Desarrollo de una herramienta web para la modularización de Ontologías

Integrantes
Juan Sebastian Baquero Grajales
Cód. 200766888
Néstor Daniel Herrera Betancourth
Cód. 201056251

Universidad del Valle Sede Tuluá
Programa de Ingeniería de Sistemas
Mayo de 2013

Desarrollo de una herramienta web para la modularización de Ontologías

Integrantes
Juan Sebastian Baquero Grajales
Cód. 200766888
Néstor Daniel Herrera Betancourth
Cód. 201056251

Trabajo de grado para optar el título de Ingeniero de Sistemas

Director
Oscar Orlando Ceballos Argote, M.Sc.

Universidad del Valle Sede Tuluá
Programa de Ingeniería de Sistemas
Mayo de 2013

Nota de aceptación

_____

_____

_____

_____

_____

Presidente del Jurado

_____

Jurado 1

_____

Jurado 2

Tuluá, Mayo del 2013

Resumen

En la actualidad, las ontologías se consideran un elemento importante en la web semántica, lo que hace importante su estudio en las ciencias de la computación. En las últimas dos décadas se han desarrollado ontologías en diferentes campos como medicina, biología, comercio electrónico, genética entre otras. Sin embargo, una dificultad que se presenta con las ontologías, especialmente en áreas como la medicina o la biología, es su gran tamaño en cuanto al número de conceptos y relaciones, por ejemplo, algunas ontologías puede contener hasta 16500 conceptos, lo que dificulta el uso y el procesamiento. Para solucionar este problema, en la literatura, se propone la modularización de ontologías, la cual se basa en dividir una ontología en subpartes.

En la modularización de ontologías existen dos enfoques: particionamiento de ontologías (Ontology Partitioning Approaches) y extracción de módulo (Module Extraction Approaches). Por cada enfoque de modularización existen diferentes técnicas tales como Swoop, PATO que se basan en el enfoque de particionamiento y Prompt y KMi en el enfoque de extracción de módulo. Estas técnicas se implementan en herramientas, que en la gran mayoría, llevan el nombre de la técnica implementada, por ejemplo Swoop, Pato, Prompt y KMi. Estas herramientas son desarrolladas en un entorno de escritorio e implementan una técnica, por consiguiente, un solo enfoque de modularización. Si en algún momento se desea modularizar una ontología con una técnica o un enfoque diferente a la implementada, no hay manera de realizarlo, la opción es elegir otra que cumpla con los nuevos requerimientos.

En este trabajo se soluciona este problema implementando una herramienta web que modulariza una ontología con el enfoque y la técnica que el usuario elija, retorna el resultado de la ontología modularizada en OWL contribuyendo al objetivo de la W3C de estandarizar el formato de las ontologías y además ofrece un informe de la técnica de modularización con la ontología del usuario para que este pueda modularizar con las técnicas que desee y según el informe que la herramienta le retorne el usuario elija la modularización que más se acople con su necesidad.

Palabras clave: modularización de ontologías, enfoques de modularización, particionamiento de ontología, extracción de módulo, técnicas de modularización de ontologías, PATO, Swoop, Prompt, KMi, OWL.

Abstract

Currently, ontologies are considered an important element in the semantic web, which makes its study important in computer science. In the last two decades have developed ontologies in different fields as medicine, biology, electronic commerce, genetics among others. However, a difficulty that arises with ontologies, especially in areas such as medicine or biology, is that its size in the number of concepts and relationships, for example, certain ontologies can contain up to 16500 concepts, making it difficult use and processing. To solve this problem in the literature, we propose the modularization of ontologies, which is based on an ontology divided into subparts.

The modularization of ontologies there are two approaches: partitioning of ontologies (Ontology Partitioning Approaches) and extraction module (Module Extraction Approaches). For each approach there are different modularization techniques such as Swoop, Pato based on the partitioning approach and Prompt and KMi in the module extraction approach. These techniques are implemented in tools, which in most bear the name of the technique implemented, for example Swoop, DUCK, Prompt and KMi. These tools are developed in a desktop environment and implement a technique, therefore, one approach to modularization. If you ever want to modularize ontology with a technique or a different approach to that implemented, there is no way to do, option is to choose one that meets the new requirements.

This paper addresses this problem by implementing a web tool that a modularized ontologies approach and technique that the user chooses, it returns the result of the OWL ontology modularized in contributing to the goal of the W3C to standardize the format of ontologies as well report provides a modularization technique with user ontology so that it can modularize the techniques you want the report that the tool will return the user to choose the most fitting modularization to your needs.

Keywords: Ontology modularization, modularization approaches, partitioning ontology module extraction, ontology modularization techniques, Pato, Swoop, Prompt, KMi, OWL.

Dedicatoria

Dedicado al Señor Gildardo Baquero y a la Señora Deysi Grajales, Infinitas gracias por su apoyo, por sus consejos, por su comprensión al estar ausente todo este tiempo, ustedes son el motor de mi vida y sin ustedes esto no sería posible. No existen palabras para expresar lo agradecido que estoy con ustedes.

Juan Sebastian

Dedicado a Zoraida Sierra, la persona que siempre estuvo a mi lado durante toda mi carrera, regañándome al momento de hacer pereza, apoyándome cuando estuve en momentos de crisis, dando su mejor esfuerzo por facilitarme mi responsabilidades para poder lograr culminar esta carrera y más importante compartiendo conmigo en los momentos dulces y amargos, por esto y mucho más, todo esto va dedicado a ella.

Néstor Daniel

Agradecimientos

Primero que todo le quiero agradecer a Dios por estar conmigo en esta etapa de mi vida, por guiarme en los momentos difíciles, por consolarme en los momentos tristes, y por las muchas alegrías que me permitió vivir. A mi padre sin su apoyo y consejo definitivamente esto no sería posible. A mi madre y hermanas quienes con su consejo y amor me alentaron a alcanzar este triunfo. A mis profesores y compañeros por todos los conocimientos y momentos que compartimos y a mi director de trabajo de grado el Magister Oscar Ceballos por el apoyo y conocimiento donado a este trabajo. Infinitas gracias a todos.

Juan Sebastian

En primer lugar agradezco a Dios por la vida que tengo y lo que he podido hacer con ella hasta el momento como realizar esta importante etapa, a mis padres Oscar Herrera y María Fabiola Betancur por estar siempre dándome su apoyo y ánimos, a mis hermanos Jenny, Emmanuel y Mateo Herrera por darme las ganas de seguir adelante para poder ayudarlos en un futuro, a mi director de trabajo de grado el Magister Oscar Ceballos por toda su ayuda y conocimiento prestado y un agradecimiento de todo corazón a mis tíos Javier Herrera y Olga Herrera porque sin ellos el ingresar a una universidad no habría sido posible para mí, por su preocupación y constante ayuda. Infinitas Gracias.

Néstor Daniel

Índice general

Índice de cuadros

Índice de figuras

Lista de anexos

Capítulo 1

Introducción

1.1 Descripción general

En la actualidad, las ontologías se consideran un elemento importante en la web semántica, lo que hace importante su estudio en las ciencias de la computación. Aunque el término ontología proviene de la filosofía, en inteligencia artificial tiene diferentes connotaciones. La definición más consolidada es la propuesta por Gruber en 1993 y extendida por Studer y colegas en 1998 [1] la cual dice: una ontología es una especificación explícita y formal sobre una conceptualización compartida. En esta definición, conceptualización hace referencia a una vista simplificada y abstracta del mundo que se desea representar para algún propósito específico, explícita se refiere a que el tipo de conceptos utilizados sean explícitamente definidos, formal se entiende como el hecho de que una ontología debe ser legible por la máquina y compartida que refleje la noción de que la ontología no es restringida solo para un individuo, sino que es aceptada por un grupo de personas.

Las ontologías cuentan con componentes para cumplir con la definición anteriormente mencionada, estos componentes fueron definidos por Gruber en [1] y son: conceptos que son los nombres utilizados para referirse un objeto o término en específico que puede incluir un conjunto de sinónimos que especifican los mismos objetos o términos. Las relaciones, se utilizan para representar correspondencias entre diferentes conceptos y para proveer una estructura jerárquica a la ontología. Las funciones se refiere a lo que hace un concepto y esto puede ayudar a describir el concepto a más detalle. Las instancias son ejemplos puntuales del concepto y los axiomas.

Además de definir estos componentes, para cumplir con la palabra formal de la definición de ontologías, se necesitan lenguajes para definirlas en el computador. Existen varios lenguajes para la representación de ontologías, entre los más representativos están RDF (ResourceDescription Framework) [2] y OWL (Web OntologyLanguaje) [3] . RDF es un lenguaje diseñado por la w3c. La w3c tiene como principal objetivo definir mecanismos para describir recursos en la web. OWL aprovecha las ventajas (independencia del editor, reutilización de datos, auto contención, modularidad del esquema y escalabilidad) de RDF y las adopta como el lenguaje estándar para representar ontologías en la web.

En las últimas dos décadas se han desarrollado ontologías en diferentes campos [4] como medicina, biología, comercio electrónico, genética entre otras. Las ontologías contribuyen a la construcción de fuentes de información y documentación relevante, organizada, clasificada y actualizada para

las comunidades virtuales de trabajo e investigación. Sin embargo, una dificultad que se presenta con las ontologías, especialmente en áreas como la medicina o la biología, es que su gran tamaño en cuanto al número de conceptos y relaciones, por ejemplo, algunas ontologías puede contener hasta 16500 conceptos, lo que dificulta el uso y el procesamiento.

Para solucionar este problema, en la literatura, se propone la modularización de ontologías, la cual se basa en dividir una ontología en subpartes. En la modularización de ontologías existen dos enfoques: particionamiento de ontologías (OntologyPartitioningApproaches) y extracción de módulo (Module ExtractionApproaches) [5] . Por cada enfoque de modularización existen diferentes técnicas tales como Swoop [6], PATO [7] que se basan en el enfoque de particionamiento y Prompt [8] y KMi [9] en el enfoque de extracción de módulo. Estas técnicas se implementan en herramientas [10], que en la gran mayoría, llevan el nombre de la técnica implementada, por ejemplo Swoop, PATO, Prompt y KMi. Estas herramientas son desarrolladas en un entorno de escritorio e implementan una técnica, por consiguiente, un solo enfoque de modularización. Si en algún momento se desea modularizar una ontología con una técnica o un enfoque diferente a la implementada, no hay manera de realizarlo, la opción es elegir otra que cumpla con los nuevos requerimientos.

Para solucionar estos problemas se diseñó e implementó WTOM, una herramienta web para la modularización de ontologías que permite seleccionar, según el caso, el enfoque y la técnica de modularización más apropiada a la necesidad del usuario, entrega un informe sobre el resultado de la modularización para que el usuario elija cual técnica es la que más le conviene según su necesidad y el resultado de la modularización se entrega en formato OWL para contribuir con el objetivo de la W3C de estandarización de formato de ontologías.

1.2 Problema

1.2.1 Descripción del problema

En la modularización de ontologías existen dos enfoques: primero, denominado particionamiento de ontologías (OntologyPartitioningApproaches) y segundo, extracción de módulo (Module ExtractionApproaches) [5]. Por cada enfoque de modularización existen diferentes técnicas. En el primer enfoque, se encuentran las técnicas PATO [7] y SWOOP [6]. En el segundo, se encuentra KMi [9] y Prompt [8].

Por lo general, por cada técnica existe una herramienta de software que lleva el mismo nombre. En la figura 1.1 se comparan estas herramientas según el enfoque y la técnica utilizada. Los criterios de comparación son: el tamaño de los módulos y las propiedades (redundancia, conectividad y distancia) de la ontología. La comparación se realiza aplicando estas técnicas a las ontologías ISWC [11], KA [12] y Portal [13]. La ontología ISWC define relaciones entre personas (investigadores), organizaciones y publicaciones. Esta ontología cuenta con 55 conceptos, 72 relaciones y 5 axiomas. La ontología KA que define una organización virtual compuesta por investigadores, universidades, proyectos, publicaciones y la ontología Portal la cual se utiliza en artículos comunitarios que impulsen la web semántica; la primera versión de esta ontología define 106 clases, 63 propiedades y 6 instancias [14]. Además, en el cuadro 1.1 se comparan las herramientas según el formato de la ontología que modularizan y la interacción de la herramienta con el usuario

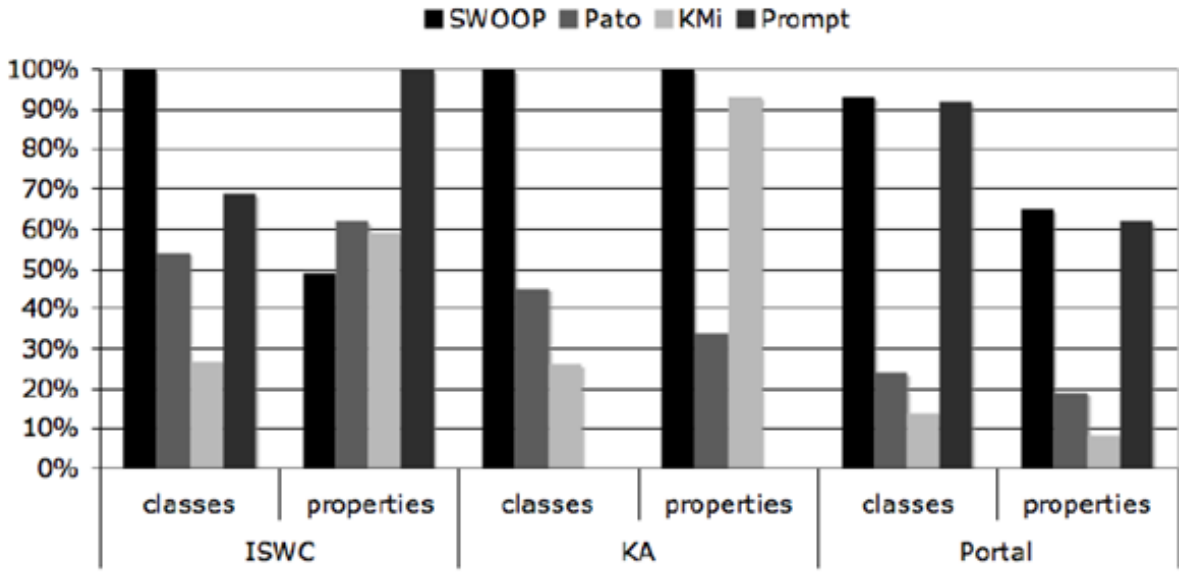

Figura 1.1: Comparación de herramientas de modularización.

