipos de canales: pools y lanes.
- **Artefactos:** los artefactos son usados para proveer información adicional sobre el proceso, estos se dividen en tres tipos: objetos de datos, grupos y anotaciones.

PROTOCOLOS SEMÁNTICOS:

WSML (WEB SERVICE MODELING LANGUAGE)

WSMO (Web Service Modeling Ontology) [6] proporciona un modelo conceptual para la descripción de servicios relacionados con la Web semántica, además ofrece cuatro componentes clave para modelar diferentes aspectos de los servicios Web semánticos en WSML (Web Service Model Language):

Ontologías: Proporcionan las especificaciones formales y explícitas de los vocabularios usados por los elementos de los modelos [30]. Este tipo de especificaciones formales permiten el procesamiento automático de las descripciones WSMO, proporcionando conocimientos básicos para el objetivo y las descripciones de Servicios Web.

Objetivos: Los Objetivos describen la funcionalidad y el estilo de interacción desde la perspectiva del solicitante.

Descripciones de Servicios Web: Las Descripciones de servicios Web especifican la funcionalidad y los medios que interactúan con el servicio, con el fin de lograr la funcionalidad requerida.

Mediadores: Conectan diferentes elementos WSMO y resuelven la heterogeneidad en la representación de datos, el estilo de la interacción y los procesos de negocio.
El Lenguaje de Modelado de Servicios Web WSML, toma en cuenta todos los aspectos de la descripción de Servicios Web identificados por WSMO. El objetivo de WSML es proporcionar un marco coherente que agrupa a las tecnologías Web con diferentes paradigmas de lenguajes de programación lógica a fin de permitir la descripción de los servicios de Web Semánticos.

WSML proporciona una sintaxis formal y una semántica para WSMO, además de brindar una sintaxis legible, ofrece una sintaxis XML y RDF (Resource Description Framework) para el intercambio de mensajes sobre la Web y para la interoperabilidad con aplicaciones basadas en RDF [31].
Los autores en [32] explican claramente la distinción que realiza WSML entre el modelado de los elementos conceptuales y la especificación de las definiciones lógicas complejas, debido a esto WSML divide su sintaxis en dos partes: Sintaxis Conceptual y Sintaxis de Expresiones Lógicas.

- **SINTAXIS CONCEPTUAL**

  La sintaxis conceptual permite el modelado de ontologías, servicios Web, objetivos, y mediadores. A continuación se explicaran más en detalle cada uno de ellos:

  Ontología: Una ontología en WSML consta de los siguientes elementos:

  - **Concept:** El concepto (Concept) forma parte de la terminología básica del universo del dominio. Este puede tener instancias (instances) y puede tener un número de atributos (attributes) asociados a estas instancias.
  - **Attribute:** La definición de atributo (Attribute) puede adoptar dos formas: Constraining (usando ofType) e Inferring (usando ImpliesType). Constraining Attribute define un tipo de restricción sobre los valores del atributo, e Inferring Attribute implica que el tipo de valor para el atributo se infiere de la definición del atributo.
  - **Relations:** Las relaciones (Relations) pueden ser organizadas en jerarquía utilizando subRelationOf, los parámetros pueden ser escritos usando definiciones de tipo parámetro como ofType type (usado para comprobar el tipo de valores de los parámetros) e ImpliesType type (usado para inferir en el concepto del valor del parámetro), donde type es un identificador del concepto.

- **Instances:** Las instancias (Instances) explícitamente en una ontología son las que comparten como parte de la ontología, un concepto puede tener un número de instancias asociadas con él.
- **Axioms:** Los axiomas (Axioms) constituyen un medio para añadir expresiones lógicas a una ontología, estas se usan para refinar un concepto o definiciones de relaciones en la ontología.

Servicios Web: Un servicio web está compuesto de una capacidad y un número de interfaces.

- **Capabilities:** Describe la funcionalidad del Servicio Web al expresar las condiciones sobre su pre-estado y post-estado usando expresiones lógicas.
- **Interfaces:** Las interfaces describen la interacción con un servicio desde el punto de vista del solicitante (Coreografía), y la interacción de un servicio con otros servicios y objetivos (goals) con el fin de cumplir con la capacidad (Orquestación) desde el punto de vista del proveedor.

Objetivos: Los objetivos (goals) son simétricos a los Servicios Web en el sentido de que los objetivos describen la funcionalidad deseada, y los Servicios Web describen la funcionalidad ofrecida, por lo tanto la descripción de los objetivos consta de los mismos elementos del modelado para la descripción de un Servicio Web, nombres, propiedades no funcionales, capacidad y un número de interfaces.

Mediadores: Los mediadores conectan objetivos, Servicios Web y ontologías, las conexiones entre los mediadores y otros elementos WSML se establecen de dos formas diferentes:

1. Cada elemento WSML para especificar un número de mediadores usados se hace mediante la palabra usesMediator.
2. Cada mediador (dependiendo del tipo de mediador) tiene una o varias fuentes y un destino. Ambos fuente y destino son opcionales con el fin de permitir mediadores genéricos.

o **SINTAXIS DE EXPRESION LOGICA**

La sintaxis general de expresiones lógicas para WSML se basa en el estilo de lógica de primer orden (First Order Logic Style), la cual contiene constantes, símbolos de función, variables, predicados, y conectores lógicos usuales. WSML se basa en extensiones de F-Logic [33] con el fin de modelar conceptos, atributos, definiciones de atributos, subconceptos y conceptos de relaciones de membrecía, además WSML tiene un número de conectores para facilitar la programación lógica basada en variantes (default negation, LP-Implication, y restricciones de integridad como las utilizadas en bases de datos).

BPMO (BUSINESS PROCESS MODELING ONTOLOGY)

BPMO es una ontología desarrollada por el proyecto SemBiz, basada en WSMO. Esta ontología proporciona un framework integrado por todos los elementos necesarios para la definición de procesos de negocio, tomando como base la infraestructura propuesta por WSMO [34]. Los elementos que componen este framework son: BPMO Ontology, BPMO Business Process, BPMO Business Goal y BPMO Mediators. Tomando como base [35, 36] a continuación se explican brevemente cada uno de estos elementos:

- **BPMO Ontology:** Es una ontología WSMO (esto respecto a la estructura y la sintaxis como está prescrito en WSMO), que utiliza la representación del lenguaje WSML. Para definir algunos procesos de negocio. El experto del dominio necesita definir todas las entidades involucradas: organizaciones, personal, software, sistemas usados, datos del negocio, y así sucesivamente. La ontología BPMO proporciona soporte para definir todas estas entidades. Los conceptos que son parte de la ontología se encuentran detallados en [36].

- **BPMO Business Process:** El punto de partida para la definición de BPMO Business Process es la definición de Servicio Web WSMO. Similar a los Servicios Web Semánticos un proceso de negocio puede importar ontologías externas y usar mediadores para superar los problemas de heterogeneidad. La principal diferencia entre un proceso de negocio (como lo definen en SemBiz) y un Servicio Web WSMO es que los Servicios Web definen una interfaz para hacer uso de la coreografía y la orquestación, mientras que los procesos de negocio se centran en como un servicio es ejecutado, que roles interactúan, si este servicio proporciona una funcionalidad en particular, y que subprocesos hacen parte de un proceso de negocio determinado. BPMO Business Process define una serie de conceptos, los cuales se encuentran detallados en [36].

- **BPMO Business Goals:** Este elemento, define lo que un solicitante desea lograr después de la ejecución de un proceso de negocio, esta definición es similar a la de un proceso de negocio, pero la principal diferencia radica en que el solicitante del proceso no está interesado en saber cómo se ejecuta el proceso, solo el resultado que este proporcione. BPMO Business Goals consta de un solo elemento llamado BusinessGoal, el cual se considera el concepto principal de la ontología, ya que define las ontologías importadas, los mediadores usados, un nombre (opcional), y la capacidad solicitada.

- **BPMO Mediators:** BPMO no define mediadores específicos, solo reúsa las definiciones de los mediadores de WSMO. BPMO solo reutiliza los mediadores: ontology-to-ontology mediation (ooMediator), y goal-to-goal mediation (ggMediator).

OWL-S (Ontology Web Language - Semantic)

OWL-S [13] es un protocolo de ejecución de procesos de negocio semántico basado en ontologías OWL. Gracias a este lenguaje los usuarios y los agentes de software son capaces de descubrir, invocar, componer y vigilar los recursos Web

que ofrecen servicios haciéndolo con un alto grado de automatización, teniendo en cuenta las categorías de servicios atómicos y compuestos [5]. OWL-S brinda características importantes que ayudan al descubrimiento, invocación, composición e interoperabilidad automática de servicios Web.

Los autores en [37] presentan los tres conceptos fundamentales que captura OWL-S sobre el servicio a describir:

¿Qué hace el servicio?, Para esta pregunta se define el ***Service Profile*** que consta de tres partes: El servicio de peticiones (identifica el comprador), el servicio proveedor (identifica el vendedor), y componentes de infraestructura.

¿Cómo trabaja el servicio?, Por medio del ***Service Model*** se describen las interacciones entre el cliente y el servicio. El ***Service Model*** le dice a un cliente cómo trabaja el servicio, detallando el contenido semántico de las solicitudes, las condiciones en que se producen determinados resultados y en caso necesario se realiza un seguimiento paso a paso de los procesos que conducen a esos resultados es decir, describe cómo solicitar el servicio y lo que sucede cuando el servicio es ejecutado [5].

¿Cómo utilizar el servicio?, El modulo ***Service Grounding***, específica los detalles del acceso a un servicio, tales como: el protocolo, el formato del mensaje, serialización, transporte y direccionamiento.

Dos conceptos claves del lenguaje OWL-S son los procesos atómicos y compuestos. Los primeros son procesos que siempre reciben un mensaje de entrada y devuelven un mensaje de salida, aclarando que así estos procesos sean muy complejos en la búsqueda de un resultado, estos no dejan de ser atómicos; mientras que los segundos son procesos que poseen un estado y pueden cambiar durante su ejecución, a este tipo de procesos se les permite estar compuestos de procesos atómicos. Esta descomposición del proceso puede adoptar distintas estructuras según cómo se realice la composición. Las principales estructuras son:

- **Sequence**: Indica que el proceso está compuesto de una secuencia de procesos.
- **Split**: Ejecución de procesos en paralelo.
- **Split-Join**: Sincroniza los procesos de un Split en su finalización.
- **Any-Orden**: Ejecuta los procesos aleatoriamente.
- **Choice**: Ejecuta los subprocesos escogidos.

Además de estas estructuras, también existen otras actividades muy conocidas por los actuales lenguajes de descripción de procesos de negocio, como: If-Then-Else, Iterate, Repeat-While, Repeat-Until.

BPEL4SWS (BUSINESS EXECUTION LANGUAGE FOR SEMANTIC WEB SERVICES)

BPEL4SWS [38] es una extensión del lenguaje BPEL con soporte para los conceptos de servicios Web semánticos, como las mediaciones y descripciones semánticas de las actividades utilizadas, definiendo un canal de comunicación

entre dos parejas de servicios, en lugar de utilizar partner links que están basados en WSDL 1.1. Este lenguaje se crea con el fin de eliminar las dos principales deficiencias que tiene BPEL: (i) El hard-coding[1] de interfaces de servicio y (ii) La falta de semántica al utilizar tipos de datos. Según lo anterior se puede decir que en BPEL no pueden ser usados servicios que proveen la misma funcionalidad y que son implementados por diferentes interfaces, reduciendo así la flexibilidad y reusabilidad del proceso de negocio descrito. BPEL4SWS, intenta superar las deficiencias mencionadas de BPEL de la siguiente manera: (i) Permitiendo descripciones semánticas en las implementaciones de las actividades (en lugar de referirse a interfaces sintácticas WSDL) y (ii) el uso de ontologías como modelo fundamental de datos, aplicando el concepto de mediación ontológica. BPEL4SWS toma como base BPEL^{light} el cual desacopla la lógica del proceso y la definición de la interfaz, además permite agregar descripciones SWS (Semantic Web Service), como frameworks WSMO y OWL-S para soportar el descubrimiento semántico [39].

BPEL4SWS introduce un número de extensiones con el fin de adicionar servicios Web Semánticos dentro de un modelo de proceso BPEL. Estas extensiones se encuentran detalladas en el Anexo A. A continuación se explica de manera general las extensiones tomando como base [40, 41]:

1. **Manejo de datos**

   Tipos de datos Ontológicos: En BPEL4SWS las variables pueden ser definidas como tipos de mensaje WSDL, elementos XSD, tipos simples o tipos complejos. BPEL4SWS utiliza anotaciones SA-WSDL[2] para tipos de datos XML con el fin de proporcionar una representación ontológica de los datos. La información dada en documentos SAWSDL/XSD se utiliza para transformar instancias de datos XML en su representación ontológica y viceversa.

2. **Mediación:** La manipulación de datos en BPEL4SWS puede ser implementado usando copia de declaraciones como se describe en el estándar de BPEL 2.0 o usando mediaciones. Las mediaciones ontológicas se definen con la etiqueta extensionAssignOperation, y se utiliza por medio del elemento mediate.

3. **Razonamiento en BPEL4SWS:** En BPEL4SWS el razonamiento ontológico puede ser utilizado para evaluar expresiones, en particular expresiones booleanas que son usadas para evaluar condiciones, como por ejemplo joinConditions, transitionConditions, condiciones de salida como las actividades while y repeatUntil, y condiciones que deciden qué camino tomar como es el caso de la actividad if.

4. **Modelo de Interacción sin WSDL:** BPEL4SWS abstrae la definición de la interfaz y proporciona un modelo de interacción sin WSDL. Esto se basa en el nuevo concepto introducido denominado conversación. Un elemento conversación es una abstracción sin WSDL de un partner link, en lugar de referirse a un partner link type, una conversación se refiere a un objetivo, o una

[1] Hard-Coding: http://www.pcmag.com/encyclopedia_term/0,2542,t=hard+coded&i=44076,00.asp

[2] Semantic Annotations for WSDL and XML Schema(SA-WSDL): http://www.w3.org/TR/sawsdl/

descripción del servicio Web definido de acuerdo a RO4SSOA [42]. BPEL4SWS define nuevos tipos de actividades que son presentadas a continuación:

- **<b4s:conversation>:** Especifica un intercambio de mensajes entre dos Partners.

- **<b4s:interactionActivity>:** Especifica cuando un proceso envía o recibe uno o muchos mensajes. Está definido como un elemento hijo de <extensionActivity>.

- **<b4s:pick>:** Indica que un proceso recibe uno o varios mensajes. Está definido como elemento hijo de <extensionActivity>.

- **EventHandler:** Un <b4s:eventHandler> especifica que un proceso recibe uno o varios mensajes, definido como un elemento hijo de <b4s:eventHandlers> el cual puede ser elemento hijo de <process> o <scope>.

- **Partner:** Un <b4s:partner> es utilizado para especificar múltiples <b4s:conversation> que tienen lugar con un solo partner. Está definido como un elemento hijo de <b4s:partners> el cual puede ser elemento hijo de <process> o <scope>.

Finalmente es importante mencionar que BPEL4SWS está altamente acoplado con su representación ontológica, llamada sBPEL [43], además BPEL4SWS describe un atributo adicional llamado modelReference como está definido en la especificación de SA-WSDL [44], el cual identifica la correspondiente ontología utilizada en el modelo de procesos sBPEL [45],[46].

2.2 TRABAJOS RELACIONADOS

Existen diferentes trabajos que abordan los núcleos temáticos del presente proyecto, en ellos se han presentado propuestas interesantes, alrededor de diferentes aspectos involucrados en la detección de equivalencias entre procesos de negocios semánticos. A continuación se presenta un análisis de los mismos, exponiendo su aporte y aplicación al momento de desarrollar este proyecto de grado.

2.2.1 RECUPERACIÓN DE PROCESOS BASADOS EN SINTAXIS.

- En [24] el emparejamiento de procesos BPEL busca recuperar servicios basándose en el comportamiento, por medio de algoritmos que permiten entregar correspondencias parciales y la evaluación de la distancia semántica entre éstas y las necesidades del cliente. En caso de que no exista un servicio que satisfaga las necesidades del usuario, se recuperará el más similar y se podrá utilizar por extensión o modificación. Bajo este enfoque los procesos utilizan una representación formal basada en grafos de tal manera que el problema de encontrar similitud entre procesos es reducido a un problema de comparaciones de grafos. Además presentan un prototipo que tiene como entradas dos modelos BPEL y evalúan la distancia semántica entre éstos. El prototipo provee un script de edición de operaciones que puede ser usado para modificar los procesos que se

encuentran en un repositorio de acuerdo al proceso de consulta suministrado por el cliente, con el ánimo de encontrar semejanzas entre ellos.

- Al igual que el artículo citado anteriormente, en [47] exponen la recuperación de servicios limitándose al emparejamiento de entradas y/o salidas. Investigaciones recientes han demostrado que estos enfoques no son completamente acertados, ya que existen casos donde la recuperación de servicios debe estar basada en el modelo de comportamiento del servicio. Dicho artículo Presenta dos escenarios que motivan la recuperación de servicios. El primero está centrado en la integración de los servicios Web y consiste en la recuperación de servicios teniendo en cuenta la compatibilidad de comportamiento, es decir, si el servicio a recuperar no es compatible con los servicios publicados por una organización, entonces existen dos posibilidades, o la empresa adapta su servicio o desarrolla un adaptador para poder interactuar con el servicio; en ambos casos, la diferencia entre los protocolos de negocio son detectadas automáticamente. El segundo contexto denominado "análisis delta" toma como base las diferencias entre dos modelos de negocio de una organización, para evaluar las actividades de reingeniería a aplicar. En éste contexto la evaluación de similitud entre los modelos es expresada en términos de las diferencias encontradas. Finalmente en este artículo explican paso a paso el algoritmo de detección de errores para el emparejamiento de protocolos WSCL y presentan un prototipo que está disponible como servicio Web.

Los trabajos de investigación explicados anteriormente muestran que el descubrimiento de procesos actúa sobre protocolos de descripción sintáctica, lo cual no garantiza la correcta recuperación, ya que al no considerar la semántica de los procesos los algoritmos de recuperación omiten componentes que son similares a los buscados por el cliente, u ofrecen al usuario resultados con baja medida de precisión denominados "falsos positivos".

- En [48] se presenta un análisis comparativo entre cuatro algoritmos para el emparejamiento de grafos:

    - ***Algoritmo Voraz:*** El algoritmo comienza marcando todos los posibles pares de nodos a partir de los dos grafos como pares abiertos. En cada iteración el algoritmo selecciona un par abierto que tenga mayor similitud, y es agregado al mapeo. El par seleccionado consta de dos nodos. Dado que cada nodo puede ser mapeado una vez, el algoritmo elimina el conjunto de los pares abiertos, en otras palabras, todas las parejas que contenga uno de los nodos seleccionados. El algoritmo se repite hasta que no exista un par abierto.
      Este algoritmo tiene una complejidad de $O(n^3)$ donde n es el número mayor de nodos de un grafo. En la primera iteración se tiene en cuenta n^2 pares abiertos, en la segunda iteración $(n - 1)^2$ pares abiertos, y así sucesivamente.

    - ***Algoritmo de Exhaustividad con poda:*** Este algoritmo explora todos los posibles mapeos de manera recursiva, sin embargo cuando el árbol de recursividad alcanza un determinado tamaño, el algoritmo se encarga de podarlo con el fin de mantener sólo los mapeos con mayor similitud. Por otra parte la complejidad del algoritmo en el peor los casos desarrolla un comportamiento exponencial, pero cuando el árbol de recursividad es podado, la complejidad del algoritmo disminuye.

- ***Algoritmo Heurístico de Procesos:*** El tercer algoritmo es una variación del algoritmo de exhaustividad. Este también construye un árbol de recursividad de los posibles mapeos, pero inicia mapeando los nodos fuente de los grafos que representan a los procesos de negocio, y así reiteradamente hasta terminar el mapeo de todos los nodos. Dado que los nodos más cercanos del inicio de un proceso deben ser mapeados con los nodos más cercanos del inicio del otro proceso (y viceversa), el método que poda el árbol de recursividad arrojara un alto desempeño, ya que el algoritmo poda los mapeos con pares de nodos que están más separados del inicio del proceso. Es importante mencionar que el algoritmo trabaja solo con grafos que contengan nodos fuente, además este algoritmo es válido para notaciones de modelado de procesos como BPMN y BPEL.

  Algoritmo A – Estrella: Este algoritmo se basa en la búsqueda Heurística A Estrella, la cual es aplicada al problema del emparejamiento de grafos en [49]. En cada paso, el algoritmo selecciona un mapeo parcial ***map*** con la similitud de edición de grafos máxima. El algoritmo toma un nodo n1 del grafo g1 que aun no ha sido mapeado, y crea un mapeo entre este nodo y cada nodo n2 de g2 tal que n2 no aparezca en ***map***. El algoritmo entonces crea m nuevos mapeos, mediante la adición de (n1, n2) a ***map***. Además se crea un mapeo donde (n1, e) es agregado a ***map*** (donde e es un nodo falso), este último par representa el caso en que el nodo n1 ha sido eliminado. Este paso se repite hasta que todos los nodos de g1 sean mapeados. La complejidad teórica del algoritmo esta dado por O (n^2m), donde n y m son el numero de nodos en g1 y g2 respectivamente.

Este análisis se realizo utilizando un repositorio de 100 modelos de procesos, arrojando como resultados que el algoritmo A - estrella ofrece un mejor promedio de precisión, pero es significativamente lento, sin embargo los tiempos de ejecución pueden ser aceptables para repositorios con pocos procesos de negocio. Las otras dos técnicas basadas en la búsqueda exhaustiva con poda, son menos atractivas debido a la calidad y escalabilidad que presentan. Mientras que el algoritmo voraz es más rápido pero su promedio de precisión no es tan bueno como el algoritmo A – estrella, ya que su promedio se aproxima al valor de los algoritmos exhaustivos.

2.2.2 RECUPERACIÓN DE PROCESOS BASADO EN SEMÁNTICA

- En [13], los autores argumentan que en muchas situaciones el emparejamiento de servicios deberá tener en cuenta el modelo de procesos de OWL-S descrito en el Service Model, el cual indica la forma de interactuar con el proveedor de servicios, ya que los enfoques actuales de recuperación de procesos utilizan sólo el perfil de usuario y en algunas aplicaciones no es suficiente el descubrir servicios solo con las entradas y salidas, las cuales pueden ser utilizadas tan solo como un primer filtro. La unidad elemental del modelo de procesos es un proceso atómico, lo que representa una operación indivisible, que el cliente puede realizar mediante el envío de un mensaje en particular a los servicios y recibir su correspondiente respuesta. En el artículo citado anteriormente los autores proponen técnicas de correspondencia que operan sobre los modelos de procesos OWL-S alojados en un repositorio con el fin de recuperar aquellos procesos que tengan mayor similitud respecto al proceso buscado por un usuario. Para lograr este propósito, el

problema de búsqueda es reducido a un emparejamiento de grafos (los procesos atómicos representan nodos y las restricciones son transformadas en aristas). La medida de similitud que permite clasificar los servicios recuperados muestra las diferencias en cuanto a estructuras y disconformidades semánticas entre entradas y salidas usadas en el proceso. Para lograr esto, usan el módulo de cotejo semántico que permite comparar dos conjuntos de entradas/salidas haciendo uso de una ontología. Las comparaciones están basadas en los siguientes parámetros: Grado de emparejamiento (MD) y solicitud semántica (SS), el primero refleja las relaciones entre entradas/salidas en la ontología, para esto los autores definen cuatro relaciones de orden entre conceptos (EXACT > PLUG-IN > SUBSUMES > FAIL), mientras que el segundo es usado para cuantificar la similitud semántica.

Finalmente, éste artículo realiza un paralelo entre BPEL y OWL-S, el cual resalta las capacidades de descripción lógica ofrecida por OWL-S, lo que daría más formalidad, y mayor riqueza y precisión en el momento de recuperar un servicio.

- En [50], los autores presentan un emparejador hibrido de servicios llamado WSMO-MX, el cual está basado en la programación lógica de F-Logic [33] y en medidas de similitud sintáctica. El proceso que se lleva a cabo para recuperar un conjunto de servicios de una consulta dada por el usuario, se hace mediante el uso de filtros de correspondencia llamados IOPE (Input, Output, Precondition, Effect); en caso de que alguno de los filtros llegara a fallar el emparejamiento sintáctico, es ejecutado. Una vez recuperado el conjunto de servicios se clasifica de forma descendente según los grados de similitud semántica, los cuales se dividen en 7 tipos: Equivalence, Plugin, Inverse-plugin, Intersection, Fuzzy Similarity, Neutral, Disjunction. Los grados Equivalence y Plugin hacen referencia a las relaciones lógicas, mientras que Inverse-Plugin, Intersection, y Disjunction se basan en el comportamiento semántico. El grado Fuzzy Similarity se refiere a una lógica de correspondencia sintáctica, y el grado neutral declara la tolerancia de error del algoritmo de emparejamiento en el cual no se declara similaridad ni falla del algoritmo.

Si bien es cierto que OWL-S y WSMO facilitan la recuperación de servicios soportados en la semántica del proceso de negocio, en la actualidad dichos lenguajes son poco usados por las grandes casas Software. Por lo tanto el presente trabajo de grado utilizará el protocolo semántico BPEL4SWS el cual es una evolución del estándar BPEL4WS, lenguaje que ha alcanzado un fuerte desarrollo en la industria TI, especialmente en grandes compañías, como Oracle y Sun, además de Novell, Adobe y SAP, entre otras.

2.3 ANÁLISIS

En esta sección se ha presentado un conjunto de mecanismos para la recuperación de procesos basados en sintaxis y en semántica. La recuperación de procesos de negocio basados en sintaxis, no garantiza el correcto descubrimiento, ya que omite componentes que son similares a los buscados por el cliente, u ofrecen al usuario resultados con baja medida de precisión, tal como lo exponen en [10],[9], mientras que los algoritmos de recuperación basados en semántica mejoran el descubrimiento de servicios al inferir sobre los conceptos de un dominio, esto gracias a la utilización de una ontología, la cual enriquece las búsquedas recuperando un conjunto de servicios útil para el usuario.

Según lo dicho anteriormente, desde la perspectiva de recuperación de procesos de negocio, los lenguajes sintácticos, no garantizan el correcto descubrimiento, ya que omite componentes que son similares a los buscados por el cliente, u ofrecen al usuario resultados con baja medida de precisión, tal como lo exponen en [10],[9], debido a esto se recomienda utilizar protocolos semánticos. Si bien es cierto que los protocolos semánticos pueden dar una mayor precisión a la hora de elegir el proceso de negocio con mayor similitud respecto al proceso buscado por el usuario, éstos inducen un aumento en el tiempo de ejecución de los algoritmos de recuperación utilizados. Por lo tanto, el presente trabajo de grado pretende guardar una proporción entre la calidad en la recuperación de procesos de negocio utilizando un protocolo de descripción semántica y el tiempo invertido para detectar las equivalencias entre ellos.

En este sentido, el presente trabajo de grado busca dar solución a las deficiencias presentadas anteriormente, detectando las equivalencias entre procesos de negocio basados en la semántica de las operaciones, entradas y salidas de los servicios compuestos, pero considerando también las propiedades sintácticas de sus descripciones. Adicionalmente la presente propuesta toma en cuenta las cuatro relaciones de orden entre las estructuras de los procesos de negocio comparados (EXACT > PLUG-IN > SUBSUMES > FAIL). Todas las consideraciones anteriores son aplicadas al protocolo BPEL4SWS ya que es la extensión semántica del protocolo BPEL, éste último considerado como uno de los lenguajes más utilizados en la actualidad.

2.4 RESUMEN

Este capítulo presento los conceptos fundamentales necesarios para comprender el campo de aplicación del presente proyecto como son: los Servicios Web, los Procesos de Negocio, La Arquitectura Orientada a Servicios y los Lenguajes para la descripción de Procesos de Negocio tanto sintácticos como semánticos. De igual manera, se expuso el estado actual sobre las investigaciones desarrolladas alrededor de la recuperación de procesos de negocio

Finalmente, tomando como referencia los conceptos y el análisis de los trabajos relacionados se presentó desde un punto de vista general la aproximación que abordará el presente trabajo de grado (en el resto del documento se detallará como se realiza dicha propuesta): detectar equivalencias entre procesos de negocio para soportar el proceso de recuperación de servicios basadas en la semántica de las operaciones, entradas y salidas de los servicios compuestos, pero considerando también las propiedades sintácticas de sus descripciones. Al mismo tiempo, se toma en cuenta las cuatro relaciones de orden entre las estructuras de los procesos de negocio comparados (EXACT > PLUG-IN > SUBSUMES > FAIL). Todas las consideraciones anteriores son aplicadas al protocolo BPEL4SWS debido a que es el protocolo semántico de última generación y además su antecesor BPEL es el lenguaje más utilizado en la actualidad debido a la gran variedad de herramientas que ofrecen las grandes casas productoras de software.

Capítulo III

3 ALGORITMO PARA LA DETECCIÓN DE SUBESTRUCTURAS

El presente trabajo toma como hipótesis que la representación formal basada en grafos es una herramienta que expresa con suficiencia el comportamiento de las orquestaciones publicadas por los proveedores de servicios. Lo anterior está fundamentado en la valoración de representaciones formales para servicios compuestos descrita en [14] en la cual se concluye que el modelo de representación de grafos ofrece una simple y madura notación para expresar la semántica de un servicio compuesto.

Tomando en cuenta la apreciación anterior, el presente capítulo está estructurado de la siguiente manera: primero se muestra una comparación entre distintos algoritmos de detección de sub-grafos isomorfos donde se refleja el porqué de la escogencia del algoritmo VF2. Seguidamente se describe en detalle el algoritmo VF2. Para comprender los puntos mencionados anteriormente en el Anexo A se explican los conceptos más relevantes referentes a la teoría de grafos, iniciando con las definiciones fundamentales de lo que son nodos, aristas, las definiciones de grafos dirigidos y no dirigidos; además del concepto general de isomorfismo de grafos.

3.1 ALGORITMOS PARA LA DETECCIÓN DE ISOMORFISMO DE SUB-GRAFOS

Los grafos son una general y poderosa estructura de datos para representar objetos y conceptos. En esta representación formal los nodos suelen constituir objetos o partes de objeto y las aristas describen las relaciones entre los objetos. Bajo este enfoque y teniendo en cuenta que los procesos de negocio BPEL4SWS tienen comportamientos similares de un grafo donde los nodos suelen ser semejantes a las actividades y el flujo de control de dichos procesos puede representarse por medio de aristas; es posible trasladar la problemática de emparejamiento de grafos a un problema de detección de isomorfismo de grafos. A continuación se exponen los más importantes algoritmos para la detección de grafos y sub grafos isomorficos con el fin de seleccionar el más adecuado para la detección de equivalencias entre procesos.

Actualmente existen múltiples algoritmos para la detección de patrones de estructuras de grafos tales como el algoritmo de Ullmann's, Nauty (basado en la teoría de grupos), algoritmo para arboles poligonales, algoritmo CDK, QuickSI, entre otros. Sin embargo, tal y como lo afirman en [56], no existen investigaciones en las cuales se presente el rendimiento de los diferentes mecanismos en términos de tiempo de ejecución. Sin embargo los algoritmos mencionados a continuación son los más utilizados y recomendados con los cuales se ha hecho un estudio de rendimiento.

