ASISTENTE PARA LA CREACIÓN DE CONSULTAS SEMÁNTICAS. APLICACIÓN A LA FABRICA DE IDEAS DE PROYECTOS DE CÓDIGO ABIERTO UBUNTU IDEAS

Autor: D. Geovanny Poveda Cardona

Tutor: D. Carlos Ángel Iglesias Fernández

Departamento: Departamento de Ingeniería Telemática (Grupo de Sistemas Inteligentes)

Tribunal nombrado por la Comisión de Ordenación Académica de la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universidad Politécnica de Madrid el día 18 de Julio de 2011.

Presidente: D. Gregorio Fernández

Vocal: D. Carlos Ángel Iglesias Fernández

Secretario: D. Ignacio Soto

Realizado el acto de defensa y lectura del Proyecto Fin de Máster el día 27 de Julio de 2011.

Calificación:

EL PRESIDENTE EL VOCAL

EL SECRETARIO

UNIVERSIDAD POLITÉCNICA DE MADRID
ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

TRABAJO FIN DE MASTER

ASISTENTE PARA LA CREACION DE CONSULTAS SEMANTICAS. APLICACIÓN A LA FABRICA DE IDEAS DE PROYECTOS DE CÓDIGO ABIERTO UBUNTU IDEAS

GEOVANNY POVEDA CARDONA

2011

Resumen.

El objetivo de este proyecto Fin de Máster, es proponer el diseño de un modelo de búsqueda semántica que basado en tecnologías como SPARQL, RDF y ARC2 permitan realizar procesos de consulta auto-asistidos. La introducción de este tipo de conceptos permitirá alcanzar escenarios de búsqueda más agiles y dinámicos, ya que los usuarios no sólo podrán adaptar y configurar sus propias fuentes de información, sino que también tendrán la capacidad de interactuar de una forma más simple con los conceptos que se manejan alrededor de cada dominio de información.

Con este objetivo, se ha desarrollado un componente de software que basado en herramientas como ARC2, SPARQL, PHP y EVOC, han permitido definir modelos de búsqueda fácilmente adaptables a sistemas gestores de ideas semánticos como GI2MO Ideas Stream **[GI2IS].** El propósito de usar un gestor de Ideas en el componente, se fundamenta en el concepto de fuente semántica de información. Dado que mediante el uso de Ideas Stream será posible realizar búsquedas que usen un dominio de información lo suficientemente amplio en términos de la cantidad de relaciones semánticas que este expone (Ideas – Comentarios – Soluciones, etc.).

Una característica adicional que se contemplo en el diseño de este componente de software llamado GI2SE, estuvo relacionada con la posibilidad de ofrecer opciones configuración sobre los puntos de acceso final SPARQL. Este tipo de opciones hacen que los usuarios tengan la posibilidad de usar GI2SE como una herramienta de búsqueda, en la que sea posible realizar consultas sobre dominios de información que no sólo se contemplen en el ámbito local donde ha sido desplegado el módulo. Por otra parte, el diseño de esta aplicación contemplo el uso de funcionalidades asociadas con la verificación de cambios en el contenido de las fuentes semánticas de información. La anterior funcionalidad permite que el resultado de las búsquedas sea conforme a los procesos de actualización que puedan llevarse acabo sobre el sistema gestor de Ideas.

El módulo de búsqueda hace uso de interfaces gráficas basadas en herramientas como Exhibit **[XEHBI]** y librerías propias del gestor de contenidos Drupal. El desarrollo metodológico adaptado para la realización del componente, sigue algunas de las recomendaciones hechas por el modelo unificado de lenguaje UML. La memoria describe las etapas, herramientas y procesos previos que fueron necesarios para la realización del módulo en mención. Igualmente describe el uso de las funcionalidades de búsqueda por medio de manuales de usuario previstos a manera de anexo.

Palabras claves: RDF, URL ENDPOINT SPARQL, Análisis de Datos, Buscador.

Agradecimientos.

Especial agradecimientos a todas las personas que me han brindado su apoyo y voz de aliento durante todo el tiempo de realización del Máster, profesores del Departamento y compañeros de clase.

En primer lugar, mis más grandes agradecimientos mi novia, mis padres y mis hermanos quienes han hecho que una vez más cumpla con otro objetivo en mi vida.

Igualmente, quiero agradecer al personal académico e investigativo del Grupo de Sistemas Inteligentes (GSI), quienes continuamente me brindaron todo el respaldo y apoyo necesario para cumplir con el trabajo fin de Máster. Especial agradecimiento a Carlos Ángel por la gran oportunidad que me brindo en hacer parte de su excelente grupo de investigación. A Adam Westerski por su infinita paciencia e incondicional apoyo en todos los temas de investigación y a todos los doctorandos del grupo, su apoyo y amistad fueron pieza clave.

1. Introducción.

El presente trabajo se desarrolla en el marco metodológico contemplado en el proyecto de investigación RESULTA (Red de Consultoría para la gestión de procesos y relaciones), el cual es una iniciativa propuesta por la Universidad Politécnica de Madrid en consorcio con empresas del sector de la consultoría en España. El consorcio propone el uso y creación de herramientas semánticas de colaboración que permitan mejorar las posibilidades de interacción e interoperabilidad en los procesos y relaciones de las organizaciones, mediante la incorporación de catalogación semántica, análisis estadístico de datos y búsqueda inteligente de datos a través de ontologías. En este sentido y teniendo en cuenta las líneas de investigación que se desarrollan en el proyecto, este trabajo fin de Máster se enmarca en el estudio y desarrollo de aplicaciones que estén en la línea de búsqueda, gestión e innovación del conocimiento.

1.1 Motivación.

Con el desbordante crecimiento de información que cada vez más se hace presente en la Web, diversas han sido las propuestas que se han lanzado con relación a la forma de consulta, acceso y manejo de los datos allí presentes. Si bien los esquemas existentes solucionan gran parte de estos problemas. Hoy por hoy y en vista a las condiciones de uso futuro que nos impone la Web, cada vez más se hace necesario pensar en herramientas y procedimientos que doten de inteligencia los actuales sistemas de información **[COLINA].** Conceptos como las tecnologías semánticas contextualizadas en el marco de la Web 2.0 y su aplicabilidad en temas como la búsqueda y la gestión de innovación, han despertado un especial interés en los actuales modelos de análisis de negocios de diferentes compañías.