Herramienta	**Formato ontología**	**Interacción**
SWOOP	RDF(S) y OWL	Automática
PATO	RDF(S).	Parametrizada
Prompt	RDF(S) y OWL	Interactiva
KMi	RDF(S) y OWL	Automática

Cuadro 1.1: Comparación herramientas de modularización

Con base en los criterios de comparación, se puede observar que SWOOP genera módulos de gran tamaño, incluso tan grandes como la ontología original y el número de propiedades que devuelve son importantes. SWOOP realiza la modularización de manera automática puesto que no necesita de parámetros adicionales. Por el contrario, KMi genera módulos más pequeños en comparación a las otras herramientas y entrega buena parte de las propiedades de la ontología. KMi requiere de un conjunto de término de la ontología para la modularización.

PATO es la más óptima de todas, puesto que, los módulos se reducen en buena proporción al igual que las propiedades de la ontología. PATO necesita de parámetros adicionales dependiendo de la ontología y los requerimientos de la aplicación para realizar la modularización. Prompt, al ser una herramienta interactiva (involucra al usuario en cada paso de la modularización), depende de cómo el usuario configure las opciones ofrecidas para obtener la modularización.

Además de los criterios de comparación establecidos, también se puede establecer que la mayoría de las herramientas para la modularización, se desarrollan bajo un entorno de escritorio [1] (normalmente implementadas en java) restringidas a un enfoque o técnica de modularización, lo que causa algunas dificultades, por ejemplo cuando se requiere realizar la modularización con otra técnica. En este caso, la solución es cambiar de herramienta. Además, no es posible ver las diferencias de la modularización según el enfoque y técnica utilizada.
También, en [15] se puede observar que el consorcio W3C define a RDF como la infraestructura para la definición de recursos en la web semántica y define a OWL como el lenguaje de las ontologías web proponiendo a la comunidad de la web semántica estandarizar todas las ontologías en OWL, (puesto que RDF es para definición de recursos y las ontologías definidas en otros entornos no son compatibles con la arquitectura World Wide Web en general y de la web semántica en particular). Por lo tanto, se puede establecer una necesidad de contar con una herramienta que implementa múltiples enfoques y técnicas de modularización, además de contar con la posibilidad de exportar el resultado a OWL.

1.2.2 Formulación del problema

¿Cómo facilitar la modularización de ontologías de tal manera que se permita seleccionar el enfoque y la técnica más adecuada según el criterio del usuario?

1.3 Objetivos

1.3.1 Objetivo general

Desarrollar una herramienta web para la modularización de ontologías que permite aplicar diferentes enfoques y técnicas según la elección del usuario.

1.3.2 Objetivos específicos

En el cuadro 1.2 se relacionan los objetivos específicos con los productos esperados de dichos objetivos y la sección de este documento que expone dicho resultado.

Objetivo específico	Sección
1. Implementar las técnicas PATO y SWOOP del enfoque de particionamiento de ontología.	3.1
2. Implementar las técnicas KMI y PROMPT del enfoque de extracción de modulo.	3.2
3. Ofrecer un reporte del resultado de la modularización según la(s) técnica(s) utilizada(s).	3.3
4. Probar que la herramienta funcione según los enfoques y técnicas de modularización de ontologías.	3.4
5. Exportar el resultado de la modularización en formato OWL.	3.5

Cuadro 1.2: Relación objetivos específicos con los productos esperados

1.4 Estructura del documento

El resto del documento está organizado de la siguiente manera:
En el *Capítulo 2: Marco referencial*, se presenta el marco conceptual y los antecedentes sobre herramientas web para la modularización de ontologías.
En el *Capítulo 3: Herramienta Web para la Modularizacion de Ontologias WTOM*, se presentan los resultados obtenidos del proceso de desarrollo.
En el *Capítulo 4: Aspectos del desarrollo de software*, se muestran las diferentes fases cumplidas en el desarrollo y el proceso de ingeniería de software llevado a cabo para cumplir con los objetivos propuestos. Además, se modelan y ejecutan diferentes escenarios de pruebas para medir la efectividad y la pertinencia de la herramienta frente a los objetivos e Historia de usuario propuestas en el proceso de desarrollo.
Este trabajo finaliza con el *Capítulo 5: Conclusiones y trabajo futuro*.

Capítulo 2

Marco Referencial

En el presente capítulo se definen los conceptos usados y discutidos en el trabajo de grado y en el desarrollo de la aplicación. Además se exponen los antecedentes de herramientas implementadas para la modularización de ontologías.

2.1 Marco Conceptual

2.1.1 Ontología

En informática la definición declarativa más consolidada de ontología es la propuesta por Gruber en 1993 y extendida por Studer y colegas en 1998 [1], la cual dice: "Una ontología es una especificación explicita y formal de una conceptualización compartida". En esta definición, conceptualización es una vista simplificada y abstracta del mundo que deseamos representar para algún propósito en específico, definiendo un vocabulario controlado. Explicita se refiere a que el tipo de conceptos utilizados sean explícitamente definidos. Formal se refiere al hecho de que la ontología debe ser legible por la máquina. Compartida refleja la noción de que la ontología no es restringida solo para un individuo, sino que es aceptada por un grupo de personas. El uso de ontologías permite el tratamiento ponderado del conocimiento, sirviendo como herramienta para recuperar información de una manera automatizada. Se considera a las ontologías como el corazón de la web semántica, estas se representan bajo varios lenguajes como OIL, DAML, pero relacionados a nuestro tema de interés hablaremos de dos de estos en específico, RDF y OWL.

OWL es un lenguaje de marcado para publicar y compartir datos usando ontologías en la World Wide Web (WWW). Actualmente OWL es el formato estándar para las ontologías de acuerdo al World Wide Web Consortium (W3C) y tiene como objetivo facilitar un modelo de marcado construido sobre RDF y codificado en XML. Tiene como antecedente DAML+OIL, en los cuales se inspiraron los creadores de OWL para crear el lenguaje. Junto al entorno RDF y otros componentes, estas herramientas hacen posible el proyecto de web semántica. RDF es un framework para metadatos en la World Wide Web (WWW), desarrollado por el World Wide Web Consortium (W3C). Es un lenguaje de objetivo general para representar la información en la web (un metadato data model). Es una descripción conceptual. Así mismo hace parte de los lenguajes de ontologías.

Profundizando en el tema de las ontologías, se puede tomar una ontología como una especificación explícita y formal de una conceptualización compartida, en lo referente a explicita seria el hecho

de que los conceptos que usamos son explícitamente definidos y la conceptualización seria en si una vista más simple y abstracta con un vocabulario controlado de aquello que deseamos representar con un determinado fin, así mismo, estas varían en su forma y disposición, pero todas poseen uno o varios componentes necesarios que les sirven para representar el conocimiento de algún dominio en específico, entre los que están [16]:

Conceptos
Todas aquellas ideas básicas que se intentan formalizar. Estos conceptos pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento, etc.

Relaciones
Representan la interacción y enlace entre los conceptos del dominio. Suelen formar la taxonomía del dominio. Por ejemplo: subclase-de, parte-de, parte-exhaustiva-de, conectado-a, etc.

Funciones
Forman un tipo concreto de relación donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología. Por ejemplo, pueden parecer funciones como categorizar-clase, asignar-fecha, etc.

Instancias
Se utilizan para representar objetos determinados de un concepto.

Axiomas
Son teoremas que se declaran sobre relaciones que deben cumplir los elementos de la ontología. Por ejemplo: “Si A y B son de la clase C, entonces A no es subclase de B”, “Para todo A que cumpla la condición C1, A es B”, etc.

Un ejemplo de ontología aplicando algunos de estos conceptos se puede visualizar en la figura 2.1.

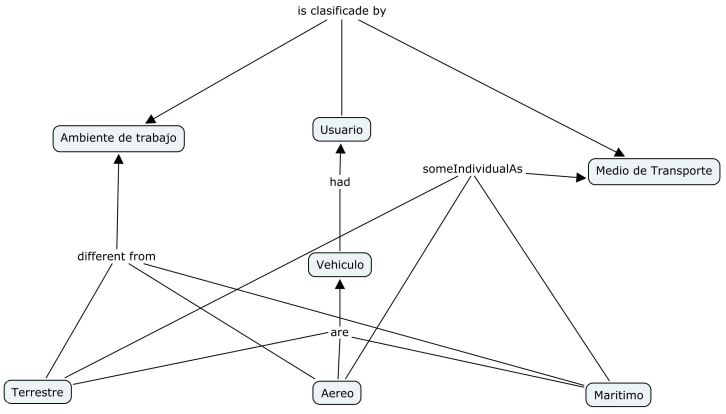

Figura 2.1: Ontologia vehículos.

2.1.2 Modularización

La Modularización de ontologías se basa en "dividir" la ontología para que un usuario trabaje con la sub parte que más le interesa de la ontología.
Para modularizar una ontología existen dos enfoques [10]: particionamiento de ontologías (Ontology Partitioning Approaches), es la tarea de particionar una ontología original (O), dividiendola en módulos M (M,. . .,Mk) cada Mi es una ontología y la unión de todos los módulos es equivalente a la ontología original O, la extracción de modulo (Module Extraction Approaches)[5] que consiste en reducir una ontología en sub-partes, los módulos se convierten en un particular sub-vocabulario. Esta tarea también se le llama segmentación. Mas precisamente una ontología O y un conjunto SV Sig(O) de términos de la ontología, un módulo en el mecanismo de extracción retorna un módulo Msv, suponiendo que es parte relevante de O que cubre el sub-vocabulario SV (sig(Msv) subSet SV).. Cada uno de estos enfoques, a su vez, implementa diferentes técnicas para la modularización.

PATO[7] que es una técnica de particionamiento, la cual realiza 3 pasos para realizar la modularización

1. Crear dependencia del grafo
En el primer paso se extrae desde el archivo de la ontología un gráfico de las dependencias de la ontología, esto se implementa utilizando las librerías SWI semantic web [17], lo anterior con el fin de que el resultado pueda ser analizado por la herramienta de análisis de redes Pajek [18].

2. Determinar la fuerza de las dependencias
En el segundo paso la fuerza de las dependencias entre los conceptos tiene que ser determinada. Después de la suposición básica del enfoque, se usa la estructura del gráfico de la dependencia para determinar los pesos de estas. En particular, se usa los resultados de la teoría de redes sociales mediante el cálculo de la red de resistencia proporcional para el grafo de dependencias. La fuerza $P_{i,j}$ proporcional de una conexión entre un nodo c_i y c_j describe la importancia de un enlace desde un nodo al otro basado en el número de conexiones de un nodo tiene ($a_{i,j}$ es el peso asignado previamente para el enlace entre el c_i y c_j.

$$P_{i,j} = \frac{a_{i,j} + a_{j,i}}{\sum_k a_{i,k} + a_{k,i}}$$

La intuición detrás de esto es que los contactos sociales individuales son más importantes si sólo hay unos pocos de ellos. En este caso, esta medida es útil porque se quiere evitar que las clases que sólo se relacionan con un bajo número de otros conceptos se separen de ellos. Esto iría en contra de la intuición de que los conceptos en un módulo deben estar relacionados.

3. Determinar módulos
La red de resistencia proporcional proporciona una base para detectar conjuntos de conceptos fuertemente relacionados. Para ello, se hace uso del algoritmo de ïsla"[19]. Un conjunto de vértices $I \subseteq C$ es una isla en la línea de red si y sólo si se induce un subgrafo conectado y las líneas dentro de la isla son más fuertes relacionados entre ellos que con los vértices vecinos.

SWOOP[6] que también hace parte de las técnicas de particionamiento, el algoritmo de esta técnica consiste en:

El algoritmo inicialmente comprueba si O es ε-safe. Si la comprobación es negativo, entonces se

devuelve O como un resultado. De lo contrario, el algoritmo comienza un paso de particionado mediante la creación de un nuevo módulo $\sum_i$ y al forzar una transición de estado inicial en una entidad arbitraria (concepto, rol o individuo) en O.

En un paso partición dada, cada relación conceptual, individual y de enlace (generado en el paso anterior) puede estar en uno de dos estados posibles: o bien como entidades en O o en $\sum_i$. Un rol, sin embargo, puede estar en uno de cuatro estados posibles: Como un rol en O (Estado 1), como una relación de O a $\sum_i$ (Estado 2), como una relación enlace de $\sum_i$ a O (Estado 3), y finalmente, como un rol en $\sum_i$ (Estado 4). Sólo las transiciones $1 \rightarrow 2$,$1 \rightarrow 3$,$1 \rightarrow 4$ $2 \rightarrow 4$ y $3 \rightarrow 4$ están permitidos.