Un procedimiento que reduce significativamente el tamaño del espacio de búsqueda de equivalencias entre dos grafos es el algoritmo Backtraking propuesto por Ullmann en [57], este algoritmo está diseñado tanto para grafos isomorfos como para sub grafos isomorfos, convirtiéndose hoy en día en uno de los más usados para el emparejamiento exacto de grafos. Otro algoritmo de Backtraking es el presentado en [58] por Schmidt and Druffel (SD), el cual hace uso del grado de la secuencia de los vértices y de un procedimiento recursivo, utilizando las matrices de distancia, para obtener la partición inicial de vértices.

La información de la matriz de distancias es usada en un procedimiento de backtraking para reducir la búsqueda de posibles mapeos. El algoritmo más reciente es conocido como VF[59], el cual está basado en la estrategia de búsqueda en profundidad junto con un conjunto de reglas que serán analizadas con detalle más adelante. Se consideran dos versiones de este algoritmo, el algoritmo VF solamente se basa en la implementación reportada en [59], mientras que el segundo llamado VF2 utiliza estructuras de datos más eficaces con el fin de optimizar el tiempo de ejecución. En cuanto a algoritmos de grafos isomorfos, se hace necesario mencionar el algoritmo McKay's Nauty [60], basado en un conjunto de transformaciones que conducen a reducir los grafos que son emparejados a una forma regular (por cada iteración sobre el algoritmo), por lo cual, resulta al final muy simple encontrar un isomorfismo. Nauty es considerado como uno de los algoritmos más rápidos para la detección de isomorfismo de grafos.

Para llevar a cabo la comparación de los 5 algoritmos mencionados con anterioridad, los autores en [61], utilizaron una base de datos compuesta por 10.000 parejas de grafos isomorfos divididos en 4 tipos de grafos, a saber:

- **Grafos conectados aleatoriamente (3000 parejas):**
  Son grafos donde las aristas se conectan con nodos sin regularidad estructural. Estos grafos fueron adoptados del mismo modelo propuesto en [57]. Fijando el valor n de la probabilidad que una arista este presente entre dos nodos distintos.

- **Redes de grafos regulares 2D (1000 parejas):**
  Son grafos que son utilizados para simular aplicaciones relacionadas con estructuras regulares. Las redes consideradas son conectadas directamente por aristas a través de 4 conectores (cada nodo es conectado solamente con un nodo norte, sur, este y oeste).

- **Redes de grafos irregulares 2D (3000 parejas):**
  Se han introducido para simular el comportamiento de los algoritmos en la presencia de redes ligeramente distorsionadas. Estas han sido obtenidas a partir de las redes regulares 2D mediante la adición de aristas. Cada conexión de nodos ha sido determinada aleatoriamente de acuerdo a una distribución uniforme.

- **Grafos con limite de grados (3000 parejas):**
  Limita aplicaciones en las que cada nodo establece un número fijo de relaciones (aristas) con otros nodos.

En la Figura 12, las graficas de análisis muestran los tiempos de emparejamiento como una función del tamaño de los grafos de entrada de los 5 algoritmos considerados para las pruebas; los tiempos son siempre medidos en segundos en escala logarítmica. Antes de su presentación cabe destacar que algunas curvas no reportan el tiempo obtenido en correspondencia con un tamaño dado. Esto ocurre cuando el algoritmo no es capaz de encontrar una solución al problema de isomorfismo en un tiempo inferior a 30 minutos.

Se define la densidad n como la probabilidad de que una arista este presente entre dos nodos distintos. Cuanto mayor sea el valor de n, el grafo es más denso. La Figura 12, muestra el emparejamiento de tiempo de los 5 algoritmos seleccionados tomando como referencia los grafos conectados aleatoriamente. Donde se muestra el valor de n cuando es igual a 0.01 y 0.1.

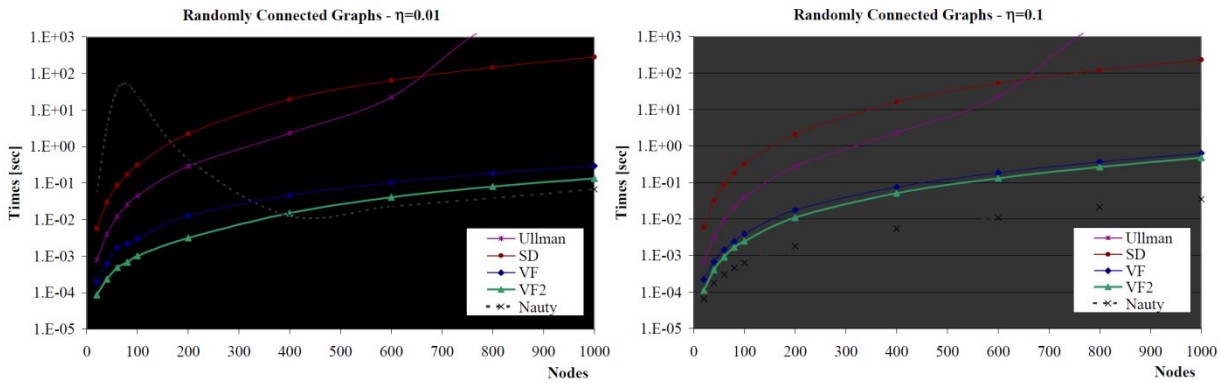


Figura 1 Grafos conectados aleatoriamente para n = 0.01 y n = 0.1

Como se puede observar, los algoritmos VF, VF2, y Nauty son mejores con respecto a SD y Ullman. Por otra parte, el rendimiento de VF2 es siempre mejor que VF, mientras que Ullmann es mejor que SD si el tamaño de los grafos es menor que 700. Después de ese tamaño, en efecto el algoritmo de Ullmann no es capaz de encontrar un isomorfismo. En conclusión el algoritmo VF2 obtiene el mejor rendimiento para grafos pequeños y para grafos demasiado dispersos mientras que para los grafos densos el algoritmo Nauty obtiene los mejores resultados.

En la Figura 13 se muestra el rendimiento de los 5 algoritmos sobre redes regulares 2D. En este caso, como el tamaño de los grafos crece hasta 100 nodos, Nauty y Ullmann no encuentran soluciones. También es posible observar que para cualquier tamaño del grafo de entrada el algoritmo VF2 es el mejor teniendo en cuenta que el algoritmo VF siempre funciona mejor que SD.

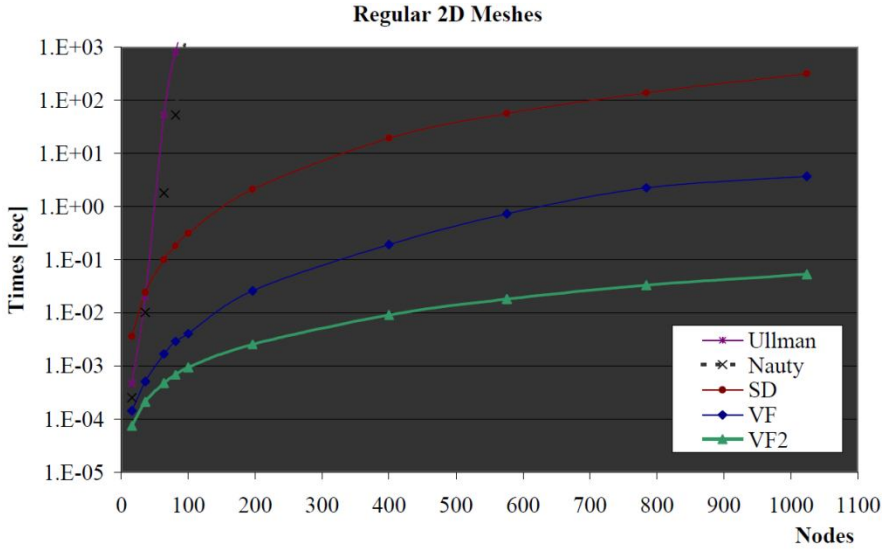


Figura 2 Redes regulares 2D

Para las redes irregulares 2D, el número de aristas adicionadas son p por cada nodo, donde p es una constante mayor que cero. La Figura 14 reporta el rendimiento de todos los algoritmos sobre las redes irregulares 2D. Donde p toma valores de 0.2 y 0.6. La principal diferencia con el caso de redes regulares es que el algoritmo Nauty siempre encuentra una solución, pero es menos eficiente que VF2. Sin embargo este

comportamiento es mejor que VF y SD. Por otro lado si el algoritmo de Ullmann encuentra una solución el tiempo de emparejamiento es siempre mejor que el obtenido por SD.

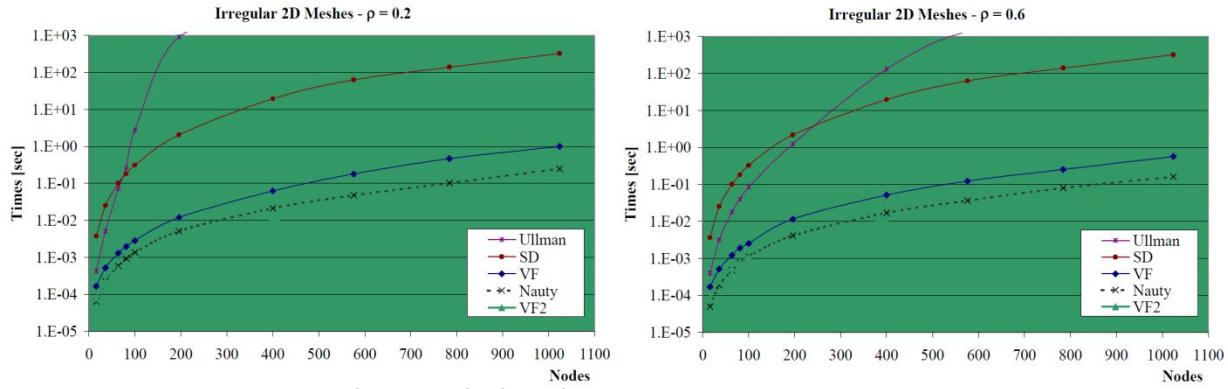


Figura 3 Redes irregulares 2D con p=0.2 y p=0.6

Finalmente la Figura 15 muestra el rendimiento de los grafos limitados de grado, donde el grado de los grafos está definido como v tomando valores de 3 y 9 respectivamente. En este caso el algoritmo de Ullmann no siempre encuentra una solución. Sin embargo cuando su tiempo es menor que el algoritmo SD y superior al obtenido por los otros algoritmos, Ullman encuentra solución. Por otra parte SD se muestra deficiente que VF, VF2 y Nauty; a su vez que VF2 es siempre mejor que VF Independientemente del valor que tome v. Ahora bien, como el número de nodos de los grafos de entrada son mayores que 600, el algoritmo VF2 tendrá el mejor desempeño.

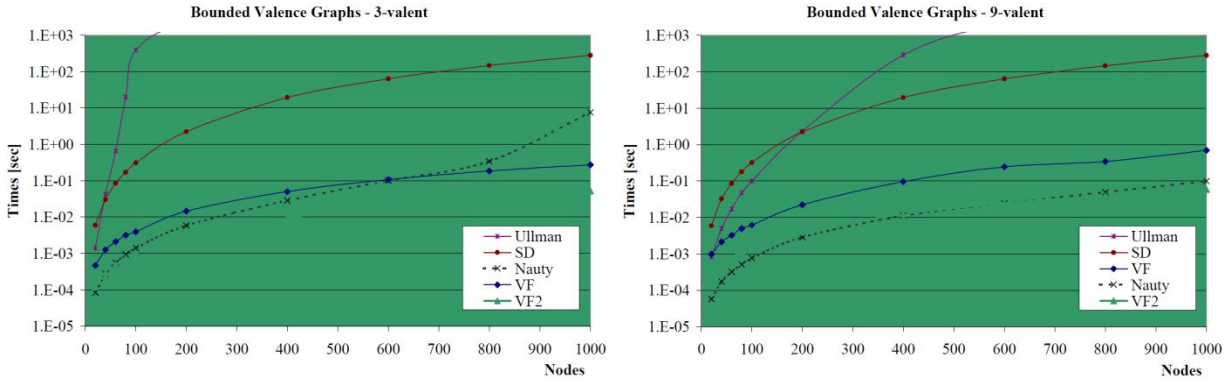


Figura 4 Grafos con grado 3 y 9

De lo anterior es válido concluir que bajo cualquier tipo de grafos (conectados aleatoriamente, con redes regulares 2D, con redes Irregulares 2D , con limite de grados), el algoritmo VF2 marca una diferencia notoria en cuanto a los otros algoritmos respecto a grafos con estructura regular, siendo este más eficiente especialmente para grafos de gran tamaño, manteniendo siempre un optimo rendimiento en el tiempo de respuesta, fiabilidad y requerimientos de memoria en comparación con otros algoritmos y más precisamente con el algoritmo de Ullmann presentado en [57].

Los procesos de negocio tienen características similares a los grafos regulares y/o irregulares, debido a que el número de aristas de los nodos en algunos casos siempre tienen el mismo grado o a veces no mantienen la misma regularidad. Al unificar los resultados de las pruebas realizadas a las redes tanto regulares como irregulares, es posible observar en las graficas que para estos dos tipos de grafos, el algoritmo que

ofrece un mejor tiempo de respuesta es VF2. Lo anteriormente dicho es posible comprobarlo en la Tabla 1, obtenida de [62], donde se comparan los algoritmos VF2 y Nauty los cuales brindaron los mejores tiempos de respuesta.

	Randomly Connected			2D (regular and irregular) Mesh				Bounded valence		
Nodes	η=0.01	η=0.05	η=0.1	regular	ρ=0.2	ρ=0.4	ρ=0.6	ν=3	ν=6	ν=9
20	VF2	VF2	Nauty	VF2	Nauty	Nauty	Nauty	Nauty	Nauty	Nauty
40	VF2	Nauty	Nauty	VF2	VF2	Nauty	Nauty	Nauty	Nauty	Nauty
60	VF2	Nauty	Nauty	VF2	VF2	VF2	Nauty	VF2	Nauty	Nauty
80	VF2	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	Nauty	Nauty
100	VF2	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	Nauty	Nauty
200	VF2	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	Nauty
400	Nauty	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	Nauty
600	Nauty	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	Nauty
800	Nauty	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	VF2
1000	Nauty	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	VF2

Tabla 1 Tabla comparativa de algoritmos

3.2 ALGORITMO VF2

VF2 es un algoritmo utilizado para la detección grafos isomorficos, el cual es un método de correspondencia deterministico que permite verificar correspondencias entre grafos. A través de este algoritmo es posible realizar las comparaciones entre diferentes grafos y determinar cuál grafo es igual a otro, en cuanto a su estructura (isomorfismo); ó cuál grafo está contenido en otro (sub grafo) [62] .El algoritmo VF2 funciona de la siguiente forma: Se consideran dos grafos Q y G, el sub grafo isomorfo de Q a G, es expresado como el conjunto de pares (n,m), donde n pertenece a G1 y m pertenece a G2. Si s representa a un estado intermedio, s denotara a un estado parcial desde Q a G, es decir un mapeo desde un sub grafo de Q a un sub grafo de G. Estos dos sub grafos son denotados como Q (s) y G (s) respectivamente.

Encontrar el isomorfismo de grafos entre Q y G, se hace bajo una secuencia de transición de estados. Por otra parte el objetivo del proceso de emparejamiento entre dos grafos G1 y G2 consiste en la determinación de un mapeo m, el cual asocia nodos de G1 con nodos de G2 y viceversa de acuerdo a algunos criterios predefinidos. El mapeo es una función biyectiva y consta de un conjunto de pares de nodos que cubren todos los nodos del grafo de consulta. Cuando los pares de nodos son adicionados al mapeo parcial, se verifica si estos nodos cumplen con ciertas condiciones de consistencia. Estas reglas permiten la reducción del espacio de búsqueda, simplificando significativamente el costo computacional del emparejamiento de grafos. A continuación es presentado el algoritmo VF2.

Algoritmo 1. VF2: Algoritmo de isomorfismo de grafos

PROCEDURE Match (s)

INPUT: an intermediate state s; the initial state; the initial state s_0 has $M(s_0) = \emptyset$
OUTPUT: the mappings between two graphs
IF $M(s)$ covers all the nodes of G_2 **THEN**
OUTPUT $M(s)$
ELSE
Compute the set $P(s)$ of the pairs candidate for inclusion in $M(s)$
FOREACH p in $P(s)$
IF the feasibility rules succeed for the inclusion of p in $M(s)$ **THEN**
Compute the state's obtained by adding p to $M(s)$
CALL $Match(s')$
END IF
END FOREACH
Restore data structures

END IF
END PROCURE MATCH

Un mapeo $M \subseteq N1\ x\ N2$ se considera isomorfo si y solo si M es una función biyectiva que preserva la estructura de dos grafos. Ahora bien dado un mapeo $M \subseteq N1\ x\ N2$ se dice que es un grafo- sub grafo isomorfo si y solo si M es isomorfo entre G2 y un sub grafo de G1. El proceso de búsqueda de la función de mapeo puede ser convenientemente descrita por un estado de representación de espacios (SSR) [63], cada estado s del proceso de emparejamiento puede ser asociado a un mapeo parcial de la solución M(s), el cual contiene solamente un subconjunto de M. M(s) identifica dos sub grafos de G1 y G2 llamados G1(s) y G2 (s) obtenidos mediante la selección de G1 y G2 respectivamente. Un estado s intermedio puede ser alcanzado a través de diferentes rutas, gracias a que el algoritmo VF2 explora el grafo mediante la búsqueda por profundidad. A través de todos los posibles estados (SSR) solamente los pequeños subconjuntos son consistentes con el tipo de estructura requerida, en el sentido de que no hay condiciones que excluyan la posibilidad de alcanzar una solución completa. Se puede mostrar que la condición de consistencia, en el caso de grafos isomorfos ocurre cuando los grafos parciales G1 (s) y G2 (s), asociados a M(s) son isomorfos. Los nodos vecinos juegan un papel importante en la búsqueda de subestructuras ya que gracias a estos es posible determinar si existe una relación de isomorfismo a través de los pares candidatos, a todos los vértices vecinos a Q (s) se indican como NQ(s), y todos los vértices vecinos a G (s) se indican como NG(s). El conjunto de los pares candidatos son un subconjunto de NQ (s) X NG(s), es decir el par de nodos (n, m) ∈ NQ (s) X NG(s).

Como se observa en el algoritmo presentado, se realiza el cálculo del conjunto de los llamados pares candidatos para incluirlos en el mapeo parcial aplicando ciertas reglas de factibilidad o viabilidad para determinar si los pares candidatos pueden ser adicionados al mapeo. Estas funciones aplicadas son mostradas en la siguiente sección, sin embargo es importante mencionar algunas notaciones: Dado un grafo G y uno de sus nodos n, se llama Pred (G, n) al conjunto de predecesores de n, es decir al conjunto de nodos de G desde el cual se origina una rama que termina en n. De forma análoga se llama Succ (G, n) al conjunto de sucesores de n, es decir al conjunto de nodos de G, que son el destino

de una rama que inicia desde n. También se define *T*out1(*s*) como el conjunto de nodos de G1 que no están en M1 (s) pero son sucesores de un nodo en M1 (s), a su vez, *T*in 1(*s*) se define como el conjunto de nodos que no están en M1 (s) pero son predecesores de un Nodo en M1 (s). De forma similar son definidos *T*out 2(*s*) y *T*in 2(*s*).

3.2.1 CONJUNTO DE LOS PARES CANDIDATOS P(S)

El conjunto p (s) de todos los posibles pares candidatos que son adicionados al estado actual son obtenidos considerando primero el conjunto de los nodos directamente relacionados a G1 (s) y G2(s). El conjunto estará formado por todos los pares de nodos (n,m), con n perteneciente a T1 Out (s) y m a T2 out (s) a menos que uno de estos dos conjuntos este vacío; en este caso el conjunto P (s) es también obtenido considerando a T1 in (s) y T2 in (s).

El conjunto P(s) es construido como se muestra: si *T*out1(*s*) y *T*out2(*s*) no son vacios, entonces

$$\boldsymbol{p(s) = T_1 out(s) \times \{min\ T_2\ out\ (s)\}}$$

Donde los min consultan al nodo en Tout2(s) con la etiqueta más pequeña (cualquier otro criterio puede ser usado). Si Tout1(s) y Tout2(s) son vacios y Tin1(s) y Tin2(s) son distintos de vacío, entonces:

$$\boldsymbol{p(s) = T_1 in(s) \times \{min\ T_2 in\ (s)\}}$$

Finalmente si Tout1(s), Tout2(s), Tin1(s) y Tin2(s) son vacios, entonces:

$$\boldsymbol{p(s) = (N_1 - M_1(s)) \times \{min(N_2 - M_2(s))\}}$$

En caso de que sólo uno de los conjuntos T in o sólo uno de los conjuntos T out este vacío, entonces el estado s no puede ser parte de un emparejamiento por lo cual queda descartado para un futuro estudio de viabilidad. La definición de P(s) garantiza que el algoritmo de búsqueda no visite un estado dos veces.

3.2.2 REGLAS DE FACTIBILIDAD

El algoritmo provee un conjunto de reglas para verificar que se cumplan ciertas condiciones de consistencia, haciendo posible la generación de estados consistentes, además el número de estos estados pueden ser reducidos gracias al conjunto de reglas mencionadas (a este conjunto de reglas se les denomina k-look-a head) que verifican si un estado s consistente no tiene sucesores consistentes después de k pasos como lo afirman en [64]. Estas reglas son validadas con la función feasibleRules(), la cual retorna verdadero si la adición del estado s del par de nodos (n,m) satisface todas las reglas de factibilidad para cada par de nodos. Seguidamente se agregan el par de nodos y el estado sucesor es procesado para luego aplicar al proceso la recursividad.
La factibilidad depende de la estructura de los grafos de entrada solamente, sin embargo si los nodos y las aristas tienen atributos, entonces estos también deben ser tenidos en cuenta, por lo que la forma general de la función de factibilidad, queda determinada de la siguiente manera:

$$F(s,n,m) = F_{structure}(s,n,m) \wedge F_{label}(s,n,m)$$

Donde, $F_{structure}$ depende solamente de la estructura del grafo, y F_{label} depende de las etiquetas del nodo. Además s representa el estado, n y m representan al conjunto de los pares candidatos. En la Figura 16 se observa la detección de sub grafos isomorficos por estructura $F_{structure}(s,n,m)$. Donde el algoritmo ha pasado por dos estados en la detección de las subestructuras s: 1 ↔ 1, s: 2 ↔ 2, y los nodos candidatos para el tercer estado son $n = 3, m = 3$.

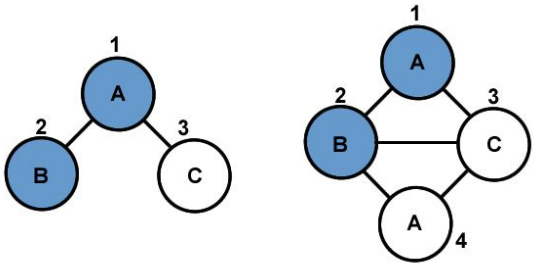


Figura 5 Detección de sub grafos isomorficos

Existen 5 reglas de factibilidad las cuales reciben los nombres: Rprev, Rsuc, Rin, Rout, Rnew. Las dos primeras reglas verifican la consistencia de la solución parcial obtenida, adicionando los pares de nodos candidatos a la actual solución parcial; las otras tres reglas son introducidas para las búsquedas en el grafo. En particular Rin y Rout realizan un look-ahead y Rnew un segundo look-ahead. Bajo este marco tenemos que un par de nodos son factibles si se cumple que Rprev, R suc , Rin, Rnew y Rout son verdaderos, es decir:

$$\text{Fstructure}(s,n,m) = \text{Rprev} \wedge \text{R suc} \wedge \text{Rin} \wedge \text{Rnew} \wedge \text{Rout}$$

Dado un grafo G=(N,B) y un nodo n que pertenece a N, los conjuntos respectivos que contienen los predecesores y los sucesores de n son denotados por Pred (G,n) y Succ (G,n) como se menciono anteriormente. En las siguientes definiciones se usaran T1 (s) = T1 in (s) U T1 out (s) y Ñ1 (s)= N1 – M1 (s) – T1 (s). De manera similar son definidas las expresiones para T2 y Ñ2. Las definiciones formales de las cinco (5) reglas de factibilidad se muestran a continuación:

$$Rpred(s,n,m) \leftrightarrow$$
$$(\forall n' \in M_1(s) \cap Pred(G_1,n) \exists\, m' \in Pred\,(G_2,m) \mid (n',m') \in M(s)) \wedge$$
$$(\forall m' \in M_2(s) \cap Pred(G_2,m) \exists\, n' \in Pred\,(G_1,n) \mid (n',m') \in M(s))$$

$$Rsucc(s,n,m) \leftrightarrow$$
$$(\forall n' \in M_1(s) \cap Succ(G_1,n) \exists\, m' \in Succ\,(G_2,m) \mid (n',m') \in M(s)) \wedge$$
$$(\forall m' \in M_2(s) \cap Succ(G_2,m) \exists\, n' \in Succ\,(G_1,n) \mid (n',m') \in M(s))$$

$$Rin(s,n,m) \leftrightarrow$$
$$(Card(Sucess(G_1,n) \cap T_1\, in\,(s)) \geq Card\,(Succ\,(G_2,m) \cap T_2\, in(s))) \wedge$$
$$Card(Pred(G_1,n) \cap T_1\, in\,(s)) \geq Card\,(Pred\,(G_2,m) \cap T_2\, in(s))$$

$$Rout(s,n,m) \leftrightarrow$$
$$(Card(Sucess(G_1,n) \cap T_1\, out\,(s)) \geq Card\,(Succ\,(G_2,m) \cap T_2\, out(s))) \wedge$$
$$Card(Pred(G_1,n) \cap T_1\, out\,(s)) \geq Card\,(Pred\,(G_2,m) \cap T_2\, out(s))$$
$$Rnew(s,n,m) \leftrightarrow$$
$$(Card(\tilde{N}_1(s) \cap Pred(G_1,n)) \geq Card\,(\tilde{N}_2\,(s) \cap Pred(G_2,n)) \wedge$$
$$(Card(\tilde{N}_1(s) \cap Succ(G_1,n)) \geq Card\,(\tilde{N}_2\,(s) \cap Succ(G_2,n))$$

3.2.3 ETIQUETAS DE LOS NODOS

A la hora de incluir los atributos de los nodos dentro del emparejamiento (fase 2), este match puede realizarse de dos maneras, dependiendo de los símbolos o atributos numéricos. Para atributos simbólicos, en algunos casos son derivados de atributos numéricos, suponiendo que una relación de compatibilidad (semejanza) es definida entre dos atributos de los nodos. En algunas aplicaciones esta relación puede coincidir con la relación de equivalencia mientras en otros casos una definición más tolerante puede ser necesaria. En cada momento se hace la verificación de factibilidad de cada nuevo par de nodos, verificando también sus atributos. La verificación de las etiquetas formalmente está definida así:

$$Fsem\,(s,n,m) \leftrightarrow n \approx m$$
$$\wedge\, \forall\, (n',m') \in M(s), (n,n') \in B_1 \rightarrow (n,n') \approx (m,m')$$
$$\wedge\, \forall\, (n',m') \in M(s), (n',n) \in B_1 \rightarrow (n',n) \approx (m',m)$$

La Figura 17 muestra como ejemplo una consulta y una red formadas por 3 y 4 nodos respectivamente, dentro de cada nodo están ubicados sus respectivos subíndices. Inicialmente el algoritmo en su primer estado toma los primeros subíndices de los grafos como nodos candidatos es decir los nodos con subíndices 0 y 0. La función de factibilidad retorna verdadero y avanza al siguiente estado, donde los nuevos pares candidatos ahora son 2 y 1. La función de factibilidad muestra que los nodos candidatos deben ser desechados ya que el nodo candidato de la red no tiene aristas que salgan de él mientras que el nodo candidato de la consulta tiene una arista de salida (Figura 18 numeral 3); por esta razón el algoritmo debe regresar a un estado anterior (backtraking) para buscar otro camino y otro par de candidatos. Como el nodo del grafo de consulta se mantiene fijo hasta que no existan más caminos entonces el algoritmo avanza al siguiente índice en el nodo del grafo de la red, ahora el nuevo par de candidatos ahora son los nodos con subíndices 2 y 2, los cuales también son desechados debido a que no tiene conexión de nodo arista nodo, es decir el nodo candidato no está conectado con el nodo seleccionado en el estado (Figura 18 literal 5). Básicamente el algoritmo trata de encontrar pares candidatos, verificar su factibilidad y recorrer un grafo en busca de caminos que tengan la estructura del grafo de consulta, si los pares candidatos son desechados el algoritmo realiza backtraking en busca de un nuevo par de candidatos. Este procedimiento es realizado hasta que el algoritmo recorra todos los posibles caminos.

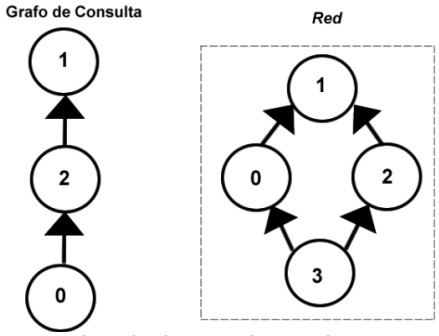
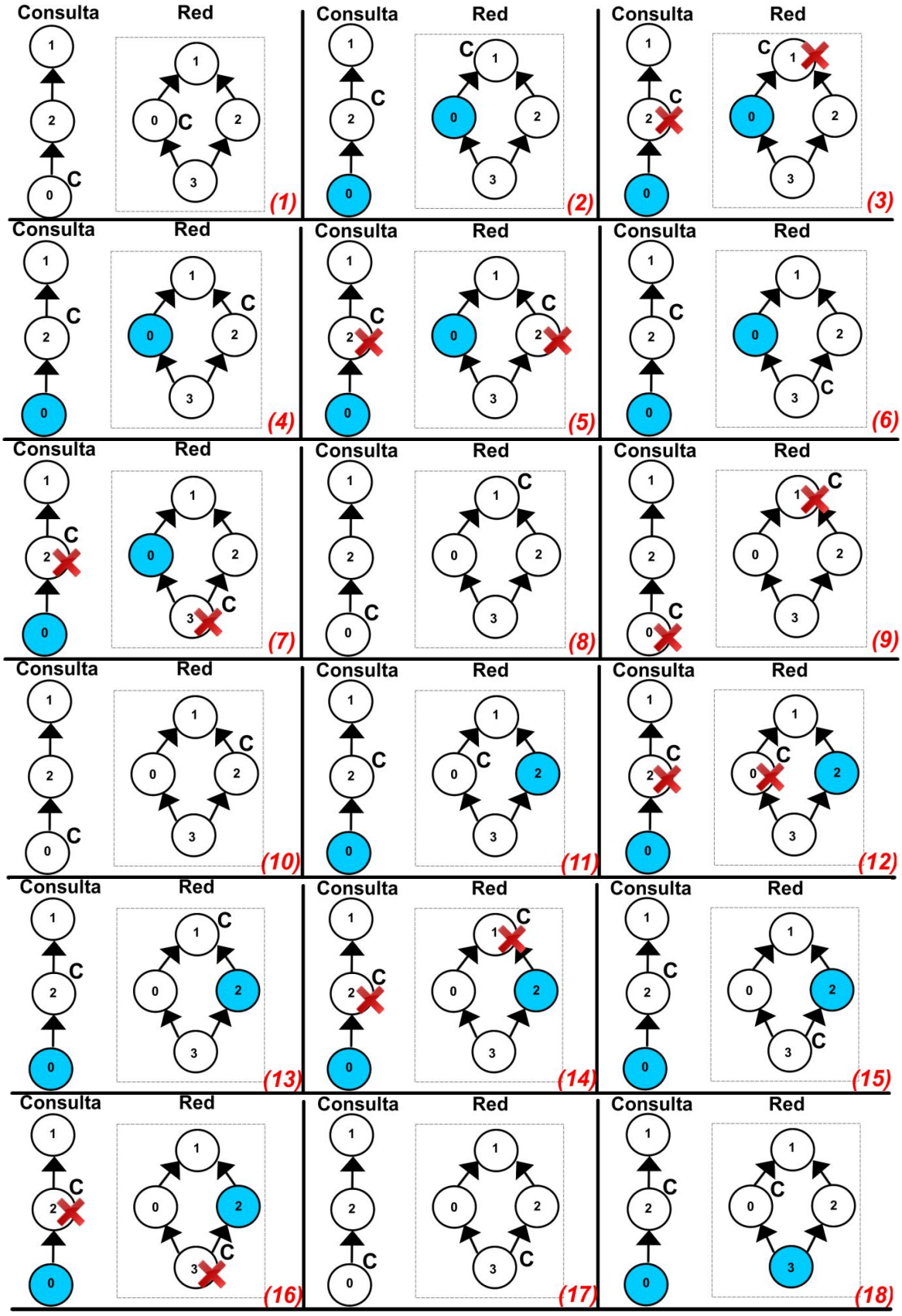


Figura 6 Ejemplo de consulta y red para VF2

En la Tabla 2 se muestra paso a paso el desarrollo del ejemplo anterior basado en la Figura 18.