En el caso de las búsquedas semánticas, indudablemente la Web 2.0 requiere la construcción de sistemas que permitan generar procesos de búsqueda más activos y personalizados. En los cuales sea posible generar relaciones entre conceptos pertenecientes a diferentes dominios de conocimiento. Esto permitirá obtener resultados más enriquecedores y verdaderamente adecuados a un contexto específico de información.

En este sentido, se hace necesario el uso de ontologías que no sólo permitan hacer descripciones semánticas de metadatos sino que también sean la base para adoptar mecanismos de intercambio e interoperabilidad de conocimiento entre los sistemas. Uno de los temas que recientemente ha despertado interés en este campo, es el desarrollo de ontologías relacionadas con sistemas

gestores de ideas, ya que estas contemplan un amplio dominio de información que no sólo sirve para ser utilizado en búsqueda recurrente de información, sino también para propósitos de gestión de Innovación **[MOOBR].** Además de lo anterior, los actuales esquemas de búsqueda exigen el uso de herramientas y librerías como ARC2 que permitan adaptar las condiciones de búsqueda tradicionalmente brindadas por medio de herramientas como SPARQL a ambientes tipo Web. De esta forma la interoperabilidad de información, se convierte en un tema clave para la construcción de nuevos modelos de servicios semánticos.

Sin duda, los gestores de ideas Semánticos representan una muy buena solución a la problemática de gestión de información, dado que estos contienen una amplia cantidad de datos que hace que temas como la inferencia de conocimiento pueda ser fácilmente aplicada sobre múltiples dominios de información. Sin embargo, temas como la escasa definición de reglas de sincronización y la ausencia de estructuras de mapeos que permitan realizar actualización automática de datos, contribuyen a que actualmente los sistemas gestores de ideas tengan limitantes de uso. Es por ello, que hoy por hoy el uso de este tipo de plataformas es un tema altamente cuestionado y muchas veces criticado, dado que la gran mayoría presentan un alto nivel de complejidad en la forma de estructurar, organizar y presentar la información.

Tal como expresaba Richard White, CEO de UserVoice, en el artículo recogido por el blog ReadWrite Web **[REAW].** La mayor parte de los empleados se encuentran a veces confusos frente al manejo de este tipo de plataformas **[TRA1].** De esta forma, el consumo de ideas y la interacción del usuario con la plataforma ocuparán posiciones críticas en la actualización de este tipo de sistemas. Un punto interesante a destacar, tiene que ver con la adopción semántica que muchas compañías actualmente esta implementado en sus Apis de desarrollo. Esto hace pensar que en un tiempo muy cercano será posible obtener repositorios globales de ideas interconectados entre sí. Un ejemplo de ello es el proyecto Drupal, el cual define una serie de métodos e interfaces de uso para la construcción de repositorio de ideas semánticos **[DRUPAL].**

En sentido el motivo de realización de este proyecto fin de Máster estará enfocado en suplir parte de las necesidades y deficiencias descritas anteriormente. De tal forma que basados en plataformas de gestión de ideas como Gi2MO Ideas Stream, sea posible diseñar componentes de software con interfaces dinámicas de búsqueda que tengan la capacidad de verificar automáticamente cambios en el contenido. De tal manera que conforme el sistema de gestión ideas se actualiza con nuevos conceptos (introducidos manual o automáticamente) los resultados de las búsquedas serán conforme a estos cambios.

Este conjunto de objetivos permitirá definir el contexto en el cual se desarrollara el presente proyecto y, ofrecerá una serie de alternativas mediante las cuales será posible definir nuevas propuestas de análisis y búsqueda de información semántica.

I.2 Estructura de la Memoria.

El presente documento está organizado a manera de capítulos, subcapítulos y secciones, la memoria cuenta con un total de cinco capítulos. La estructura se describe a continuación:

Capítulo I – Introducción.

Inicialmente a manera de introducción se expondrá el marco contextual en el que se desarrollara el proyecto, así como también los motivos por los cuales se realiza esta memoria.

Capítulo II – Estado del Arte.

Por otra parte, el capítulo II del documento describe el estado actual del arte de las tecnologías empleadas durante la realización del proyecto. En este capítulo se encuentra el marco teórico que sustenta las diferentes temáticas abordadas en este trabajo de fin de Máster. Inicialmente y con el fin de contextualizar al usuario en los conceptos que se desarrollaran a lo largo de la memoria, se ofrecerá una pequeña introducción respecto los principales conceptos que contempla la web Semántica. Así mismo, se ofrece una breve descripción acerca de los sistemas de gestión de ideas, técnicas de análisis de información basadas en representaciones de clúster y el uso de algoritmos tipo K-means. Igualmente se hará referencia al concepto de minería de datos como principal componente de alimentación de sistemas basados en representaciones ontológicas.

Capítulo III – Análisis.

El capítulo III del documento presenta la fase de análisis realizada en la construcción del componente de software propuesto. Este análisis establece las diferentes interacciones que se llevan a cabo entre el usuario y el sistema mediante la representación de casos de uso y otras metodologías.

Capítulo IV – Arquitectura.

Este capítulo describe la etapa de diseño e implementación realizada durante el desarrollo del componente de software semántico. Sobre este capítulo se hace una descripción global de la

arquitectura del servicio puesto en marcha. Allí se especifican las diferentes interacciones que se establecen entre los componentes de la arquitectura por medio de los diagramas más representativos del modelo estructural.

Capítulo V – Conclusiones y trabajos futuros.

El capítulo final de esta memoria, describe de manera general los resultados alcanzados durante la realización del proyecto. Además de ello propone una serie de conclusiones y líneas futuras de investigación para la mejora de la solución expuesta.