Una vez que todas las transiciones de estado en un proceso de particionado se han completado, los axiomas correspondientes se mueven. Cada axioma en la ontología de entrada se desplaza una sola vez, es decir, cada vez que se exporta desde O a un componente recién creado, nunca se pondrá de nuevo en O, ni se movía a un componente diferente.

Pasando a las técnicas te extracción de módulo tenemos, PROMPT [9], el algorito teoricamente funciona tomando una signatura S y una ontología Q, este recupera un fragmento de Q1 perteneciente a Q de la siguiente forma: Primero, los axiomas en Q que mencionan alguno de los símbolos en S son añadidos a Q1, en segundo lugar, S se amplía con los símbolos en Sig (Q1), estos pasos se repiten hasta que se alcanza un punto fijo.

La idea principal de este algoritmo es podar axiomas irrelevantes, desafortunadamente, este algoritmo no detecta otras dependencias. Y el principal problema con estos algoritmos es que ignoran la semántica de las ontologías. Como consecuencia de ello, pueden, por un lado, extraer axiomas irrelevantes y, por otra parte, pérdida de axiomas esenciales. Estos algoritmos, sin embargo, no estaban destinados para extraer módulos de acuerdo a una colección formal de los requisitos; en cambio, tenían la intención de extraer "partes pertinentes"de ontologías que son “probablemente relacionados” a la firma dada, y no garantizan la exactitud de los resultados.

Mas en la aplicación la tecnica PROMPT crea dos versiones de la ontología Vo y Vn por cada concepto en Vo, Prompt determina una correspondencia de conceptos en Vn si estos son uno . Usando esta información se crea una vista T, el cual es definido por Vo para determinar cuáles términos son válidos en Vn. Si por cada término de Vo esta explícitamente en T como un término de arranque o nombre de una propiedad entonces este se define en Vn.

Finalmente tenemos a KMI, la cual propone computarizar por aparte los conceptos de la ontología, las propiedades, las instancias y los axiomas todo esto recursivamente. Al final obtendrá un sub vocabulario de todos estos conformando una nueva ontología más pequeña, al crear el nuevo sub vocabulario es fiel su enfoque de modularización: extracción de módulo.

2.1.3 JENA

Framework Java para construir aplicaciones basadas en ontologías[20]. Se desarrolló en HP Labs en el 2000, en 2009 HP cedió el proyecto a la fundación Apache que decidió adoptarlo en noviembre de 2010. Su arquitectura incluye API para trabajar (leer, procesar, escribir) ontologías

RDF y OWL, motor de inferencia para razonar sobre ontologías RDF y OWL, estrategias de almacenamiento flexible para almacenar tripletas RDF en memoria o fichero, motor de queries compatible con especificación SPARQL.

Al momento de gestionar una ontología usando Jena, ya sea para crearla, leerla o modificarla, hay varios aspectos a tener en cuenta, estos son[20]:

Clases
Son un conjunto de conceptos relacionados jerárquicamente que nos permitirá identificar y clasificar elementos de un dominio de forma concisa y concreta. Estas clases son muy similares a las clases que normalmente usamos en POO (Programación Orientada a Objetos) con la diferencia que las clases de la ontología tienen un significado semántico. Además de que esta puede tener subclases, clases hermanas y una clase principal de donde parten todas las demás (owl:Thing).

Instancias
Son una representación concreta de una clase. Una instancia quedará identificada y definida por la clase a la que pertenezca y por toda la jerarquía de clase que haya por encima. Esta también puede definir valores concretos para todas las propiedades que tenga la clase a la que pertenece.

Propiedades
Son la forma de asignar más contenido a una clase. Por tanto, las propiedades se definen a nivel de clase.

Veamos algunos casos del uso de Jena directamente aplicado a la herramienta WTOM:

En la figura 2.2 se ve el codigo requerido para mostrar la jerarquía de sub-clase codificada por el modelo dado:

```
1  import com.hp.hpl.jena.ontology.*;
2  import com.hp.hpl.jena.rdf.model.*;
3  import com.hp.hpl.jena.shared.PrefixMapping;
4  import com.hp.hpl.jena.util.iterator.Filter;
5
6  import java.io.PrintStream;
7  import java.util.*;
8
9      public void showHierarchy( PrintStream out, OntModel m ) {
10
11         Iterator i = m.listHierarchyRootClasses()
12                       .filterDrop( new Filter() {
13                                       public boolean accept( Object o ) {
14                                           return ((Resource) o).isAnon();
15                                       }} );
16
17         while (i.hasNext()) {
18             showClass( out, (OntClass) i.next(), new ArrayList(), 0 );
19         }
20     }
```

Figura 2.2: Jerarquía Jena.

En la figura 2.3 se visualiza como cargar y leer los datos de una ontología:

```
1  package modonto;
2
3  import com.hp.hpl.jena.ontology.OntModel;
4  import com.hp.hpl.jena.ontology.OntModelSpec;
5  import com.hp.hpl.jena.rdf.model.ModelFactory;
6
7      public static void main(String[] args) {
8          OntModel m = ModelFactory.createOntologyModel( OntModelSpec.OWL_MEM, null );
9
10         m.getDocumentManager().addAltEntry( "http://www.prueba.org/pizza",
11                                             "file:ontologias/pizza.owl" );
12
13         m.read( "http://www.prueba.org/pizza" );
14
15         new ClassHierarchy().showHierarchy( System.out, m );
16     }
17 }
18
```

Figura 2.3: Carga con Jena.

2.1.4 GWT

GWT o Google Web [21] es un entorno open source para el desarrollo en Java creado por Google que permite ocultar la complejidad de varios aspectos de la tecnología AJAX. Con GWT se pueden usar herramientas de desarrollo de Java como es el caso de Eclipse o netbeans, El concepto de Google Web Toolkit es bastante sencillo, básicamente lo que se debe hacer es crear el código en Java usando cualquier IDE, este código luego es compilado a muy alto nivel y el compilador lo traducirá a HTML y JavaScript. La primera versión de este Framework salió el 17 de mayo del 2006 y el más reciente el 15 de enero del 2013. Entre otras de sus características está el soportar todos los navegadores Dispone de una colección de componentes de interfaz de usuario (widgets) dinámicas y reusables.

El ciclo de vida del desarrollo con GWT es[21]:

1. Escribir y depurar la aplicación en Java, usando tantas librerías de GWT como sean necesarias y en cualquier entorno de desarrollo de Java.

2. Usar el compilador Java-to-JavaScript de GWT para convertir la aplicación en un conjunto de ficheros JavaScript y HTML, que se pueden llevar a cualquier servidor web.

3. Comprobar que la aplicación funciona en cada navegador al que se quiera dar soporte.

2.2 Antecedentes

2.2.1 Extracción de Módulo

PROTEGE[22] es un software para la manipulación de ontologías, este cuenta con muchos plugins, entre ellos, uno en especifico llamado PROMPT el cual soporta el manejo de múltiples ontologías,

permitiéndonos comparar versiones de una misma ontología, mezclar dos ontologías en una sola, entre otros aspectos, o extraer una parte de la ontología, como se puede observar en la figura 2.4, se debe iniciar en la configuración de PROTEGE la promptTab.

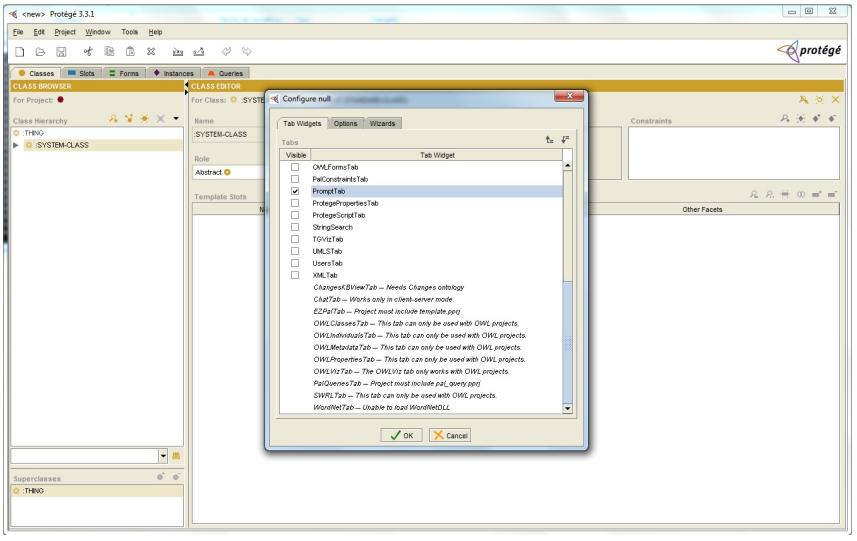

Figura 2.4: PromptTab.

Una vez activada la promptTab, este plugin nos da varias opciones a elegir, de las cuales es de nuestro interés la extracción como se muestra en la figura 2.5.

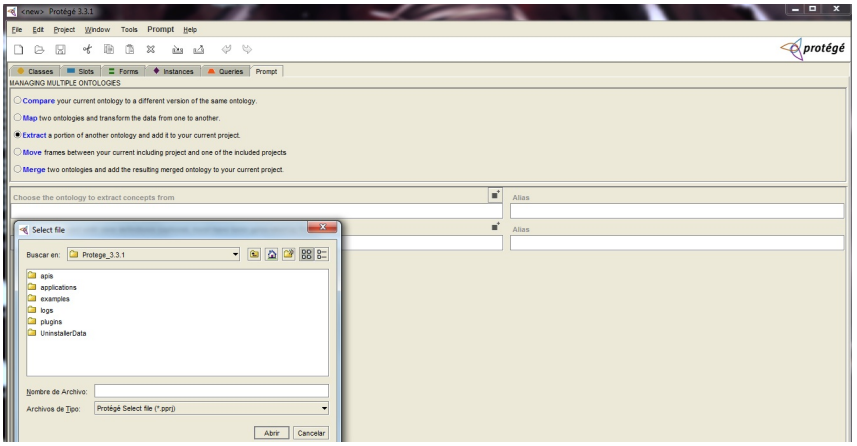

Figura 2.5: Extracción.

En esta herramienta, el plugin PROMPT muestra un gran percance, y es que al momento de extraer de una determinada ontología, esta debe estar guardada en el formato pprj, que es la extensión de los proyectos generados por PROTEGE, ya que si por ejemplo, se intenta cargar una ontología owl para realizarle una extracción esto genera errores en PROTEGE como se muestra en la figura 2.6.

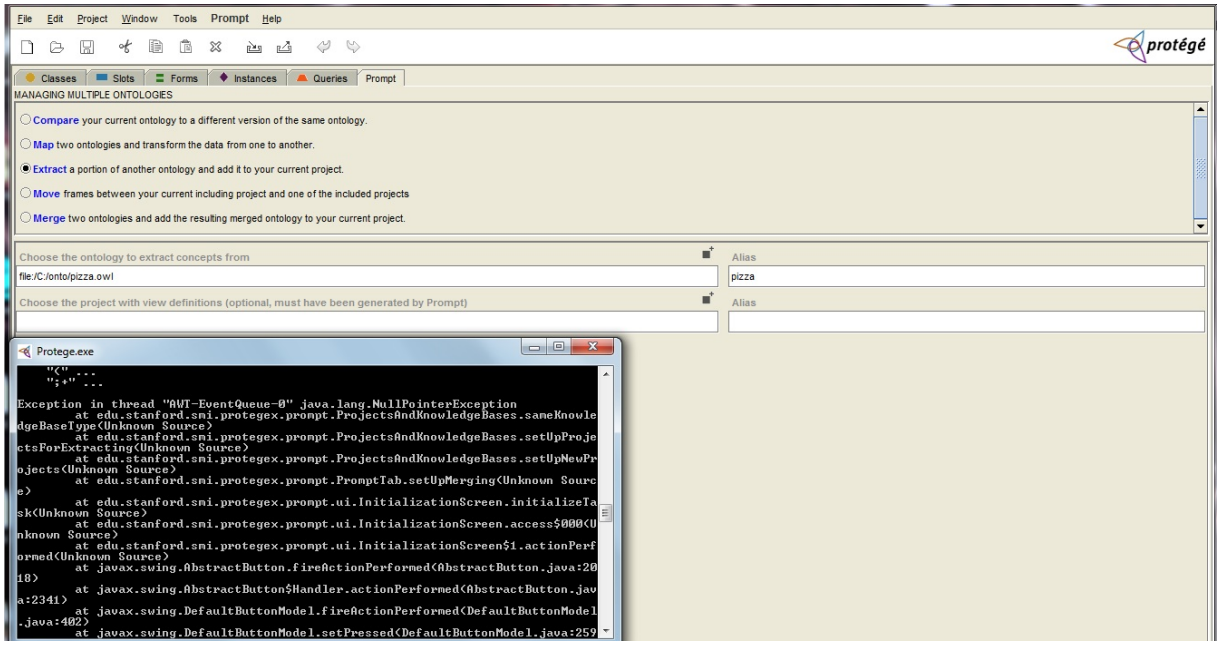

Figura 2.6: Error de Carga.

En ocasiones, es posible que se desee guardar sólo una parte de una ontología, ya sea para utilizar un subconjunto de la base de conocimientos en un proyecto diferente, para distribuir sólo una parte de dicha ontología, para depurar una porción más pequeña de la misma, hasta finalizar. PROMPT permite seleccionar y elegir los marcos y luego copiarlos en su proyecto local. Se puede elegir un marco y copiar todo relacionado con el.