Numeral	Estado	Nodos Candidatos	Factibilidad
1		0,0	Verdadero
2	0↔0	2,1	
3	0↔0	2,1	Falso
4	0↔0	2,2	
5	0↔0	2,2	Falso
6	0↔0	2,3	
7	0↔0	2,3	Falso
Backtracking			
8		0,1	
9		0,1	Falso
10		0,2	Verdadero
11	0↔2	2,0	
12	0↔2	2,0	Falso
13	0↔2	2,1	
14	0↔2	2,1	Falso
15	0↔2	2,3	
16	0↔2	2,3	Falso
Backtracking			
17		0, 3	Verdadero
18	0↔3	2, 0	Verdadero
19	0↔3 2↔0	1, 1	
20	0↔3 2↔0 1↔1		
Backtracking			
21	0↔3 2↔0		
22	0↔3 2↔0	1,2	
23	0↔3 2↔0	1,2	Falso
Backtracking			
24	0↔3		
25	0↔3	2,1	
26	0↔3	2,1	Falso
27	0↔3	2,2	Verdadero
28	0↔3 2↔2	1,0	
29	0↔3 2↔2	1,0	Falso
30	0↔3 2↔2	1,1	Verdadero
31	0↔3 2↔2 1↔1		
Backtracking			
32	0↔3 2↔2		
33	0↔3		

Tabla 2 Resultados del Ejemplo para el Algoritmo VF2


Consulta
Red
(1)
(2)
(3)
(4)
(5)
(6)
(7)
(8)
(9)
(10)
(11)
(12)
(13)
(14)
(15)
(16)
(17)
(18)

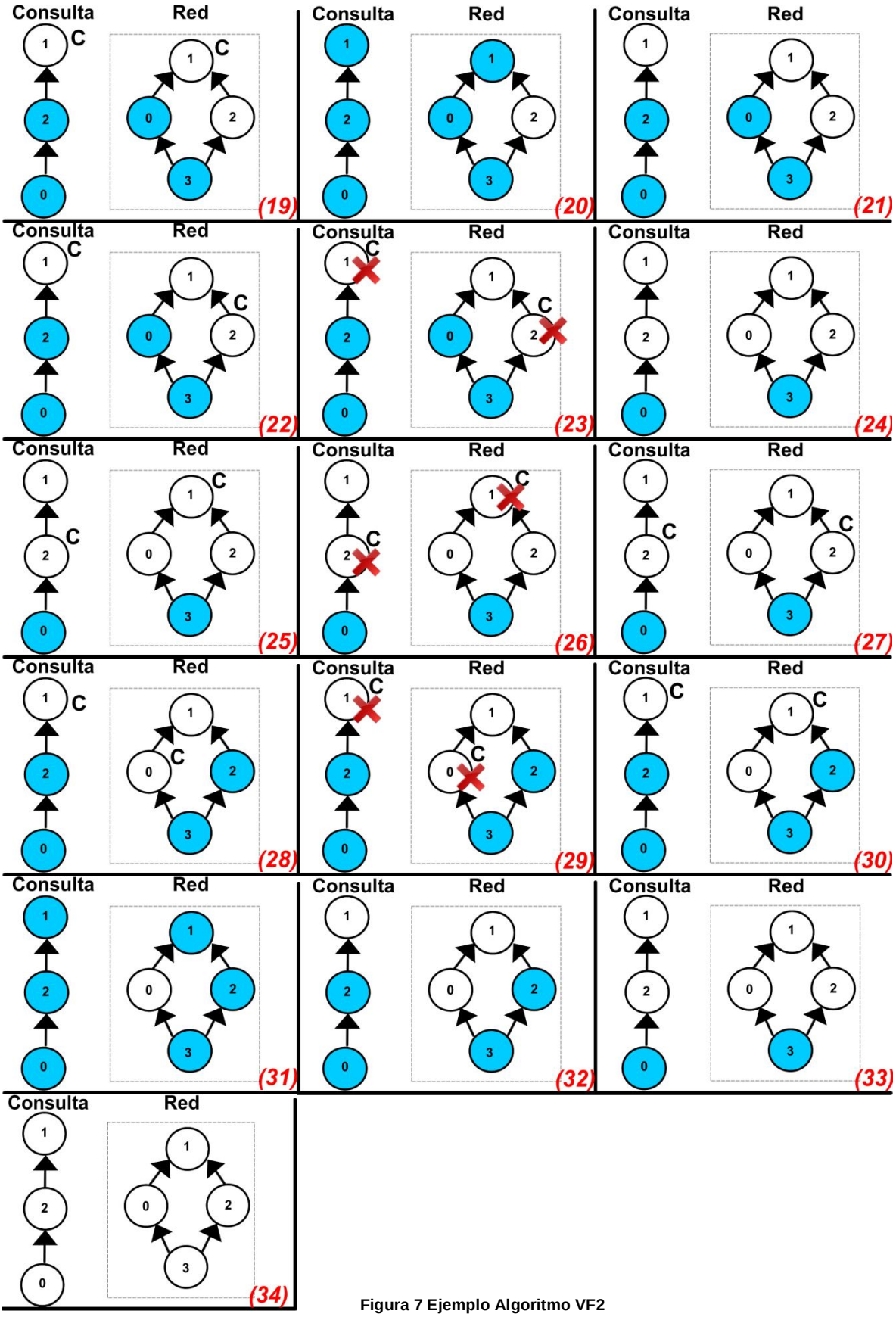


Figura 7 Ejemplo Algoritmo VF2

3.3 NETMATCH

NetMatch [65] es una herramienta que permite la detección de sub estructuras dentro de una red, dada una consulta. Diferentes sistemas biológicos tales como personas, organismos, células, proteínas, DNA, RNA y otras pequeñas moléculas surgen de interacciones complejas entre componentes. Estas redes son naturalmente modeladas como grandes grafos. Los cuales pueden ser analizadas usando técnicas de la teoría de grafos. En biología celular puede ser utilizada para observar si la conectividad de un tipo de gen es similar en algunas características a otros genes. En epidemiológica, la relación entre grafos y personas puede especificar patrones específicos de enfermedades y pueden ser usados para optimizar la entrega de vacunas. Las consultas a NetMatch se hacen a través de grafos y los resultados de las consultas son sub grafos de la consulta original que tienen la misma forma del grafo de consulta. Netmatch provee un eficiente algoritmo de emparejamiento de grafos con extensiones para manejar múltiples etiquetas por nodo, múltiples aristas entre pares de nodos y consultas aproximadas. Además NetMatch ha sido implementado como un plug-in para Cytoscape (software bioinformático de código abierto, para la visualización y análisis de redes biológicas) [66] permitiendo un rápido desarrollo de características adicionales.
Al igual que GraphBlast [67], la cual es otra herramienta para la detección de sub estructuras, el algoritmo utilizado por la herramienta NetMatch para la detección de sub grafos isomorfos es VF2; Sin embargo GraphBlast maneja únicamente grafos no dirigidos. Caso contrario ocurre con NetMatch donde la detección de sub estructuras es manejada a través de grafos dirigidos orientados hacia redes, de tal forma que pueda llevar un orden secuencial de los nodos.

Su funcionamiento consiste en primera instancia en cargar una consulta y una red, las cuales pueden ser dibujadas gracias a una de las funcionalidades que brinda NetMatch, recordando que las aristas entre los nodos son exclusivamente orientadas (dirigidas). Los nodos y las aristas pueden ser editados en cuanto a su estructura o las etiquetas. Seguidamente el algoritmo VF2 toma como entrada la configuración estructural tanto de la consulta como de la red para hallar la detección de todas las estructuras exactas del grafo de consulta que se encuentren en la red.

3.4 RESUMEN

Este capítulo corresponde a una contextualización en torno a la temática de detección de sub estructuras de grafos, comenzando por realizar una definición formal de la técnica de detección de grafos isomorfos y los distintos algoritmos que implementan esta técnica. Seguidamente se verifican estudios de rendimiento en tiempo de ejecución de los algoritmos de detección de isomorfismo, el algoritmo que mejor se adapta a las necesidades del presente proyecto de grado es VF2. El algoritmo VF2 es explicado en detalle con ejemplos básicos donde se muestran cada una de las funciones que utiliza VF2, tales como reglas de factibilidad, los pares candidatos entre otras funciones. Finalmente en este capítulo se describe una de las principales herramienta que hace uso de algoritmo VF2 explicando la escogencia de la misma. A partir de la definición y posterior explicación del algoritmo VF2 y de la herramienta que lo utiliza, se hace necesario detallar la forma como fue adaptado el mencionado algoritmo dentro de la aplicación final, lo cual es mencionado en el siguiente capítulo.

Capítulo IV

4 MECANISMO PARA LA DETECCIÓN DE EQUIVALENCIAS ENTRE PROCESOS DE NEGOCIO SEMÁNTICOS

Este capítulo describe el conjunto de algoritmos que conforman el mecanismo de emparejamiento de procesos de negocio BPEL4SWS, basados en el flujo de control del proceso, entradas, salidas y nombre de operación de cada una de las actividades básicas que componen un proceso de negocio.

El mecanismo utilizado para el emparejamiento de procesos de negocio se divide en 5 fases:

- ***Fase 1. Recuperación de procesos de negocio según su flujo de control:*** Esta fase implementa los algoritmos de recuperación de procesos de negocio según el flujo de control. Además ofrece un filtro opcional para el usuario, el cual consiste en recuperar procesos de negocio según el tipo de actividad que los componen.

- ***Fase 2. Evaluación de los procesos de negocio según los nombres de operación:*** Una vez recuperado un conjunto de procesos en la fase 1, se procede a evaluarlos, según la similitud del nombre de operación de cada una de las actividades básicas que los componen.

- ***Fase 3. Evaluación de los procesos de negocio según las entradas:*** Una vez recuperado un conjunto de procesos en la fase 1, se procede a evaluarlos según la similitud de las entradas de cada una de las actividades básicas.

- ***Fase 4. Evaluación de los procesos de negocio según las salidas:*** Una vez recuperado un conjunto de procesos en la fase 1, se procede a evaluarlos según la similitud de las salidas de cada una de las actividades básicas.

- ***Fase 5. Evaluación total de los procesos recuperados:*** En esta última fase se obtiene el resultado total que permite clasificar los procesos de negocio BPEL4SWS utilizando el mecanismo de emparejamiento propuesto.

Cabe enfatizar que el mecanismo de emparejamiento por fases permite reducir el tiempo de ejecución, además ofrece al usuario clasificaciones parciales de los resultados de emparejamiento.

La Figura 19 presenta el diagrama de fases del mecanismo de emparejamiento de procesos.

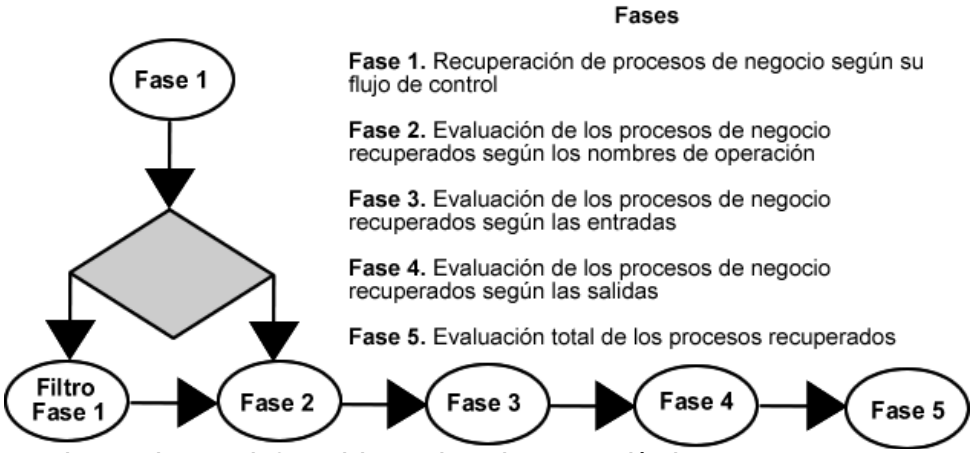


Figura 8 Diagrama de fases del mecanismo de recuperación de procesos BPEL4SWS

A continuación se explica detalladamente el banco de repositorios utilizado por el mecanismo de emparejamiento, la técnica utilizada para lograr la equivalencia entre descripciones BPEL4SWS a una representación forma de grafos, y el proceso desarrollado para cada una de las fases descritas anteriormente.

4.1 REPOSITORIO DE PROCESOS DE NEGOCIO BPEL4SWS

El banco de procesos de negocio (descritos en BPEL4SWS) para el prototipo implementado pertenece al dominio de las telecomunicaciones. Donde cada uno de las entradas y/o salidas de las actividades básicas de un proceso de negocio son enriquecidas semánticamente con conceptos propios del sector de las telecomunicaciones, utilizando la ontología SSID [68].

De acuerdo a [18], los procesos de negocio están clasificados en 5 dominios correspondientes a 5 áreas diferentes, representando un total de sesenta (60) procesos de negocio semánticamente enriquecidos.

Debido a que los procesos de negocio mencionados anteriormente fueron diseñados en BPMO fue necesario transformar dichos procesos a descripciones BPEL4SWS. Esta transformación fue realizada en su totalidad de forma manual ya que actualmente no existe una herramienta para modelar procesos de negocios BPEL4SWS. Aunque existe un plugin para WSMOSTUDIO [8], el cual permite la transformación de procesos BPMO a sBPEL [69], y permite transformar documentos sBPEL a procesos BPEL4SWS, este presenta ciertas deficiencias, por lo cual se opto por descartar el uso de esta herramienta. Estas deficiencias ocurren cuando se intenta transformar compuertas como Exclusive Choice, Deferred Choice, Multiple Choice.

Los procesos de negocio diseñados en el presente trabajo de grado son abstractos, por lo tanto solo tienen en cuenta la estructura modular sin analizar detalles de la ejecución de los mismos. Por otra parte, tal como lo afirman en [70], cabe mencionar que las actividades invoke, receive, reply e interactionActiviy tienen un solo atributo input (entrada) y output (salida), que sirven como canal de comunicación para el intercambio de mensajes, mientras que los procesos modelados en BPMO pueden tener una o más entradas y salidas. Por este motivo al definir el atributo de entrada y salida de los

procesos BPEL4SWS solo se tuvo en cuenta una de las entradas y/o salidas enriquecidas en los procesos BPMO.

Según el análisis realizado en el actual trabajo de grado la Tabla 3 presenta los elementos BPMO y su correspondencia al estándar BPEL4SWS.

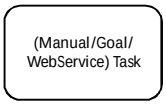
Elementos BPMO		Etiqueta BPEL4SWS	Descripción
Eventos:			
Start	 Start	<process>	Apertura de un proceso
End	 End	</process>	Cierre de un proceso
Timer	 Timer	<wait >	Especifica un tiempo de espera
Error	 Error	<terminate>	Termina la ejecución de un proceso
Message Receive	 Receive Message	<receive>	Espera por la llegada de un mensaje
Message Send	 Send Message	<invoke>	Invocación de una operación de un Servicio
Tareas:			
ManualTask GoalTask WebServiceTask	 (Manual/Goal/ WebService) Task	<receive> <invoke> <reply> <interactionActivity>	Las tareas de BPMO pueden ser representadas a través de estas etiquetas debido a que sobre estas tareas es que se hacen los llamados y se recibe por los servicios.
Compuertas:		**Compuertas**	
ExclusiveChoice	 Exclusive Choice	<switch> <case> </case> <otherwise> </otherwise> </switch>	Representa un switch con dos ramas.
Discriminator	 Discriminator	<pick> </pick >	Esta actividad espera que ocurra uno de varios eventos y ejecuta la actividad asociada a dicho evento
ParallelSplit	 Parallel Split	<flow>	Especifica un conjunto de actividades que deben ser ejecutadas concurrentemente.
Sequence		<sequence >	Reúne un conjunto de actividades para ser ejecutadas secuencialmente.
Synchronisation	 Synchronize	</flow >	Especifica el cierre de las actividades de tipo flow
While		<while >	Ejecuta iterativamente una o varias actividades, mientras se cumpla una condición
MultipleChoice	 Multiple Choice	<switch> <case> </case> <case> </case> .. .<otherwise> </otherwise> </switch>	Representa un switch con múltiples casos.

Tabla 3 Elementos BPMO y su correspondiente etiqueta en BPEL4SWS

4.2 TRANSFORMACIÓN DE BPEL4SWS A GRAFOS

En esta sección se describirá el método utilizado para lograr la equivalencia entre una descripción BPEL4SWS y una representación formal de Grafos para procesos de negocio, utilizando como base la técnica encontrada en [14]. A continuación se explicara la técnica tomada como base.

Un grafo tiene un nodo de comienzo y puede tener múltiples nodos de finalización. Un grafo también posee dos clases de nodos: los nodos regulares, que constituyen las actividades y los conectores, que representan las reglas de separación (*Split*) o unión (*Join*) del tipo *XOR* o *AND*. Los nodos son conectados mediante aristas, las cuales pueden tener un resguardo (guard*)* opcional. Los resguardos (*guards*) son condiciones que pueden ser evaluadas como verdaderas o falsas.

En esta transformación, se implementa la estrategia de barrido presentada en [71] para transformar un documento BPEL a un grafo BPEL. La idea general del barrido es mapear Actividades Estructuradas (*SA*) a sus respectivos fragmentos de grafo BPEL (Algoritmo 2). Éste algoritmo recorre la estructura anidada del flujo de control BPEL (BCF) de arriba hacia abajo y aplica recursivamente un proceso de transformación específico para cada tipo de actividad estructurada. Primero, los nodos *Start* y *End* son identificados en el documento BPEL, luego por cada actividad estructurada *Structured Activity graph* (*SAgraph*$_i$) es calculada ejecutando la función *BCF transform* en *SA*$_i$ (Algoritmo 3). Seguido el *SAgraph*$_i$ es adicionado al *BPEL graph*. Por cada *SAgraph*$_i$ adicionado al *BPEL graph,* el Algoritmo 2 calcula sus aristas con otras actividades estructuradas de grafos (*SAgraph*$_j$) considerando el mapeo *tc*, el cual es definido de acuerdo con las condiciones de transición de las Actividades Estructuradas (*SA*) en el documento BPEL.

Algoritmo 2. Función FlatteningBCF

```
INPUT: BPELdocument
OUTPUT: BPELgraph
ADD Start node to BPELgraph
ADD End nodes to BPELgraph
for each SAi do
        Calculate SAgraphi = BCF transform(SAi)
        ADD SAgraphi to BPELgraph
        Calculate the edge (SAgraphi ,SAgraphj ) = tc(SAi , SAj )
        ADD the edge (SAgraphi ,SAgraphj ) to BPELgraph
end for
return BPELgraph
```

El Algoritmo 3 presenta el procedimiento BCFtransform, el cual es invocado recursivamente de acuerdo al número de elementos anidados. En este algoritmo, el primer parámetro representa la actividad estructurada a ser procesada, seguida por el nodo predecesor y sucesor del grafo estructurado (SAgraph), entre los cuales, la estructura anidada está anclada (predecesor y sucesor).

El procedimiento BCFtransform, empieza comprobando si la actividad estructurada actual (SA) sirve como objeto y fuente de relaciones. Si es así, serán adheridos respectivamente conectores al comienzo y al final del bloque actual de la actividad. De este modo las relaciones en el flujo de control de BPEL son mapeadas a aristas y las respectivas

conexiones de unión (*Join*) o separación (*Split*), son adicionadas alrededor de las actividades básicas anidadas. Existen cinco sub-procedimientos para manejar las cinco actividades estructuradas: *Sequence*, *Flow*, *Switch*, *While*, y *Pick*. En éste punto, se asume que *Pick* es únicamente utilizada para modelar eventos alternativos de comienzo. La transformación de *Scopes* simplemente llama al procedimiento para esa actividad anidada. Actividades *Empty* son mapeadas a una arista entre el nodo predecesor y el nodo sucesor. *Terminate* es mapeada a un evento *End*. Además, cada actividad Básica BPEL (*Receive*, *Reply*, *Invoque*, *Wait*) es transformada a un nodo de grafo BPEL. Puesto que el control de flujo BPEL es de importancia, la actividad *Assign* es mapeada a una arista entre nodos sucesores y predecesores.

Algoritmo 3. Función BCFtransform

INPUTS: *SA*
OUTPUT: *SAgraph*
Calculate *BCF transform*(*SA, Predecessor, Successor, Connector*)
if ∃(*Link*$_i$, *SA*) **then**
    ADD a *SplitConnector*$_i$
    ADD an edge between the *P redecessor Activity* and the *SplitConnector*$_i$
end if
if ∃(*SA, Link*$_j$) **then**
    ADD a JoinConnector$_j$ (Link$_j$)
    ADD an edge between the *JoinConnector*$_j$ and *Successor activity*
end if
if*activity*∈*Seq***then**
    Calculate BCFtransformSeq(SA,Predecessor,Successor)
else if *activity* ∈ *F* low **then**
    Calculate BCFtransformFlow(SA,Predecessor,Successor,AND)
else if *activity* ∈ *P ick* **then**
    Calculate BCFtransformPick(SA,Predecessor,Successor,XOR)
else if*activity* ∈ *Switch* **then**
    Calculate BCFtransformSwitch(SA,Predecessor,Successor,XOR)
else if *activity*∈*W hile* **then**
    Calculate BCFtransformWhile(SA,Predecessor,Successor,XOR)
else if *activity* ∈ *Basic* **then**
    Calculate (Activity,Predecessor,Successor)
else if *activity* ∈ *Empty* **then**
    Calculate (Predecessor,Successor)
else if *activity* ∈ *Assign* **then**
    Calculate (Predecessor,Successor)
else if *activity* ∈ *Terminate* **then**
    Calculate (Predecessor,End)
end if
return *SAgraph*

Los procedimientos de BCFtransformSeq, BCFtransformFlow, BCFtransformPick, BCFtransformSwitch y BCFtransformWhile, generan los elementos del grafo BPEL que corresponden a sus respectivas actividades estructuradas BCF. BCFtransformSeq transforma un *Sequence*, conectando todas las actividades anidadas con las aristas del grafo. Aunque no esté definida explícitamente, esta transformación requiere un orden definido en las actividades anidadas. Para cada sub-actividad el procedimiento

BCFtransform es invocado nuevamente. La representación de grafos de Switch (BCFtransformSwitchprocedure), en el grafo BPEL, consiste de un bloque de ramas alternativas entre una separación (*Split*) *XOR* y una unión (*Join*) *XOR*. Las condiciones de ramificación son asociadas a las aristas.

La actividad *Pick* tiene cierta similitud a la actividad *Switch*. Pero, en lugar de evaluar una expresión, la actividad *Pick* espera por la ocurrencia de un conjunto de eventos y ejecuta las actividades asociadas. Esos eventos pueden ser relacionados a tiempo o a mensajes recibidos. Sintácticamente, el BCFtransformPick procedure mapea a los mismos elementos del flujo de control correspondientes a la actividad *Switch*. En el caso de las condiciones de *OnMessage*, el mensaje es especificado con elementos de flujo no controlable, similares a una actividad *Receive*. En el caso de un evento *OnAlarm*, el tiempo es modelado de la misma manera que para una actividad *Wait*. Cada evento alternativo es seguido por actividades anidadas combinadas con una unión (*Join*) *XOR*. El BCF procedure es transformado a un bloque de ramas paralelas, comenzando con una separación (*Split*) *AND* y sincronizada con una unión (*Join*) *AND*. Para la actividad While, BCFtransformWhile crea un bucle entre una unión (*Join*) *XOR* y una separación (*Split*) *XOR*. La condición es adicionada a la arista. Para la transformación de *Scopes* simplemente se llama al procedimiento de esta actividad anidada.

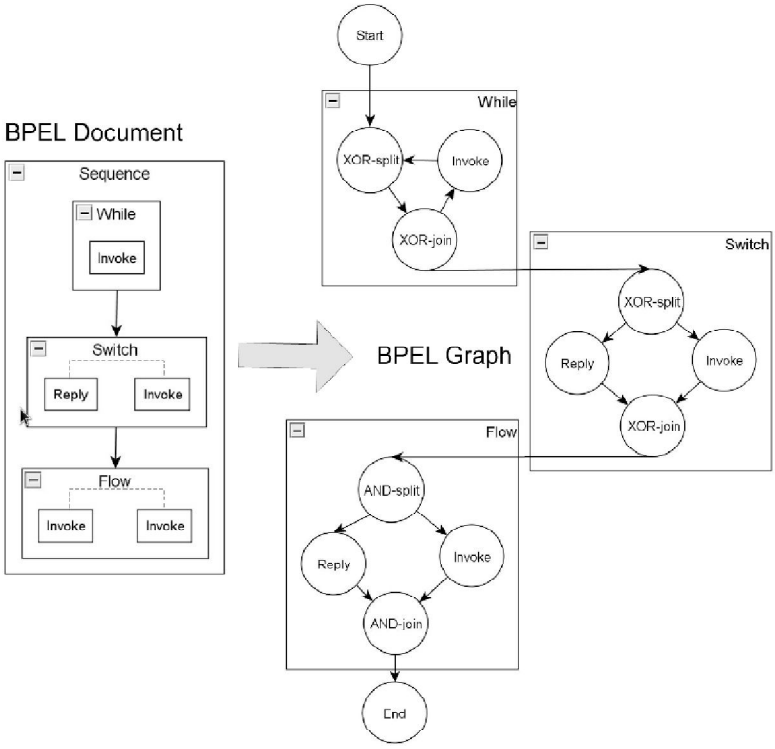


Figura 9 Correspondencia entre elementos BPEL y elementos de Grafos

Los nodos que representan las actividades básicas tienen los siguientes atributos: *ActivityType*(*AT*), *Operation* (*Op*), *PortType* (*PT*) y *PartnerLink* (*PL*). Los nodos conectores tienen dos atributos: *ConnectorType* (*AND-Split*, *AND-Join*, *XOR-Split*, *XOR-Join*) y *ActivityType* (la actividad estructurada BPEL4SWS desde la cual fue transformada). La Figura 20 muestra la correspondencia entre un esquema BPEL y elementos de grafos.
Con base en la técnica explicada anteriormente para transformar archivos BPEL a grafos, se considero agregar nuevas actividades y atributos que pertenecen al Lenguaje de Ejecución de Procesos de Negocio para Servicios Web semánticos (BPEL4SWS). Las

actividades agregadas fueron <extensionActivity> y <b4s:interactionActivity>, además del atributo saModelReference a cada una de las actividades básicas.

La actividad <extensionActivity> se decidió agregar ya que se considera la actividad padre de todas las nuevas actividades ofrecidas por BPEL4SWS, gracias a esto es posible mezclar actividades del estándar de BPEL 2.0 con actividades de BPEL4SWS. Por otro lado la actividad <b4s:interactionActivity> es la encargada de especificar cuando un proceso envía o recibe uno o muchos mensajes, además de invocar una conversación. La conversación es el nuevo elemento introducido por BPEL4SWS, encargado de desacoplar la WSDL de la lógica del proceso.

Finalmente se tomo la decisión de implementar el atributo saModelReference ya que este enriquece las actividades básicas, en otras palabras, específica el concepto ontológico al que pertenece cada actividad.

4.3 FASE 1: RECUPERACIÓN DE PROCESOS DE NEGOCIO POR FLUJO DE CONTROL

En esta fase se describirá el método utilizado para la recuperación de procesos en cuanto a su estructura o también llamado flujo de control, utilizando como base el algoritmo VF2 el cual se explico detalladamente en el capítulo III. Además de la búsqueda por estructura, esta fase permite emparejar el flujo de control del proceso teniendo en cuenta los tipos de cada actividad que componen el proceso de negocio. Cabe aclarar que la recuperación de procesos por tipo de actividad es opcional y solo es válida siempre y cuando se realice la recuperación de procesos según su flujo de control. Estos mecanismos reducen la búsqueda de procesos en el repositorio de acuerdo al proceso de consulta realizado por el usuario.

El algoritmo Netmatch utilizado para el desarrollo de esta fase, recupera grafos que tengan la estructura idéntica al grafo de consulta y grafos en los cuales este contenido el grafo de consulta (subgrafos), pero no considera el caso inverso, por este motivo se decidió realizar una adaptación al algoritmo el cual permita detectar grafos publicados que estén contenidos en el grafo de consulta. Para realizar estas modificaciones se decidió utilizar la representación basada en categorías como la presentada en [72].

En el contexto del presente proyecto las categorías son adaptadas de la siguiente forma:

- **Exact**: Dos grafos son iguales o exactos cuando el número de nodos del grafo de consulta tiene exactamente el mismo número de nodos del grafo publicado, además de tener la misma estructura (Figura 21 literal a).
- **Plugin**: Cuando el grafo de consulta está contenido en el grafo publicado. (Figura 21 literal b).
- **Subsume**: Esto ocurre cuando el grafo publicado está contenido en el grafo de consulta (Figura 21 literal c).
- **Fail**: En cualquier otro caso, la correspondencia es establecida como fail (Figura 21 literal d).

La Figura 21 presenta un ejemplo de las categorías de correspondencia para grafos explicadas anteriormente.