Los anexos de esta memoria son los siguientes:

- Manual de instalación y configuración de módulo GI2SE.
- Manual de usuario del módulo GI2SE.

Este anexo describe el proceso de instalación y configuración del módulo de búsqueda semántico llamado Gi2SE, sobre el gestor de contenidos Drupal.

2. Capítulo II Estado del Arte

2.1 Introducción.

El concepto de gestión de ideas hace referencia al conjunto de actividades que permiten promover y facilitar la adquisición de ideas de innovación sobre entornos tecnológicos que adaptan el concepto de social media. Basado en el concepto de inteligencia colectiva, proporciona ideas que al ser filtradas y procesadas por equipos de evaluación y análisis se convierten en elementos de valor relevante en la toma de decisiones y en la evolución de nuevas ideas o proyectos en las organizaciones **[BGLI]**. No obstante, este tipo de conceptos está empezando a percibirse como una herramienta en la que los sistemas de información no sólo encuentran mejoras y desarrollo de nuevos productos, procesos o servicios, sino que también recogen las propuestas relativas a la estructura de la organización, los procesos internos y las técnicas de gestión. Tanto así, que cerca del 66 % de las 1000 organizaciones más grande del mundo a 2010**[GDES]**, han adaptado el concepto de gestión de ideas mediante el desarrollo de sistemas gestores.

Por otra parte, es importante hacer referencia al concepto de minería de datos como el proceso mediante el cual es posible descubrir y extraer datos de diferentes fuentes de información, a través de herramientas de búsqueda e identificación de patrones y tendencias. Su principal objetivo es servir como elemento base en la toma de decisiones y estrategias en las organizaciones. Una característica relevante de este tipo de procesos y en especial el realizado en este proyecto, tiene que ver con el formato de salida obtenido. Básicamente corresponde a un formato RDF, el cual al estar basado en estructuras XML facilita la interoperabilidad de información entre las diferentes tecnologías utilizadas.

Igualmente, debe tenerse en cuenta el concepto de clúster como elemento clave en la representación visual del análisis realizado a los datos obtenidos en la etapa de extracción. Un proceso de clúster consiste en la división de datos pertenecientes a una muestra en grupos de objetos que preservan características de similitud. En este sentido, pretende medir la similitud entre objetos haciendo uso de diferentes formas de distancia como la distancia Euclídea y la de Manhatan **[GYAL]**. Sin duda, el elemento más importante de este trabajo fin de Máster es el concepto de asistente de consultas semántico. Este concepto representa el punto de atención central de este proyecto, dado que con la implementación de funcionalidades de estas

características, es posible lograr escenarios en los que los procesos de consulta puedan brindar mayor dinamismo y mayor grado de interoperabilidad con otros sistemas. En el caso de este proyecto, el termino dinamismo estará representado a través de los distintos procesos de configuración que se pueden realizar por parte de los usuarios. En lo que se refiere a interoperabilidad, este término se representa por medio de la opción de búsqueda remota que es posible alcanzar cuando se hace uso de librerías como ARC2 sobre tecnologías como SPARQL.

En otras palabras, la idea que se pretende alcanzar con este tipo de escenarios, es que en el momento en que un usuario realice una consulta sobre cualquier dominio de información, que en teoría no es de la incumbencia de una aplicación, estos sistemas deberán incorporar algoritmos que basados en criterios de similitud, contexto y uso, tengan la capacidad de enlazarse con otros dominios que contenga la información requerida. De esta forma, la consulta no retornara siempre la misma información de búsqueda (buscador convencional), sino que a través de uso de reglas de inferencia sea posible acercarse a lo que el usuario realmente desea.

Los siguientes apartados muestran una introducción acerca de las principales características de las herramientas de gestión empleadas y tomadas como referencia para la realización del proyecto. Igualmente ofrece una descripción acerca de las técnicas de extracción de datos utilizadas para la alimentación y generación de contenidos sobre sistemas de gestión semántica.

2.2 Web Semántica

2.2.1 Introducción

La Web Semántica se basa en la idea de añadir metadatos semánticos y ontológicos a la World Wide Web. Este concepto nace de la visión de Tim Bermers-Lee sobre la posibilidad de introducir información semántica en la Web, como respuesta a su omisión en el primer diseño de la misma **[TIMB2]**. La principal meta de la Web Semántica es cubrir las deficiencias que muestra la actual versión Web. La mayor parte de los documentos de la World Wide Web están escritos en HTML. Este lenguaje, basado en etiquetas, fue concebido con la idea de ofrecer herramientas para adecuar el aspecto visual del documento e incluir objetos multimedia en el texto, es decir, su objetivo era proporcionar características similares a lenguajes de maquetación convencionales como LaTeX. En su definición se omitió la posibilidad de incluir metadatos en el contenido, deficiencia en el lenguaje que es objeto de estudio de la Web Semántica.

La importancia de la inclusión de metadatos reside en la posibilidad de conectar la información

dentro de un documento. Así pues, se puede asignar propiedades al contenido o establecer relaciones entre contenidos independientes. A modo de ejemplo, se puede pensar en un catálogo de productos deportivos. Sin la posibilidad de añadir metadatos, la información extraída del documento por un máquina no tendrá ninguna relación, es decir, se podrán obtener los datos "camiseta", "pantalón", "Nike" y "50€" pero no podrán establecerse relaciones entre ellos. Sin embargo, gracias a la Web Semántica y la inclusión de información semántica, una máquina podrá inferir que la camiseta y el pantalón son del mismo fabricante, Nike, y que el precio de la camiseta es de 50€. Así pues, este pequeño ejemplo muestra la importancia de los metadatos semánticos como evolución de la actual World Wide Web.