Hay una gran variedad de puntos que complican el uso de PROMPT en la herramienta PROTEGE, comenzando por el formato de los archivos, al momento de extraer datos de una ontología para generar una nueva (extracción de módulo), dicha ontología inicial debe estar guardada bajo la extensión de proyectos de PROMPT .pprj de lo contrario este generara errores al cargar la ontología. Otro aspecto a tener en cuenta es la compatibilidad entre versiones, ya que el plugin de PROMPT solo es soportado hasta ciertas versiones de PROTEGE, es decir la versión más reciente de PROTEGE no cuenta con el plugin de prompt para dicha modularización.

En la figura 2.7 podemos observar la carga de la ontología inicial Vo en la llamada vista T, en la cual el usuario pasa a realizar la elección de las clases, subclases y demás requeridos para su nueva ontología, para más detalle del manejo y funcionamiento de esta herramienta y la aplicación de la técnica ver el punto 3.2.1 .

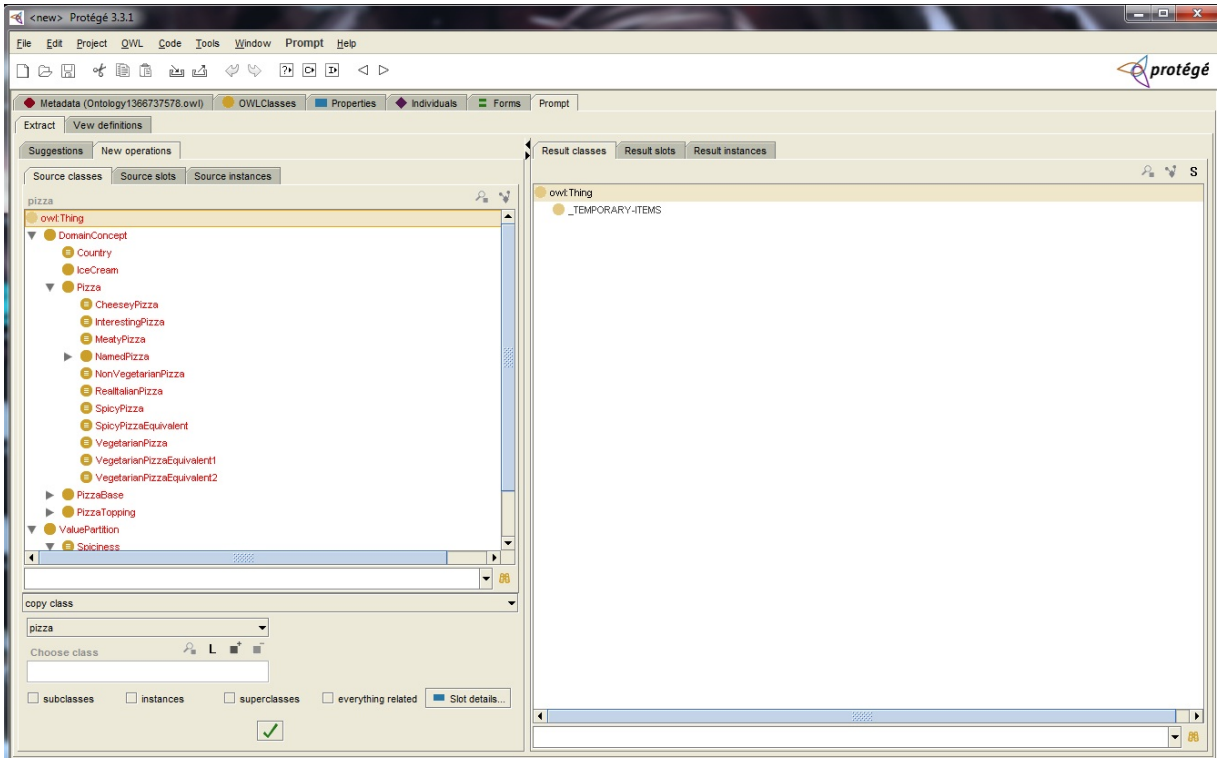

Figura 2.7: Prompt Protege.

En cuanto a KMI, la herramienta no se encuentra disponible para su descarga o prueba del funcionamiento de dicha modularización, se empleó una gran cantidad de tiempo en la búsqueda de esta, pero todo conlleva a que esta se retiró por parte de sus desarrolladores en el Knowledge Management Institute, en este caso para la aplicación de la técnica, todo lo referente a dicha modularización es respaldado científicamente por documentación, estudios y teoría del funcionamiento de esta, como es el caso de [10], aquí se hace referencia al modo en que se pasa de un vocabulario complejo a un su vocabulario conformado por las principales clases y relaciones requeridas..

2.2.2 Particionamiento de Ontología

En cuanto el enfoque de particionamiento de Ontología el cual consiste en modularizar la ontología y que al unir los módulos que se generan en la modularización este conjunto tiene que ser equivalente a la ontología original.

De este enfoque existen dos técnicas implementadas las cuales son: Swoop y Pato.

En cuanto a SWOOP, está desarrollada bajo el enfoque de escritorio en lenguaje Java, se enfoca a ontologías en formato OWL y crea un conjunto de ontologías locales(figura 2.8).

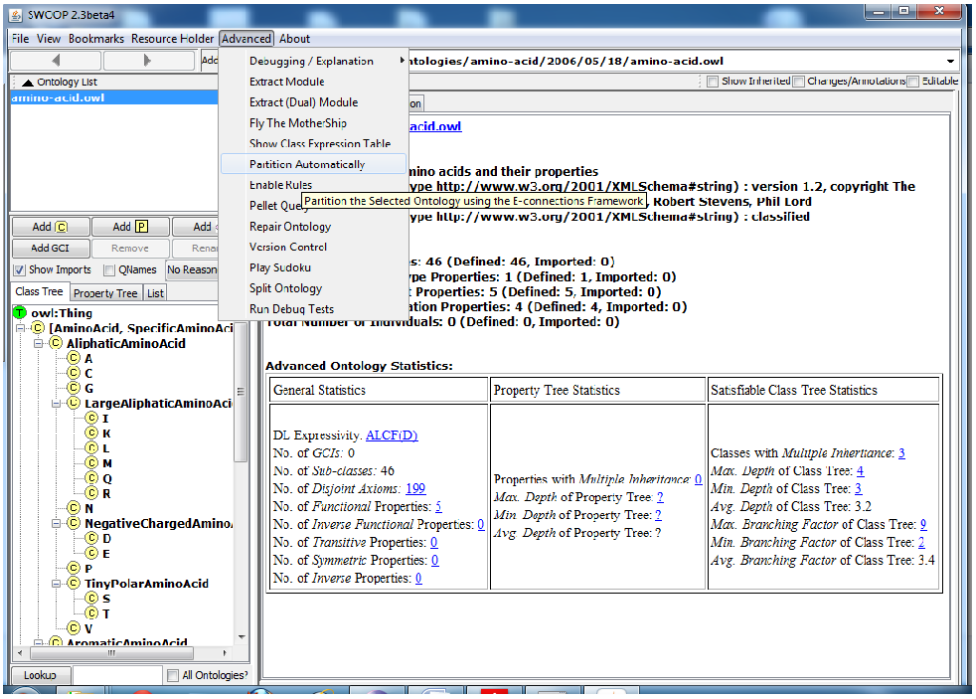

Figura 2.8: Swoop Escritorio.

PATO por su parte es una aplicación que también está bajo el enfoque de escritorio, es una herramienta desarrollada en java y trabaja con ontologías en formato RDF. Esta herramienta divide la ontología en un conjunto de módulos, para ello, primero procesa una dependencia de grafos entre las entidades de la ontología (figura 2.9).

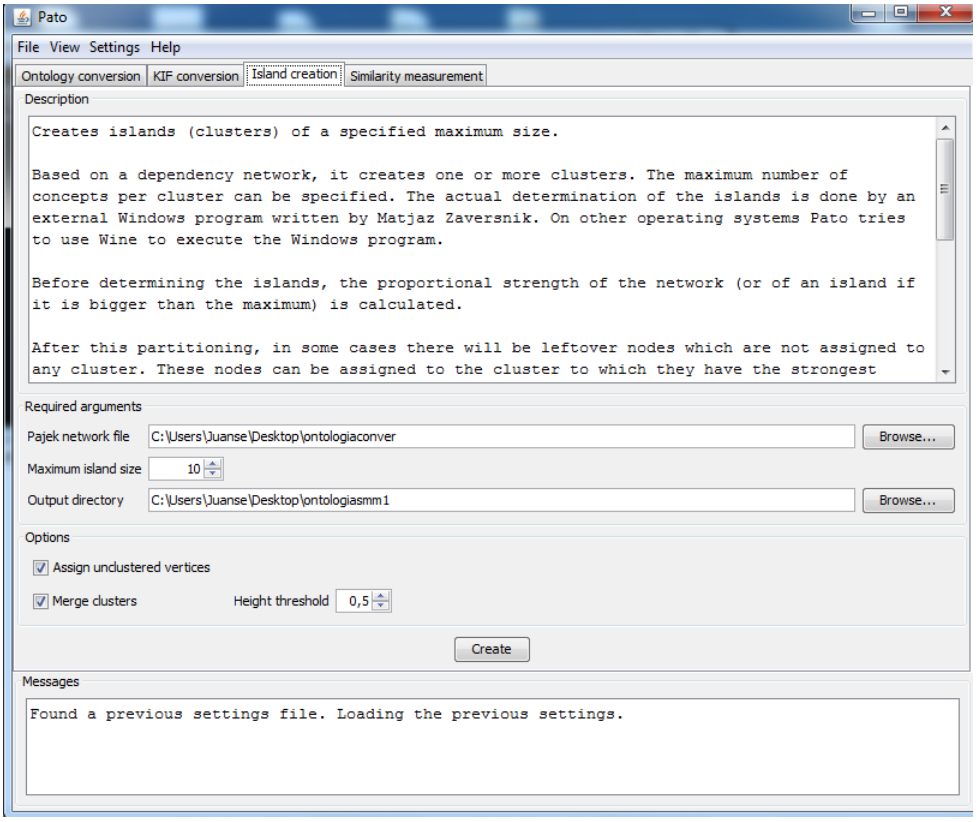

Figura 2.9: Pato Escritorio.

Capítulo 3

Herramienta Web para la Modularización de Ontologías *WTOM*

En el presente capítulo se propone *WTOM* (*Web Tool Ontolgies Modularization*) para facilitar la tarea de modularización de ontologías. *WTOM* es una herramienta Web que reúne los dos enfoques de modularización de ontologías y dos técnicas de cada enfoque acaparando cuatro técnicas de modularización de ontologías. En la figura 3.1, se visualiza en Index principal de WTOM

Figura 3.1: Index WTOM.

WTOM se compone de 10 paneles: principal, Registro, Login, Carga de Ontología, Modularización, Panel Pato, Panel Swoop, Panel Prompt, Panel Kmi, Panel descarga. Después de realizada la modularización en la carpeta de usuario se crean los archivos OWL generados de la modularización los grafos de estos módulos y el reporte, toda esta carpeta queda a disposición del usuario por si este desea descargarla.

La herramienta después de que el usuario se identifique procederá a solicitarle al usuario cargar la ontología que desea trabajar en la herramienta, el usuario tiene la opción de subirla desde su equipo (figura 3.2) o desde un repositorio (figura 3.3).

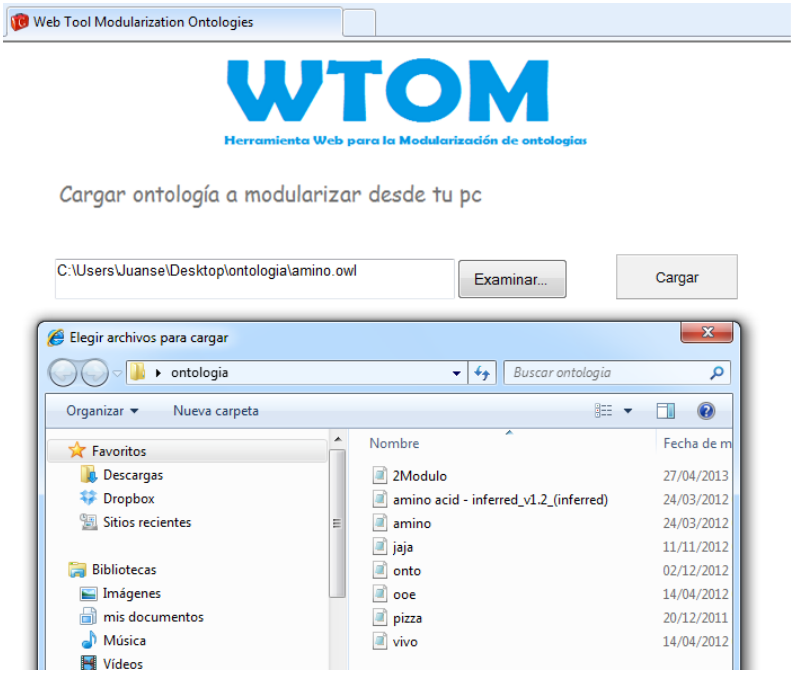

Figura 3.2: Carga de Ontología desde PC.

Figura 3.3: Carga de Ontología desde Repositorio.