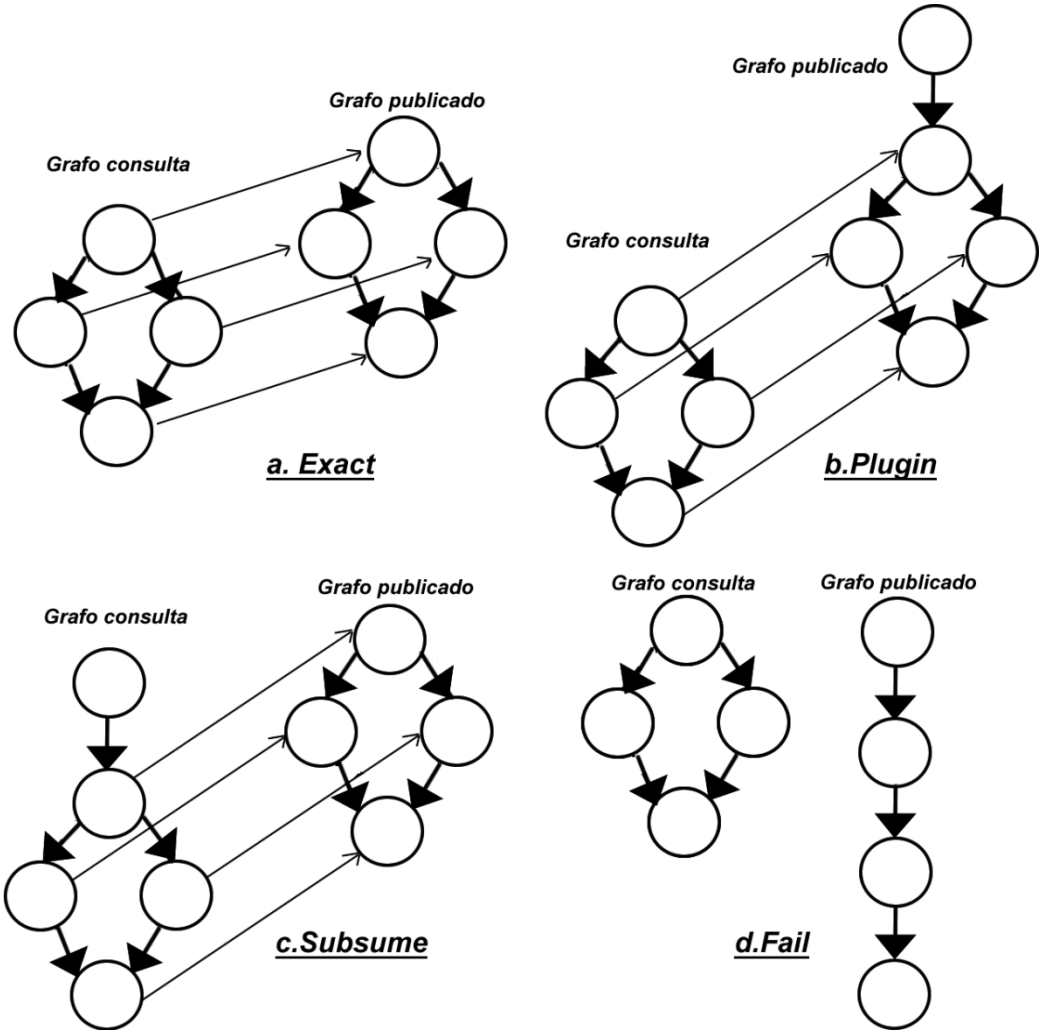


Figura 10 Categorías de correspondencia para grafos

La función SS formaliza las categorías antes mencionadas:

$$SS = \begin{cases} 1, & si\ (P_c \cap P_b = P_c = P_b): Exact \\ m_1, & si\ (P_c \subseteq P_b): Plugin \\ m_2, & si(P_b \subseteq P_c): Subsume \\ 0, & si\ (P_c \nsubseteq P_b) \wedge (P_b \nsubseteq P_c): Fail \end{cases}$$

$$Donde: \quad m_1 = \frac{|N_q \cap N_t|}{N_t} * \frac{1}{1+(N_t - N_q)} \quad Y \quad m_2 = \frac{|N_q \cap N_t|}{\#N_{Eliminar} + N_t} * \frac{1}{1+(\#N_{Eliminar})}, \ \textit{Ec. 1 y 2.}$$

$$p_c: Proceso\ de\ consulta, \qquad p_b: Proceso\ Publicado$$

En dicha expresión N_q y N_t representan el número de nodos del grafo de consulta y el grafo publicado respectivamente, y $|N_q \cap N_t|$ como su intersección.

La ecuación general m_1 está basada en el trabajo realizado en [14], donde calcula la similitud entre dos grafos basándose en medidas de distancia. De esta forma dos grafos son considerados similares cuando el valor de la medida de distancia entre ellos es

pequeño. A diferencia de [14], la presente expresión (m_1) modifica el cálculo de edición distancia por el número de nodos necesarios para transformar el grafo de consulta en el grafo publicado. De manera análoga, m_2 realiza la misma funcionalidad que m_1, pero aplicada a la categoría subsume, por esta razón la fórmula original sufre un cambio en los denominadores.

De acuerdo a lo anterior, para la expresión m_2 es necesario determinar el número de nodos del grafo publicado, que deben eliminarse para convertirlo en el grafo de consulta. En el primer denominador de m_2 este valor es adicionado al número de nodos de la consulta, esto se hace con el fin de penalizar los procesos que pertenecen a la categoría Subsume, ya que desde la perspectiva del usuario los procesos que se encuentran en la categoría Plugin requieren menos modificaciones que los procesos categorizados en Subsume.

Finalmente cabe resaltar que si la estructura del proceso de consulta está contenida más de una vez en el proceso publicado, el algoritmo generará en esta fase un solo resultado del emparejamiento que representara a todo el proceso en general. Sin embargo para las posteriores fases de emparejamiento son consideradas cada una de las subestructuras detectadas en el proceso publicado, con el fin de seleccionar cual es la mejor subestructura según el criterio de emparejamiento en análisis y así ubicarlo en la tabla de clasificación. En caso de encontrar un proceso que pertenece a la categoría Subsume, el procedimiento funciona de manera similar.

4.4 FASE 2: EVALUACIÓN DE LOS PROCESOS DE NEGOCIO SEGÚN LOS NOMBRES DE OPERACIÓN

El mecanismo para la evaluación por el atributo operación de las actividades básicas que componen un proceso de negocio BPEL4SWS, toma como referente el trabajo realizado en [14], donde son definidos un conjunto de algoritmos que permiten calcular la similitud entre dos etiquetas.

Cabe resaltar que dicho emparejamiento actúa únicamente sobre las actividades básicas de BPEL4SWS, debido a que los otros tipos de actividades no tienen el atributo operación. A continuación se explica detalladamente la técnica utilizada para detectar la similitud entre operaciones.

Como base para el desarrollo de esta fase se utilizo el modulo ***Linguistic Similarity (LS)***, tomado de [14], el cual calcula la similitud lingüística entre dos etiquetas basándose en sus nombres. Para obtener esta medida se utilizan los algoritmos Ngram, Check Synonym y Check Abbreviation. El algoritmo Ngram estima la similitud de acuerdo al número común de qgramas entre las etiquetas [73]. El algoritmo Check Synonym utiliza el diccionario lingüístico WordNet [74], para identificar sinónimos, mientras el algoritmo Check Abbreviation usa un diccionario de abreviaciones adecuado al dominio de aplicación. Si todos los algoritmos entregan un valor de 1, existe una coincidencia exacta entre las etiquetas, si entregan un valor de 0 no hay similitud entre las palabras. Si los valores entregados por Ngram y CheckAbbreviation son iguales a 0 y el valor de Check synonym está entre 0 y 1, el valor total de la similitud es igual a Check Synonym. Finalmente, si los tres algoritmos arrojan un valor entre 0 y 1, la similitud lingüística es el promedio de los tres. La función LS es descrita en el Algoritmo 4.

Algoritmo 4. Función Linguistic Similarity (LS).

$$LS = \begin{cases} 1 & Si(m1 = 1 \vee m2 = 1 \vee m3 = 1) \\ m2 & si(0 < m2 < 1 \wedge m1 = m3 = 0) \\ 0 & si(m1 = m2 = m3 = 0) \\ \dfrac{m1 + m2 + m3}{3} & si(m1, m2, m3 \in (0,1)) \end{cases}$$

Donde:

$m1 = Sim(NGRAM), m2 = Sim(Synonym\ Matching), m3 = Sim(Abbreviation\ Expansion)$

Antes de utilizar el modulo LS debe clasificarse cada nodo de los grafos de los procesos recuperados y los nodos del grafo de proceso de consulta, por sus respectivos tipos de actividad. De esta forma solo los nodos que pertenecen a un tipo de actividad básica entre el grafo de consulta y los grafos publicados son comparados. El algoritmo implementado, inicia ordenando en dos estructuras de datos cada uno de los nodos de los grafos, tanto de consulta como el proceso de publicación. Una vez ordenados se realiza la comparación del tipo de actividades, donde solo pueden ser evaluados bajo la función de similitud lingüística aquellos nodos que pertenezcan a una actividad básica (interactionActivity fue considerada actividad básica, y el atributo por el cual se emparejo fue el nombre, ya que esta actividad no tiene el atributo operación). Este emparejamiento se realiza uno a uno sobre la estructura de datos ordenada, por cada proceso publicado. A continuación se describe el algoritmo propuesto.

Algoritmo 5. Calculo promedio comparación lingüística por proceso de negocio

```
Inputs: BPELgraph: Strict (graph_input, graph_output)
Output: set of nodes organized by Activity Type: Invoke, Receive, InteractionActivity and Reply
begin
        AcumSS = 0
        Structure query =  orderGraph(graph_input,)
        Structure target =  orderGraph(graph_output)
        For( i := 0 to query.length)
           If((query[i].typeActivity =  Invoke OR query[i].typeActivity = Receive OR query      [i].typeActivity
           = InteractionActivity OR query[i].typeActivity  = Reply) AND (target [i].typeActivity =  Invoke OR
           target[i].typeActivity = Receive OR target[i].typeActivity = InteractionActivity OR
           target[i].typeActivity =  Reply))
                AcumSS = AcumSS + Linguistic_Similarity(query[i].nameOperation, target
                [i].nameOperation)
           end if
        end for
        return (AcumSS / target.length)
end
```

Una vez realizado el emparejamiento de cada nodo por el atributo operación, son obtenidos sus respectivos valores de similitud, con el objetivo de calcular el promedio de similitud, el cual es hallado por medio de la media aritmética, arrojando como resultado el promedio ponderado [75] de dicho proceso. La formula general de la media ponderada está determinada por:

$$\bar{X} = \frac{(X_1+X_2+X_3+\cdots+X_n)}{n} \qquad \textit{Ec. 3.}$$

Donde X, representa cada uno de los valores del cálculo de similitud obtenidos y n representa al número total de nodos. Luego de realizar estos cálculos, si dos o más procesos tienen el mismo valor de similitud, existe un segundo criterio para su clasificación que consiste en aplicar la varianza sobre todos los valores de similitud de los nodos. La varianza permite identificar la diferencia promedio que hay entre cada uno de los valores respecto a su punto central (Media $\bar{X}$) [76], sin embargo esta varianza es calculada de diferente manera tanto para una población como para una muestra. En el contexto actual del trabajo de grado, los procesos recuperados por la fase 1 hacen parte de una muestra del repositorio de procesos, por lo cual se ha determinado que la ecuación involucre la muestra de una población, y está determinada por la siguiente ecuación:

$$S^2 = \frac{(X_1-\bar{X})^2+(X_2-\bar{X})^2+(X_3-\bar{X})^2+\cdots+(X_n-\bar{X})^2}{(n-1)} = \frac{\sum(X_i-\bar{X})^2}{(n-1)} \qquad \textit{Ec. 4.}$$

Como ejemplo se tiene dos grafos recuperados (con 3 nodos cada uno), cuyo cálculo de similitud está determinado por: 0.1, 0.8, 0.9 y 0.7, 0.6, 0.5 respectivamente como se muestra en las tablas 4 y 5. Puede verificarse que el valor de la media de cada uno de estos grafos tiene un valor de 0.6, por lo cual los dos grafos estarían en la misma posición de clasificación, sin embargo no es posible elegir cuál de los dos grafos es más conveniente para el usuario, por lo tanto es viable tomar como referencia los valores que estén más cercanos a un punto medio. En el ejemplo la varianza para la Tabla 4 toma el valor de 0.11, mientras que para la tabla 5 es 0.01. Con esto se concluye que el grafo que tiene menor variabilidad respecto a la media, es el grafo presentado en la Tabla 5; Esto es evidente al analizar el valor inicial del primer grafo donde 0.1 está muy lejos de la media.

Nodos Grafo Consulta: Q Atributo: Operación	Nodos Grafo Publicado: P1 Atributo: Operación	Similitud
Comprar	Intercambiar	0.1
Vender	Revender	0.8
Rentar	Alquilar	0.9

Tabla 4 Similitud de grafos por el atributo operación – Q vs P1

Nodos Grafo Consulta: Q Atributo: Operación	Nodos Grafo Publicado: P2 Atributo: Operación	Similitud
Comprar	Adquirir	0.7
Vender	Ofrecer	0.6
Rentar	Permutar	0.5

Tabla 5 Similitud de grafos por el atributo operación – Q vs P2

4.5 FASE 3 Y 4: EVALUACIÓN DE LOS PROCESOS DE NEGOCIO SEGÚN ENTRADAS/SALIDAS

Esta sección describe el mecanismo utilizado para la evaluación de los procesos de negocio recuperados en la fase 1 de acuerdo a los atributos de entrada y salida de las actividades básicas que componen el protocolo BPEL4SWS. Como referente teórico es tomado el trabajo realizado en [18], donde son definidos un conjunto de algoritmos que permiten calcular la similitud de las entradas y salidas entre dos tareas BPMO. Así, a continuación serán expuestos los algoritmos que soportan éste proceso, presentando la

modificación realizada, para que el algoritmo de emparejamiento trabaje sobre los nodos que representan las actividades básicas.

Para llevar a cabo la técnica desarrollada en [18] es necesario que previamente los conceptos ontológicos estén asociados correctamente a los atributos que definen la funcionalidad de las actividades básicas (entradas y salidas). Una vez verificado este requisito, es esencial definir en primer lugar la distancia semántica existente entre los conceptos que conforman dicho enriquecimiento, puesto que de ella depende la clasificación de las relaciones en categorías o niveles de correspondencia y la posterior determinación de la similitud semántica entre los parámetros que definen la funcionalidad de las actividades.

A continuación es presentada una descripción detallada de los algoritmos empleados para definir la distancia y la correspondencia semántica entre los conceptos que representan las entradas y salidas de dos actividades básicas que pertenecen a diferentes procesos de negocio.

4.5.1 DISTANCIA SEMÁNTICA

La distancia puede ser entendida como una medida que refleja el grado de diferencia en el significado o sentido de dos conceptos pertenecientes a la misma ontología de dominio.

Para calcular la distancia semántica, previamente deben estar enriquecidas las entradas y salidas con conceptos de la ontología SSID, estos conceptos están distribuidos en una jerarquía de 10 niveles. En este sentido, es necesario definir unos criterios para garantizar que el valor de la distancia entre los conceptos comparados, refleje efectivamente la diferencia del significado implícita en la jerarquía de la ontología [77]. Estos criterios se enumeran a continuación:

1. La diferencia semántica entre conceptos de niveles superiores es mayor que la de conceptos de niveles inferiores en la ontología de dominio, en otras palabras, dos conceptos generales son menos similares que dos conceptos especializados.
2. La distancia entre conceptos hermanos (horizontalmente adyacentes) es mayor que entre conceptos padre e hijo (verticalmente adyacentes).
3. La profundidad del concepto raíz es cero, y la profundidad de los otros conceptos es igual a la longitud de su trayectoria (camino) hasta el nodo raíz.

Teniendo como referencia estos criterios, se determinó el siguiente procedimiento para calcular la distancia semántica entre dos conceptos correspondientes a dos entradas/salidas de los procesos de negocio de consulta y publicado:

En primer lugar es calculada la Distancia por Salto (D_{salto}) en cada uno de los niveles de la ontología que separan a los dos conceptos comparados, lo cual es realizada tomando un concepto de cada nivel y calculando su profundidad (depth), para posteriormente aplicar la siguiente fórmula:

$$D_{\text{salto}} = 1 + \frac{1}{2^{\text{depth}}} \qquad \textit{Ec. 5.}$$

Donde depth es el número de saltos en la ontología contados a partir del concepto raíz[3] hasta el concepto objetivo[4]. De esta manera, D_{salto} es calculado para cada concepto presente en el trayecto entre el concepto del enriquecimiento de la actividad de consulta y el concepto del enriquecimiento de la actividad publicada.

Posteriormente se calcula la Distancia Semántica (SD) entre los conceptos comparados a través de la sumatoria de los valores de D_{salto} asignados a cada uno de los trayectos que los separa:

$$SD = \sum D_{salto} \qquad \textit{Ec. 6.}$$

Para ilustrar mejor este procedimiento considere un ejemplo donde es calculada la distancia semántica entre una entrada de la actividad de consulta, enriquecida mediante el concepto A, y una entrada de la actividad publicada enriquecida con el concepto D de la ontología mostrada en la Figura 22.

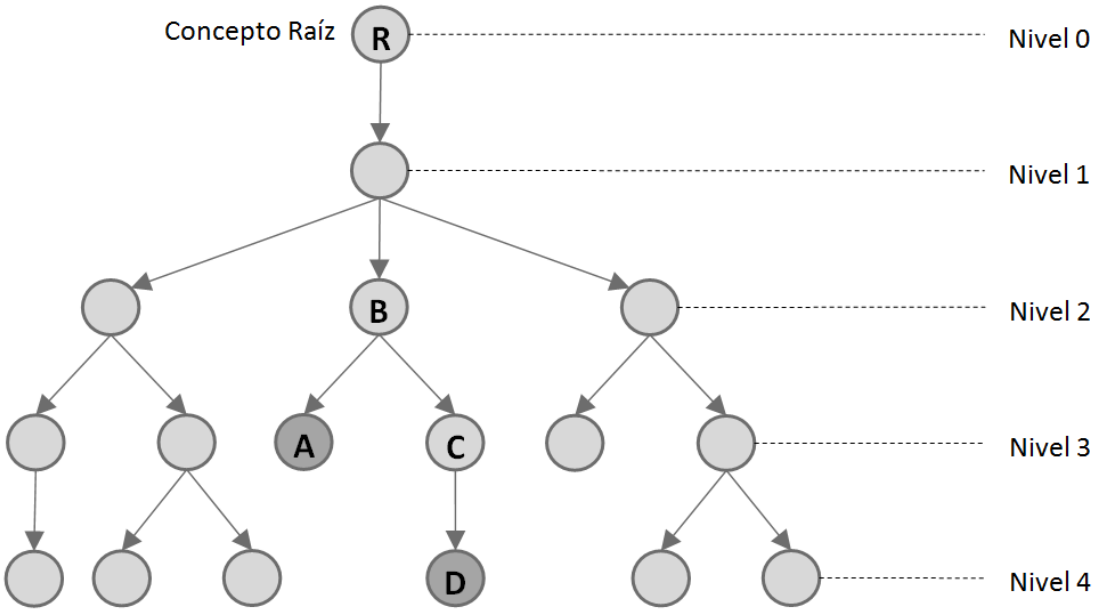


Figura 11 Ejemplo de Ontología de dominio por niveles

De acuerdo a lo descrito anteriormente, es necesario en primera instancia calcular las distancias (D_{salto}) que hay entre los segmentos A-B, B-C y C-D, para posteriormente sumarlas y obtener la Distancia Semántica total entre A y D. Estas distancias por salto se obtienen de la siguiente manera:

- Para el segmento A-B: el concepto objetivo es A dado que es el más alejado del nodo raíz, por lo cual la profundidad es 3 ya que corresponde al número de saltos que hay desde A a R, es decir $depth_A = 3$. Aplicando la ecuación 1 se tiene:

$$D_{salto\ (A-B)} = 1 + \frac{1}{2^3} = 1.125$$

[3] Es el primer concepto en la jerarquía de la ontología a partir del cual se ramifican los demás conceptos.
[4] Entre los dos conceptos de cada trayecto, el concepto objetivo es el que está más alejado del nodo raíz.

- Para el segmento B-C: el concepto objetivo es C, por lo cual la profundidad es el número de saltos que hay desde C a R, es decir $depth_C = 3$. Aplicando la ecuación 1:

$$D_{salto\ (B-C)} = 1 + \frac{1}{2^3} = 1.125$$

- Para el segmento C-D: el concepto objetivo es D, por lo cual $depth_D = 4$. Aplicando la ecuación 1 se tiene:

$$D_{salto\ (C-D)} = 1 + \frac{1}{2^4} = 1.0625$$

Finalmente la Distancia semántica entre A y D se obtiene sumando los valores de estos segmentos (ecuación 2), así:

$$SD = \sum D_{salto} = D_{salto\ (A-B)} + D_{salto\ (B-C)} + D_{salto\ (C-D)} = 3.3125$$

Mediante este ejemplo es posible comprobar el cumplimiento de cada uno de los criterios anteriormente definidos teniendo en cuenta que: la distancia entre el segmento A-B es mayor que la del segmento C-D (criterio 1), la distancia entre A-C es mayor que entre A-B o B-C (criterio 2), y finalmente los valores de profundidad para el concepto raíz y los nodos involucrados en la comparación son $depth_R = 0$, $depth_A = 3$, $depth_C = 3$ y $depth_D = 4$ (criterio 3).

La forma como es implementado este procedimiento dentro del prototipo, fue a través de la definición de un mecanismo para calcular la distancia semántica entre dos conceptos pertenecientes a la misma ontología de dominio (SSID), dicho mecanismo consiste en la ejecución de consultas sobre la ontología para encontrar los superconceptos[5] de los dos términos comparados. Para llevar a cabo estas consultas (las cuales están definidas en sintaxis wsml) se emplea el razonador WSML2Reasoner[78] soportado sobre el motor de inferencia de Datalog[6] denominado IRIS [79] .

Como resultado de las consultas realizadas, el razonador arroja un conjunto de superconceptos por cada concepto comparado, el cual almacena en un arreglo de cadenas de caracteres (Strings). Posteriormente, los arreglos obtenidos son analizados para determinar la distancia semántica entre los dos conceptos comparados. Sin embargo, para proceder con el análisis es necesario tener en cuenta que el razonador no entrega los resultados de la consulta de superconceptos en un orden coherente, por lo que es necesario ordenar los conceptos contenidos en estos arreglos en forma ascendente (es decir desde el término más específico al más general), de acuerdo con la disposición establecida en la ontología SSID. De esta forma, el concepto comparado siempre queda en el primer campo del arreglo, seguido por el concepto inmediatamente superior y así sucesivamente hasta llegar al concepto más general perteneciente a la misma "línea de ascendencia" del concepto comparado.

[5] Son los conceptos que se encuentran por encima del concepto a comparar y en la misma línea en la ontología de dominio.

[6] Datalog es un lenguaje de reglas y consultas para bases de datos deductivas. Las consultas están basadas en lógica de primer orden.

Este mecanismo es realizado con el fin de facilitar la comparación de los dos conceptos correspondientes al enriquecimiento semántico de las actividades de consulta y publicada, además, este mecanismo constituye en el punto de partida para determinar también el grado de correspondencia y la similitud semántica entre los dos conceptos.

Una vez explicado el funcionamiento para hallar la distancia semántica, a continuación es presentado el algoritmo utilizado (Algoritmo 6). Este empieza por evaluar si el concepto del enriquecimiento de la actividad del proceso de consulta (C_Q) es diferente al concepto de la actividad del proceso publicado (C_T), ya que si esta condición no se cumple significa que los dos enriquecimientos corresponden exactamente al mismo concepto, por lo cual la distancia entre ellos toma el valor de 0.

En el caso en que C_Q y C_T sean distintos se procede a encontrar sus superconceptos (denotados por SC_Q y SC_T respectivamente) y el concepto que es común en estos dos arreglos (CC), esto es realizado con el fin de definir el trayecto más corto entre los dos conceptos comparados. Posteriormente es calculada la sumatoria de todas las distancias por salto que hay entre CC y C_Q (en el arreglo SC_Q) y entre CC y C_T (en el arreglo SC_T). Por último se suma estos dos valores para obtener la Distancia Semántica total (SD) entre los dos conceptos.

Algoritmo 6. Calcular Distancia Semántica entre parejas de entradas/salidas

```
procedure findSemanticDistance(C_Q, C_T)
SD := 0
if C_Q ≠ C_T then
begin
        hops := 0
        {hops is the number of hops in the domain ontology that there are between C_Q and C_T}
        SC_Q [ ] := getSuperConcepts(C_Q)
        SC_T [ ] := getSuperConcepts(C_T)
        CC := findCommonConcept(SC_Q, SC_T)
        {CC is the closest common concept between SC_Q and SC_T in the domain ontology}
        dist1 := 0
        {dist1 is the number of hops between C_Q and CC in the domain ontology}
        dist2 := 0
        {dist2 is the number of hops between C_T and CC in the domain ontology}
        for j := 1 to indexOf(CC, SC_Q)
                {indexOf(CC, SC_Q) returns the index value for CC in SC_Q}
                dist1 := dist1 + 1 + 1/(2^(size(SC_Q)-j))
        for j := 1 to indexOf(CC, SC_T)
                dist2 := dist2 + 1 + 1/(2^( size(SC_T)-j))
        SD := dist1 + dist2
end
```

4.5.2 EMPAREJAMIENTO SEMÁNTICO

Teniendo en cuenta el mecanismo para calcular la distancia semántica explicado anteriormente, se uso el enfoque desarrollado en [18], para clasificar el grado de correspondencia entre los conceptos del enriquecimiento de las entradas/salidas de las actividades básicas comparadas:

- **Exact:** Cuando los dos términos del enriquecimiento comparados pertenecen al mismo concepto en la ontología SSID.
- **Plugin:** Cuando el concepto del enriquecimiento de la actividad del proceso publicado pertenece al conjunto de superconceptos del enriquecimiento de la actividad del proceso de consulta, y la separación entre conceptos es menor a dos saltos en la ontología de dominio.
- **Subsume:** Cuando el concepto del enriquecimiento de la actividad del proceso de consulta pertenece al conjunto de superconceptos del enriquecimiento de la actividad del proceso publicado, y la separación entre conceptos es menor a cuatro saltos en la ontología de dominio.
- **Intersect:** Cuando los conceptos del enriquecimiento semántico de las actividades de los procesos de consulta y publicación tienen superconceptos comunes, y la separación entre conceptos es menor a cuatro saltos en la ontología de dominio.
- **Fail:** Cuando la separación entre conceptos es mayor a cuatro saltos en la ontología de dominio (una distancia semántica de cuatro implica una disimilitud semántica considerable, teniendo en cuenta que la ontología de SID tiene diez niveles de jerarquía).

Tomando como referencia estas categorías el algoritmo presentado en [18] es utilizado para determinar la relación entre parejas de entradas/salidas de las actividades basicas comparadas (Algoritmo 7). De acuerdo con este algoritmo, cuando el concepto de la actividad del proceso de consulta (C_Q) sea igual al concepto de la actividad del proceso publicado (C_T) el grado de correspondencia es establecido como *exact*. De lo contrario se procede a calcular la distancia semántica entre los dos conceptos C_Q y C_T, si este valor resulta mayor que cuatro el grado de correspondencia se define como *fail*.

Si ninguna de las anteriores condiciones se cumple es necesario encontrar los superconceptos de C_Q y C_T (los cuales se denominan SC_Q y SC_T respectivamente). Luego es necesario evaluar si el concepto de la actividad del proceso publicado pertenece al conjunto de superconceptos del concepto de la actividad del proceso de consulta (es decir si C_T está contenido en SC_Q) y verificar si la distancia semántica entre C_Q y C_T es menor que 2, si ambas condiciones se cumplen es definido el grado de correspondencia como *plugin*. En caso de que estas condiciones tampoco se cumplan, es necesario examinar si el concepto de la actividad del proceso de consulta es superconcepto del concepto de la actividad del proceso publicado (C_Q está contenido en SC_T), si el resultado es positivo el grado de correspondencia es determinado como *subsume*.

De no cumplirse las condiciones para las cuales el grado de correspondencia sea *exact*, *plugin*, *subsume o fail,* el grado de correspondencia se define como *intersect*.

Algoritmo 7. Grado de Correspondencia entre parejas de entradas/salidas

procedure *SemanticMatchmaking*(C_Q,C_T)
if $C_Q = C_T$ **then** *matchingDegree* := exact
else
begin
    if *findSemanticDistance*(C_Q,C_T) < 4 **then**
    begin
        SC_Q := *getSuperConcepts*(C_Q)
        {SC_Q is a set of superconcepts of C_Q}
        SC_T := *getSuperConcepts*(C_T)
        {SC_T is a set of superconcepts of C_T}
        if ($C_T \subseteq SC_Q$ **and** SD < 2) **then** *matchingDegree* := *plugin*
        else if $C_Q \subseteq SC_T$ **then** *matchingDegree* := *subsume*
            else *matchingDegree* := *intersect*
        end
    else *matchingDegree* := *fail*
end
{the relation between C_Q and C_T is defined by *matchingDegree*}

Este mecanismo es utilizado por cada par de actividades del proceso de consulta y el publicado. Una vez calculado el emparejamiento semántico para cada par de actividades, se realiza un promedio entre todos los emparejamientos calculados anteriormente y el total se asigna como atributo al proceso publicado con el cual fue comparado el proceso de consulta. Este procedimiento es realizado para todos los procesos recuperados en la fase 1, además de calcular la varianza para dichos procesos. En caso de que el promedio sea igual para algunos procesos de negocio, estos procesos son ordenados por el valor obtenido en la varianza. Ubicando en los primeros lugares los procesos con varianza más cercana a cero. Este mecanismo de clasificación de procesos se realizó de la misma forma como se clasificaron los procesos en la fase 2.

4.6 FASE 5: EVALUACIÓN TOTAL DE LOS PROCESOS RECUPERADOS

Para la evaluación total de los procesos recuperados es necesario tener en cuenta cada una de las fases anteriores, para posteriormente calcular una ponderación sobre dichas etapas. Sin embargo no es viable elegir una ponderación basada en la media aritmética ya que los valores de cada fase no tienen el mismo peso, debido a esto se busca una aproximación de ponderación que sea útil para encontrar el promedio entre porcentajes. La media geométrica [80] fue elegida entre opciones como media, mediana, media aritmética, debido a que su principal uso es promediar porcentajes, encajando perfectamente en la evaluación total de los procesos recuperados ya que esta fase recibe como entrada los resultados de similitud de los procesos obtenidos por las diferentes fases. A continuación es presentada la expresión general de la media geométrica, la cual es definida como la raíz n- ésima del producto de los n valores.

$$MG = \sqrt[n]{X_1} * X_2 * X_3 * \ldots . * X_n \qquad \textit{Ec. 7.}$$

Donde el valor n-ésimo del radical así como la variable X, es igual a 4 (en caso de que se ejecuten las fases: 1, 2, 3,4) o 5 (en caso de que se ejecuten las fases: 1 por estructura, 1 por tipo de actividad, 2, 3 y 4). Una vez evaluados los procesos se procede a clasificarlos ordenándolos de mayor a menor, en caso de que existan procesos con el mismo valor, son ordenados por la medida de dispersión llamada varianza como se explico en la fase 2.

4.7 RESUMEN

Inicialmente este capítulo explica el mecanismo manual para transformar procesos de negocio BPMO a BPEL4SWS y el enriquecimiento de las entradas y salidas de las actividades básicas. Estos procesos hacen parte del repositorio que consta de 60 procesos de negocio. Seguidamente es presentado el mecanismo utilizado para transformar un proceso BPEL4SWS a una representación formal de Grafos y sus respectivas adaptaciones.