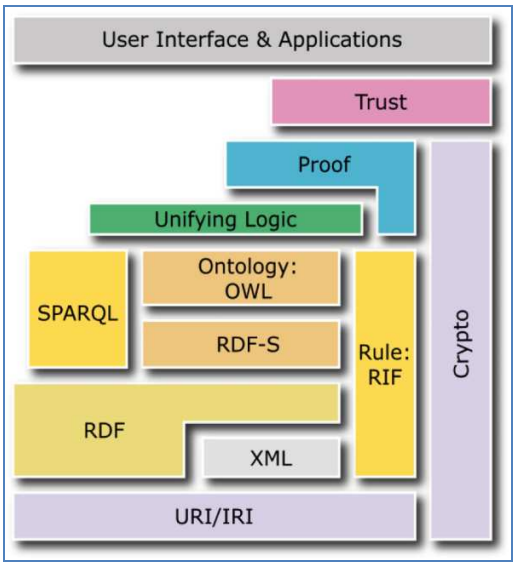


Figura 2.1. Estructura de la WEB Semántica.

.

La figura 2.1 muestra la capa de lenguajes y tecnologías recogidas por el campo de la Web Semántica. Estos esfuerzos por la definición de estándares se llevan a cabo en el contexto de la W3C. Aunque muchos de sus componentes están finalizados y son estables, algunas de sus partes aún están en desarrollo.

Las capas de la parte inferior proporcionan los estándares Web básicos para el desarrollo de otras tecnologías. Las siguientes capas definen lenguajes para describir recursos de una forma adecuada para ser procesado por máquinas. Estos lenguajes y las capas superiores son:

- **XML**: Este componente aporta la sintaxis superficial para los documentos estructurados, pero sin dotarles de ninguna restricción sobre el significado.
- **XML Schema:** Este capa define un lenguaje para especificar la estructura de los

documentos XML.

- **Resource Description Framework (RDF):** esta capa define un modelo de datos abstracto para el intercambio de información en la Web.
- **RDF Schema (RDFS):** este componente extiende RDF con un vocabulario que puede ser empleado para describir propiedades RDF y relaciones entre esas propiedades y otros recursos **[DBRV]**. Algunos ejemplos de construcciones definidas por RDFS son: *Class, Property, List, range, domain, type, subclassOf* y *subPropertyOf.*
- **Web Ontology Language (OWL):** esta capa extiende RDFS para obtener un lenguaje completo para la definición de ontologías **[SBFHJ]**. OWL proporciona construcciones más expresivas como la posibilidad de expresar relaciones entre clases (p.ej. Disyunción), cardinalidad (por ejemplo "únicamente uno"), igualdad, tipologías de propiedades más complejas, caracterización de propiedades (por ejemplo simetría) o clases enumeradas.
- **Rule Interface Format (RIF):** esta capa tiene como meta proporcionar un lenguaje común para representar y expresar reglas. La recomendación del W3C consiste en un conjunto de reglas definidas en el núcleo de RIF que pueden ampliarse por medio de extensiones. Así pues, RIF define los requisitos de representación de conocimiento en términos de reglas.
- **SPARQL Protocol and RDF Query Language (SPARQL):** este componente proporciona un protocolo y lenguaje de consulta para la gestión de los datos semánticos de la Web **[SPALR]**.
- **Unifying Logic:** esta capa tiene como objetivo ofrecer una interfaz lógica consolidada para todas las capas lógicas existentes (reglas y ontologías). Otros lenguajes lógicos serán subconjuntos de este lenguaje unificado.

La principal barrera que presenta la Web Semántica es la adaptación de todo el contenido Web existente a información semántica. Aunque se ofrece al usuario todas las herramientas necesarias para la gestión de metadatos, no se ha ofrecido todavía un componente tecnológico que permita convertir de forma automática el abundante volumen de información de Internet al modelo de datos RDF. Por lo tanto, los creadores de páginas web están obligados a traducir “a mano” su contenido. Desde un punto de vista positivo, las mayor parte de las páginas web actuales tienden a almacenar la información formateada en bases de datos, característica que permite la conversión de los datos de forma automática, empleando herramientas RDB-RDF.

Por último, hay que destacar la gran acogida de esta tecnología por parte de la comunidad de usuarios de la Web. Existe un gran número de herramientas que buscan la integración del campo de la Web Semántica con el mundo de la World Wide Web. Algunos ejemplos de estas herramientas son la DBpedia, FOAF o SIOC.

El primero de ellos es un intento de publicar la estructura de datos presente en la Wikipedia, siguiendo el modelo RDF. Así pues, este proyecto trata de ofrecer toda la información de la enciclopedia en línea en un formato adecuado para que agentes inteligentes puedan realizar consultas avanzadas o inferir conocimiento. Los otros dos proyectos mencionados definen vocabularios estándar para la transmisión de información. SIOC (Semantically-Interlinked Online Community) se centra en los términos relacionados con la web, como los foros de discusión, blogs, hilos de noticias o listas de correo. FOAF trata de describir las relaciones que un individuo tiene con otra persona o conceptos de su entorno. Es un intento de expresar las relaciones de cada usuario en un contexto social.

En resumen, la Web Semántica es la "Web de datos". Esta tecnología mejorará el actual modelo de datos y redefinirá el modo en el que está organizada la información. Como enunció su creador, la Web Semántica va a ser una revolución en el tratamiento de los datos de la World Wide Web, afectando a todos los campos, desde la gestión de las relaciones sociales hasta el pago de sus facturas.

2.2.2 Resource Description Framework.

RDF es un modelo de datos abstracto para el intercambio de información en la Web. Este modelo fue creado por parte de la familia de la World Wide Web Consortium (W3C) y parte de la idea de presentar la información (recursos web) empleando un lenguaje unificado, basado expresiones sujeto-predicado-objeto conocidas como triplas RDF. Este concepto es similar a otros modelos como los diagramas de clases o de Entidad-Relación. Puede interpretarse como un grafo en términos matemáticos, ya que consiste en un conjunto de nodos (sujetos y objetos) relacionadas por propiedades o arcos (predicados, aunque estas relaciones sean unidireccionales). Además, este modelo extiende la estructura de interconexión empleada en la Web (por medio de enlaces URL) para identificar los elementos y relaciones, denominados recursos, expresados en la tripla RDF.