Este proceso se muestra más detalladamente a continuación: Se selecciona una ontología en formato owl desde el disco del PC(figura 3.4)

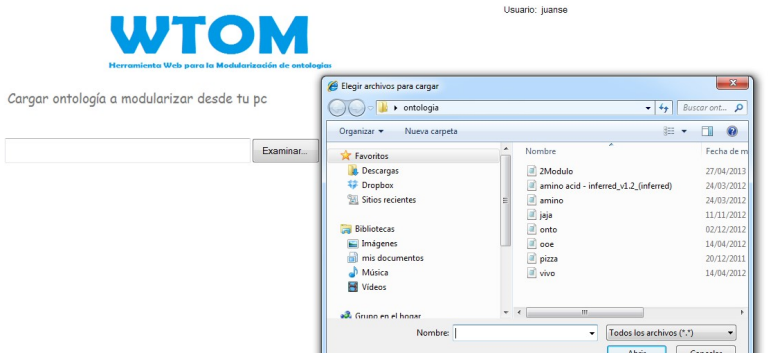

Figura 3.4: Carga de Ontología.

Se verifica que la ruta de carga sea la correcta(figura 3.5)

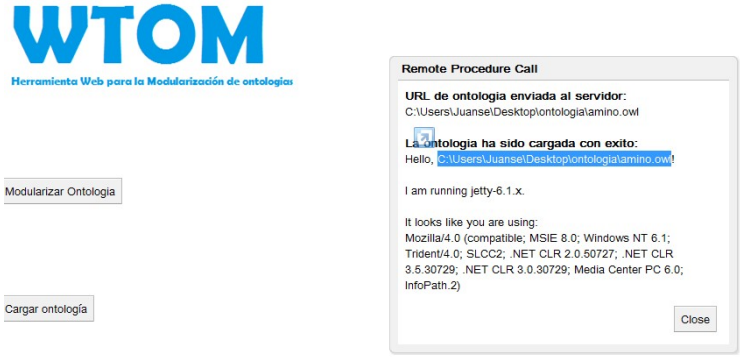

Figura 3.5: Confirmación de Ruta.

Por último se verifica que el archivo se encuentre en la carpeta del usuario en el servidor(figura 3.6)

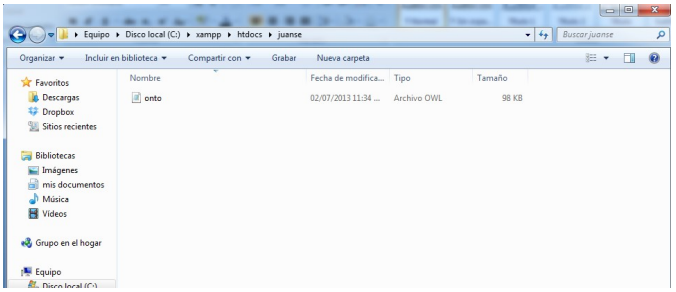

Figura 3.6: Verificación en el Servidor.

En la carga desde repositorio, se establece la URL (figura 3.7)

Figura 3.7: Verificar url.

Aqui se muestra la pagina en donde se encuentra ubicada la ontología en la red(figura 3.8)

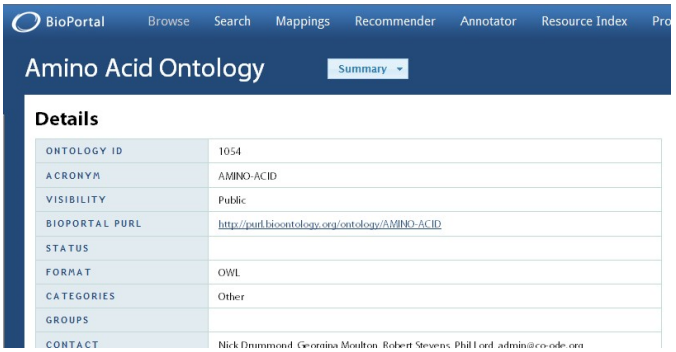

Figura 3.8: Carga Repositorio

Luego se verifica la dirección de la carga al servidor sea la URL de la ontología(figura 3.9)

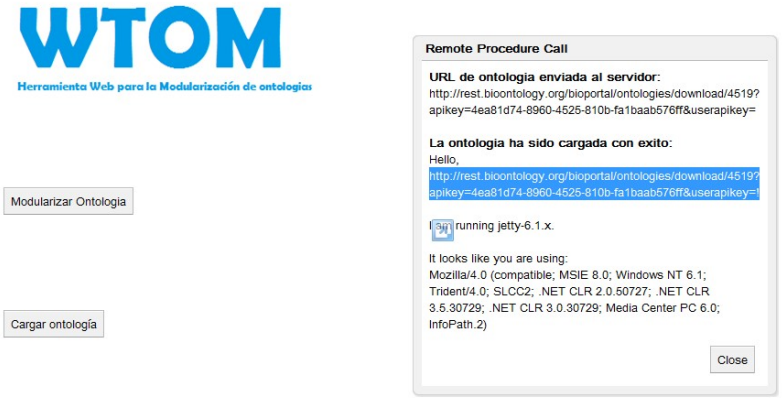

Figura 3.9: Ver Repo.

Y por último se verifica que el archivo se encuentre en la carpeta del usuario en el servidor como se hizo anteriormente.

3.1 Implementación las técnicas PATO y SWOOP del enfoque de particionamiento de ontología

Para cumplir con este objetivo específico se consultó la documentación oficial de estas técnicas de modularización entendiendo científicamente como operaban, posteriormente se buscan las herramientas en entorno de escritorio, se evaluaron, y por último se reutilizarón los algoritmos que se proponían en la documentación de cada técnica para ofrecerlas en entorno web, obteniendo los siguientes resultados:

3.1.1 Implementación Técnica Pato

Una vez cargada la ontología en el panel de modularización se selecciona el enfoque de particionamiento de ontología el cual despliega en su menú las opciones de modularizar con Pato o Swoop, al elegir pato se obtiene la siguiente interfaz:

Figura 3.10: Modularizar con Pato.

Despues de elegir modularizar con pato se obtiene los resultados de la siguiente forma:

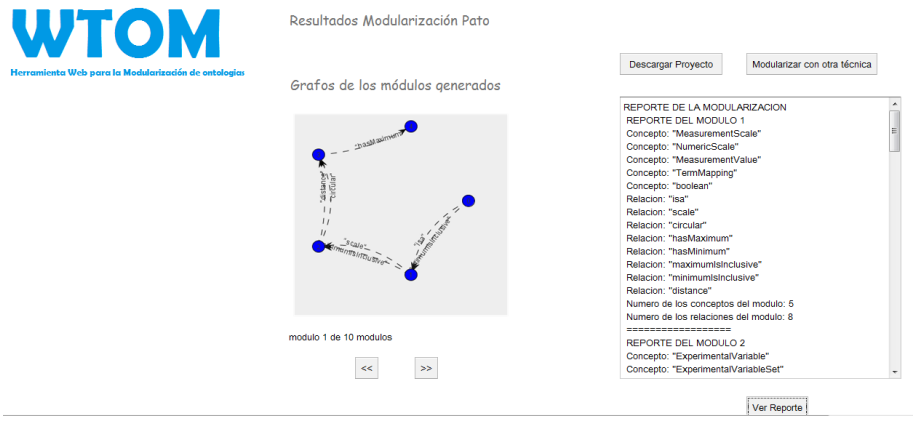

Figura 3.11: Resultados Pato.

En esta interfaz se puede apreciar los grafos de los módulos generados por la modularización, además se obtiene un reporte donde se informa al usuario los conceptos, las relaciones y el numero de conceptos y relaciones de cada módulo, además da la opción de descargar proyecto el cual permitirá al usuario descargar a su equipo los módulos en formato OWL.

3.1.2 Implementación Técnica Swoop

Igualmente que en la técnica Pato, una vez cargada la ontología, en el panel de modularización se selecciona el enfoque de particionamiento de ontología el cual despliega en su menú las opciones de modularizar con Pato o Swoop, al elegir Swoop se obtiene la siguiente interfaz:

Figura 3.12: Modularizar con Swoop.

Despues de elegir modularizar con Swoop se obtiene los resultados de la siguiente forma:

Figura 3.13: Resultados Swoop.

En esta interfaz se puede apreciar los grafos de los módulos generados por la modularización, además se obtiene un reporte donde se informa al usuario los conceptos, las relaciones y el numero de conceptos y relaciones de cada módulo, además da la opción de descargar proyecto el cual permitirá al usuario descargar a su equipo los módulos en formato OWL.

3.2 Implementar las técnicas KMI y PROMPT del enfoque de extracción de módulo

Para cumplir con este objetivo se consultó la documentación oficial de estas técnicas de modularización entendiendo como operaban en especial para el caso de la técnica de KMI que no fue posible encontrar la herramienta, posteriormente para el caso de prompt se buscan las herramientas en entorno de escritorio, se evaluarón, y por último se reutilizarón los algoritmos que se proponían en la documentación de cada técnica para ofrecerlas en entorno web, obteniendo los siguientes resultados:

3.2.1 Implementación Técnica Prompt

Una vez cargada la ontología, en el panel de modularización se selecciona el enfoque de extracción de módulo el cual despliega en su menú las opciones de modularizar con Prompt o KMi, al elegir Prompt se obtiene la siguiente interfaz:

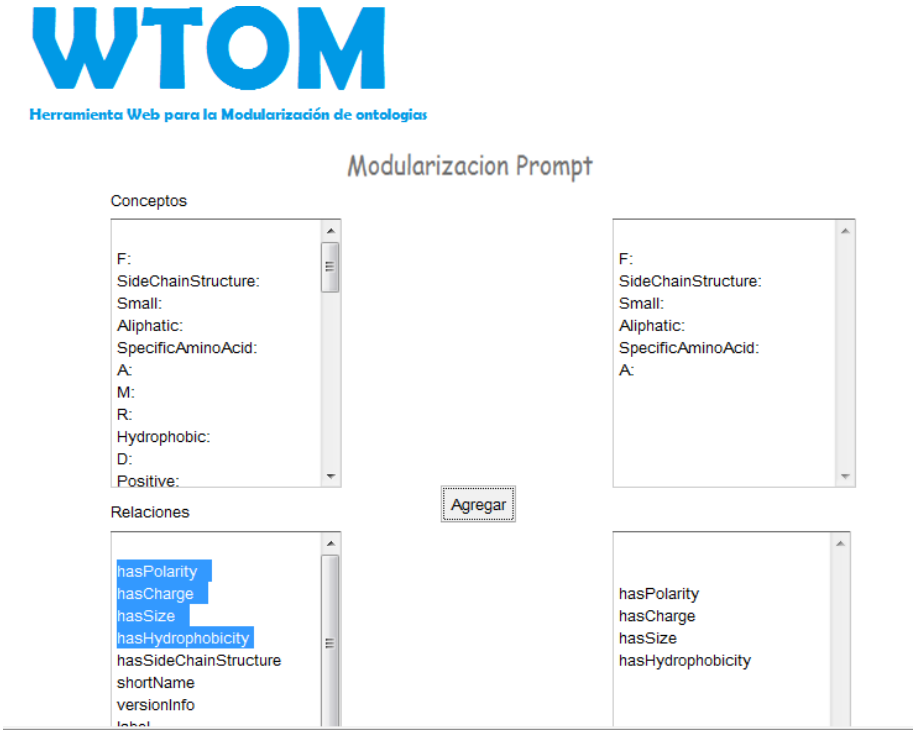

Figura 3.14: Modularizar con Prompt.

Esta Interfaz en las aéreas de texto del lado derecho lista los conceptos y las relaciones de la ontología cargada, el usuario procede a seleccionar tanto los conceptos como las relaciones que desea en el nuevo módulo y después va a la opción crear ontología en donde se genera el nuevo módulo, los resultados se visualizan de la siguiente forma:

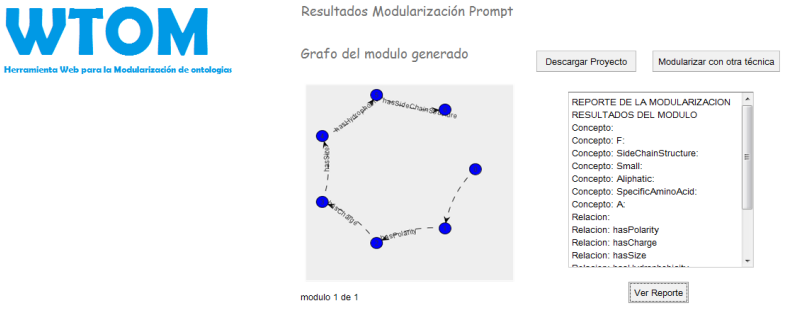

Figura 3.15: Resultados Prompt.

En esta interfaz se puede apreciar el grafo del módulo generado por la modularización, además se obtiene un reporte donde se informa al usuario los conceptos, las relaciones y el numero de conceptos y relaciones del módulo, además da la opción de descargar proyecto el cual permitirá al usuario descargar a su equipo el módulo en formato OWL.

3.2.2 Implementación Técnica KMi

Una vez cargada la ontología, en el panel de modularización se selecciona el enfoque de extracción de módulo el cual despliega en su menú las opciones de modularizar con Prompt o KMi, al elegir KMi se obtiene la siguiente interfaz:

Figura 3.16: Modularizar con KMi.

Despues de elegir modularizar con KMi se obtiene los resultados de la siguiente forma:

Figura 3.17: Resultados KMi.

En esta interfaz se puede apreciar el grafo del módulo generado por la modularización, además se obtiene un reporte donde se informa al usuario los conceptos, las relaciones y el numero de

conceptos y relaciones del módulo, además da la opción de descargar proyecto el cual permitirá al usuario descargar a su equipo el módulo en formato OWL.