Finalmente se explican las etapas que conducen a la adaptación del algoritmo propuesto, donde se detalla cada una de las diferentes fases explicando los algoritmos utilizados y sus respectivos métodos para clasificar los procesos encontrados. A continuación es presentada una breve descripción de los mecanismos utilizados por cada fase:

- **Fase 1: Recuperación de procesos de negocio por flujo de control:** En esta fase se describió el método utilizado para la recuperación de procesos basado en su estructura, utilizando como base el algoritmo VF2.
- **Fase 2: Evaluación de los procesos de negocio según los nombres de operación:** Como base para el desarrollo de esta fase se utilizo el modulo Linguistic Similarity (LS), el cual calcula la similitud lingüística entre dos etiquetas. Para obtener esta medida se utilizan los algoritmos Ngram, Check Synonym y Check Abbreviation.
- **Fase 3 y 4: Evaluación de los procesos de negocio según entradas/salidas:** Para el desarrollo de estas fases se utilizaron algoritmos que calculan la distancia semántica entre dos conceptos de la ontología SSID, para posteriormente poder hallar la similitud semántica.
- **Fase 5: Evaluación total de los procesos recuperados:** Esta fase recibe como entrada los resultados de similitud de los procesos obtenidos en las fases anteriores, y calcula la media geométrica para cada uno de estos. Una vez terminado se clasifican los procesos dependiendo del valor de la media geométrica.

CAPITULO V

5 PROTOTIPO Y ARQUITECTURA DE LA PLATAFORMA

En este capítulo se presentan los resultados y artefactos generados en el proceso de ingeniería llevado a cabo para el desarrollo del prototipo.

El proceso de desarrollo de software fue guiado por la metodología del proceso unificado racional RUP (Rational Unified Process) [81]. Además se aprovecho la flexibilidad que brinda esta metodología para ajustarla a las necesidades específicas de este proyecto. Por otro lado, RUP es iterativo e incremental, dirigido por la realización de casos de uso y centrado en la arquitectura. Las fases que componen RUP son: iniciación, elaboración, construcción y transición. En cada una de estas fases se realizan un número determinado de iteraciones según el alcance del proyecto. Estas iteraciones están constituidas por ciertas actividades (análisis, diseño, implementación y pruebas) siguiendo un modelo en cascada y su esfuerzo varía dependiendo de la fase en la cual se encuentra el proyecto. A continuación se describe cada una de las fases mencionadas anteriormente.

- **Fase de Iniciación**: Esta fase tiene como propósito definir el alcance del proyecto y los requisitos funcionales para el desarrollo del prototipo. Se elabora la propuesta inicial del plan de trabajo, y se establecen las fases e iteraciones a seguir. Además se identificar los riesgos y todas las entidades externas con las que se trata (actores), definiendo sus interacciones a un alto nivel de abstracción (identificación de casos de uso).
- **Fase de elaboración**: En la fase de elaboración son seleccionados los casos de uso que permiten definir la arquitectura base del sistema, además de la especificación de los casos de uso.
- **Fase de Desarrollo:** Esta fase está enfocada en la construcción de un producto completamente operativo y eficiente.
- **Fase de Transición:** El propósito de esta fase es asegurar que el software esté disponible para los usuarios finales, ajustar los errores y defectos encontrados en las pruebas de aceptación, capacitar a los usuarios y proveer el soporte técnico necesario. Se debe verificar que el producto cumpla con las especificaciones entregadas por las personas involucradas en el proyecto.

A continuación son detalladas cada una de las fases, aplicadas al presente trabajo de grado.

5.1 FASE DE INICIACIÓN

En esta fase se determina la viabilidad y alcance del presente proyecto, mediante las bases conceptuales, investigación de presuntas alternativas y demás actividades que permiten estructurar el presente trabajo. Esta fase se compone de las siguientes actividades: Consideraciones iniciales, Captura y análisis de requerimientos funcionales e identificación de los casos de uso.

Consideraciones iniciales: Tomando como base la investigación documental realizada, donde se presentó la información acerca de las bases teóricas y trabajos relacionados

(capítulo II), surgieron diferentes consideraciones para tener en cuenta en el desarrollo del prototipo a construir. Dichas consideraciones son presentadas a continuación:

- **Metodología:** Para las fases de Iniciación, elaboración y transición, se ha definido una (1) iteración. Mientras que para la fase de construcción se han definido dos (2) iteraciones. Como resultado final de cada fase, surgen una serie de artefactos. En la fase de iniciación los artefactos generados son: lista de requerimientos, casos de uso detectados, diagrama de casos de uso, prototipos de interfaz de usuario, cronograma de actividades y lista de riegos, por otro lado los artefactos generados en la fase de elaboración son: Casos de uso extendidos, diseño de plan de pruebas, arquitectura del sistema y requisitos no funcionales, mientras que los artefactos generados en la fase de construcción son: diagramas de secuencia, diagrama de colaboración ó comunicación, diagrama de clases, diagrama de paquetes, reporte de ejecución de pruebas y código fuente del prototipo desarrollado. Finalmente en la fase de despliegue solo se genera los manuales de usuario.

  En el presente capitulo se muestran los principales resultados obtenidos en la aplicación de la metodología de desarrollo RUP, sin embargo, la totalidad de los artefactos se encuentran en el ANEXO A.

- **Descripción formal de procesos de negocio:** En el ANEXO E, se encuentra un articulo generado en la etapa investigativa denominado: *"Lenguajes para composición de procesos de negocio"*, donde se muestran en detalle, lenguajes de descripción sintácticos (WS-BPEL, WS-CDL, BPMN, YAWL, EPC) y semánticos (WSML, BPMO, OWL-S, BPEL4SWS). Además se realiza una comparación de estos lenguajes, sobre ocho (8) requerimientos, tomados de [82]. Según los resultados de comparar estos lenguajes, es posible afirmar que desde la perspectiva de recuperación de procesos de negocio, los lenguajes sintácticos, no garantizan el correcto descubrimiento, ya que omite componentes que son similares a los buscados por el cliente, u ofrecen al usuario resultados con baja medida de precisión, tal como lo exponen en [9, 10], debido a esto se recomienda utilizar protocolos semánticos. Por otro lado, los lenguajes basados en semántica mejoran el descubrimiento de servicios al inferir sobre los conceptos de un dominio, esto gracias a la utilización de una ontología, la cual enriquece las búsquedas recuperando un conjunto de servicios útil para el usuario. Sin embargo los protocolos semánticos presentados son poco usados en la actualidad por las grandes casas productoras de software a diferencia de BPEL4SWS, el cual representa la evolución de BPEL, para descripciones semánticas. Por estas razones en el presente proyecto de grado, se eligió a BPEL4SWS como lenguaje para describir los procesos de negocio.

  En [14], los autores plantean la transformación de procesos descritos en BPEL en una descripción formal basada en grafos (donde los nodos son las actividades y la aristas tienen las condiciones de transición que expresan las dependencia del flujo de control entre las actividades), con la intención de trasladar el problema de emparejamiento de procesos de negocio, a un problema de emparejamiento de grafos a través de reglas ya establecidas. Por consiguiente se ha decidido manipular y adaptar dicha herramienta, con el objetivo de transformar los procesos descritos en BPEL4SWS en una representación formal basada en grafos.

- **Detección de sub grafos isomorfos:** El proyecto está encaminado a la detección de equivalencias entre procesos de negocio, basado en criterios de similitud; uno

de estos criterios está orientado a la detección de similitud por estructura de los procesos de negocio. Por este motivo es necesario determinar un mecanismo que haga posible la detección de sub grafos isomorfos. Los autores en [56], comparan los actuales métodos y herramientas para la detección de sub grafos isomorfos, basado en pruebas de rendimiento que actúan sobre distintos tipos de grafos, como se menciona en el capítulo III. El resultado de dicha comparación muestra que el algoritmo VF2 es el mejor para la detección de grafos isomorfos.
Por otro lado, es preciso ubicar una herramienta que haga uso del algoritmo VF2, para su posterior adaptación al algoritmo propuesto en este trabajo de grado. NetMatch es un algoritmo de detección de sub grafos isomorfos que hace uso de VF2, el cual ha sido seleccionado debido a que utiliza grafos dirigidos, con un enfoque basado en redes, por eso esta herramienta recibe el nombre NetMatch, en español emparejamiento de redes.

- **Comparaciones lingüísticas y semánticas:** La detección de similitudes por entradas y salidas es realizado a través de inferencia semántica. Mientras que el emparejamiento por nombre de operación es desarrollado a través de comparaciones lingüísticas.
  Cuando se recuperan procesos de negocio basado en semántica implica más tiempo en la búsqueda de procesos al estar ligado a una ontología del dominio. Por otro lado las comparaciones lingüísticas no son tan precisas como las semánticas, pero realizan las comparaciones en un tiempo mucho menor que las comparaciones semánticas.
  El método utilizado para la comparación lingüística se basa en [14], el cual exponen un método donde calculan la similitud lingüística entre dos etiquetas, mediante el uso de los algoritmos: NGram, Check synonym y Check abbreviation. El algoritmo NGram calcula la similitud de acuerdo al número de caracteres comunes entre las dos etiquetas, en otras palabras registra y verifica cuantas coincidencias existen al comparar dos cadenas de entrada. El algoritmo Check synonym hace uso de un diccionario lingüístico, el cual es usado para hallar los sinónimos entre las etiquetas de los nombres, mientras que Check abbreviation usa un diccionario de abreviación de acuerdo al dominio de la aplicación. Por otra parte las comparaciones semánticas se basan en el método desarrollado en [18], el cual calcula la distancia semántica entre un par de conceptos pertenecientes a un ontología (SSID), para posteriormente calcular la similitud semántica.

Alcance del sistema: El prototipo permite el descubrimiento de equivalencias entre procesos de negocios, dado un proceso de consulta frente a un repositorio elegido. La única entrada del sistema corresponde a un documento descrito en BPEL4SWS. Seguidamente se hace la transformación a una representación formal basada en grafos. Los nodos representan las actividades del proceso de negocio (básicas y estructuradas). Los nodos que representan a las actividades básicas tienen los atributos: tipo de actividad, nombre de la actividad, nombre de la conversation (solo para la actividad interactionActivity), entrada y/o salida (dependiendo del tipo de actividad). A partir de este punto se realizan una serie de fases que conducen a comparar y encontrar equivalencias que tienen dos procesos. Las fases mencionadas son: i) Recuperación de procesos de negocio por flujo de control, ii) Evaluación de los procesos de negocio según los nombres de operación, iii) Evaluación de los procesos de negocio según las entradas, iv) Evaluación de los procesos de negocio según las salidas, y v) Evaluación total de los procesos recuperados. Los resultados obtenidos en cada una de las fases, son visualizados en diferentes tablas, ordenadas de acuerdo al grado de similitud de cada

proceso. Las tablas de clasificación tienen las siguientes columnas: Posición, Nombre del proceso, lista de nodos que componen el sub grafo encontrado, imagen del sub grafo encontrado y una opción que muestra el grafo encontrado dentro del repositorio. Además debe permitir visualizar tanto el proceso de consulta como el repositorio de procesos.

- **Captura y análisis de requisitos funcionales:** Los requisitos funcionales identificados para la plataforma son los siguientes:

  RF1. Publicar repositorios de procesos de negocio. El usuario puede crear diferentes repositorios de procesos de negocio.

  RF2.Visualización de los grafos del repositorio. Una vez cargado el proceso de consulta y seleccionado uno de los repositorios, el sistema muestra gráficamente el proceso de consulta y cada uno de los procesos que contiene el repositorio seleccionado.

  RF3. Recuperación de procesos de negocio por flujo de control. El prototipo recupera los procesos que más se ajusten a un determinado proceso de consulta en cuanto a su estructura (similitud estructural).

  RF4. Recuperación de procesos de negocio por flujo de control y tipo de actividad. El usuario tiene la opción de elegir si desea realizar el emparejamiento por tipo de actividad que compone un proceso. Este requerimiento es visto como un filtro de procesos para tener resultados más exactos. Previamente el RF3 debe haberse ejecutado.

  RF5. Evaluación de los procesos de negocio según los nombres de operación. El sistema permite realizar un emparejamiento teniendo en cuenta el nombre de la operación de las actividades básicas que compone un proceso. Previamente el RF3 debe haberse ejecutado.

  RF6. Evaluación de los procesos de negocio según las entradas. El sistema permite realizar un emparejamiento teniendo en cuenta las entradas de las actividades básicas que compone un proceso. Previamente el RF5 debe haberse ejecutado.

  RF7. Evaluación de los procesos de negocio según las salidas. El sistema permite realizar un emparejamiento teniendo en cuenta las salidas de las actividades básicas que compone un proceso. Previamente el RF6 debe haberse ejecutado.

  RF8. Clasificar los procesos de negocio. El sistema debe ser capaz de ordenar los procesos recuperados y evaluados en los requerimientos: RF4, RF5, RF6, RF7, para realizar una clasificación de estos procesos de acuerdo a su grado de similitud frente a un proceso de consulta.

  RF9. Evaluación total de los procesos recuperados. El sistema debe realizar una ponderación total teniendo en cuenta los porcentajes y la clasificación de todos los procesos realizados en RF8. Una vez ponderados, dichos procesos deben ser organizados ascendentemente.

Identificación de casos de uso: En esta sección es presentado el diagrama de casos de uso para el prototipo desarrollado. Los casos de uso se identificaron a partir de las funcionalidades que ofrece el sistema, a continuación se realiza una breve descripción de cada uno de ellos:

- **CU-001 Crear repositorio y publicar procesos:** Este caso de uso permite publicar documentos BPEL4SWS dentro de un repositorio de procesos de negocio. Además permite crear nuevos repositorios.

- **CU-002 Recuperar procesos por estructura:** Este caso de uso permite recuperar un conjunto de procesos de negocio de acuerdo a un proceso de consulta, según su estructura.

- **CU-003 Recuperar procesos por tipo de actividad**: Una vez ejecutado el CU-002, el usuario podrá recuperar procesos de negocio según los tipos de las actividades que los componen.

- **CU-004 Evaluar procesos por operación:** Este caso de uso permite al usuario ejecutar un emparejamiento de acuerdo al nombre de las operaciones de las actividades que componen un proceso de negocio. Previamente debe haberse ejecutado el caso de uso: CU-002.

- **CU-005 Evaluar procesos por entradas:** Este caso de uso permite al usuario ejecutar un emparejamiento de acuerdo al nombre las entradas de las actividades que componen un proceso de negocio. Previamente debe haberse ejecutado el caso de uso: CU-004.

- **CU-006 Evaluar procesos por salidas:** Este caso de uso permite al usuario ejecutar un emparejamiento de acuerdo al nombre las salidas de las actividades que componen un proceso de negocio. Previamente debe haberse ejecutado el caso de uso: CU-005.

- **CU-007 Clasificar procesos recuperados y evaluados**: Ordena los procesos de negocio recuperados y evaluados en los casos de uso: CU-002, CU-004, CU-005, CU-006.

- **CU-008 Evaluación total de procesos:** Este caso de uso realiza una evaluación total de los procesos de negocio recuperados y evaluados en los casos de uso: CU-002, CU-004, CU-005, CU-006. Previamente debe haberse ejecutado el caso los casos de uso: CU-002, CU-004, CU-005, CU-006.

- **CU-009 Abrir documento del proceso seleccionado:** Este caso de uso permite abrir y visualizar el documento BPEL4SWS de un proceso de negocio en particular.

- **CU-010 Visualizar proceso de consulta y repositorio de procesos**: El usuario puede visualizar los procesos de negocio que están contenidos en el repositorio, además del proceso de consulta.

- **CU-011 Visualizar procesos seleccionados**: El usuario puede visualizar un proceso de negocio seleccionado en la tabla de clasificación, distinguiéndolo de los demás procesos del repositorio.

En la Figura 23, se presenta el diagrama de casos de uso del sistema.

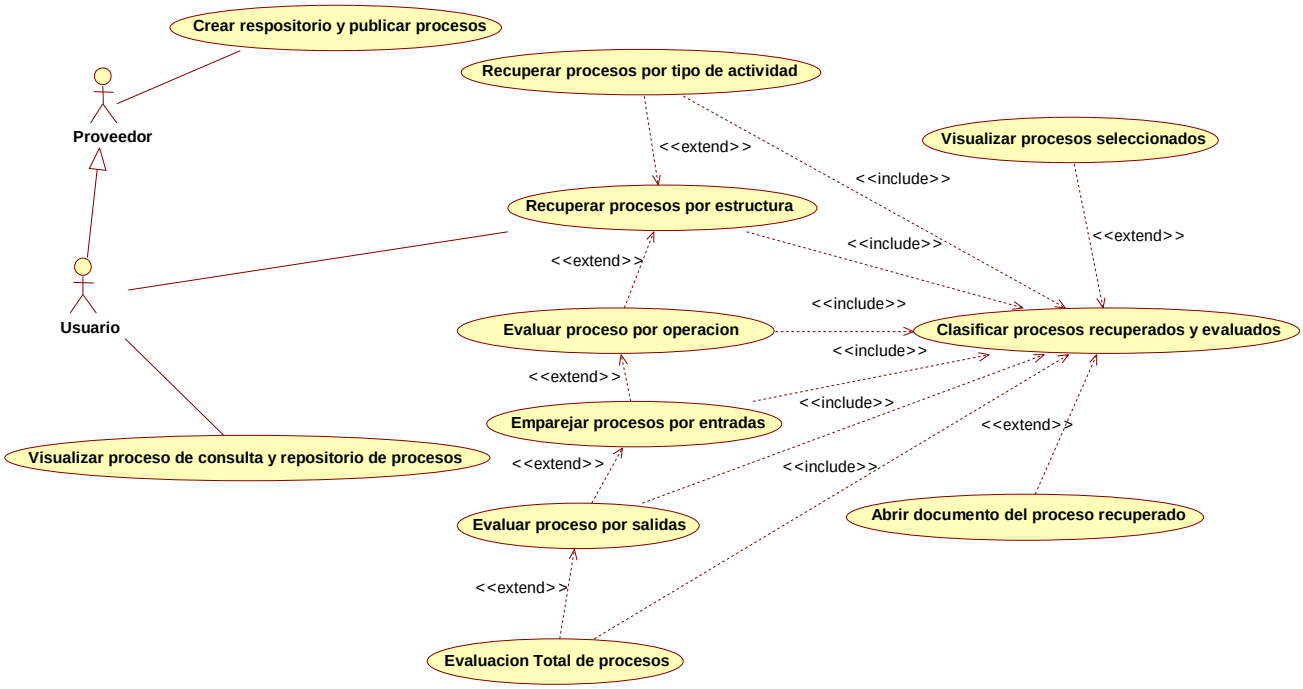


Figura 12 Diagrama de casos de uso del sistema

5.2 FASE DE ELABORACIÓN

Definición de la arquitectura

A continuación es presentada una breve descripción de cada uno de los subsistemas que la componen.

- **Usuario:** Corresponde a los usuarios que interactúan con el prototipo. Los usuarios ingresan al sistema un proceso de consulta, con el fin de recuperar un conjunto de procesos de negocio útiles para el usuario.

- **Transformador BPEL4SWS a Grafos**: Este modulo permite transformar descripciones BPEL4SWS a grafos dirigidos. El algoritmo emplea un proceso de transformación recursivo para cada tipo de actividad estructurada. Las actividades básicas de BPEL y las adiciones realizadas para BPEL4SWS, son transformadas en nodos, las secuencias son transformadas conectando los nodos requeridos por medio de aristas. Las actividades estructuradas son representadas por medio de operadores lógicos XOR y AND.

- **Repositorio de Procesos**: Representa el conjunto de procesos de negocio (BPEL4SWS) que contiene un repositorio.

- **Detector de estructuras:** Este modulo recupera los grafos mas similares al grafo de consulta en cuanto a su estructura. Dentro de este modulo se encuentran los sub módulos: detector de estructuras y detector de estructuras por tipo de actividad. El modulo detector de estructuras encuentra similitudes en cuanto a la estructura del grafo, y el modulo detector de estructuras por tipo de actividad como su nombre lo indica recupera grafos que contenga la misma estructura teniendo en cuenta que los nodos del grafo sean del mismo tipo que los nodos del grafo de consulta.

- **Comparador lingüístico**: Recibe como parámetros el grafo de consulta y los grafos recuperados en el modulo: detector de estructuras. Este modulo evalúa los grafos por el atributo operación que contiene cada nodo, utilizando el mecanismo implementando en [14], denominado similitud lingüística (Linguistic Similarity ó LS).

- **Comparador semántico:** Este modulo tomado como base de [18], realiza un emparejamiento uno a uno entre los atributos de entradas y salidas de los nodos que componen los grafos de consulta y los grafos del repositorio respectivamente. Es considerado un módulo importante ya que hace uso de la semántica de los procesos. En este bloque se realiza la implementación de los algoritmos que hacen posible la evaluación de entradas y salidas de las actividades que componen los procesos de negocio. El comparador está compuesto por los siguientes módulos:

    - **Determinador de Similitud:** Este módulo analiza los resultados obtenidos de las comparaciones de entradas/salidas suministradas por el modulo inferencia semántica, y establece un valor total de similitud semántica para la comparación de las entradas o salidas.

    - **Inferencia semántica:** La función principal de este módulo consiste en extraer de la ontología los súper conceptos, dado un par de conceptos (el par de conceptos son los nombres de las entradas o salidas). Seguidamente este módulo calcula la Similitud Semántica (SS) de acuerdo a la distancia Semántica (SD). Los resultados obtenidos en este módulo se entregan a los componentes superiores de la arquitectura para determinar la similitud existente entre los atributos de entradas y/o salidas de los grafos comparados.

- **Analizador de similitudes**: Este bloque representa al eje central de la arquitectura, el cual coordina el intercambio de parámetros entre los módulos, además clasifica y analiza los resultados obtenidos por cada proceso a partir de los módulos: detector de estructuras, comparador lingüístico, y comparador semántico, calculando medidas estadísticas como el promedio, varianza, y la media geométrica. Igualmente es el encargado de realizar la evaluación total de los procesos según los módulos explicados anteriormente.

- **Archivo BPEL4SWS de entrada:** Representa un documento BPEL4SWS el cual es la entrada del prototipo.

- **Rankings de resultados obtenidos:** Representa los resultados obtenidos a partir del comparador de procesos, el cual son mostrados al usuario.

En la Figura 24, es presentada la arquitectura planteada que da soporte al prototipo desarrollado.

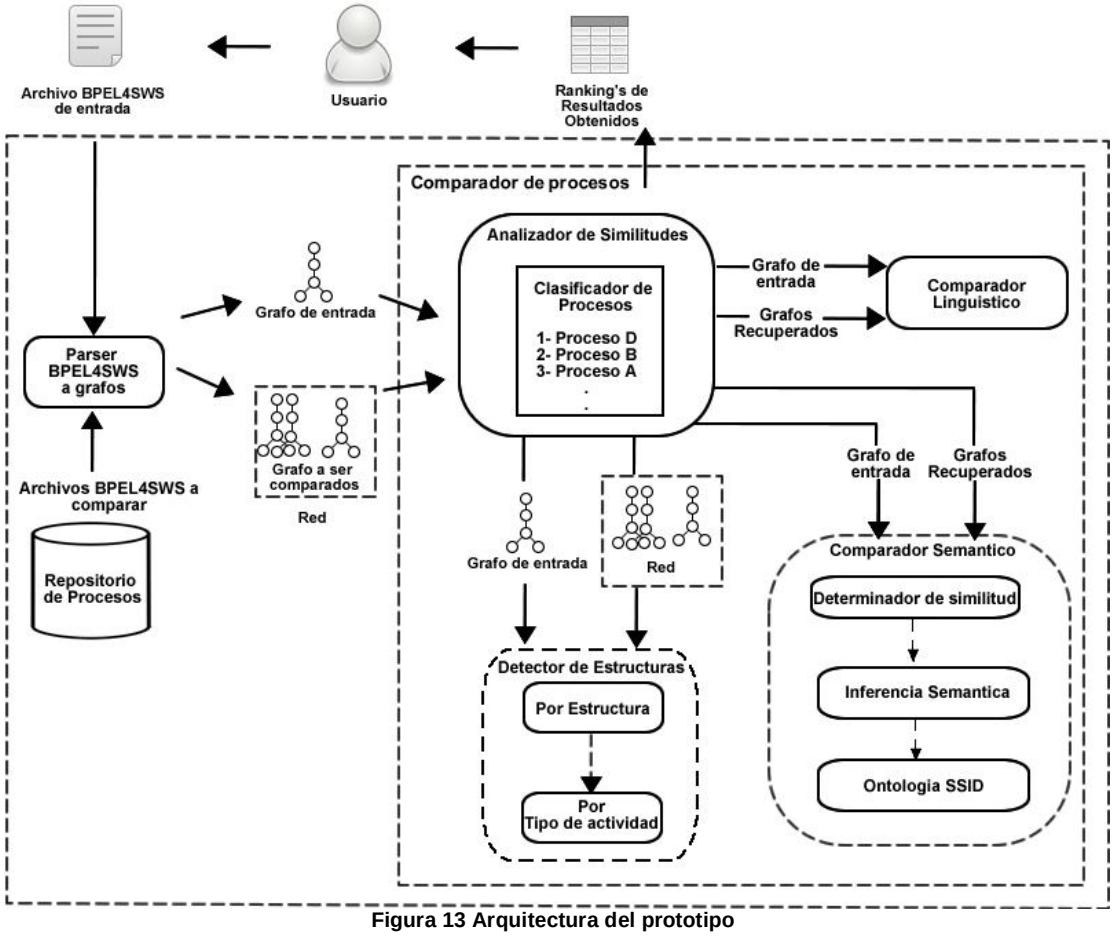


Figura 13 Arquitectura del prototipo

5.3 FASE DE CONSTRUCCIÓN

En esta sección se llevan a cabo los aspectos relacionados con el desarrollo y la construcción de la plataforma final. Para ello se hace uso de UML (Lenguaje unificado de modelado) [83].

Las dos iteraciones definidas anteriormente para la fase de construcción, han sido divididas de acuerdo a los casos de uso detectados, de la siguiente manera:

Iteración 1:

- CU-001 Crear repositorio y publicar procesos.
- CU-010 Visualizar proceso de consulta y repositorio de procesos.
- CU-002 Recuperar procesos por estructura.
- CU-011 Visualizar procesos seleccionados.
- CU-009 Abrir documento del proceso seleccionado.
- CU-003 Recuperar procesos por tipo de actividad

Iteración 2:

- CU-004 Evaluar procesos por operación.
- CU-005 Evaluar procesos por entradas.
- CU-006 Evaluar procesos por salidas.
- CU-007 Clasificar procesos recuperados y evaluados.
- CU-008 Evaluación total de procesos

Al final de cada una de las iteraciones se debe realizar la ejecución de las pruebas diseñadas en la fase de elaboración. Las pruebas que son diseñadas en la fase de elaboración y ejecutadas en la fase de construcción, están enfocadas en las funcionalidades y validación del sistema (ANEXO A).

- **Diagrama de Clases del Sistema**

  Esta sección se enfoca en el desarrollo del diagrama de clases del prototipo implementado. El diagrama de clases completo (con todos sus atributos y métodos) se encuentra en el ANEXO A. A continuación se realiza una breve descripción de cada una de las clases que componen el diagrama

  - **BpelPathRepository:** Contiene la lógica que permite cargar los procesos de negocio que se encuentran en un repositorio BPEL4SWS.

  - **BpelGraph**: Esta clase contiene la lógica suficiente y necesaria para realizar la transformación de las actividades BPEL a grafos.

  - **VisualStyleGraph:** Esta clase permite cambiar el estilo a los nodos de los grafos, en cuanto a su color y forma.

  - **HierarchicalLayoutAlgorithm**: Permite visualizar los grafos de manera ordenada, organizando los nodos de los grafos en jerarquía.

  - **ProcessGraph:** Permite almacenar en una representación de objetos la estructura resultante de la serialización del documento BPEL4SWS, representando un grafo mediante arcos, funciones, conectores y nodos. Las funciones son los nodos del grafo que representan acciones específicas. Los nodos representan cada una de las actividades del grafo.

  - **BpelTransform**: Implementa las estrategias de transformación de cada actividad BPEL4SWS. Esta clase es responsable de transformar el metadato de BPEL4SWS en un objeto java de tipo ProcessGraph.

  - **BpelReader:** Carga los archivos BPEL encontrados en el repositorio, para luego leerlos mediante lectores especializados que actúan sobre cada actividad, almacenando toda la información necesaria para el proceso de transformación. Para cada parte del archivo de entrada, se extraen sus componentes y se crean objetos que representan finalmente el proceso de negocio.

  - **Bbpel4swsReader:** Adiciona las nuevas actividades del lenguaje BPEL4SWS con respecto a su antecesor BPEL 2.0.

- **Activity:** Representa la actividad/nodo/función de un grafo. La cual puede ser: FuncionInvoke, FunctionReceive, FunctionReply, FunctionInteractionActivity. Cada función posee cinco (5) atributos: Tipo de actividad, nombre de la actividad, conversación, entrada y/o salida (dependiendo del tipo de actividad).

- **LinguisticSimilarity:** Calcula la similitud lingüística entre dos cadenas. Para obtener esta medida de similitud se hace uso de los algoritmos Ngram, Check Synonym y Check Abreviation, los cuales son implementados por las clases NgramMatcher, CheckSynonym y CheckAbreviation respectivamente.

- **CheckSynonym:** Implementa el algoritmo que permite calcular la similitud de dos etiquetas, usando un diccionario lingüístico para identificar sinónimos. Para este caso el diccionario utilizado es WordNet [74].

- **NgramMatcher:** Implementa el algoritmo que permite calcular la similitud de dos etiquetas, de acuerdo al número común de qgramas (tokens) entre las etiquetas.

- **CheckAbreviation:** Implementa el algoritmo que permite calcular la similitud de dos etiquetas, usando un diccionario de abreviaciones de acuerdo al dominio de la aplicación.

- **GUI:** Implementa la lógica necesario para la presentación de la interfaz grafica del prototipo.

- **Total:** Esta clase contiene toda la lógica necesaria para clasificar el total de las equivalencias como resultado total de la evaluación de las fases.

- **Matching:** Representa una clase abstracta para realizar todas las posibles fases de detección de equivalencias.

- **NetMatchTask**: Esta clase define el inicio y monitoreo de la aplicación Netmatch. Además define el emparejamiento por tipo de actividad.

- **NetMatch:** Esta clase es la encargada administrar las detecciones por estructura. Entre sus métodos se llama el algoritmo VF2, además de guardar el total de detecciones encontradas por dicho algoritmo.

- **VF2Monostate:** Esta clase implementa los métodos necesarios para detectar sub grafos isomorfos, tales como: buscar un par de nodos candidatos, verificar si dos nodos son factibles para formar un posible emparejamiento, agregar dos nodos candidatos en una estructura que almacena los grafos isomorfos detectados, backtracking, entre otras funciones. Además esta clase se modifico para permitir detectar similitudes de tipo Subsume (definida en el capítulo IV).

- **MyMatch:** La función principal de esta clase consiste en detectar las rutas por las cuales el algoritmo VF2 tiene la posibilidad de encontrar subestructuras. Además en esta clase se implementan las funciones que no permiten recorrer más de una vez, un camino descartado.

- **Graph:** Esta clase implementa las funcionalidades necesarias para que el algoritmo VF2 manipule los elementos de un grafo cuando este encuentra sub grafos isomorfos.