El concepto de recurso puede asociarse a toda información que puede ser obtenida directamente de la World Wide Web, como una página web, un correo electrónico o una

imagen. Esta definición es una primera introducción al término y resulta incompleta debido a que el modelo RDF no limita a la descripción de recursos sólo a los elementos presentes en la Web, sino que establece que un recurso puede referirse a cualquier concepto, como, por ejemplo, relaciones entre recursos. Para terminar, todo recurso se identifica por medio de URIs (Uniform Resource Identifier). La URI de un recurso no necesita ser accesible por medio de HTTP o representar un recurso tangible y accesible a través de la red. Por lo tanto, una URI puede identificar cualquier cosa que se desee emplear como recurso.

El sujeto de la tripla RDF tiene asociado una URI o un nodo en blanco (blank nodes) que representa un recurso. Los recursos identificados por nodos en blanco se traducen en recursos anónimos. El predicado también se representa por un recurso, pero en este caso establece las propiedades que tiene un recurso o las relaciones entre ellos. Por último, el objeto es el campo más variable, ya que acepta tanto recursos definidos por URIs o nodos en blanco, como valores literales.

2.3 Sistemas de gestión de ideas

2.3.1 Introducción.

Los sistemas de gestión de ideas tienen como objetivo proporcionar herramientas que faciliten la recopilación de ideas y su administración. Este tipo de plataformas surgieron como respuesta a las necesidades de innovación que demandaban las empresas y se posicionaron rápidamente como una herramienta interesante para la implementación y explotación económica de nuevas ideas y descubrimientos. Estos sistemas fomentaban una nueva cultura en las organizaciones basada en la innovación que favorecía la participación de los empleados y el descubrimiento de nuevas oportunidades de negocio. Así, la gestión de innovación empezó a formar parte de la estrategia de negocio de las compañías **[TL03]** . Además, este tipo de sistemas no sólo se centran en la mejora y desarrollo de nuevos productos, procesos o servicios, sino que también recogen las propuestas relativas a la estructura de la organización, los procesos internos y las técnicas de gestión .

Los sistemas de gestión de ideas están recibiendo una buena acogida en el entorno empresarial, demostrando las ventajas de este tipo de plataformas. Esta acogida a provocado que las herramientas de gestión de innovación adquieran otros nombres, entre ellos los más comunes son: Bancos de Ideas **[TAYLC]** , Idea pools, Idea War Chest **[WERC]**, Idea

Refrigerator y Idea Archives **[ZBSA]** .

Estas plataformas de gestión han actualizado el antiguo "Buzón de sugerencias" extendiendo sus funciones por medio de sistemas y métricas que permiten gestionar tanto la generación, captura y colaboración de ideas como su evaluación, desarrollo, implementación y seguimiento de los resultados.

Para conseguir esta solución estructurada y disciplinada de administración de la innovación, los sistemas de gestión de ideas deben incorporar dos componentes básicos :

- Un *front-end* que favorezca la captura y desarrollo de ideas. El proceso de introducción empleado debe ser claro, siendo el formato de entrada más habitual un formulario que recoge el contenido junto con el nombre del autor y una clasificación previa de la idea. Además, este componente tiene que proporcionar un método de búsqueda de ideas y una vista general de las innovaciones propuestas tanto por el usuario (para ver el estado de sus contribuciones) como por el resto de la comunidad de usuario de la plataforma.
- Un *back-end* que defina e implemente procesos de evaluación y filtrado de ideas que faciliten la elección de ideas con un mayor potencial de desarrollo e implementación. El proceso de valoración debe ser sencillo y el formato más

2.3.2 Drupal.

Corresponde a un sistema de gestión de contenidos de código abierto escrito en lenguaje PHP, en el cual es posible la construcción de sitios web dinámicos haciendo uso del concepto de extensibilidad. El cual contempla la posibilidad de incorporación de nuevas funcionalidades en forma de módulos a partir de un conjunto de operaciones básicas previstas en forma de API. Además de ello, se caracteriza por ser el primer manejador de contenidos que incorpora capacidades semánticas mediante el uso de módulos como SPARQL, RDF y SIOC **[DSEPA]**. Este manejador de contenidos también tiene la habilidad de realizar clasificación colaborativa a través de *Folksnomias* con las cuales es posible establecer relaciones de jerarquía no predeterminas. Indudablemente esta es una característica que afianza los objetivos de la WEB semántica dentro de los lineamientos de la WEB 2.0. Otra característica de relevancia que posee este manejador de contenidos, es la versatilidad de adecuación de contenido, con la que es posible crear no sólo páginas web convencionales sino páginas personal, foros, blogs, etc.

En aras del uso de la Web Semántica, es importante hacer referencia a la alta aceptación que ha

tenido este gestor de contenidos dentro de diversos ámbitos comerciales como es el caso de Yahoo Research, Ubuntu y Sony Music.

A continuación se detallan algunos de los componentes base del sistema de gestión de contenidos Drupal:

- **Nodos**.

Corresponde a la unidad básica de información en la que Drupal almacena elementos como Autor, titulo, fecha y creación. De manera general representan una abstracción de datos que pueden llegar a extenderse a diferentes tipos de contenido como páginas o *Stories.* Un punto importante de esta representación tiene que ver con la capacidad que tiene de añadir cualquier tipo de información que sea extensible desde un módulo o plugin **[DGHP]**.

- **Módulos**.

Corresponde a un tipo de elemento que permite ampliar las funcionalidades del gestor de contenidos por medio de agregación de componentes. Suelen clasificarse en dos categorías, la primera de ellas corresponde a los *Core Modules* cuya instalación y configuración se encuentran por defecto en la distribución inicial de Drupal. La segunda son los *Contributed Modules* los cuales son módulos soportados por la comunidad de desarrolladores del proyecto Drupal.

- **Bloques**.

Corresponde a un conjunto de componentes que permiten predeterminar el lugar de ubicación de los módulos y nodos a partir de características de valoración, de uso o actividad de cierto tipo de información. Este tipo de componentes también puede llegar a definir la visibilidad de algún elemento en función de los roles del usuario.

- **Hooks**.

Corresponde aquel tipo de entidad mediante la cual es posible interactuar con las funcionalidades que ofrece el API de desarrollo de Drupal. Es invocada cada vez que se hace uso de un módulo por medio de eventos internos o *callbacks*.