3.3 Reporte del resultado de la modularización según la(s) técnica(s) utilizada(s).

Para cumplir este objetivo se realizó un análisis de los reportes que ofrecían las herramientas de modularización para que WTOM ofreciera reportes mas completos obteniendo los siguientes resultados:

3.3.1 Reporte WTOM Enfoque Particionamiento de Ontología

Puesto que el enfoque de particionamiento de ontología genera varios módulos el reporte se organiza de la siguiente manera tanto para Pato como para Swoop:

Por cada módulo se realiza en reporte en el cual se empieza identificando el módulo, se listan los conceptos de este módulo, las relaciones, y por último se informa al usuario el numero de conceptos y relaciones del módulo, este proceso con todos los módulos que se hayan generado en la modularización.

3.3.2 Reporte WTOM Enfoque Extracción de Módulo

Puesto Que el enfoque de Extracción de módulo solo genera un módulo el reporte se organiza de la siguiente manera tanto para Prompt como para Swoop:

Para el módulo se realiza en reporte en el cual se listan los conceptos del módulo, las relaciones, y por último se informa al usuario el número de conceptos y relaciones del módulo.

3.4 probar que la herramienta funcione según los enfoques y técnicas de modularización de ontologías.

Trabajando con las herramientas que se encontraron de escritorio como referentes para las pruebas de funcionamiento de las técnicas y además en la literatura científica de estas se puede concluir:

3.4.1 Prueba de funcionamiento Pato

Con respecto a la técnica pato puesto que su implementación se realizo reutilizando código de la herramienta original se observan que el algoritmo implementado realiza los 3 pasos básicos para que propone la información científica para realizar la modularización con esta técnica, quedando faltante el paso de optimización que realiza la herramienta pero que no se propone como necesario para la modularización por lo cual con respecto a la herramienta no da el mismo número de módulos pero si se realiza correctamente la modularización.

3.4.2 Prueba de funcionamiento Swoop

Igualmente Swoop se implemento reutilizando código de la herramienta original y se verifica que este algoritmo cumpla con los pasos básicos expuestos en la literatura por lo cual se verifica que los módulos generados por Swoop de escritorio son exactamente iguales a los que genera WTOM.

3.4.3 Prueba de funcionamiento Prompt

Al probar la modularización con Prompt en el gestor de ontologías Protege se observa que el usuario navega por la ontología siendo el quien decide que conceptos y que relaciones conformaran el nuevo modulo por lo cual se copia ese mismo modelo para WTOM obteniendo el mismo resultado.

3.4.4 Prueba de funcionamiento KMi

Puesto que no se encontró la herramienta que modulariza ontologías con la técnica KMi la prueba solo se realiza basándose en la información científica de esta la cual a grandes rasgos propone tener en cuenta en el nuevo modulo los conceptos y las relaciones de mayor jerarquía en la ontología, por lo cual basados en ese concepto se realiza la implementación de esta técnica obteniendo el resultado propuesto para esta.

3.5 Resultado de la modularización en formato OWL

En las herramientas originales al intentar exportar el resultado de la modularización en OWL se encontró que no todas las herramientas realizaban ese proceso (por ejemplo en la herramienta de escritorio Pato la cual entrega los módulos en formato .net los cuales listan los conceptos y las relaciones del nuevo modulo mas no lo entrega en un formato de ontología). En WTOM con aprovechando la potencia de la librería Jena convierte los vectores tanto de conceptos como de relaciones que se generan en la modularización en archivos OWL este proceso se realiza con todas las técnicas implementadas en WTOM, por lo cual se da un parte de cumplimiento de este objetivo al garantizar que el resultado de la modularización es una ontología en formato OWL.

Capítulo 4

Aspectos del desarrollo de software

En el presente capítulo se expone el proceso de desarrollo de la aplicación y se documenta la misma. Se discuten y presentan las diferentes tecnologías usadas para la implementación de la herramienta y se realiza una breve reseña de dichas tecnologías.

El desarrollo realizado en el presente trabajo de grado se llevó a cabo siguiendo directrices de la metodología de desarrollo ágil XP[23]. La metodología ágil XP define reglas para la gestión de las iteraciones en el proceso desarrollo. Estas reglas están definidas como: planeación, gestión, codificación, diseño y pruebas.

Se eligió la metodología de desarrollo XP por las características del proyecto de software, la necesidad de un prototipado constante y la calidad cambiante de los requerimientos del proyecto de software[23]. También se usa XP a causa del reducido número de desarrolladores, en este caso dos personas, situación que va muy bien con el principio de "Programación en parejas" propuesto por la metodología.

4.1 Planeación

XP define como regla fundamental la planeación constante y completa durante el proceso de desarrollo[23]. Esta planeación tiene como artefacto fundamental la redacción de historias de usuario. Para WTOM se elaboraron historias iniciales que describen las funcionalidades principales de la aplicación de forma modular. Igualmente, cada módulo y sus historias de usuarios fueron definidas de acuerdo a las necesidades presentadas.

4.1.1 Historias de Ususario iniciales

Puesto que la metodología seleccionada para la implementación del proyecto (XP) propone la creación de historias de usuario se generan las siguientes:

Historia de Usuario	
Id: 27354369	Nombre: Técnica Swoop
Usuario: cliente	Prioridad: 3 puntos
Descripción: Como usuario quisiera que la herramienta ofrezca una interfaz en la cual pueda realizar la modularización con los pasos de la técnica Swoop, la cual pertenece al enfoque de particionamiento de ontología.	
Observaciones:	

Figura 4.1: H.U. Técnica Swoop.

Historia de Usuario	
Id: 27354421	Nombre: Técnica Prompt
Usuario: Cliente	Prioridad: 3 puntos
Descripción: Como usuario quisiera que la herramienta ofrezca una interfaz en la cual pueda realizar la modularización con los pasos de la técnica Prompt, la cual pertenece al enfoque de extracción de módulo.	
Observaciones:	

Figura 4.2: H.U. Técnica Prompt.

Las demás historias de usuario se adjuntan en el Anexo D: Historias de usuario para la herramienta web de modularización de ontologías WTOM.

4.1.2 Velocidad del proyecto

Se dividirá el proyecto en 4 iteraciones como se muestra en la figura 4.3

	Iteración 1	Iteración 2	Iteración 3	Iteración 4
Horas	108	108	108	108
Semanas	6	6	6	6
Horas semanales	18	18	18	18
Historias de Usuario	5	4	5	4

Figura 4.3: Velocidad del Proyecto.

4.1.3 División del proyecto

El proyecto fue dividido en 4 interacciones cada una de 6 semanas. Cada iteración desarrolla las siguientes historias de usuario:

Iteración 1(cuadro 4.1):

Historia de Usuario	Id
Registrar usuario	01
Login usuario	02
Carpeta de Usuario	03
Carga local	04
Carga desde repositorio	05

Cuadro 4.1: Iteración 1

Se codifican las historias de usuario propuestas para esta iteración desarrollando en el Framework GWT y utilizando el Framework de ontologías Jena. Se implementa el registro de usuario en la cual la herramienta en una ventana de registro pide los siguientes datos: Nombre, Apellido, Apodo, email y contraseña, en el cual cuando el usuario termina de introducir estos datos estos se guardan en la base de datos llamada WTOM en la tabla usuarios que esta implementada en MySQL.

Se implementa el Login de usuarios en el cual se le solicita al usuario el apodo y la contraseña y la herramienta busca en la base de datos llamada WTOM en la tabla usuarios si el Apodo y contraseña suministradas por el usuario son correctas. Se implementa la carga local de la ontología en el cual se utiliza el Framework de ontologías Jena. Se solicita la dirección de la ontología en el disco y por medio de Jena se carga la ontología en la herramienta y se guarda el archivo en el servidor. Se implementa la carga desde repositorio de la ontología en el cual se utiliza el Framework de ontologías Jena. Se solicita la url de la ontología en el repositorio y por medio de Jena se carga la ontología en la herramienta y se guarda el archivo en el servidor.

Iteración 2(cuadro 4.2):

Historia de Usuario	Id
Técnica Pato	06
Técnica Promp	07
Elección de Enfoque	08
Elección de Técnica	09

Cuadro 4.2: Iteración 2

Se codifican las historias de usuario propuestas para esta iteración desarrollando en el Framework GWT y utilizando el Framework de ontologías Jena, Para Implementar la técnica Pato se realiza la búsqueda de la herramienta de modularización de ontologías Pato (Entorno de escritorio),

de la cual se encuentran los archivos .class a los cuales se les descompila con el descompilador jd-gui y luego de obtener todas los .java de la herramienta (23 clases en total y unas de ellas con más de 1000 líneas de código) se entra a depurar y a tener total control sobre este código.

Igualmente con Prompt se carga la ontología a trabajar con el Framework Jena con lo cual se listan para el usuario los conceptos y las relaciones de la ontología, el usuario procede a seleccionar que conceptos y relaciones quiere tener en el nuevo módulo y la herramienta realiza el proceso.

En cuanto a la elección de enfoque y de técnica se realiza un panel llamado PanelModularizacion en el cual se puede elegir uno de los dos enfoques y después de que el usuario elige el enfoque este muestra las técnicas relacionadas a cada uno para que el usuario decida con cual quiere modularizar su ontología.

Iteración 3(cudro 4.3):

Historia de Usuario	Id
Técnica Swoop	10
Técnica KMi	11
Ingreso de datos	12
Reporte modularización	13

Cuadro 4.3: Iteración 3

Para implementar la primera historia de usuario de la iteración 3 la cual se refiere a la técnica de modularización Swoop se utilizo la tecnología SVN la cual nos evita la transcripción manual de las clases del proyecto en un editor (proceso que no se pudo llevar a cabo con la técnica Pato al no existir el proyecto en SVN), por esto se agiliza la depuración de la herramienta y se procede a pasar a la implementación de la técnica separándola totalmente de la interfaz (implementada en Java con la librería Swing).

Con relación a KMi al no conocer la herramienta la cual implementa esta técnica, para la implementación de esta se fundamento en la literatura de esta en la cual se propone sacar tanto las clases principales de una ontología como sus relaciones y efectivamente fue posible obtener este resultado con el Framework de ontologías Jena.

En cuanto al ingreso de datos la única técnica que permite iteración con el usuario es Prompt en la cual el usuario selecciona tanto los conceptos como las relaciones que quiere en el nuevo módulo. Y con respecto a los reportes de modularización depende el enfoque se da un reporte, para el enfoque de particionamiento como se generan varios módulos, el reporte que se entrega se listan los conceptos y las relaciones de cada módulo y el consolidado de conceptos y relaciones por cada módulo. En cuanto a el enfoque de extracción de módulo al solo generar un módulo se listan los conceptos y las relaciones del módulo generado y el total de estos datos.

Iteración 4(cuadro 4.4):

Historia de Usuario	Id
Visualización ontología	14
Formato de ontología	15
Descarga de ontología	16
Cerrar sección	17

Cuadro 4.4: Iteración 4

Para realizar la visualización del modulo o los módulos generados por la modularización se uso la librería Jung representando los conceptos como nodos y las relaciones como aristas direccionadas.

En cuanto al formato de devolución del módulo en formato OWL se utiliza el framework de gestión de ontologías Jena con el cual se crea el modelo OWL y con vectores de conceptos y relaciones se escriben los módulos en formato OWL tal como lo propone la W3C.

Para la descarga de los módulos en formato OWL se comprime la carpeta donde el modulo o los módulos se almacenan para que se descargue el archivo .ZIP con los módulos OWL.

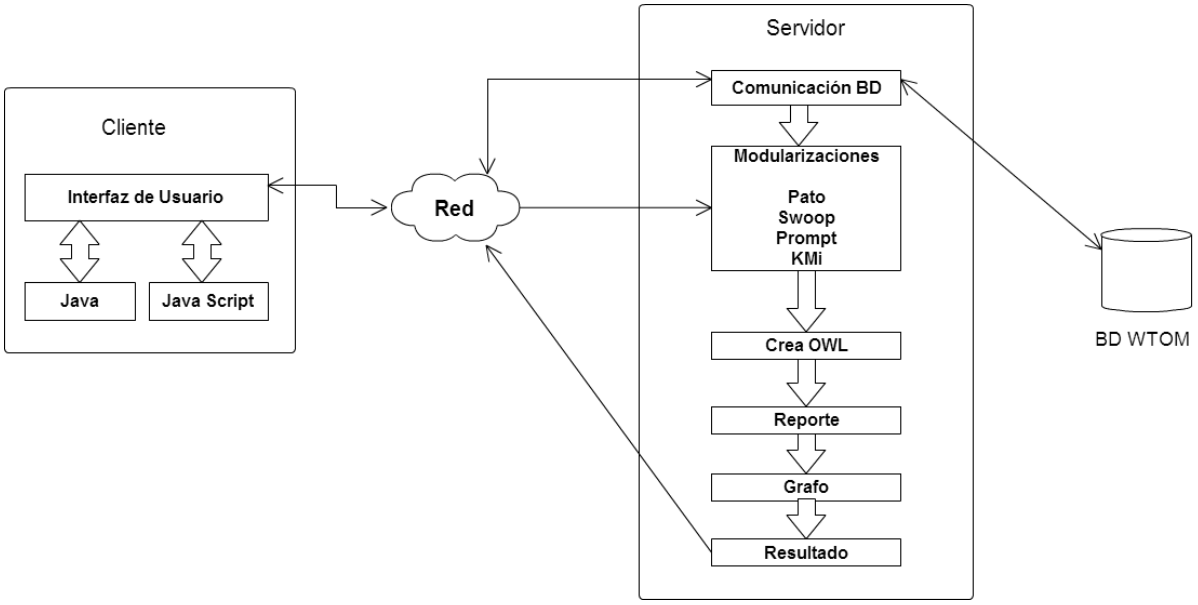

Figura 4.4: Arquitectura del Sistema.