- **Input_output_similarity:** Esta clase tiene toda la lógica necesaria que permite obtener la similitud entre las entradas y/o salidas de las actividades de un proceso de negocio. Dichas entradas y/o salidas han sido previamente enriquecidas con conceptos ontológicos, por esta razón los métodos implementados tienen la facultad de detectar la similitud basándose en funciones semánticas.

- **Structure:** Esta clase tiene toda la lógica necesaria para clasificar los procesos recuperados en la fase de emparejamiento por estructura de grafos.

- **Operation:** Esta clase tiene toda la lógica necesaria para clasificar los procesos en la fase de emparejamiento por nombre de operación, basado en la similitud lingüística.

- **FunctionInteraction**: Manipula los atributos propios de una actividad BPEL4SWS, tales como: PartnerLinks, PortType, Operation, saModelReference y conversation.

La Figura 25 presenta el diagrama de clases de la aplicación, las cuales son descritas a continuación.

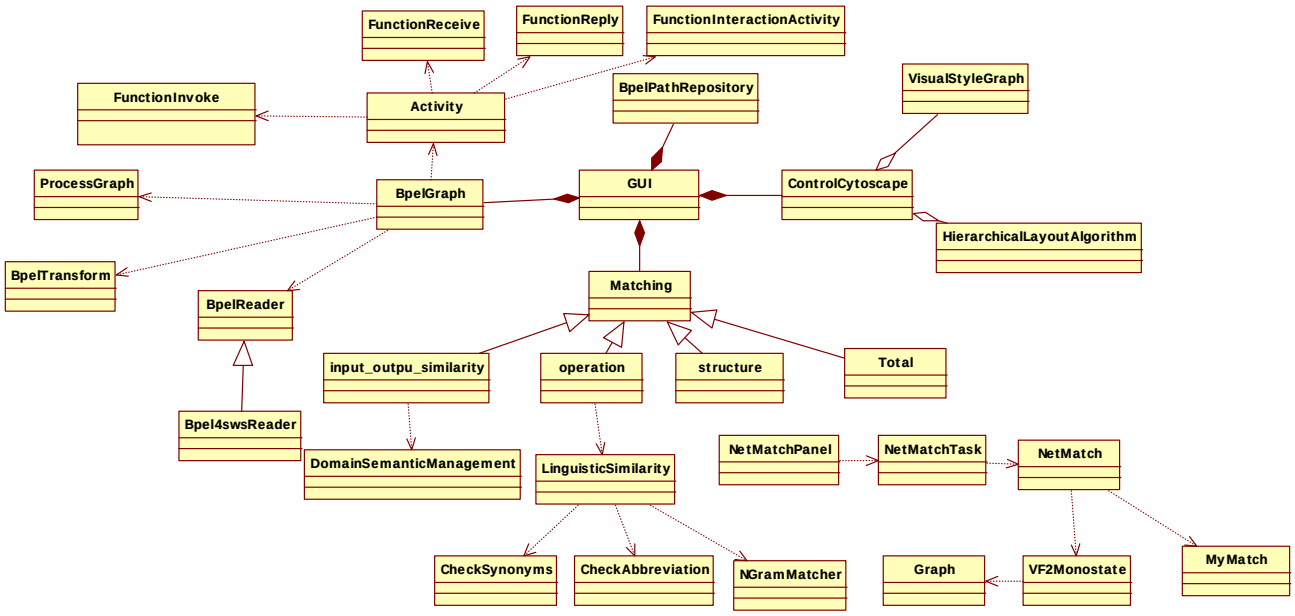


Figura 14 Diagrama de clases de la aplicación

- **Diagrama de paquetes:** Según [84], los diagramas presentados en esta sección, exponen la vista lógica de las aplicaciones software que componen el sistema. Dichos diagramas están organizados en paquetes, subsistemas y capas (Presentación, lógica del negocio y acceso a datos), mostrando la interacción existente entre capas así como también los paquetes más relevantes que las

componen. La Figura 26 presenta el diagrama de paquetes del prototipo. A continuación se describen las capas y su interacción.

- **Capa de presentación:** Esta capa es implementada mediante:

  - **USER_GUI:** Ofrece una interfaz grafica al usuario, que permite cargar un proceso con el fin de recuperar los procesos más similares al proceso de consulta.

- **Capa de lógica del negocio:** Los paquetes que implementan esta capa, representan las funcionalidades más importantes del sistema propuesto. En esta sección se realiza una breve descripción de cada uno de estos paquetes.

  - **Xerces:** Representa la serialización XML. Está compuesto por un conjunto de paquetes software, que permiten analizar y manipular los archivos XML. La biblioteca implementa una serie de API que incluye a DOM[7] y SAX[8].
  - **JWNL (Java WordNet Library**): Representa una API que permite tener acceso al diccionario WordNet. Además provee funcionalidades que van más allá del acceso a datos, tales como el descubrimiento de relaciones y procesamiento morfológico [85].
  - **WSML2Reasoner**: Esta capa, provee una plataforma para el razonamiento sobre las variables flight y rule de WSML, soportada en la infraestructura para referencia sobre formalismos basado en reglas. La plataforma opera, a partir de una transformación sintáctica de ontologías WSML a programas Datalog, por esta razón el razonamiento se realiza por medio de consultas Datalog aplicadas sobre una ontología transformada, en este caso SSID.
  - **VisualStyle:** Este paquete maneja todo lo referente al estilo visual de los grafos, permitiendo modificar la apariencia de los nodos en cuanto a su color y forma.
  - **Giny:** Al igual que la librería jgraph [85], contiene todas las funcionalidades de visualización e interacción gráfica. Éste paquete permite la visualización de los grafos tanto del repositorio como del proceso de consulta.
  - **Detector de equivalencias:** Este paquete contiene las clases encargadas de realizar la detección de equivalencias realizadas en las diferentes fases del algoritmo. Una vez realizadas cualquiera de estas fases se hace uso del paquete ranking.
  - **Ranking:** Representa a un paquete contenedor de las clases que implementan los diferentes algoritmos utilizados para clasificar los procesos encontrados en las fases de detección de equivalencias.
  - **Semantic:** Contiene las clases necesarias para hacer uso de la ontología SSID y razonar sobre los conceptos a través de WSMO2Reazoner. Además de esto permite el cálculo de la distancia semántica y la similitud semántica, aspectos que son necesarios en el cálculos de equivalencias por entradas y salidas de una actividad básica de un proceso de negocio.

[7] DOM: Document Object Model. Es un modelo en objetos para la representación de documentos HTML y XML
[8] SAX: Simple API for XML. Provee mecanismos que permiten leer datos de un document XML.

- **NetMatch:** Contiene las clases para interactuar con la herramienta NetMatch, la cual brinda diferentes métodos para la detección de sub grafos isomorfos, haciendo uso del algoritmo VF2.
- **Cytoscape:** Este paquete contiene los archivos JAR que permiten convertir los grafos BPELGraph a grafos Cytoscape para poder ser manipulados por NetMatch.
- **BpelGraph:** Representa las clases necesarias que hacen posible la transformación de los documentos XML a grafos BPEL4SWS.

o **Capa de acceso a Datos:** Esta capa tiene por objetivo servir como puente entre la capa lógica de negocio y el proveedor de procesos. Los paquetes que hacen parte de esta capa son descritos a continuación:

- **RepositoryBPEL4SWS:** Es un repositorio de procesos de negocios descritos en BPEL4SWS, sobre el cual se van a realizar consultas por parte de los diferentes usuarios de la aplicación.
- **Wordnet:** Base de datos léxica en ingles que agrupa las palabras en conjuntos de sinónimos llamados "synsets", proporcionando definiciones cortas y generales, almacenando las relaciones semánticas entre estos conjuntos de sinónimos. Este paquete es utilizado por el comparador lingüístico (a través del paquete JWNL) para verificar la relación semántica entre dos palabras.
- **SSID:** Representa a la ontología del dominio de las telecomunicaciones utilizada en este proyecto para el enriquecimiento de las entradas y salidas de las actividades básicas de los procesos de negocio.
- **CsPlugin:** Este paquete contiene las clases necesarias para ordenar el grafo de forma visual.

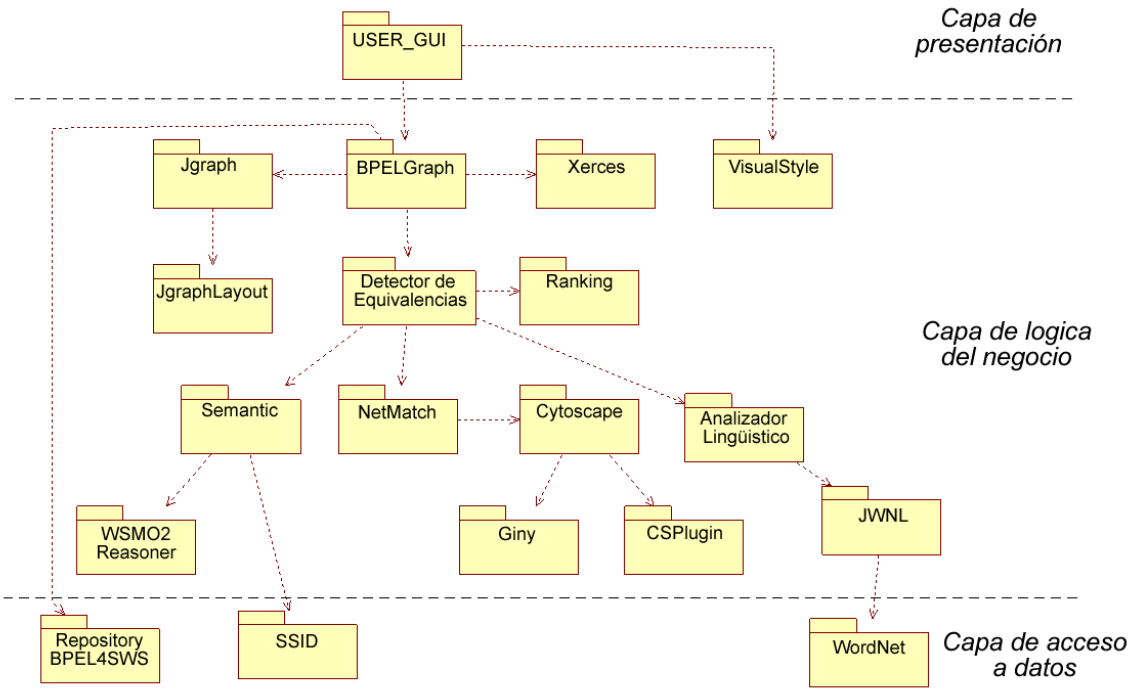


Figura 15 Diagrama de paquetes

- **Interfaces de usuario:** En esta sección son presentadas las interfaces graficas de usuario (GUI), del prototipo desarrollado basado en la metodología RUP.

  En la Figura 27 se puede observar la interfaz grafica de usuario del prototipo, la cual contiene los siguientes controles principales:

  - **Botón Select:** Es utilizado para cargar el proceso de consulta.
  - **Botón Match:** Muestra los resultados del emparejamiento por estructura, en la tabla ubicada en la parte superior derecha. La tabla tienen las siguientes columnas: Posición, Nombre del proceso, lista de nodos que componen el sub grafo encontrado, imagen del sub grafo encontrado.
  - **Botón Next Match:** Permite al usuario avanzar a la siguiente fase de evaluación de procesos. Los resultados son desplegados en la tabla superior derecha.
  - **Botón Back Match:** Permite regresar a una fase previa. Los resultados son desplegados en la tabla superior derecha.
  - **Botones Zoom:** Permiten aumentar o disminuir el tamaño del grafo de consulta y los grafos del repositorio.
  - **Combo REPOSITORY DOMAIN:** es usado para seleccionar un repositorio en particular

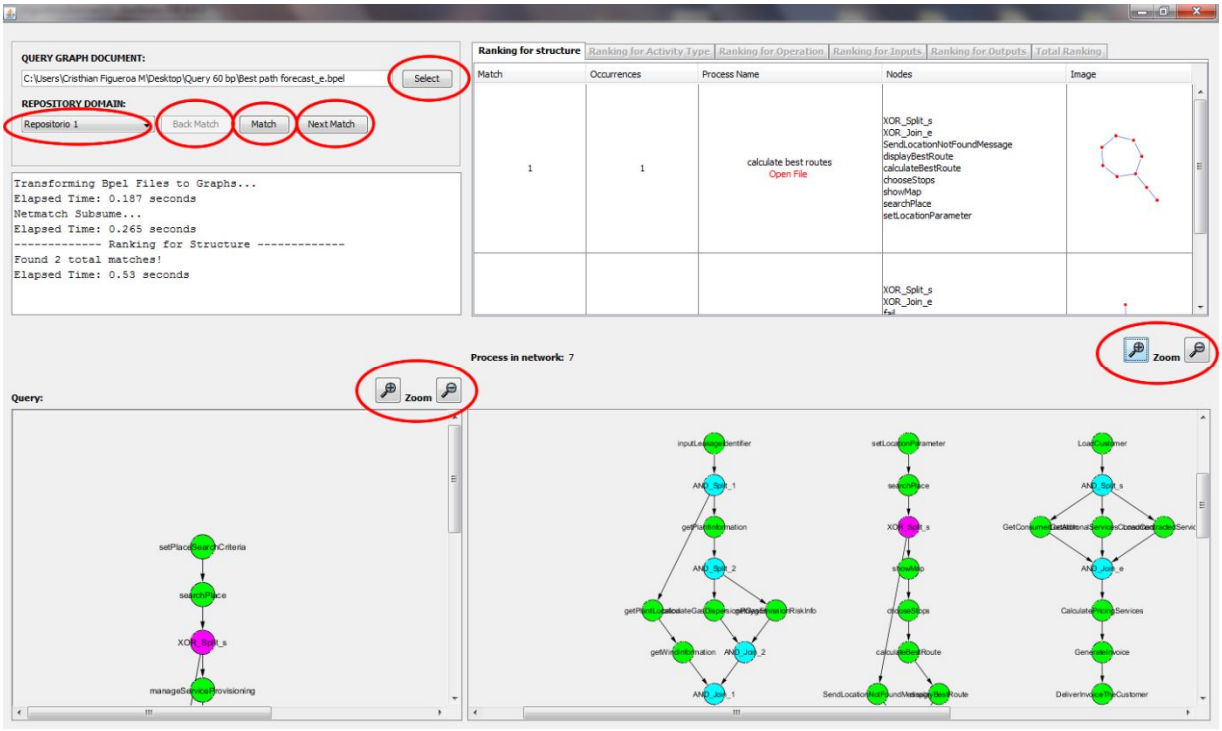


Figura 16 Interfaz Grafica de Usuario – Resultados

Por otra parte el panel ubicado en la esquina inferior izquierda muestra el proceso de consulta transformado a su representación en grafos, de la misma forma se muestra el repositorio en el panel ubicado en la esquina inferior derecha. Cabe resaltar que el conjunto de grafos representados en el panel derecho se denomina red.

Finalmente los colores de los nodos que componen un grafo, indican lo siguiente:

- ***Color verde:*** Actividades básicas (interactionAtctivity, receive, invoke, reply, wait, terminate, empty)
- **Color Magenta:** Actividades de tipo switch, While.
- **Color Agua marina:** Actividades de tipo flow, pick.

La Figura 28 refleja la implementación del *caso de uso visualizar procesos seleccionados.* Cuando el usuario presiona click izquierdo, sobre la imagen del grafo pintado situado en la tabla de clasificación, inmediatamente, este grafo es ubicado y pintado de color amarillo en el repositorio, mientras que los otros grafos del repositorio toman un color blanco, con el objetivo de diferenciarse con el grafo seleccionado.

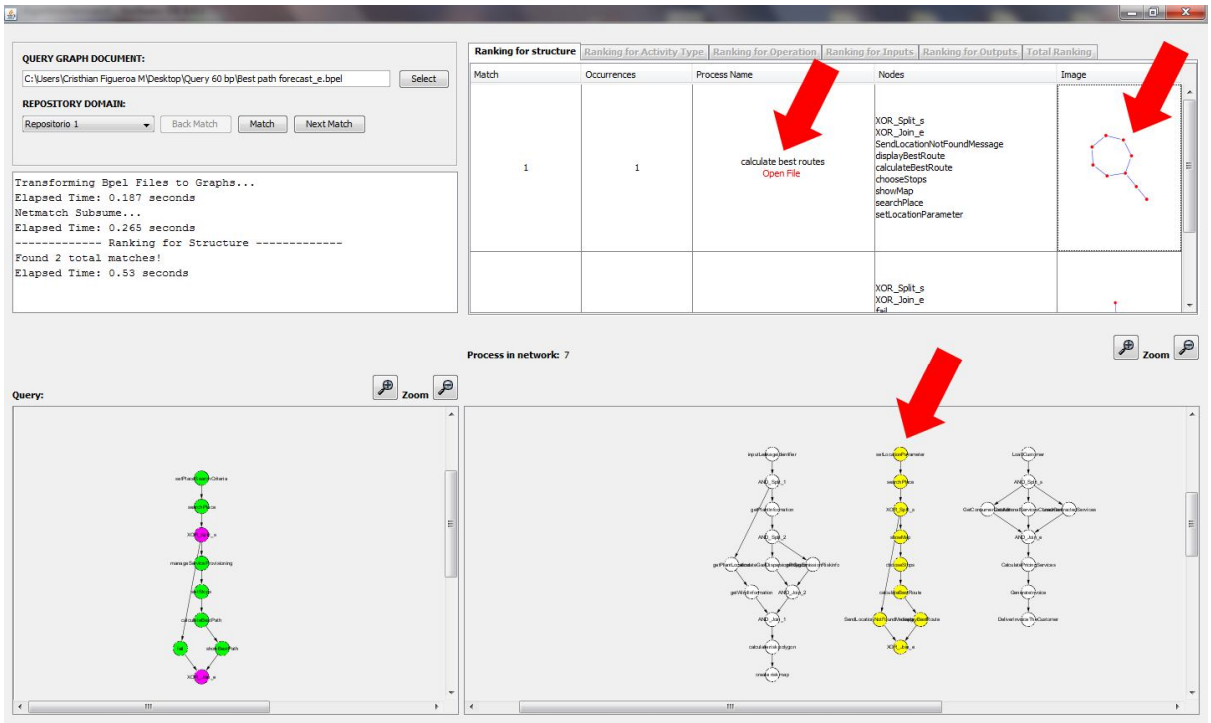


Figura 17 Interfaz Grafica de Usuario – Resultados

Por otra parte, el usuario puede seleccionar la opción *open file*, ubicada en la columna Process name. Una vez presionada esta opción, la aplicación abre en un editor de texto el proceso de negocio seleccionado.

5.4 RESUMEN:

En este capítulo, se expone el proceso de ingeniería realizado para definir la arquitectura que soporta el prototipo desarrollado de detección de equivalencias entre procesos de negocios semánticos. Entre los aspectos teóricos expuestos en este capítulo se describieron las consideraciones previas a la definición de la arquitectura y desarrollo de la misma. Como resultado de aplicar la metodología de desarrollo RUP en el presente proyecto surgieron una serie de artefactos para cada una de las fases concernientes a la metodología.

Por otro lado, en este capítulo solo fueron mostrados los artefactos más relevantes junto con su descripción, sin embargo, la totalidad de los artefactos están plasmados en el ANEXO A, estos artefactos están distribuidos en diferentes fases de la siguiente forma:

- **Fase iniciación**: Lista de requisitos, lista de riesgos, Diagrama de casos de uso, prototipos de interfaz de usuario, cronograma de actividades.

- **Fase elaboración:** Diagrama de casos de uso extendidos, plan de pruebas (ANEXO B), arquitectura, requisitos no funcionales.

- **Fase Construcción:**
    - **Análisis:** Diagramas de secuencia, Diagramas de colaboración
    - **Diseño:** Diagrama de clases, Diagrama de paquetes.
    - **Implementación:** Código fuente (aplicación)
    - **Pruebas:** Ejecución de las pruebas

- **Despliegue:** Manual de usuario, Manual de Instalación.

Finalmente en este capítulo se muestran las diferentes interfaces de usuario que hacen posible el entendimiento de la aplicación.

Capítulo VI

6 EXPERIMENTACIÓN Y EVALUACIÓN

En este capítulo, inicialmente se expone la herramienta de evaluación desarrollada en el trabajo de grado *"**Plataforma para la Evaluación de Sistemas de Recuperación de Servicios Basados en Comportamiento**"* que se llevó a cabo en la Universidad del Cauca, la cual es utilizada para valorar la calidad del proceso de emparejamiento, en el prototipo definido en el presente trabajo de grado. Además se enseña la metodología de evaluación usada y las pruebas ejecutadas al prototipo, para establecer la calidad de sus resultados y su rendimiento. Finalmente el análisis de los resultados, junto con las graficas asociadas y los trabajos futuros son presentados.

6.1 METODOLOGÍA DE EVALUACIÓN

Para la valoración del prototipo realizado se utilizo una herramienta de evaluación de técnicas de recuperación de procesos de negocio desarrollada en el trabajo de grado *"**Plataforma para la Evaluación de Sistemas de Recuperación de Servicios Basados en Comportamiento**",* que se llevo a cabo en la Universidad del Cauca. Antes de exponer y detallar las funcionalidades de esta plataforma, es conveniente aclarar los conceptos más relevantes entorno a la metodología empleada para la evaluación de la aplicación software, desarrollada en el presente trabajo de grado.

6.1.1 BENCHMARKING

Benchmarking se define como un proceso sistemático y continuo para evaluar los productos, servicios y procesos de trabajo de las organizaciones que son reconocidas como representantes de las mejores prácticas, con el propósito de realizar mejoras organizacionales [86]. En otras palabras un benchmarking es el proceso encargado de comparar benchmarks.

En el mismo sentido, un Benchmark puede definirse como un punto de referencia a partir del cual es posible evaluar comparativamente el rendimiento de un sistema respecto de otro [87]. En informática un benchmark, es entendido como el resultado de la ejecución de un programa o un conjunto de programas en una máquina, que brindan información sobre el rendimiento de un elemento concreto o la totalidad del sistema.

A continuación se explica, la adaptación de los conceptos expuestos anteriormente a la ***"Plataforma para la Evaluación de Sistemas de Recuperación de Servicios Basados en Comportamiento"***, los cuales son detallados en la siguiente sección.

6.2 DESCRIPCIÓN DEL BENCHMARK DE REFERENCIA

El benchmark de referencia utilizado, consta de un conjunto de resultados de similitud obtenidos a partir de las comparaciones manuales realizadas por diferentes usuarios. Los cuales evaluaron los flujos de control de los modelos de procesos de negocio publicados, además de las entradas, salidas, nombres y descripciones de las tareas pertenecientes a cada uno de los modelos de procesos de negocio. Esta evaluación fue realizada por un grupo de 8 evaluadores con amplio conocimiento del dominio de los procesos de negocio.

Para generar este benchmark fue necesaria la utilización de una herramienta que permite llevar a cabo la comparación manual entre Modelos BP a través de una interfaz gráfica de usuario, la cual soporta el acceso simultáneo de varios evaluadores y almacena los resultados de las comparaciones en una base de datos.
La "Plataforma para la Evaluación de Sistemas de Recuperación de Servicios Basados en Comportamiento" contiene un total de 700 comparaciones manuales entre parejas de procesos de negocio (7 procesos de consulta x 100 procesos publicados).

Los procesos publicados se encuentran descritos en bpmo. Para el presente trabajo de grado se transformaron dichos procesos a bpel4sws, con el fin de poder utilizar el benchmark de referencia para la evaluación de los resultados obtenidos por el prototipo desarrollado.

Para determinar el nivel de rendimiento del sistema se compararon los resultados registrados en el benchmark del prototipo con aquellos consignados en el benchmark de referencia, utilizando 5 procesos de consulta y 60 procesos publicados en el repositorio, además se tuvo en cuenta los resultados obtenidos por flujo de control de los modelos de procesos de negocio, entradas, salidas, nombre de las operaciones de cada una de las actividades pertenecientes a los procesos de negocio.

A partir de lo anterior se realizo un análisis estadístico basado en la aplicación de un conjunto de medidas empleadas en la evaluación del desempeño de sistemas de recuperación de información. Dichas medidas se describen en la siguiente sección.

6.3 MEDIDAS DE DESEMPEÑO

Como lo afirman en [85] el desempeño general de un sistema se establece utilizando las siguientes medidas:

- **Precisión (p):** Calcula la fiabilidad de los procesos relevantes recuperados por el algoritmo**.**

$$p = \frac{|I|}{|P|} \qquad \textit{Ec. 8.}$$

- **Recall (o):** Especifica el porcentaje de los procesos relevantes recuperados.

$$r = \frac{|I|}{|R|} \qquad \textit{Ec. 9.}$$

- **Overall (r):** Valora la calidad del emparejamiento, teniendo en cuenta el esfuerzo necesario para la eliminación de falsos positivos y los servicios no recuperados.

$$o = r * \left(2 - \frac{1}{p}\right) \qquad \textit{Ec. 10.}$$

- **Top-k precisión (Pk):** Evalúa la proporción de procesos relevantes respecto al número total de procesos clasificados en los primeros k niveles del ranking del prototipo.

$$p_k = \frac{|Retrel_k|}{k} \qquad \textit{Ec. 11.}$$

- **P-presición (Pp):** Determina el número de procesos relevantes cuya clasificación en cuanto a niveles coincide con la disposición del ranking base para un nivel k definido.

$$p_p = \frac{[Retrel_k]\, Rel-p}{k} \qquad \textit{Ec. 12.}$$

A diferencia de [85] el presente trabajo de grado utiliza la **Medida-F (f)** la cual unifica las medidas Precision y Recall en una única medida. La medida F es la media armónica ponderada de la Precision y el Recall [88].

$$f = \frac{2}{\frac{1}{r}+\frac{1}{p}} \qquad \textit{Ec.13.}$$

Para evaluar la calidad del algoritmo de recuperación, se comparan los Procesos (P) retornados por el Algoritmo con los Procesos (R) obtenidos en el Banco de Procesos. De esta forma se puede determinar un conjunto de verdaderos positivos (I), servicios correctamente identificados; igualmente se determina un conjunto de falsos positivos, Procesos falsos recuperados (F = P/I), y falsos negativos, es decir Procesos relevantes no recuperados (M = R/I) [14]. $Retrel_k$ es el conjunto de Procesos relevantes para un top k de Procesos recuperados, mientras Rel-p determina cuantos de los Procesos de $Retrel_k$ están en la misma posición del ranking de referencia del Banco de Procesos [89]. Con base en la cardinalidad de estos conjuntos se tiene:

A partir de las medidas establecidas anteriormente, se realiza un completo análisis estadístico que permite determinar la calidad de un algoritmo de emparejamiento de procesos. Este análisis se fundamenta en la evaluación de las siguientes relaciones: Precision vs. Recall, Overall, Medida-F, K-Precision vs. K, P-Precision vs. K, aplicadas reiteradamente en las cinco fases que constituye el prototipo:

1. Recuperación de procesos de negocio según su flujo de control.
2. Evaluación de los procesos de negocio recuperados, según el nombre de operación de cada actividad del proceso.
3. Evaluación de los procesos de negocio recuperados, según las entradas de cada actividad del proceso.
4. Evaluación de los procesos de negocio recuperados, según las salidas de cada actividad del proceso.
5. Evaluación total de los procesos recuperados.

Además de realizar el análisis mencionado anteriormente, se plasmó un nuevo razonamiento estadístico, donde se modificaron algunos elementos del benchmark de referencia con el fin de analizar con mayor detalle los resultados obtenidos por el prototipo. La modificación consiste en calcular las medidas: K-Precisión vs. K, y P-Precisión vs. K, para los procesos recuperados por el prototipo en la Fase 1 (Recuperación de procesos de negocio según su flujo de control). Lo anterior se debe a que el análisis sobre el universo completo del benchmark de referencia no representa una muestra significativa de la calidad de recuperación del algoritmo, ya que dicho benchmark no fue construido con un enfoque basado en fases progresivas tal como fue construido el prototipo (cinco fases). Después de la Fase 1 el espacio de búsqueda para las fases subsiguientes se reduce, esto quiere decir que la evaluación por el nombre de la

operación, y los identificadores de entradas y salidas son analizadas para los resultados obtenidos por la Fase 1 (selección basada en estructura), mientras que el benchmark analiza operaciones, entradas y salidas sobre el total de las comparaciones realizadas por los usuarios. Los dos estudios completos se encuentran consignados en el ANEXO D.

6.4 CRITERIOS DE EVALUACIÓN

Para garantizar el correcto funcionamiento del prototipo, se realizaron pruebas que permiten determinar la calidad de los resultados (eficacia) y el rendimiento la plataforma (eficiencia). Es importante mencionar que el rendimiento del sistema está estrictamente ligado con el hardware del equipo, ya que éste determina el desempeño del mismo.

Para los módulos más importantes del prototipo se definieron pruebas enfocadas a la evaluación de dos aspectos fundamentales: la calidad de los resultados y el rendimiento del prototipo. El primero, evalúa la eficacia de los algoritmos, que intervienen en el proceso de recuperación y emparejamiento del prototipo, y el segundo evalúa los tiempos de respuesta de las solicitudes realizadas al prototipo [85].

Finalmente la evaluación y clasificación de los resultados de las pruebas, se realizó a partir de un conjunto de reglas, que determinan los criterios de tiempos de respuesta para aplicaciones que están directamente relacionadas a las características cognitivas de los humanos [90]:

- Los usuarios no notan un retardo de menos de 0.1 segundos. Por tanto, un tiempo de respuesta que esté bajo el umbral de 0.1 segundos se clasifica como óptimo.

- Un tiempo de respuesta de menos de 1 segundo no interrumpe el flujo del pensamiento de un usuario, pero deja notar un retardo. Por tanto, un tiempo de respuesta que esté bajo el umbral de 1 segundo se clasifica como bueno.

- Los usuarios aún esperarán la respuesta si está por debajo del umbral de 10 segundos, pero el retardo es notorio. Por tanto, un tiempo de respuesta que esté bajo el umbral de 10 segundos se clasifica como aceptable.

- Después de 10 segundos, los usuarios pierden la concentración y no continúan esperando la respuesta del sistema. Por tanto, un tiempo de respuesta que esté por encima del umbral de 10 segundos se clasifica como deficiente.

En caso que el equipo no pueda proporcionar una respuesta inmediata o el tiempo de respuesta de la aplicación sea mayor a 10 segundos se debe utilizar obligatoriamente, las barras de progreso [91]. Las barras de progreso ofrecen ventajas como: tranquilizar al usuario manteniéndolo informado ya que el sistema no se ha caído, indicándole el tiempo aproximado que puede esperar, lo que permite al usuario realizar otras actividades durante las largas esperas.

6.5 PLAN DE PRUEBAS

Las pruebas de calidad y rendimiento, se hacen sobre los módulos que realizan las funcionalidades más importantes del prototipo, estas son: Transformación de los procesos BPEL4SWS almacenados en el Repositorio a grafos, Recuperación de procesos de negocio según su flujo de control, Evaluación de los procesos de negocio recuperados según su flujo de control mediante los criterios: nombre de operación, entradas, salidas, de las actividades que componen los procesos.

A continuación en la Tabla 6 se describen las pruebas de eficacia y eficiencia realizadas a los módulos que conforman el prototipo.