2.3.3 Gi2mo Ideas Stream.

Corresponde a un sistema de gestión de contenidos de código abierto, desarrollado por el grupo de sistemas inteligentes de la Universidad Politécnica de Madrid. Aborda una arquitectura basada en componentes, en la cual se extiende y se mejora parte de las funcionalidades que ofrece el gestor de contenidos Drupal. Su principal objetivo es proveer las funcionalidades

básicas presentes en los actuales sistemas gestores de Ideas, incorporando capacidades de gestión semántica que hagan que la información creada y recolectada pueda ser utilizada de manera automática en otros procesos como el análisis estadístico de información, buscadores semánticos, etc [**GSI2**]. Otra característica de este sistema gestor de ideas, es la filosofía de uso bajo la cual ha sido creado, siendo esta de tipo *Open Source.* Lo cual lo marca como una de las mejores alternativas existentes hoy por hoy en materia de aplicaciones de gestión de ideas de licencia abierta.

Al hacer re-uso de muchos de los componentes presentes en el gestor de contenidos Drupal. Gi2mo Ideas Stream se convierte en una aplicación extensible, lo que hace que pueda ser integrada a futuro con muchos módulos desarrollados y orientadas al mismo dominio de aplicación. Sin duda esta característica convierte a gi2mo Ideas Stream en uno de los sistemas gestores de ideas más recomendado.

Ideas Stream consiste de las siguientes partes:

- **Ideas Stream Module:**

Es un componente de software que actuando como núcleo central de la aplicación permite adaptarse al manejador de contenidos Drupal reconfigurando las funcionalidades básicas. Así mismo, crea los datos necesarios para que el manejador de contenidos pueda funcionar correctamente.

- **Ideas Modules Elements (dependencias):**

Módulos de software desarrollados en forma de dependencias, necesarios para el funcionamiento de algunas de las funcionalidades de ideas Stream.

- **Ideas Stream Theme:**

Módulo de software desarrollado en forma de tema que reorganiza el estilo y diseño de Drupal con el fin de ofrecer una interface de usuario acorde a las funcionalidades del manejador.

Entre algunas de las funcionalidades básicas ofrecidas por este manejador de contenidos, se encuentran la creación, voto y comentarios de ideas. Así mismo, la organización y reconocimientos de ideas por medio de categorías y *tags.* Sin embargo, tal vez una de las funcionalidades más importantes es aquella que está relacionada con el seguimiento de ideas a partir del estado de una Idea (Aceptada, en revisión, etc.). Igualmente la exportación e importación de ideas es otro de los temas claves de Gi2mo Ideas Stream, lo cual lo convierte en un sistema de fácil interoperabilidad. Algunas de sus funcionalidades más importantes se aprecian en la figura 2.2.

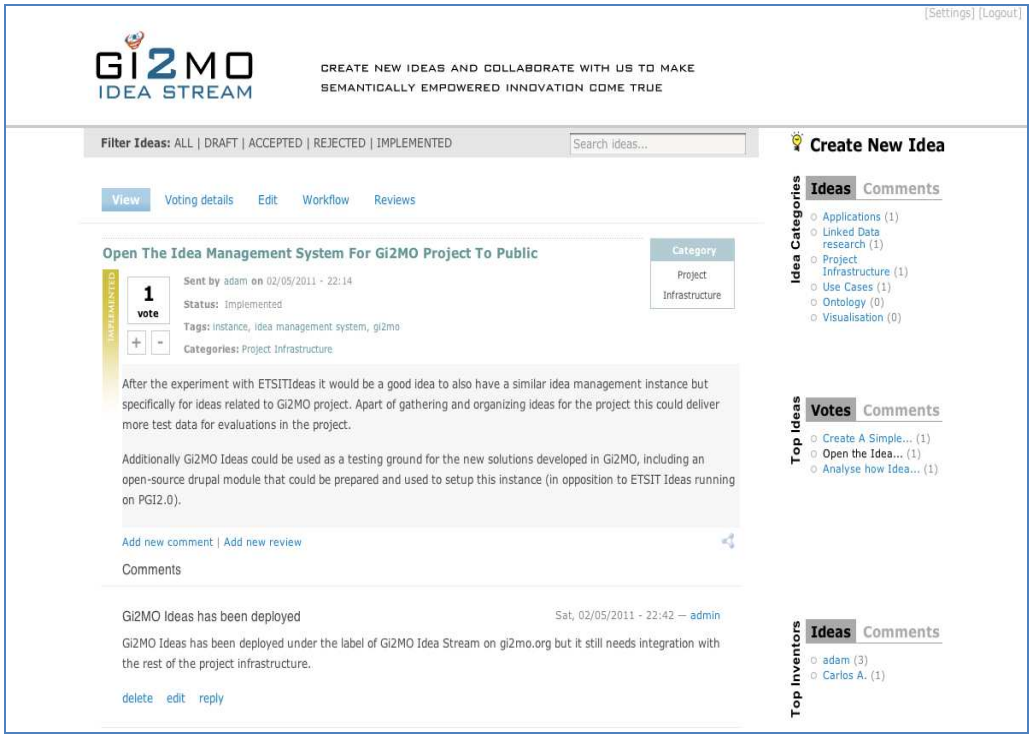


Figura 2.2. Sistema de gestión de ideas - Gi2mo Ideas Stream.

2.3.4 Ideas Brainstorm Ubuntu. – Ideas Torrent

Brainstorm Ubuntu corresponde a una plataforma de gestión de ideas que proporciona una forma fácil y simple de contribuir con ideas que mejoren las líneas de productos que provee el proyecto Ubuntu. El éxito de esta plataforma se centra en escoger las ideas que tengan un alto índice positivo de votación por parte de los usuarios de la comunidad. Con el fin de que sean analizadas por los equipos de desarrollo, quienes después de un análisis consensuado valoran y deciden el rumbo de la idea. Actualmente este sistema cuenta con cerca de 20 000 propuestas que se encuentran en diferentes estados (Aceptado – Revisión – Implementadas, etc.).