4.2 Diseño

4.2.1 Arquitectura

La arquitectura (figura 4.4) propuesta para la herramienta web para la modularización de ontologías WTOM es una arquitectura basada en el modelo cliente servidor en la cual se propone que el usuario desde la vista la cual se encuentra en el cliente cargue sus datos (ontologías, proyectos) al servidor en donde estarán también los algoritmos de modularización.

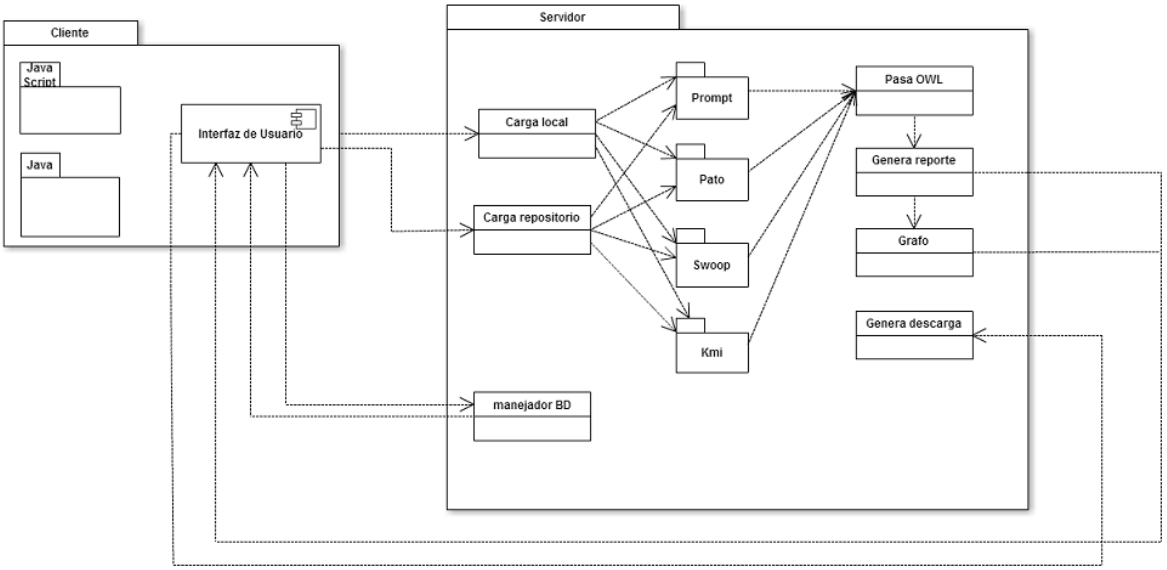

Figura 4.5: Diagrama de Paquetes.

4.2.2 Diagrama de Paquetes

La metodología de desarrollo *XP* es flexible frente a los artefactos de diseño a ser utilizados en esta etapa.
Para la aplicación web para la modularización de ontologías se propone un conjunto de paquetes (figura 4.5) con la siguiente organización:
Paquete Cliente en el cual se encontrara todo lo que tenga que ver con la interfaz del sistema la cual estará en la parte del usuario (Java, Java Script).

Paquete Servidor en el cual estará todo lo que corresponde a los algoritmos para la modularización de ontologías, carpetas de proyectos de usuario y cargas de ontologías (Java, Modularización, Pato, Prompt; KMi, Swoop, Carpeta usuario, Carga Ontología).

Paquete Compartida en el cual ira todos los datos que deben compartir el usuario y el servidor como los manejos de la base de datos para el registro, login, dirección de carpeta de usuario (Java, Java Script, Consulta, MySql).

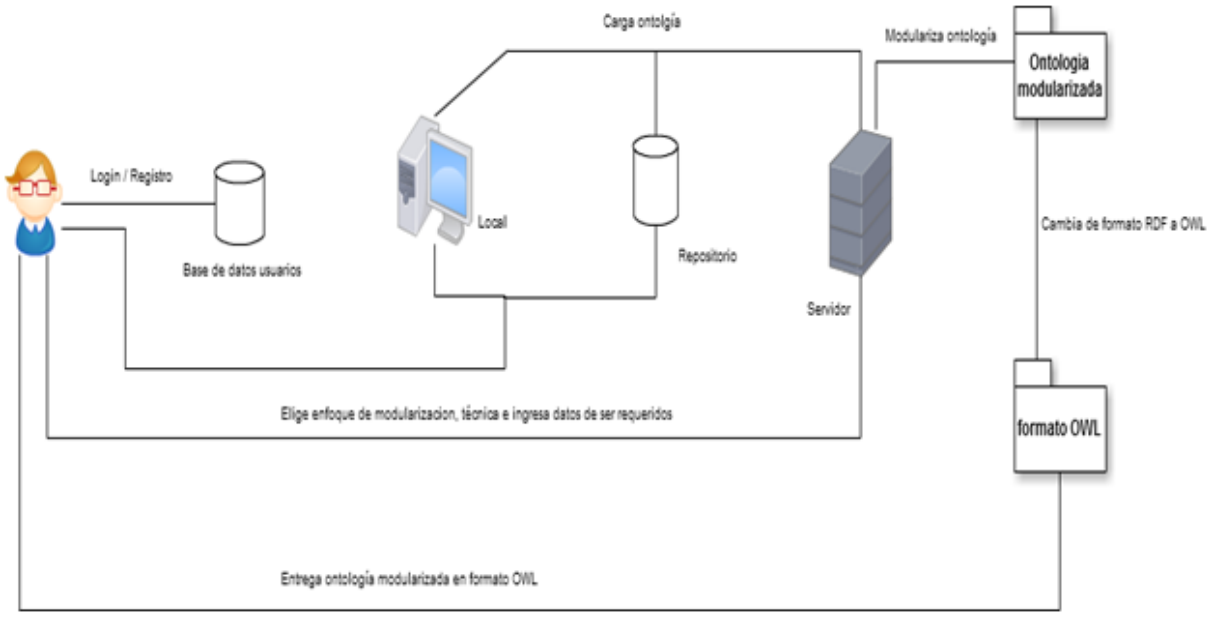

Figura 4.6: Metafora del Sistema.

4.2.3 Metáfora del Sistema

El usuario digita el dominio con el cual se identifica la aplicación en la web, en este se despliega un índex de bienvenida donde para utilizar la aplicación debe identificarse y si no tiene cuenta aún deberá registrarse. Después de haber pasado por este proceso el usuario tendrá acceso a un espacio de trabajo propio donde cargara la ontología que desea trabajar, tiene la opción de cargar una ontología desde repositorio o desde su equipo. Después de que haya elegido la ontología a trabajar podrá elegir el enfoque de modularización y según esté la técnica de modularización, cada enfoque y técnica tendrá una breve explicación de su funcionamiento. Después de que el usuario elija la técnica según esta se hará la modularización automáticamente o si la técnica lo requiere se le solicitarán algunos datos al usuario. Después de que se modulariza la ontología, el usuario recibe un informe de la modularización de los nuevos módulos o la nueva ontología (según sea el caso), una representación grafica de los módulos y descargar las ontologías en formato OWL. El usuario podrá volver acceder a su espacio y de allí repetir el proceso cuanto más lo requiera. Cuando el usuario desee salir de la aplicación simplemente cerrará sección.

4.3 Codificación

La codificación en *XP* se lleva a cabo por medio de un proceso de prototipado constante que genera una versión, módulo o componente de software por cada iteración del proceso de desarrollo. En esta sección se discuten las particularidades de la codificación y el método de programación usado.

Para la codificación se uso el framework GWT el cual permite desde la parte del servidor utilizar métodos y librerías complejas de Java lo cual se acomodaba a la realización de este proyecto puesto que se requería utilizar librerías complejas como Jena, Sesamo semantic Web, API OWL, Jung. Por lo cual fue necesaria una amplia investigación y aprendizaje del frameWork GWT.

Igualmente el FrameWork para la gestión de ontologías Jena fue de suma importancia en la codificación de la Herramienta WTOM, el cual es utilizado en la carga de las ontologías a trabajar, para la generación de los reportes, para la realización de la modularización de KMi y la creación de las ontologías generadas por la modularización en formato OWL.

El método de programación utilizado fue el enfoque orientado a objetos.

4.4 Pruebas

Este capítulo muestra el proceso de pruebas sobre los resultados obtenidos con relación a los objetivos planteados y las historias de usuario. Para probar las funcionalidades de la aplicación se llevo a cabo pruebas de aceptación.

4.4.1 Pruebas de aceptación y Comparación

Para éste trabajo de grado se decidierón probar los casos de prueba básados en las historias de usuarios que requieren mayor trabajo de procesamiento; Carga local, carga desde repositorio, técnica Pato, técnica Swoop, técnica Prompt y técnica KMi. Las demás historias de usuario se considerarón de prioridad baja debido a que requieren poco trabajo de maquina entre ellos se encuentra: Registrar usuario, login usuario, carpeta de usuario, eleccion de enfoque, elección técnica, ingreso de datos, reporte modularización, visualización de ontología, formato de ontología, descarga de ontología, cerrar sección.

ID de la Prueba	04 Carga local			
Objetivo de la prueba	Cargar una ontología a la aplicación WTOM que se encuentra almacenada en el disco duro del equipo y verificar que esta quede almacenada en la carpeta del servidor para que esta pueda ser utilizada posteriormente para la modularización			
Software requerido	Servidor de aplicaciones apache y web browser			
Datos de prueba	ontología ooe.owl			
Procedimiento de la prueba	En el panel de carga local se busca el archivo ooe.owl se selecciona y se oprime el botón cargar, esperando un mensaje de la aplicación que informe que la ontología se cargo correctamente			
Resultado esperado	Que la ontología quede almacenada en la carpeta del servidor			
Resultado obtenido	la ontología se cargo correctamente en la carpeta del servidor			
Aprobado	SI	X	NO	

Figura 4.7: H.U. 04 Carga local.

ID de la Prueba	05 Carga desde repositorio			
Objetivo de la prueba	Cargar una ontología a la aplicación WTOM que se encuentra almacenada en un repositorio en internet y verificar que esta quede almacenada en la carpeta del servidor para que esta pueda ser utilizada posteriormente para la modularización			
Software requerido	Servidor de aplicaciones apache y web browser			
Datos de prueba	ontología ooe.owl disponible en la web			
Procedimiento de la prueba	En el panel de carga desde repositorio se pega la url donde se encuentra almacenada la ontología y se oprime el botón cargar, esperando un mensaje de la aplicación que informe que la ontología se cargo correctamente			
Resultado esperado	Que la ontología quede almacenada en la carpeta del servidor			
Resultado obtenido	la ontología se cargo correctamente en la carpeta del servidor			
Aprobado	SI	X	NO	

Figura 4.8: H.U. 05 Carga desde repositorio.

ID de la Prueba	06 Técnica Pato			
Objetivo de la prueba	Verificar que la herramienta WTOM modulariza con la técnica Pato del enfoque de particionamiento de ontología			
Software requerido	web browser			
Datos de prueba				
Procedimiento de la prueba	se elige modularizar la ontología con la técnica Pato y se obtiene el resultado de este proceso			
Resultado esperado	módulos en formato OWL generados por la modularización de la ontología, reporte de la modularización y visualización de los módulos generados			
Resultado obtenido	módulos en formato OWL generados por la modularización de la ontología, reporte de la modularización y visualización de los módulos generados			
Aprobado	SI	X	NO	

Figura 4.9: H.U. 06 Técnica Pato.

ID de la Prueba	07 Técnica Prompt			
Objetivo de la prueba	Verificar que la herramienta WTOM modulariza con la técnica Prompt del enfoque de extracción de módulo			
Software requerido	web browser			
Datos de prueba				
Procedimiento de la prueba	se elige modularizar la ontología con la técnica Prompt, se seleccionan los conceptos y las relaciones que se desean en el nuevo módulo y se obtiene el resultado de este proceso			
Resultado esperado	módulo en formato OWL generado por la modularización de la ontología, reporte de la modularización y visualización del módulo generado			
Resultado obtenido	módulo en formato OWL generado por la modularización de la ontología, reporte de la modularización y visualización del módulo generado			
Aprobado	SI	X	NO	

Figura 4.10: H.U. 07 Técnica Prompt.

ID de la Prueba	11 Técnica KMi			
Objetivo de la prueba	Verificar que la herramienta WTOM modulariza con la técnica KMi del enfoque de extracción de módulo			
Software requerido	web browser			
Datos de prueba				
Procedimiento de la prueba	se elige modularizar la ontología con la técnica KMi y se obtiene el resultado de este proceso			
Resultado esperado	módulo en formato OWL generado por la modularización de la ontología, reporte de la modularización y visualización del módulo generado			
Resultado obtenido	módulo en formato OWL generado por la modularización de la ontología, reporte de la modularización y visualización del módulo generado			
Aprobado	SI	X	NO	

Figura 4.11: H.U. 11 Técnica KMi.

ID de la Prueba	11 Técnica KMi			
Objetivo de la prueba	Verificar que la herramienta WTOM modulariza con la técnica KMi del enfoque de extracción de módulo			
Software requerido	web browser			
Datos de prueba				
Procedimiento de la prueba	se elige modularizar la ontología con la técnica KMi y se obtiene el resultado de este proceso			
Resultado esperado	módulo en formato OWL generado por la modularización de la ontología, reporte de la modularización y visualización del módulo generado			
Resultado obtenido	módulo en formato OWL generado por la modularización de la ontología, reporte de la modularización y visualización del módulo generado			
Aprobado	SI	X	NO	

Figura 4.12: H.U. 10 Técnica Swoop.