PLAN DE PRUEBAS

Transformación de los procesos BPEL4SWS almacenados en el Repositorio a grafos	**Prueba de Rendimiento PR1:** Determina el tiempo que tarda el analizador BPEL-Graph en transformar los procesos de negocio BPEL4SWS, encontrados en el repositorio de procesos, en Grafos.
Fase 1: Recuperación de procesos de negocio según su flujo de control	**Prueba de Calidad de los resultados PC1:** Permite evaluar la calidad de los resultados de similitud obtenidos por el sistema al recuperar los procesos de negocio más similares según el flujo de control del proceso buscado. **Prueba de Rendimiento PR2:** Determina el tiempo que tarda el modulo de recuperación, en encontrar los procesos más similares según su flujo de control, para un determinado proceso solicitado por el usuario.
Fase 2: Evaluación de los procesos de negocio, según el nombre de operación de sus actividades.	**Prueba de Calidad de los resultados PC2:** Determina la calidad de los resultados evaluados en esta fase, y por ende la eficacia de los algoritmos que intervienen en el proceso de emparejamiento. **Prueba de Rendimiento PR3:** Determina el tiempo que tarda esta fase, en evaluar los procesos recuperados en la fase 1 y ordenarlos según su similitud de nombre de operación de las actividades de los procesos.
Fase 3: Evaluación de los procesos de negocio, según las entradas de sus actividades.	**Prueba de Calidad de los resultados PC3:** Determina la calidad de los resultados evaluados en esta fase, y por ende la eficacia de los algoritmos que intervienen en el proceso de emparejamiento. **Prueba de Rendimiento PR4:** Determina el tiempo que tarda esta fase, en evaluar los procesos recuperados en la fase 1 y ordenarlos según su similitud de entradas de las actividades de los procesos.

Fase 4: Evaluación de los procesos de negocio, según las salidas de sus actividades.	**Prueba de Calidad de los resultados PC4:** Determina la calidad de los resultados evaluados en esta fase, y por ende la eficacia de los algoritmos que intervienen en el proceso de emparejamiento. **Prueba de Rendimiento PR5:** Determina el tiempo que tarda esta fase, en evaluar los procesos recuperados en la fase 1 y ordenarlos según su similitud de salidas de las actividades de los procesos.
Fase 5: Evaluación total de los procesos de negocio según las fases anteriores	**Prueba de Calidad de los resultados PC5:** Determina la calidad de los resultados evaluados en esta fase, y por ende la eficacia de los algoritmos que intervienen en el proceso de emparejamiento. **Prueba de Rendimiento PR6:** Determina el tiempo que tarda esta fase, en evaluar los procesos recuperados en la fase 1 y ordenarlos según las fases 2,3 y 4.

Tabla 6 Plan de Pruebas

6.6 ESPECIFICACIONES TÉCNICAS DEL SERVIDOR

Los resultados de las pruebas de rendimiento están condicionados por las prestaciones del equipo donde se ejecuta el software del prototipo (Servidor DELL Precision 3400). En este caso, el equipo empleado para desplegar el plan de pruebas descrito cuenta con las siguientes especificaciones técnicas (Tabla 7):

Procesador	Intel(R) Core (TM) 2 Duo CPU E4500 @ 2.20 Ghz 2.19 Ghz
Caché del procesador	16MB de caché de nivel 2
Memoria RAM	4096 MB (4 dimm)
Disco duro	250GB (5400 RPM)
Sistema Operativo	Windows 7 32 bits

Tabla 7 Especificaciones Técnicas del Equipo empleado para las Pruebas del Prototipo

6.7 RESULTADOS

A continuación se presentan las gráficas asociadas a los resultados de las pruebas de calidad (eficacia) y rendimiento (eficiencia) definidas en la sección 6.3, las tablas donde se consignan los datos de tiempos de respuesta, calidad de los resultados y el plan de pruebas completo con todas sus pruebas de rendimiento y calidad por cada modulo, se encuentran en el ANEXO D.

- **Transformación de los procesos BPEL4SWS almacenados en el Repositorio a grafos**

  Prueba de rendimiento PR1: En ésta prueba se desea determinar el rendimiento de la etapa de transformación de las descripciones BPEL4SWS publicadas en el repositorio. Para esto se cuenta con 60 archivos BPEL4SWS publicados en el Repositorio de Procesos, y 5 procesos de negocio de consulta (ANEXO C). A continuación se muestran los resultados de los 5 procesos de negocio de consulta promediados.

  La Figura 29 presenta los tiempos de respuesta de la ejecución del mecanismo de transformación de BPEL4SWS a grafos. Para lo cual se concluye que tiene un comportamiento creciente y directamente proporcional a la cantidad de procesos contenidos en el Repositorio de Procesos. Éste comportamiento es esperado y el tiempo de respuesta es bueno ya que el mecanismo de transformación de procesos BPEL4SWS a Grafos es una tarea muy compleja.

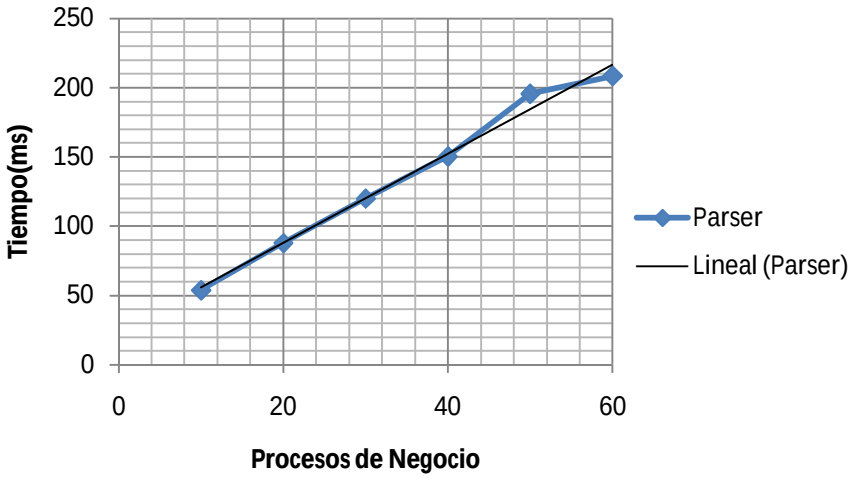


Figura 18 Rendimiento del mecanismo de transformación de los documentos BPEL4SWS del repositorio de procesos en grafos

- **Fase 1: Recuperación de procesos de negocio según su flujo de control**

  Prueba de Calidad de los resultados PC1: Los resultados obtenidos en la fase de recuperación de procesos de negocio según su flujo de control se puede apreciar claramente en la Figura 30, donde la precisión y la medida - f para los 5 procesos de consulta superan el 58,82%, y no sobrepasa el 82.35%, mientras que el valor mínimo que alcanza la medida recall es 50%. Estos valores indican que el funcionamiento del algoritmo tiene un buen desempeño y el porcentaje de procesos relevantes recuperados está por encima del 50%, sin embargo el desempeño del algoritmo mejoraría en gran medida si este detectara partes del grafo de consulta en los procesos publicados y viceversa. Los resultados arrojados por la medida overall no son los esperados ya que el máximo valor que toma uno de sus procesos de consulta es 55%, esto indica que la calidad de emparejamiento y recuperación de procesos no son muy buenos.

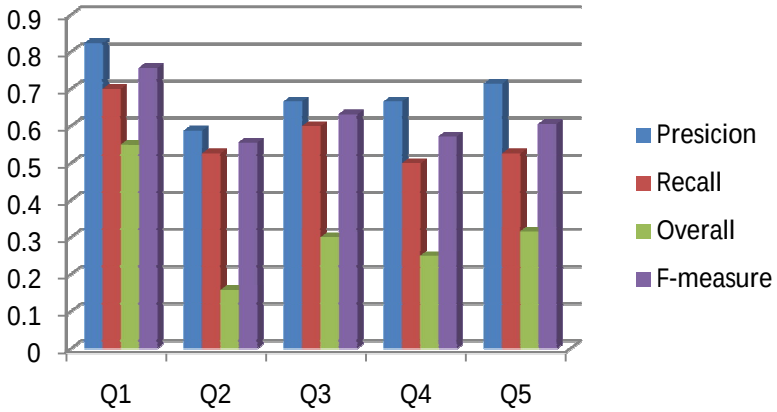


Figura 19 Precision, Recall, Overall, F-Measure - Flujo de control

La medida de precisión respecto al número de procesos de negocio recuperados por el algoritmo es presentada en la Figura 31. Para valores k = 1 y k = 2 el algoritmo es plenamente preciso, pero a medida que el k se incrementa, la precisión disminuye levemente, sin tomar valores menores a un 68%. De acuerdo a esto el mejor desempeño del algoritmo se obtiene entre los rangos k = 1, k = 4 y k = 6, k = 8. Tomando estos rangos como referencia el k óptimo del algoritmo se encuentra entre 1 y 4 procesos recuperados, valores en los que la precisión se ajusta a los parámetros requeridos para el mejor desempeño.

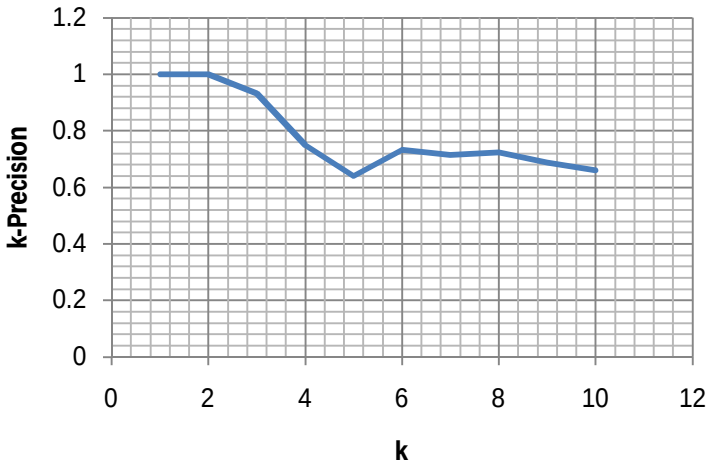


Figura 20 K-Precisión vs. K – Flujo de Control

La gráfica de P-Precisión acierta con una mayor eficacia la posición y los procesos recuperados de acuerdo al banco de pruebas generado. Para esta medida la curva de la Figura 32, decrece a medida que se incrementa el número de procesos k, sin embargo para k = 10 incrementa la precisión a 56%. Dados estos resultados el mejor desempeño del algoritmo se obtiene para los primeros 3 procesos recuperados, debido a esto el algoritmo no se considera preciso, de acuerdo a la posición de los procesos recuperados.

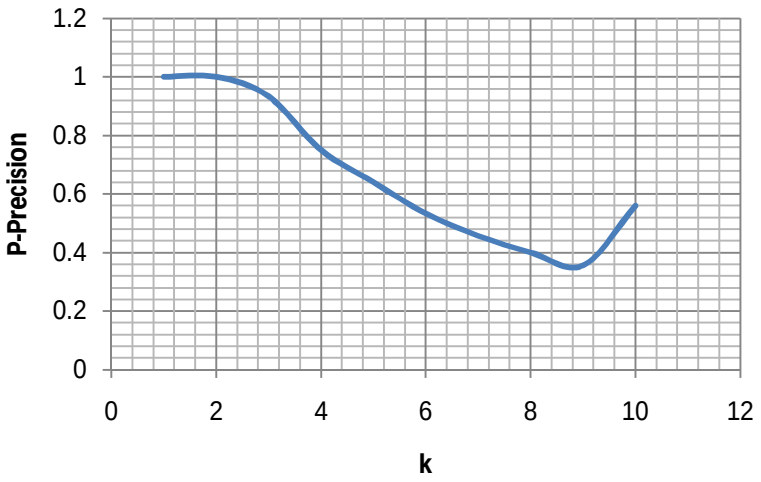


Figura 21 P-Precisión vs. K – Flujo de Control

Prueba de Rendimiento PR2: En esta parte, se presenta los resultados del estudio del tiempo de ejecución del algoritmo para la recuperación de procesos de negocio según su flujo de control. Para esta medida se cuenta nuevamente con 60 archivos BPEL4SWS publicados en el repositorio de Procesos, y 5 procesos de negocio de consulta. Estas pruebas se dividen en tres casos: El caso uno muestra los tiempos, para la detección de grafos y sub grafos encontrados en los procesos publicados (Plugin, Exact), según los procesos de consulta. El segundo caso expone los tiempos obtenidos para la detección de grafos y sub grafos encontrados en los procesos de consulta (Subsume) según los procesos publicados.

El primer caso se muestra en la Figura 33, donde se concluye que el rendimiento del sistema es proporcional al número de documentos BPEL4SWS recuperados en el repositorio de procesos específicamente, el tiempo de respuesta del algoritmo de recuperación de procesos, varía con relación a los servicios publicados en el repositorio. El transformador BPEL4SWS a grafos presenta éste mismo comportamiento, a medida que los procesos de entrada del sistema se van incrementando, el tiempo también lo hace.

Posteriormente se evaluó el rendimiento de acuerdo a [85], concluyendo que tiene el siguiente comportamiento:

- **Optimo** (tiempo de respuesta menor o igual a 0,1s): cuando el número de procesos comparados es menor o igual a 5.
- **Bueno** (tiempo de respuesta entre 0,1s y 1s): cuando el número de procesos comparados es mayor a 5 y menor o igual a 15.
- **Aceptable** (tiempo de respuesta entre 1s y 10s): cuando el número de procesos comparados es mayor a 15 y menor o igual a 32.
- **Deficiente** (tiempo de respuesta mayor a 10s): cuando el número de procesos comparados es mayor a 32.

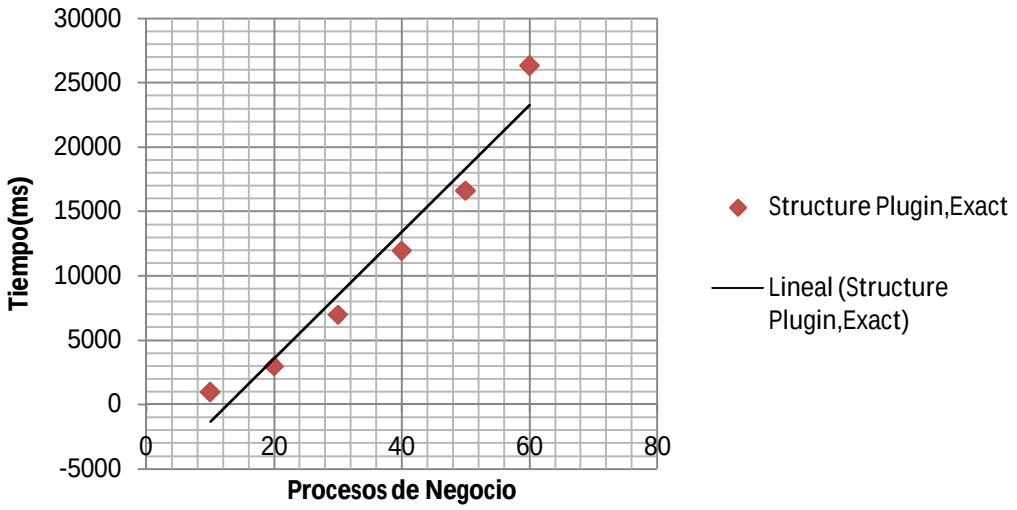


Figura 22 Rendimiento del mecanismo para la recuperación de procesos de negocio según su flujo de control – caso 1 (Plugin, Exact)

Por otra parte los resultados obtenidos para el segundo caso se muestran en la Figura 34. Estos resultados tiene el mismo comportamiento obtenido en el caso uno, por lo tanto el rendimiento del sistema es proporcional al número de documentos BPEL4SWS recuperados en el repositorio de procesos, y el tiempo de respuesta varía con relación a los servicios publicados en el repositorio.

En seguida se evaluó el rendimiento de acuerdo a [85], y se determino que tiene el siguiente comportamiento:

- **Optimo** (tiempo de respuesta menor o igual a 0,1s): cuando el número de procesos comparados es menor o igual a 6.
- **Bueno** (tiempo de respuesta entre 0,1s y 1s): cuando el número de procesos comparados es mayor a 6 y menor o igual a 16.
- **Aceptable** (tiempo de respuesta entre 1s y 10s): cuando el número de procesos comparados es mayor a 16 y menor o igual a 54.
- **Deficiente** (tiempo de respuesta mayor a 10s): cuando el número de procesos comparados es mayor a 54.

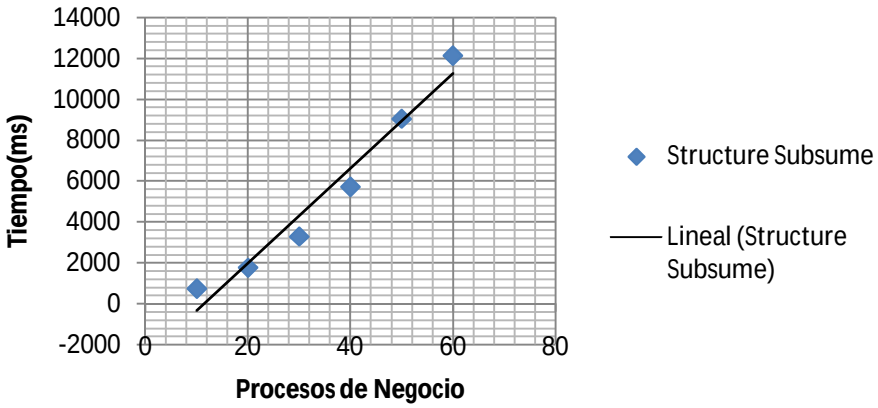


Figura 23 Rendimiento del mecanismo para la recuperación de procesos de negocio según su flujo de control – caso 2 (Subsume)

Adicionalmente se concluye que para el caso uno, los tiempos de ejecución son aceptables ya que los procesos de consulta van a ser emparejados con cada uno de los procesos publicados en el repositorio (60 emparejamientos por proceso de consulta).

Ahora, el caso dos es mucho más rápido que el caso uno debido a que antes de emparejar verifica que exista subsume, en caso contrario no realiza el emparejamiento.

- **Fase 2: Evaluación de los procesos de negocio, según el nombre de operación de sus actividades.**

  Prueba de Calidad de los resultados PC2: Los resultados obtenidos para la fase 2 se pueden observar claramente en la Figuras 35 y 36. La medida de precisión respecto al número de procesos de negocio recuperados por el algoritmo es presentada en la Figura 35. Para valores bajos de k (número de procesos recuperados) se tiene que el algoritmo es más preciso, además se puede observar que la precisión más baja es de 65.78%. De acuerdo a esto el mejor desempeño del algoritmo se obtiene en el rango k=1, k=4 y k=8, k=10. Tomando estos rangos como referencia el k óptimo del algoritmo se encuentra entre 1 y 4.

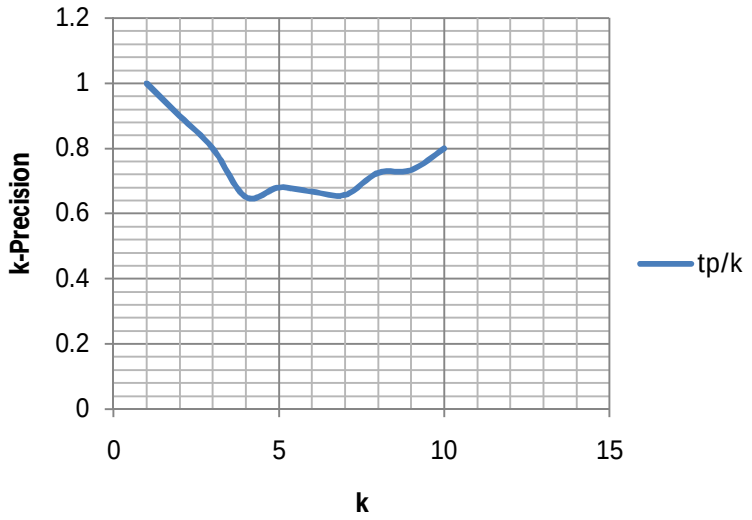


Figura 24 K-Precision vs. K –Nombre de operación de las actividades de los procesos

La medida de desempeño top-p precisión, es más exigente que la anterior en tanto que evalúa la precisión del sistema en la distribución de los procesos relevantes dentro de los niveles del ranking del prototipo. Los resultados de la aplicación de esta medida sobre el sistema implementado se ilustran en la Figura 36. La precisión es ligeramente inferior pero significativamente favorable, ya que alcanza un valor máximo del 100% en el primer y segundo nivel del ranking, el cual disminuye hasta el 73,33% en el nivel 3 y se mantiene superior al 45,71% en los niveles restantes. De esta manera, el desempeño del sistema respecto a la medida de top-k precisión es óptimo para valores de k inferiores a 3 y es en general aceptable para los demás niveles del ranking.

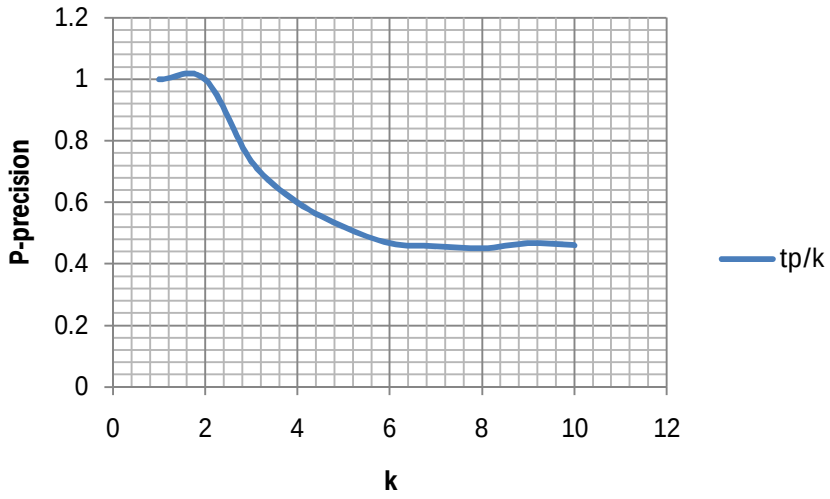


Figura 25 P-Precisión vs. K –Nombre de operación de las actividades de los procesos

- **Fase 3 y 4: Evaluación de los procesos de negocio, según las entradas y salidas de sus actividades.**

  Prueba de Calidad de los resultados PC3: La medida de precisión con respecto al número de procesos de negocio recuperados por el algoritmo es presentada en la Figura 37. Para valores bajos de k (número de procesos recuperados) se tiene que el algoritmo es más preciso (hasta k = 2), además se puede observar que la precisión más baja es de 52.17%. De acuerdo a esto el mejor desempeño del algoritmo se obtiene en el rango k=1, k=4. Tomando estos rangos como referencia el k óptimo del algoritmo se encuentra entre 1 y 4, considerando el desempeño del algoritmo como aceptable.

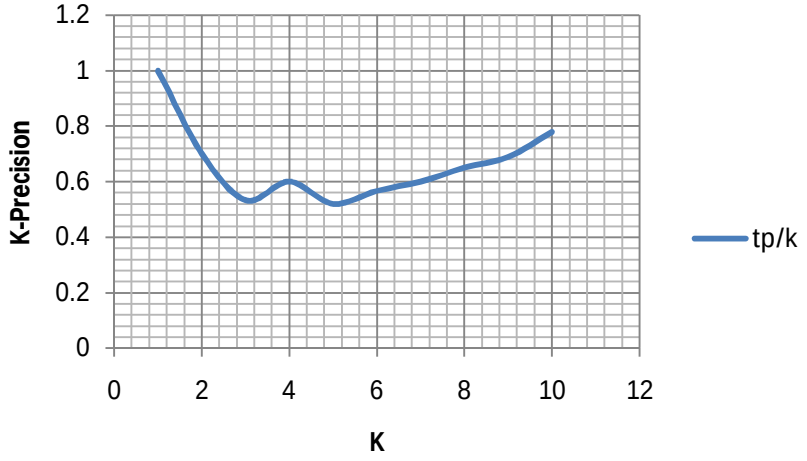


Figura 26 Precisión vs. Recall – Entradas de las actividades de los procesos

Prueba de Calidad de los resultados PC4: Los resultados de la aplicación de la medida de desempeño top-p precisión sobre el sistema implementado se ilustran en la Figura 38. La precisión es ligeramente inferior con tendencia a mantenerse constante,

puesto que, alcanza un valor máximo del 100% en el primer nivel del ranking, el cual disminuye hasta el 70% en el nivel 2 y oscila entre 33.33% y 46,71% en los niveles restantes. De esta manera, el desempeño del sistema respecto a la medida de top-k precisión es bueno para valores de k inferiores a 2 y es en general aceptable para los demás niveles del ranking.

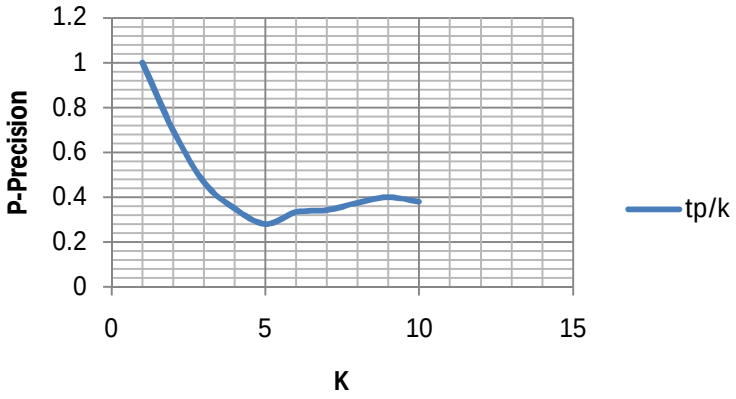


Figura 27 Precisión vs. Recall – Salidas de las actividades de los procesos

El comportamiento del sistema proyectado en las Figuras 37 y 38, se justifica en el carácter semántico del mecanismo de comparación. Es así como, en determinados casos el sistema puede llegar a estimar relaciones de similitud entre actividades del proceso, las cuales no son inmediatamente evidentes para una persona común e incluso para un evaluador experto, debido a que los términos que identifican los atributos de dichas actividades difieren significativamente desde el punto de vista sintáctico. De esta manera, el sistema puede clasificar en niveles primarios del ranking, procesos que semánticamente son muy semejantes, pero que sintácticamente no lo son.

- **Fase 5: Evaluación total de los procesos de negocio según las fases anteriores**

Prueba de Calidad de los resultados PC5: Los resultados obtenidos para la evaluación total de los procesos de negocio se puede apreciar claramente en la Figura 39, donde la precisión y la medida - f para los 5 procesos de consulta supera el 43,24%, mientras que el valor mínimo que alcanza la medida recall es 35%. Estos valores indican que los procesos evaluados para la fase total tiene un desempeño aceptable y el porcentaje de procesos relevantes recuperados está entre 35% y 60%. Los resultados arrojados por la medida overall no son los esperados ya que los procesos de consulta Q1 y Q4 arrojan valores negativos, esto indica que para estos procesos de consulta la calidad de emparejamiento y recuperación de procesos son deficientes. Además se puede concluir que el proceso de consulta que recupera mayor procesos relevantes es Q3, seguido de Q2 y Q4.

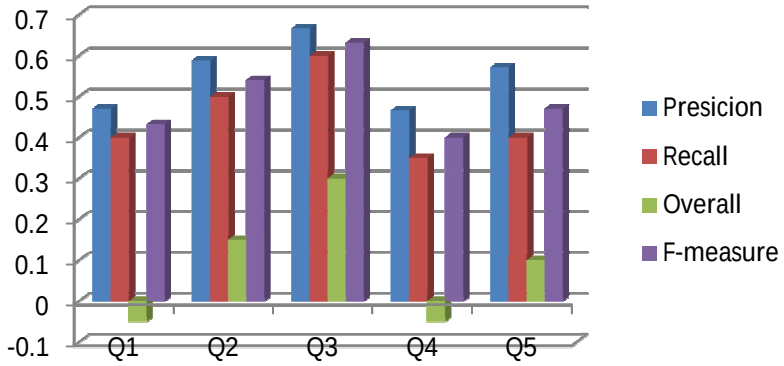


Figura 28 Precisión, Recall, Overall, F-Measure – Evaluación total

La Figura 40 ilustra la relación existente entre las medidas de precisión y recall. En esta gráfica se visualiza como el desempeño del sistema está en general ubicado en una zona intermedia (superior al 40%) y recall considerable (en promedio del 48% y 68%), lo cual implica que el mecanismo implementado, es adecuadamente selectivo, recupera un porcentaje considerable de los procesos relevantes disponibles en el banco de procesos publicados.

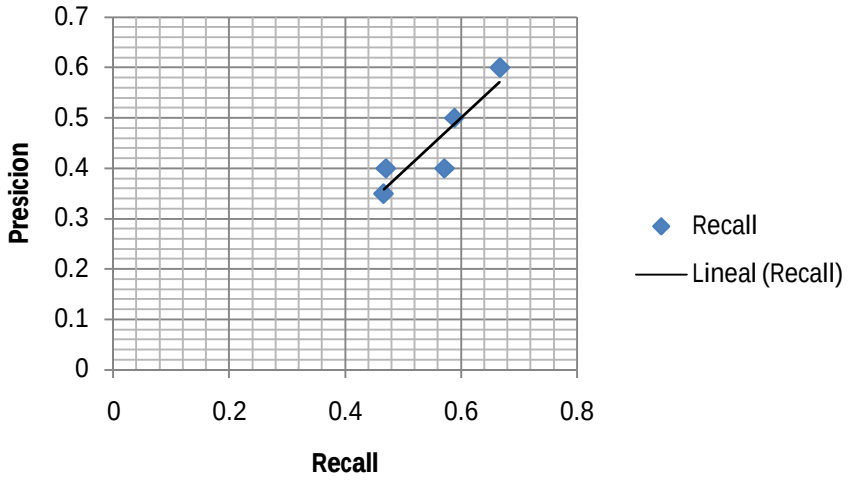


Figura 29 Precisión vs. Recall – Total

La medida de precisión respecto al número de procesos de negocio recuperados en la fase de evaluación total, es presentada en la Figura 41. Para valores bajos de k se tiene que el algoritmo es más preciso (hasta k=2), además se puede observar que la precisión más baja es de 40%. Desde k=6 hasta k=10, el desempeño del algoritmo se muestra creciente incrementando su precisión de un 40% a un 74%. De acuerdo a esto el k óptimo del algoritmo se encuentra en 1, esto indica que el proceso ubicado en la primera posición es el proceso de consulta. Según lo anterior el desempeño del algoritmo se considera aceptable.

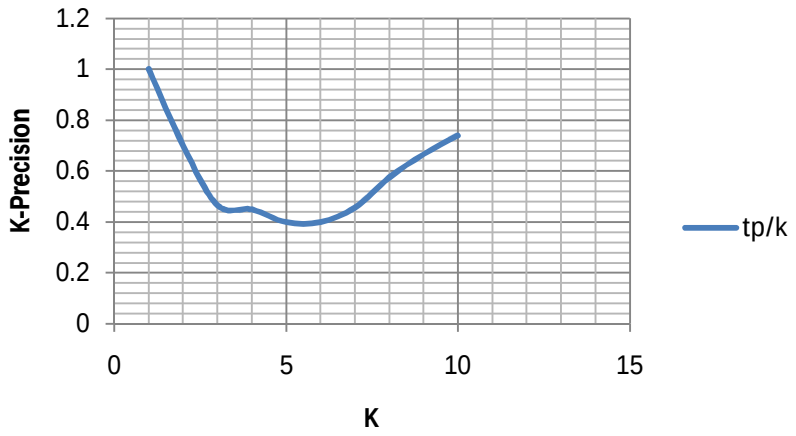


Figura 30 K-Precisión vs. K – Total

Los resultados de la aplicación de la medida de desempeño top-p precisión sobre el sistema implementado se ilustran en la Figura 42. La precisión alcanza un valor máximo del 100% en el primer nivel del ranking, el cual disminuye hasta 70% en el nivel 2 y decrece desde el nivel 2 hasta el nivel 8 en un 20%. Luego intenta mantenerse constante desde el nivel 8 hasta el nivel 10 con una precisión del 20%. De esta manera, el desempeño del sistema respecto a la medida de top-k precisión es óptimo para valores de k inferiores a 2 y es deficiente para los demás niveles del ranking.