Un punto importante aclarar acerca de este gestor de ideas, es que su funcionamiento se basa en el *Engine* del *Framework* Idea Torrent, el cual básicamente ofrece un sistema de gestión de ideas basado en Drupal. Al ser una plataforma de código abierto su principal objetivo es facilitar un canal de comunicación entre los usuarios de entornos web como Ubuntu Brainstorm y los equipos de desarrollo del mismo y no para fines comerciales.

Al igual que cualquier otro sistema gestor de ideas, su propósito es proveer plataformas en las

que los usuarios pueden expresar sus opiniones e ideas sobre un servicio, producto o compañía. Para ello implementa mecanismos en los que los usuarios pueden crear ideas, emitir comentarios y opiniones acerca de las mismas. Igualmente puede otorgar valoraciones de acuerdo al grado de innovación de una idea. Sin embargo, Idea Torrent presenta dos desventajas que dificultan su elección como sistema gestor de ideas de las organizaciones de cara a la integración con ambientes semánticos.

La primera desventaja tiene que ver con la forma en que gestiona el contenido, dado que este gestor de ideas realiza la publicación de información con estructura no basada en nodos, es decir, la gestión de contenidos como ideas, comentarios las realiza sobre sistemas relacionales. Por lo tanto, para sistemas que estén basados en Drupal y en que en algún momento tengan relación con Ideas Torrent, es posible la incompatibilidad de datos. Por otra parte, se tiene el problema de incompatibilidad con motores de bases de datos como Mysql y Oracle **[IDTP]**, ya que este sistema de ideas sólo soporta Postgres SQL.

Idea Torrent es una solución creada por un desarrollador independiente, Nicolas Deschildre, a comienzos del 2009. Su uso está protegido bajo licencia GNU Public License. Su versión actual es la 0.9.1 y es compatible con las versiones 5.x y 6.x de Drupal. La versión 2.0 está todavía en desarrollo (el proyecto está suspendido temporalmente) y su objetivo es solucionar las deficiencias mencionadas anteriormente.

Sin embargo y pese a que no cuenta con una organización en forma de nodos, el tipo de información que contiene es una muy buena fuente de datos que eventualmente podría alimentar los repositorios de información de otros gestores de ideas. Tal es el caso de Gi2mo Ideas Stream. Es interesante destacar la forma en la cual se estructura la información publicada sobre este portal. Por ejemplo, ante la publicación de una Idea, esta además de componerse de elementos básicos como título, descripción y fecha tiene una estructura medianamente jerárquica en la que se organiza información como alternativas de soluciones que a su vez se estructuran con comentarios que al tiempo tienen una serie de atributos únicos que lo identifican como nombre del autor, fecha de publicación. La figura 2.3 muestra una aproximación a estos detalles.

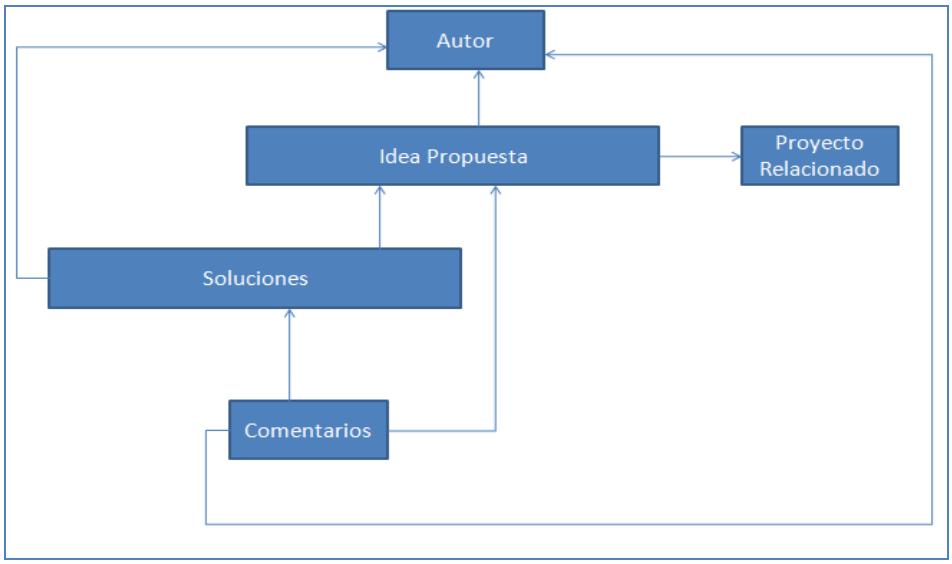


Figura 2.3. Estructura Publicación Sistema Idea Torrent.

2.4 Minería de Datos.

2.4.1 Introducción.

Corresponde a uno de los procesos de mayor importancia dentro de este trabajo fin de Máster, ya que mediante él es posible obtener los datos necesarios para realizar la labor de análisis, representación e inferencia de información a través de búsquedas. Hace uso de técnicas basadas en el concepto de Clúster, el cual se abordara en el apartado 2.5. El significado del término Minería de Datos, extiende de aquel famoso proceso comparativo realizado por Rakesh Agrawal a mediados de los 80 entre la acción de buscar información de carácter relevante en los sistemas de información y la acción de picar una montana en búsqueda de metales valiosos [RKAW]. En el que aducía que ambos procesos exigen examinar una gran cantidad de información y material, hasta tal punto que el objetivo primordial de estas dos acciones era encontrar en donde reside la importancia de estas muestras.

En esta forma, es posible definir la minería de datos como una técnica de manejo y análisis de información que aprovecha capacidades de procesamiento, almacenamiento y transmisión de datos con el fin de encontrar patrones repetitivos, tendencias o reglas que expliquen el comportamiento de los datos en un determinado contexto. Esto permite que los responsables de

las empresas u organizaciones puedan tomar las mejores decisiones sobre la base del conocimiento obtenido. Así la Minería de Datos hace uso de diferentes tipos de ciencias como la inteligencia artificial, la estadística y redes neuronales para intentar comprender el contenido de un determinado repositorio de información.