Técnica	Herramienta de escritorio	WTOM	Diferencias
Pato	Al modularizar la ontología ooe.owl en la herramienta de escritorio Pato se obtienen 2 módulos producto de la modularización	en WTOM al modularizar la ontología ooe.owl en WTOM se obtienen 6 módulos resultado de la modularización	Las diferencias se dan puesto que la herramienta de escritorio tiene implementado un método de optimización de la ontología que no se propone en la documentación científica de la técnica, por lo cual la modularización realizada por WTOM es correcta y haciendo una comparación entre los módulos generados por ambas herramientas son equivalentes (puesto que los módulos generados por WTOM de más son muy pequeños, 2 conceptos y dos relaciones).

Figura 4.13: H.U. 10 Comparación Pato.

Técnica	Herramienta de escritorio	WTOM	Diferencias
Swoop	Al modularizar la ontología aminoAcid.owl en la herramienta Swoop de escritorio se obtienen dos módulos cada uno con 28 conceptos y 16 relaciones y 18 conceptos y 0 relaciones.	Al modularizar la ontología aminoAcid.owl en la herramienta WTOM se obtienen dos módulos cada uno con 28 conceptos y 16 relaciones y 18 conceptos y 0 relaciones.	Ninguna

Figura 4.14: H.U. 10 Comparación Swoop.

Técnica	Herramienta de escritorio	WTOM	Diferencias
Prompt	La herramienta Protege con el plugin de Prompt le permite al usuario elegir cuáles conceptos y cuáles relaciones requiere en el nuevo módulo.	WTOM con la técnica Prompt le permite al usuario elegir cuáles conceptos y cuáles relaciones requiere en el nuevo módulo.	En Protege la interfaz gráfica tiene mayor cantidad de detalles.

Figura 4.15: H.U. 10 Comparación Prompt.

Técnica	Herramienta de escritorio	WTOM	Diferencias
KMi	Según la documentación científica esta técnica elige los conceptos principales en jerarquía de la ontología y las relaciones que existen entre estas	al modularizar alguna ontología en WTOM la herramienta elige los conceptos principales en jerarquía de la ontología y las relaciones que existen entre estas	KMi solo se implementa en la herramienta WTOM ya que esta solo se encuentra su teoría más no su implementación mediante una herramienta propiamente dicha.

Figura 4.16: H.U. 10 Comparación KMI.

Capítulo 5

Conclusiones y trabajos futuros

En este capítulo se presentan las conclusiones del trabajo de grado con respecto a los objetivos cumplidos y en general al proceso de desarrollo de la herramienta. Finalmente se presentan algunas líneas de trabajo futuro.

5.1 Conclusiones

Se ha presentado WTOM herramienta web para la modularización de ontologías la cual tiene como finalidad facilitar la modularización de ontologías fomentando y facilitando la investigación en el campo de la web semántica, esta herramienta implementa los 2 enfoques propuestos para la modularización de ontologías y dos técnicas de cada enfoque, Pato y Swoop por particionamiento de ontología Prompt y Kmi del enfoque de Extracción de modulo, de igual forma también ofrece reportes de la modularización y se entrega los módulos modularizados en formato estándar de ontologías OWL tal y como lo propone la W3C.

En el presente trabajo de grado se propusieron los siguientes objetivos específicos, de los cuales se concluye:

1. Implementar las técnicas PATO y SWOOP del enfoque de particionamiento de ontología.

Se implementan las Técnicas de Particionamiento de ontología basadas en las herramientas existentes y desarrolladas en Java obteniendo la mejora de que WTOM modulariza ontologías con Pato y Swoop que las herramientas de escritorio no modularizan, por lo cual se puede concluir que en WTOM puede superar notablemente a estas herramientas de escritorio en su funcionamiento, WTOM modulariza con las técnicas de enfoque de particionamiento de ontología permitiendo descargar los módulos creados por la modularización, entregando un informe completo de estos módulos y una representación grafica de los módulos creados.

2. Implementar las técnicas KMI y PROMPT del enfoque de extracción de modulo.

Se implementan las técnicas del enfoque de extracción de módulo, Prompt se implementa basándose en la herramienta Protege en la cual se seleccionan los conceptos y las relaciones que

el usuario requiere en el nuevo módulo por lo cual se al cargar la ontología se le da la opción de al usuario de seleccionar tanto los conceptos como las relaciones que se van a escribir en el nuevo módulo. Con respecto a la técnica KMi no se fue posible conocer la herramienta de modularización que implementa esta técnica por lo cual se implemento basándose en la información científica que había de esta en la cual decía que el módulo se creaba con los conceptos y las relaciones de mayor jerarquía de la ontología original. Por lo cual se concluye que WTOM modulariza con las técnicas del enfoque de extracción de módulo con las cuales después de la modularización WTOM ofrece un reporte completo del modulo creado, una representación grafica de la nueva ontología y la opción de descárgala en formato OWL.

3. Ofrecer un reporte del resultado de la modularización según la(s) técnica(s) utilizada(s).

WTOM realiza un reporte de los módulos creados después de una modularización (sin importar la técnica ni enfoque) en el cual se informa los nombres de los conceptos del modulo, los nombres de la relación del modulo y se reporta un total de conceptos y relaciones de los módulos creados, por lo cual en comparación con las herramientas de escritorio (solo la herramienta Swoop ofrece un reporte de modularización en el cual solo informa el total de conceptos y relaciones de cada módulo nuevo), se puede concluir que WTOM ofrece un reporte muy completo de la modularización realizada facilitándole al usuario la elección del módulo que más le conviene elegir para su aplicación de web semántica.

4. Exportar el resultado de la modularización en formato OWL.

La herramienta WTOM después de realizar una modularización crea los módulos en el formato estándar para las ontologías OWL propuesto por la W3C. Después de crear los módulos en OWL da la opción al usuario de descargar los módulos OWL. En comparación a otras herramientas de modularización de ontologías (solo Swoop permite descargar los módulos generados en formato OWL) WTOM permite descargar la ontología resultante de cualquier técnica de modularización en formato OWL por lo cual se puede concluir que WTOM presenta superioridad en comparación a las herramientas existentes en cuanto a este tema.

5. Probar que la herramienta funcione según los enfoques y técnicas de modularización de ontologías.

Basándose en las herramientas de modularización de ontologías y sobretodo en la literatura científica que existe sobre los enfoques y técnicas de modularización de ontologías se puede concluir que se comprobó que las técnicas implementadas en WTOM cumplieran con los lineamientos científicos que proponen las técnicas por lo cual se concluye que las técnicas implementadas en WTOM funcionan según la literatura científica de estas. Igualmente se realizaron pruebas de aceptación a la herramienta concluyendo que WTOM funciona perfectamente según las historias de usuario definidas para la implementación de la herramienta.

6. Determinar las ontologías con las que se puede trabajar.

Basándose en el hecho de que la herramienta WTOM es una aplicación web para la modula-

rización de ontologías, se concluye que debido al proceso de carga de las ontologías, ya sea de forma local o desde repositorio, es recomendable el no uso de ontologías que superen los 10 mil términos o clases, en caso de que el usuario requiera modularizar una ontología superior a la recomendada, es preferible el uso de las aplicaciones de escritorio.

7. WTOM y Jena.

En el proceso de desarrollo se descubrió que el framework GWT no es compatible con librerías que usen herramientas de creación de log, por lo cual se recomienda esto a la hora de trabajar en GWT (Librería Jena 6 usa log por el cual esta librería no es compatible con GWT).

5.2 Trabajos futuros

En el desarrollo de la herramienta se identificaron diferentes líneas de trabajo que extienden o complementan *WTOM*. Como trabajos futuros se consideran:

- **Generalizar formato de ontologías para modularizar:**. Puesto que las herramientas existentes y WTOM no modulariza el total de los formatos de ontologías (en ocasiones no modularizan el total de las ontologías OWL y RDF), se propone trabajar la herramienta para que se pueda cargar y trabajar con las técnicas una ontología sin importar su formato.
- **Arreglar visualización de los reportes de modularización:**. Se propone en una nueva intervención a la herramienta WTOM poder organizar la forma en cómo se muestran los reportes de la modularización, de una forma que sea más fácil y entendible para el usuario.
- **Arreglar visualización de los módulos generados por la modularización:**. Puesto que se podría mejorar la visualización de los grafos de los módulos generados por la modularización en cuanto a organización, nombre de conceptos y relaciones y tal vez interacción del usuario con el grafo.
- **Implementar optimización de la técnica Pato**. La herramienta de escritorio Pato implementa un paso después de la modularización, el cual es: optimización de la modularización. El cual se encarga de unir módulos pequeños a módulos más grandes obteniendo así menos módulos. Sería recomendable en un futuro implementar esta optimización en la herramienta WTOM la cual no se implemento ahora porque este paso no está se menciona en los 3 pasos fundamentales de la modularización con la técnica Pato.

Referencias

[1] Adolfo Lozano Tello. Ontologías en la web semántica. *Área de Lenguajes y Sistemas Informáticos. Departamento de Informática. Escuela Politécnica. Universidad de Extremadura. España.*, 2001.

[2] Dan Brickley and R.V. Guha. Rdf vocabulary description language 1.0: Rdf schema. *Technical report, W3C Recommendation*, 2011.

[3] Frank van Harmelen.] Deborah L. McGuinness. Owl web ontology language. *W3C Recommendation.*, 2004.

[4] Heiner Stuckenschmidt and Michel Klein. Integrity and change in modular ontologies. *Vrije Universiteit Amsterdam*, 2003.

[5] Heiner Stuckenschmidt Mathieu d'Aquin, Anne Schlicht and Marta Sabou. Ontology modularization for knowledge selection: Experiments and evaluations. *Knowledge Media Institute (KMi), The Open University, Milton Keynes, UK, University of Mannheim, Germany*, 2007.

[6] Evren Sirin Aditya Kalyanpur Bernardo Cuenca Grau, Bijan Parsia. Automatic partitioning of owl ontologies using e-connections. *University of Maryland, College Park, MD, USA*, 2005.

[7] Heiner Stuckenschmidt and Michel Klein. Structure-based partitioning of large concept hierarchies. *Vrije Universiteit Amsterdam*, 2004.

[8] Natalya F. Noy and Mark A. Musen. Specifying ontology views by traversal. *Stanford Medical Informatics, Stanford University,251 Campus Drive, x-215, Stanford, CA 94305, USA*, 2004.

[9] Yevgeny Kazakov Bernardo Cuenca Grau, Ian Horrocks and Ulrike Sattler. Just the right amount: Extracting modules from ontologies. *The University of Manchester,School of Computer Science*, 1998.

[10] Heiner Stuckenschmidt Mathieu d'Aquin, Anne Schlicht and Marta Sabou. Criteria and evaluation for ontology modularization techniques. *Knowledge Media Institute (KMi), The Open University, Milton Keynes, UK, University of Mannheim, Germany*, 2009.

[11] servogrid. Toolreview:theontmatannotator. *www.servogrid.org/slide/GEM/SW/Tool*, [consulta: diciembre del 2011].

[12] Benjamins R. Fensel D. Gomez A. Knowledge management through ontologies. *Technical University of Madrid, Department of Articial Intelligence.*, 1998.

[13] Semantic web portal ontology. Documentation for swportal ontology. *http://swportal.deri.org/ontologies/swportal*, [consulta: diciembre del 2011].

[14] Rubén Lara Hernández Ying Ding Sung-Kook Han Dieter Fensel Holger Lausen, Michael Stollberg. Semantic web portals – state of the art survey. *Next Web Generation Group, IFI – Institute for Computer Science University of Innsbruck, Austria*, 2004.

[15] W3C. El consorcio world wide web publica las recomendaciones rdf y owl. *http://www.w3c.es/prensa/2004/nota040210.html.*, [consulta: diciembre del 2011].

[16] Silvia Esther Sánchez López. Modelo de indexación de formas en sistemas vir basado en ontologías . *Universidad de las Américas, Escuela de Ingeniería y Ciencias, Puebla.*, 2007.

[17] Schreiber G. Wielinga B. Wielemaker, J. Prolog-based infrastructure for rdf: performance and scalability. *In Fensel, D., Sycara, K., Mylopoulos, J., eds.: The Semantic Web - Proceedings ISWC'03, Sanibel Island, Florida, Berlin, Germany.*, 2003.

[18] Mrvar A. Batagelj, V. Pajek - analysis and visualization of large networks. *In Jnger, M., Mutzel, P., eds.: Graph Drawing Software.*, 2003.

[19] V. Batagelj. Analysis of large networks - islands. *Presented at Dagstuhl seminar 03361: Algorithmic Aspects of Large and Complex Networks.*, 2003.

[20] Matthias Hert. Semantic web engineering. *University of Zurich, Departament of Informatics, Suiza.*, 2010.

[21] Javier Mejías Real. Manual de gwt. *http://bitacorasigloxxi.files.wordpress.com/2008/11 /breveguiagwt.pdf*, 2008.

[22] standford. protege main page. *http://protege.stanford.edu/*, [consulta: abril del 2013].

[23] Patricio Letelier y Mª Carmen Penadés. Métodologías ágiles para el desarrollo de software: extreme programming (xp). *Universidad Politécnica de Valencia, España.*, 2004.