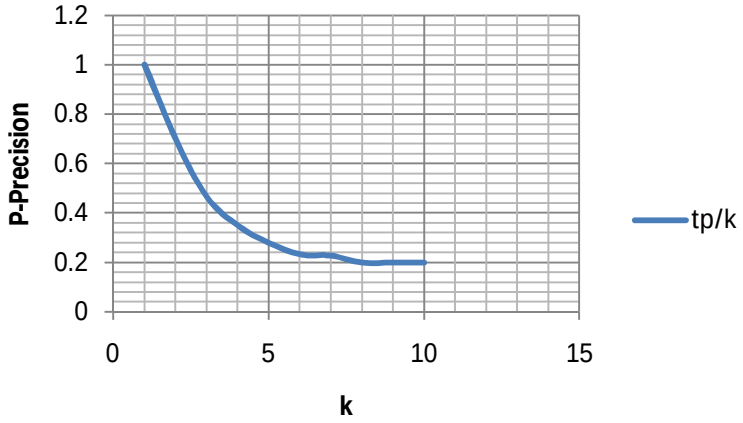


Figura 31 P-Precision vs. K – Total

6.8 CONCLUSIONES

- Según los resultados explicados anteriormente, se puede concluir que la recuperación de procesos de negocio según su flujo de control tiene un buen desempeño respecto al trabajo desarrollado en [3] ya que la precisión obtenida toma valores entre 82.35% y 66.66%, mientras que los resultados de precisión obtenidos en el trabajo antes mencionado oscilan entre un 52.94% y 89.36 %. Ahora bien los resultados obtenidos para la medida recall respecto a [3] no son tan buenos, ya que estos se encuentran entre 50% y 70%, mientras que para [3] está entre 53.45% y 100%. Los resultados antes mencionados fueron obtenidos para cuatro procesos de consulta. En la Tabla 8 se pueden observar claramente dichos resultados.

Procesos de consulta	Precisión Obtenida en el prototipo	Precisión Obtenida en [3]	Recall Obtenida en el prototipo	Recall Obtenida en [3]
Best path forecast	82.35%	89.36%	70%	57.53%
Activate Service	66.66%	59.26%	60%	100%
Launching a new product	66.66%	63.27%	50%	53.45%
Service provisioning process	71.42%	52.94%	52.63%	100%

Tabla 8 Comparación de las medidas Precisión, Recall según su flujo de control, para el prototipo desarrollado y para la plataforma desarrollada en [3]

Si se calcula el promedio para la medida precisión según la tabla 1, para las dos herramientas comparadas, el prototipo toma un valor del 71.77%, mientras que [3] toma un valor del 66.20%, esto indica que el prototipo desarrollado tiene una precisión alta respecto a la herramienta comparada. Por otra parte el prototipo arrojo un valor promedio para la medida recall del 58.15% y la herramienta desarrollada en [3] 77.74%, indicando que los procesos recuperados por [3] son más relevantes que los procesos recuperados por el prototipo desarrollado.

Sin embargo la recuperación de procesos de negocio según su flujo de control para el prototipo realizado en el presente trabajo de grado comparado con los trabajos realizados en [3, 14], tiene un excelente desempeño en tiempo de respuesta, teniendo en cuenta que permite emparejar un proceso de consulta contra 60 procesos publicados, mientras que los trabajos mencionados anteriormente solo permiten el emparejamiento uno a uno.

- Por otro lado como se explico en el capítulo IV el mecanismo para la evaluación de los procesos de negocio por entradas/salidas basado en semántica hace uso de una ontología, el cual obtiene mejores resultados en la medida precisión, para los procesos de consulta Q4 y Q5, respecto al mecanismo que evalúa los procesos por nombre de operación basado en la comparación lingüística, como se puede observar en la Figura 43. Ahora bien los mejores resultados obtenidos para la medida recall fueron encontrados en el mecanismo basado en semántica, específicamente para los procesos de consulta Q1, Q2, Q4, y Q5. A partir de estos resultados se puede concluir que el mecanismo basado en semántica es mucho más eficiente que el mecanismo basado en la comparación lingüística, sin embargo no se descarta la opción de utilizar el segundo mecanismo mencionado ya que para algunos procesos de consulta arroja resultados positivos.

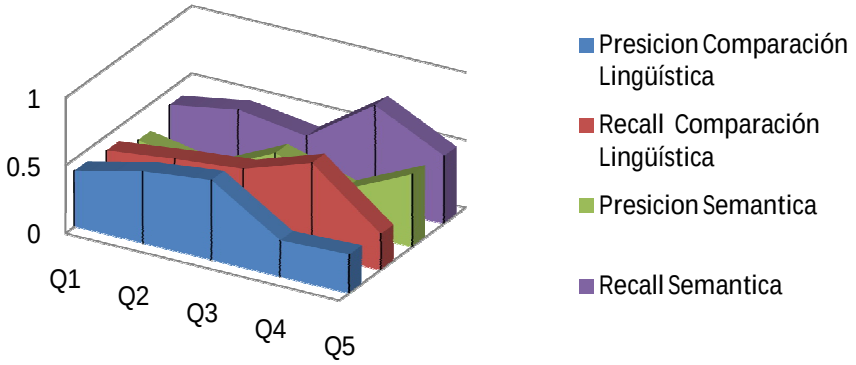


Figura 32 Precisión y Recall para los mecanismos basados en semántica y lingüística.

Finalmente en la Figura 44 se muestran los tiempos de ejecución obtenidos para el mecanismo basado en semántica y comparación lingüística, donde se puede observar claramente que es mucho más rápido el mecanismo basado en comparación lingüística, ya que los resultados al comparar 60 procesos publicados negocio no sobrepasa un tiempo de 1000 milisegundos, mientras que el mecanismo basado en semántica comparando los mismos 60 procesos publicados tarda un tiempo de 6000 milisegundos. Los tiempos obtenidos por ambos mecanismos no se consideran deficientes según [90], sin embargo la comparación lingüística sigue siendo mucho más rápida que un mecanismo basado en semántica, lo que implica que trabajar con una ontología conlleva a obtener un alto tiempo de ejecución.

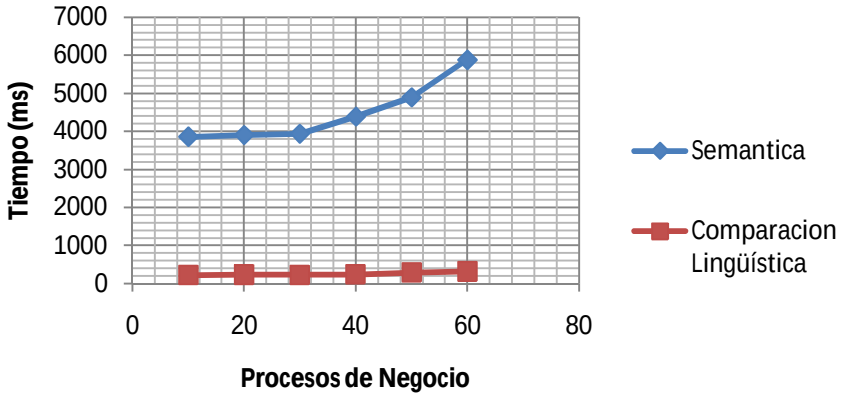


Figura 33 Tiempo de ejecución para los mecanismos basados en semántica y lingüística

6.9 RESUMEN

En este capítulo se describió detalladamente el benchmark de referencia utilizado: ***"Plataforma para la Evaluación de Sistemas de Recuperación de Servicios Basados en Comportamiento"***, y las modificaciones realizadas para adaptar el benchmark a los resultados arrojados por el prototipo. Seguido, se expuso las diferentes pruebas realizadas al sistema. Estas permitieron determinar la calidad (eficacia) y el rendimiento (eficiencia) de los módulos que componen el prototipo.

Para el proceso de transformación de los archivos BPEL a grafos y la recuperación de procesos de negocio según su flujo de control (fase 1), se obtuvo un comportamiento lineal, proporcional al número de servicios almacenados en el repositorio.

El algoritmo en la fase 1 presenta un buen desempeño ya que el porcentaje de los procesos relevantes recuperados está por encima del 50%. Para el algoritmo en la fase 2 la precisión del sistema en la distribución de los procesos relevantes dentro de los niveles del ranking del prototipo es considerada buena para los tres primeros niveles del ranking. La fase 3 y 4 presentan un comportamiento similar debido a esto la precisión para los primeros cuatro procesos recuperados tiende a decrecer y para el resto el sistema intenta mantenerse constante. Para la evaluación en la fase 5 presenta un desempeño aceptable, ya que los procesos relevantes recuperados están por encima del 46.66%. Finalmente se expuso unas conclusiones obtenidas a partir de los resultados explicados en las secciones anteriores.

Capítulo VII

7 CONCLUSIONES Y TRABAJOS FUTUROS

Este capítulo recopila las principales contribuciones del presente proyecto de grado, mostrado en la sección 7.1. Además expone las conclusiones extraídas de la realización del mismo en la sección 7.2. Finalmente en la sección 7.3 se proponen los trabajos futuros.

7.1 CONTRIBUCIONES

Entre las principales contribuciones de éste proyecto de grado se destacan las siguientes:

- El uso de un lenguaje para la ejecución de procesos de negocio semánticos BPEL4SWS, el cual es una evolución del estándar BPEL4WS, lenguaje que ha alcanzado un fuerte desarrollo en la industria TI, especialmente en grandes compañías, como Oracle y Sun, además de Novell, Adobe y SAP, entre otras.

- La implementación de actividades BPEL4SWS, en el transformador de archivos BPEL a grafos, permite lograr la equivalencia entre una descripción BPEL4SWS y una representación formal de Grafos para procesos de negocio.

- La reutilización del algoritmo Netmatch en entornos de desarrollo orientados a servicios, ya que el área de desempeño de este algoritmo es la bioinformática, basado en el emparejamiento de redes biológicas.

- Permitir al algoritmo Netmatch, detectar grafos publicados que estén contenidos en el grafo de consulta (subsume).

- La recuperación basada en el modulo de comparación lingüística, el cual ayuda al descubrimiento de procesos basado en la búsqueda de palabras en un diccionario de abreviaciones y sinónimos.

- El algoritmo adaptado al prototipo para la recuperación basado en semántica, el cual mejora el descubrimiento de procesos al inferir sobre los conceptos de un dominio, esto gracias a la utilización de una ontología (SSID), la cual enriquece las búsquedas recuperando un conjunto de procesos útil para el usuario.

7.2 CONCLUSIONES

- Dentro del presente proyecto de grado, se tuvo en cuenta las necesidades que presenta el descubrimiento de procesos basado en la semántica y las falencias existentes en las técnicas propuestas para éste dominio. Encontrando que la principal carencia son los protocolos semánticos, poco usados en la actualidad por las grandes casas productoras de software.

  Al concluir el trabajo desarrollado en el presente proyecto de grado, se propuso un mecanismo que recupera procesos de negocio de forma sintáctica y semántica, para procesos BPEL4SWS el cual es una evolución del estándar BPEL4WS, lenguaje que ha alcanzado un fuerte desarrollo en la industria TI, especialmente en

grandes compañías, como Oracle y Sun, además de Novell, Adobe y SAP, entre otras. El prototipo desarrollado consta de 5 fases. En la primera fase se recuperan procesos de negocio según el flujo de control del proceso de consulta, además si el usuario desea, el algoritmo realiza un filtro que recupera procesos que además de tener la misma estructura, contenga los mismos tipos de actividades que el proceso de consulta. En la segunda fase se evalúan los procesos recuperados en la fase 1 por el atributo operación de las actividades básicas que componen cada proceso. La tercera y cuarta fase nuevamente evalúa los procesos recuperados en la fase 1 por los atributos entrada y salida respectivamente. Finalmente la última fase realiza una evaluación total de los procesos analizados en las anteriores etapas.

Teniendo en cuenta lo anterior, se concluye que el trabajo realizado es un avance en la problemática del descubrimiento de procesos de negocio, asociado a la calidad y relevancia de los procesos retornados.

- Una etapa fundamental en éste trabajo es el recuperación de procesos. Con relación a esta temática, en el estado del arte se encontró las siguientes falencias :

    - La recuperación de procesos de negocio basados en sintaxis, no garantiza el correcto descubrimiento, ya que omite componentes que son similares a los buscados por el cliente, u ofrecen al usuario resultados con baja medida de precisión.
    - Los algoritmos de recuperación basados en semántica mejoran el descubrimiento de servicios al inferir sobre los conceptos de un dominio. Sin embargo los protocolos presentados en el estado del arte son poco usados en la actualidad por las grandes casas productoras de software.

  Teniendo en cuenta las anteriores consideraciones el prototipo cumplió a cabalidad con las deficiencias presentadas.

- El procedimiento para transformar 60 procesos de negocio BPMO a descripciones BPEL4SWS, se realizo de forma obligatoria debido a que BPEL4SWS carece de procesos de negocio publicados en la web y el proyecto que desarrolló este lenguaje [11] restringe el acceso a repositorios de gran relevancia, ya que son considerados confidenciales. Para poder construir un proceso BPEL4SWS, se siguió el único ejemplo que se encuentra disponible en [39].

- Desarrollar el prototipo por fases obtuvo resultados positivos ya que reduce el tiempo de ejecución del mecanismo empleado para la recuperación de procesos de negocio, además ofrece diferentes resultados al usuario dependiendo de la fase que este analizando.

- El benchmark de referencia utilizado no fue la mejor opción para comparar los logros obtenidos por el prototipo, ya que el análisis por fases que realiza el sistema propuesto necesita que el benchmark esté construido de la misma manera. Mientras que los resultados de la fase 1 del prototipo recupera procesos de negocio basado en la estructura, y en las fases siguientes se evalúan únicamente los procesos obtenidos en la fase 1; los resultados suministrados por el benchmark de referencia se obtuvieron evaluando al mismo tiempo y no por fases procesos

tanto por estructura, nombre, entradas y salidas de las actividades. Por este motivo fue necesario adaptar el benchmark de referencia, fijando como espacio de muestreo para las fases 2,3 y 4 del algoritmo el número de procesos de negocio recuperados por la fase 1. Una vez fijadas estas reglas, se calcularon las medidas: K-Precision vs. K, y P-Precision vs. K, ya que las otras métricas no pueden ser utilizadas debido a que los falsos positivos siempre son cero. A pesar de esto se realizo el cálculo de las métricas: Precision vs. Recall, Overall, Medida-F, K-Precision vs. K, P-Precision vs. K, aplicadas al benchmark de referencia original.

- Los resultados obtenidos en las pruebas de calidad para las fases de evaluación basadas en entradas y salidas, no fueron los esperados, debido a que las actividades básicas en BPEL4SWS tienen un solo atributo input (entrada) y output (salida), mientras que los procesos modelados en BPMO pueden tener una o más entradas y salidas, debido a esto el mecanismo manual de transformar los procesos BPMO a BPEL4SWS solo tiene en cuenta una entrada y una salida elegidas aleatoriamente para cada una de las actividades básicas en BPEL4SWS. Esto reduce la precisión de los resultados obtenidos por el prototipo al ser comparados con el benchmark de referencia, el cual trabajó con procesos BPMO.

- Los resultados entregados por el prototipo en la fase 1 permiten concluir, a través de medidas como precisión, Recall, Overall, Medida-F, k-precision y P-Precision, que el algoritmo de emparejamiento de servicios utilizado, asegura un cálculo de similitud de gran calidad, apto para un usuario que necesite encontrar un proceso por su estructura y los tipos de actividades que lo componen.

- El prototipo realizado en el presente trabajo de grado comparado con los trabajos realizados en [3, 14], tiene un excelente desempeño en tiempo de respuesta, al emparejar un proceso de consulta contra 60 procesos publicados, mientras que los trabajos mencionados anteriormente solo permiten el emparejamiento uno a uno y no 1:N, además los tiempos de ejecución de [3, 14] son considerablemente altos.

7.3 TRABAJOS FUTUROS

Esta tesis ha aportado soluciones al problema del descubrimiento de procesos, adaptando sus algoritmos a las nuevas necesidades y restricciones impuestas en los entornos de desarrollo actuales. Así, con relación al campo de estudio de éste proyecto de grado se propone los siguientes trabajos futuros:

Mejorar la recuperación de procesos de negocio por su flujo de control.

Para mejorar la recuperación de procesos de negocio por su estructura, surge como trabajo futuro añadir un mecanismo de búsqueda semántico de procesos de negocio basado en patrones de flujo de control como lo desarrollan en [92]. La adición del factor semántico en las búsquedas, permite inferir sobre las instancias de conceptos pertenecientes a una ontología y calcular la distancia semántica entre procesos asociando patrones de flujo de control.

Trabajar con procesos de negocio ejecutables.

El prototipo desarrollado en el presente trabajo de grado detecta equivalencias entre procesos abstractos, a partir de esto, como trabajo futuro se plantea adicionar

equivalencias entre procesos de negocio ejecutables, que tengan en cuenta aspectos como la WSDL de los servicios Web, partnerlinks en el caso de BPEL, y conversaciones, para BPEL4SWS.

Utilizar diferentes ontologías por niveles.

En las fases de evaluación de procesos de negocio recuperados según las entradas y salidas de las actividades, como mecanismo principal se aplico la distancia semántica, sobre la ontología SSID, la cual tiene conceptos distribuidos en una jerarquía de 10 niveles. Con base en el mecanismo mencionado anteriormente surge como trabajo futuro adaptar el mecanismo mencionado anteriormente para que funcione sobre cualquier ontología del dominio que esté compuesta por niveles.

Desarrollar una plataforma que permita evaluar satisfactoriamente los resultados del prototipo.

Como se menciono en las conclusiones de la sección anterior, el benchmark de referencia utilizado no cumple satisfactoriamente los requisitos para evaluar completamente el prototipo. Para solucionar este problema nace como trabajo futuro desarrollar una plataforma para la evaluación de procesos de negocio BPEL4SWS.

Experimentación del prototipo desarrollado en un ambiente real
Por último se propone realizar la experimentación del prototipo desarrollado en un ambiente real, para medir el grado de satisfacción de los usuarios, y especialmente determinar su desempeño dentro en un entorno de desarrollo orientados a servicios.

8 REFERENCIAS

1. Corporation, S.S., et al. *A NEW SERVICE-ORIENTED ARCHITECTURE (SOA) MATURITY MODEL*. 2005 Marzo 1 de 2010]; Available from: www.omg.org/soa/Uploaded%20Docs/SOA/SOA_Maturity.pdf.
2. Mongiello, M. and D. Castelluccia, *Modelling and Verification of BPEL Business processes*. 2006 IEEE Computer Society: Bari Italia.
3. Figueroa, C., *Descubrimiento automático de procesos de negocio basado en semántica del comportamiento*. 2009, Universidad del cauca: Popayán.
4. Tortosa, S.O., *PROPUESTA DE UNA ARQUITECTURA SOFTWARE BASADA EN SERVICIOS PARA LA IMPLEMENTACIÓN DE REPOSITORIOS DE OBJETOS DE APRENDIZAJE DISTRIBUIDOS*, in *Ciencias de la computación*. 2006, Universidad del Alcalá: Alcalá de Henares. p. 313.
5. Martin, D., et al. *OWL-S: Semantic Markup for Web Services*. 2004 Febrero 12 de 2010]; Available from: http://www.w3.org/Submission/OWL-S/.
6. Roman, D., H. Lausen, and U. Keller, *Web Service Modeling Ontology (WSMO).* 2005.
7. Delgado, I.N., A. Kerzazi, and J.F.A. Montes, *Un Editor de Modelos OWL-S: OWL-S Modeller*. 2009, Universidad de Málaga.
8. Dimitrov, M., et al. *WSMO Studio*. 2009; Available from: http://www.wsmostudio.org/.
9. Gacitua-Decar, V. and C. Pahl, *Towards Reuse of Business Processes Patterns to Design Services*, in *Emerging Web Services Technology Volume III*. 2008, Birkhäuser Basel. p. 15-36.
10. Eshuis, R. and P. Grefen, *Structural Matching of BPEL Processes*. 2007, IEEE Computer Society. p. 171-180.
11. Pfannendörfer, M., et al., *Semantics Utilised for Process Management within and between Enterprises.* 2009.
12. Zaremba, M., et al. *Web Service Execution Enviroment*. 2008; Available from: http://www.wsmx.org/.
13. Gater, A., D. Grigori, and M. Bouzeghoub, *Matching and similarity evaluation of OWL-S process models*. 2008, Universite de Versailles Saint-Quentin.
14. Corrales, J.C., *Behavioral matchmaking for service retrieval*. 2008, University of Versailles Saint-Quentin-en-Yvelines.
15. W3C. *Guía Breve de Servicios Web*. 2010 [cited 2011 12-01-2001]; Available from: http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb.
16. Duarte, G.G., *Desarrollo de plano de gestión para una red mpls*. 2005, Universitat Politècnica de Catalunya: Catalunya.
17. Aguilar, R.S., *Business process modelling: Review and framework*, in *Department of Production Economics, Linköping Institute of Technology, SE 581 83, Linköping, Sweden*. 2003.
18. Ordoñez, L. and A. Bastidas, *Comparación semantica de tareas entre dos procesos de negocio de telecomuncaciones.* 2010.
19. Corporation, M. *Arquitectura Orientada a Servicios (SOA) de Microsoft Aplicada al Mundo Real*. 2008 5-01-2011]; Available from: http://download.microsoft.com/download/c/2/c/c2ce8a3a-b4df-4a12-ba18-7e050aef3364/070717-Real_World_SOA.pdf.
20. Papazoglou, M.P., *Service-oriented computing* ACM Press, 2003.
21. keller, U., et al., *Automatic Location of Services*. 2008: Digital Enterprise Research Institute.

22. Mercadillo, A.L.C., *Bailando en la web: Coreografía y orquestación de los servicios web.* 2010.
23. Jordan, D., et al., *Web Services Business Process Execution Language Version 2.0*. 2007, OASIS Standard.
24. Corrales, J.C., D. Grigori, and M. Bouzeghoub, *BPEL Processes Matchmaking for Service Discovery*. 2005, Universite de Versailles Saint-Quentin.
25. Soto, A.R.d. and E.C. Fernández, *Nuevas Tendencias en Sistemas de Información: Procesos y Servicios 1.* 2006.
26. Botaro, A.E., F.P. Lozano, and I.M. Bulo, *Operadores de mutación para WS-BPEL 2.0.* 2008.
27. White, S.A., *Introduction to BPMN.* 2004.
28. Owen, M. and J. Raj, *BPMN and Business Process Management Introduction to the New Business Process Modeling Standard.* 2004.
29. Agi, B., *BPMN Business Process Modeling Notation.* 2006.
30. Fensel, D., *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*. 2nd edition ed. 2003, Berlin: Springer-Verlag.
31. Bruijn, J.d., et al. *Web Service Modeling Language (WSML)*. 2005; Available from: http://www.w3.org/Submission/WSML/.
32. Bruijn, J.d., et al., *The Web Service Modeling Language WSML: An Overview.* 2007.
33. Kifer, M., G. Lausen, and J. Wu, *Logical foundations of object-oriented and frame-based languages*. 1995, ACM. p. 741 - 843.
34. Yan, Z., et al., *D1.1 BPMO Requirements Analysis and Design.* 2008.
35. Cimpian, E., Z. Yan, and T.i. Holmes, *D1.2. Business Process Modeling Ontology (BPMO) version 1.* 2008.
36. Cimpian, E., et al., *D1.3. Business Process Modeling Ontology BPMO final version.* 2008.
37. Martin, D. *OWL-S: Semantic Markup for Web Services*. 2003; Available from: http://www.ai.sri.com/daml/services/owl-s/1.0/owl-s.html.
38. Lessen, T.v., J.o. Nitzsche, and M. Dimitrov, *An Execution Engine for Semantic Business Processes.* 2007.
39. Nitzsche, J., et al., *BPEL for Semantic Web Services (BPEL4SWS).* 2008.
40. Pfannendörfer, M., et al., *Semantics Utilised for Process Management within and between Enterprises.* 2008.
41. Karastoyanova, D., et al., *WS-BPEL Extension for Semantic Web Services (BPEL4SWS), Version 1.0*. 2008.
42. TC, O.S.E.E. *Reference Ontology for Semantic Service Oriented Architectures (RO4SSOA)*. 2008; Available from: http://www.oasis-open.org/committees/document.php?document_id=25381&wg_abbrev=semantic-ex.
43. Carenini, A., J. Nitzsche, and T.v. Lessen, *sBPEL to BPEL4SWS Lifting and Lowering.* 2008.
44. Farrell, J. and H. Lausen. *Semantic Annotations for WSDL and XML Schema*. 2006; Available from: http://www.w3.org/TR/2007/REC-sawsdl-20070828/.
45. Belecheanu, R., et al., *Business Process Ontology Framework*, in *SUPER Project Deliverable D1.1*. 2006.
46. Nitzsche, J., D. Wutke, and T.v. Lessen, *An Ontology for Executable Business Processes*, in *Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007), in conjunction with ESWC 2007*. 2007.
47. Corrales, J.C., D. Grigori, and M. Bouzeghoub, *Behavioral matchmaking for service retrieval.* 2005.

48. Dijkman, R., M. Dumas, and L.G. Bañuelos, *Graph Matching Algorithms for Business Process Model Similarity Search.* 2010.
49. Messmer, B., *Efficient Graph Matching Algorithms for Preprocessed Model Graphs*. 1996, University of Bern: Switzerland.
50. Klusch, M. and F. Kaufer, *WSMO-MX: A Hybrid Semantic Web Service Matchmaker* 2009, IOS Press Amsterdam, The Netherlands.
51. Diestel, R., *Graph Theory*, ed. S.-. Verlag. 2000, New york, USA.
52. Robin. *Artificial Intelligence Articles*. 2009; Available from: http://intelligence.worldofcomputing.net/ai-search/breadth-first-search.html.
53. Nishihara, T. and Y. Minamide, *Depth-First Search.* 2010.
54. Han, W.-S., J. Lee, and M.D. Pham, *i Graph: A Framework for Comparisons of Disk Based Graph Indexing Techniques.* 2008.
55. Zampelli, S.e., *A Constraint Programming Approach to Subgraph Isomorphism.* 2008.
56. Foggia, P., C.Sansone, and M. Vento, *A Performance Comparison of Five Algorithms for Graph Isomorphism.* 2002.
57. Ullmann, J.R., *An Algorithm for Subgraph Isomorphism.* 1976.
58. Schmidt, D.C. and L.E. Druffel, *Fast Backtracking Algorithm to Test Directed Graphs for Isomorphism Using Distance Matrices.* 1976.
59. Cordella, L.P., P. Foggia, and C. Sansone, *Evaluating Performance of the VF Graph Matching Algorithm.* 1999.
60. McKay, B.D., *Practical Graph Isomorphism.* 1981.
61. P. Foggia, C. Sansone, and M.Vento, *A Database of Graphs for Isomorphism and Sub-Graph Isomorphism Benchmarking.* 2001.
62. Cordella, L.P., P.F.C. Sansone, and M. Vento, *A (Sub)Graph Isomorphism Algorithm for Matching Large Graphs.* 2004.
63. Nilsson, N.J., *Principles of Artificial Intelligence.* 1982.
64. Cordella, L.P., et al., *An Improved Algorithm for Matching Large Graphs.* 2002.
65. Ferro, A., et al., *NetMatch: a Cytoscape plugin for searching biological networks.* 2007.
66. Ferro, A., et al. *NetMatch is a Cytoscape plugin to query networks for patterns*. 2007; Available from: http://baderlab.org/Software/NetMatch.
67. Ferro, A., et al., *Graphblast: Multi-Feature Graphs Database Searching, in Workshop On Network Tools And Applications In Biology* 2007.
68. Martinez, J. and N. Pérez, *YATOSP: Marco de referencia semántico para el sector Telco.* 2008.
69. Norton, B., *Semantic BPEL: An Overview for Conceptual Models for Services Working Group.* 2008.
70. Andrews, T., et al., *Business process execution language for web services, version 1.1.* 2003.
71. Mendling, J. and J. Ziemann, *Transformation of bpel processes to epcs* 2005, Hamburg. Germany
72. Paolucci, M., et al., *Semantic Matching of Web Service Capabilities.* 2002.
73. Angell, R., G. Freund, and P. Willett, *Automatic spelling correction using a trigram similarity measure* 1983.
74. Miller, G., *Wordnet: A lexical database for english* 1995.
75. Rubio, A., *Estadística descriptiva.*
76. Fernández, P. and P. Díaz, *Estadística descriptiva de los datos.* 2001.
77. Al-Mubaid, H. and H. Nguyen, *A cluster-based Approach for Semantic Similarity in the Biomedical Domain.* IEEE Eng Med Biol Soc, 2006.
78. Grimm, S., et al., *A Reasoning Framework for Rule-Based WSML*. 2006: Austria.

79. Bishop, B. and F. Fischer, *IRIS - Integrated Rule Inference System*. 2008: Austria.
80. Paz, K., *MEDIA ARITMÉTICA SIMPLE.* Facultad de Ingeniería - Universidad Rafael Landívar, 2006.
81. Prado, J.R.M. and J.c.r. Chacon, *APLICACIÓN DE LA METODOLOGÍA RUP PARA EL DESARROLLO RÁPIDO DE APLICACIONES BASADO EN EL ESTÁNDAR J2EE.* Universidad de San Carlos de Guatemala, 2006.
82. Bucchiarone, *A Survey on Services Composition Languages and Models.* 2004.
83. Maldonado, D., *Metodología de Desarrollo UML.* 2008.
84. Ambler, S.W., *UML Package Diagrams* 2005.
85. Suarez, L. and L. Rojas, *Descubrimiento de servicios en ambientes ubicuos.* 2010.
86. Spendolini, M., *Benchmarking* ed. G.E. Norma. 1994.
87. Chillarege, R., *Software Testing Best Practices.* Center for Software Engineering IBM Research, 1999.
88. Manning, C.D., P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, ed. C.U. Press. 2008.
89. Zhang, M. and N. Hurley, *Novel Item Recommendation by User Profile Partitioning* in *International Joint Conference on Web Intelligence and Intelligent Agent Technology* 2009: Milan, Italy
90. Joines, S., R. Willenborg, and K. Hygh, *Performance Analysis for Java Websites* ed. Addison-Wesley. 2002.
91. Myers, B.A., *The importance of percent-done progess indicators for computer-human interfaces*, in *CHI Proceedings of the SIGHCI conference on Human factors in computing systems*. 1985: New York, USA.
92. Burbano, D.F.R. and D.S.C. Castro, *Busqueda semantica en un repositorio de procesos de negocio*, in *Departamento de Telematica*. 2010, Universidad del Cauca: Popayan.