Con el actual crecimiento de información, cada vez más se hace necesario contar con nuevos métodos de procesamiento de datos y nuevas tecnologías de análisis de información. Las cuales no sólo optimicen la forma de operar sobre las búsquedas tradicionales, sino que también permitan crear procesos de búsqueda de conocimiento a partir de fuentes datos no explotadas. Esto permitiría reafirmar el concepto de que la Web está a punto de convertirse en una enciclopedia universal de conocimiento humano**[CWS9]**, y es aquí donde precisamente entran en juego conceptos como la WEB Semántica y la Minería Web para dar interpretación sobre la información encontrada y generar modelos que representen algún tipo de valor agregado.

La idea de concebir valor agregado sobre la información y modelos generados en los procesos de minería de datos, cada día aumenta tomando mas fuerza sobre diferentes campos de aplicación. Por ejemplo en una publicación realizada en el mes de agosto de 2010, el *Wall Street* **[GPIAB]** revelo información acerca del uso de técnicas de minería de datos realizadas por la compañía Google en la construcción de herramientas de software que permitirán predecir tendencias de comportamiento.

Puntualmente el desarrollo de un sistema de información que le permitiría conocer el estado de ánimo de sus trabajadores y averiguar cuáles de ellos plantean marcharse prontamente. La anterior decisión ha sido tomada como medida preventiva ante el alto número de marchas de algunos de los cargos estratégicos más importantes de la compañía durante los últimos meses. No obstante, las técnicas y patrones utilizados para este tipo de actividades no son nuevas, pero por primera vez se empiezan a aplicar a gran escala sobre la gestión de recursos humanos. Es interesante destacar que esta tendencia empieza a vislumbrarse también en campos como la Medicina, procesos electorales y lucha contra el terrorismo. De esta forma es posible ver que la Web Semántica y la minería de datos son dos escenarios muy importantes dentro del marco definición de la estructura semántica que promulga la WEB 3.0.

2.4.2 Descripción de Etapas de Minería de Datos.

En general la técnica de minería de datos sigue una serie de procedimientos que se caracterizan por:

1. **Determinación de los objetivos.** Corresponde a la primera etapa del proceso de la minería de datos, en la cual se establecen los objetivos y alcances del dominio de información que será analizado.

2. **Selección / Pre procesamiento de datos.** Es la etapa en que se hace la selección de datos a partir de múltiples fuentes de información. Esto con el fin de tener un repositorio de datos consolidado en el que sea posible reducir y transformar la información manejada.

3. **Preparación de los datos**. Cuando los datos han sido seleccionados, estos deben preservar la característica de integridad referencial, intentado preservar un formato de estandarización por medio de transformaciones que permitan corregir inconsistencias y diferencias de codificación entre los datos.

4. **Construcción del Modelo**. En esta fase se lleva a cabo la construcción de modelos en función de los objetivos planteados durante la primera etapa. Los modelos pueden ser de tipo descriptivo, en los que se especifican patrones con base a comportamientos previamente definidos **[DTMT]**.

5. **Descubrimiento de patrones.** Corresponde a la etapa en la cual es posible descubrir patrones de comportamiento a partir del uso de técnicas, herramientas y algoritmos que haciendo uso del modelo previamente definido. De esta manera permiten encontrar una serie de características de similitud que ayuden agrupar las actividades por grupos de técnicas.

6. **Despliegue de patrones.** Una vez obtenido los resultados relacionados con el descubrimiento de patrones, éstos deben ser evaluados, e interpretados por medio de técnicas de validación de modelos.

Un aspecto importante en la minería de Datos, son las interpretaciones que pueden llegar a tomar sus métodos y procedimientos dependiendo del contexto de aplicación, Teniendo en cuenta el marco conceptual en el que se define este proyecto, es necesario considerar la minera de datos como la etapa de exploración presente en el proceso de descubrimiento de conocimiento *Knowledge Discovery from Databases* **[CJOD].** Este

concepto hace referencia a la forma a través de la cual es posible identificar patrones, parámetros y modelos de comportamiento para que al ser analizados y procesados incorporen el conocimiento obtenido en algún sistema real. La figura 2.4 muestra en detalle las fases de composición de KDD.

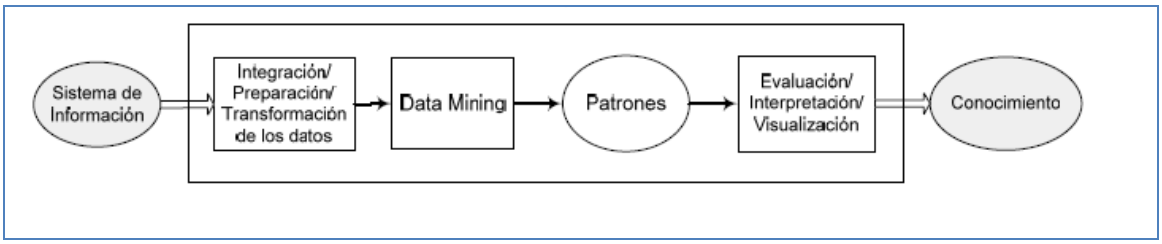


Figura 2.4. Fases del Descubrimiento de Conocimiento.

Centrado en esta idea, KDD aborda diferentes tipos de técnicas para el análisis y procesamiento de la información. Entre las cuales es posible distinguir métodos probabilísticos y estadísticos, técnicas de visualización, métodos de clasificación bayesiana, etc. A su vez, este tipo de técnicas se encuentran asociadas con la utilización de algoritmos que basados en procedimientos de verificación o descubrimiento permiten definir el modelo, criterio y algoritmo de búsqueda.

2.4.3 Minería WEB.

Por su parte, la minería WEB se puede definir como aquella disciplina extendida de la minería de datos en la que es posible descubrir y extraer información automáticamente de los datos presentes en el contexto de la WEB. En este sentido es posible establecer que la minería WEB posee tres dominios de extracción de conocimiento, los cuales son:

1. **Minería de Uso.** Corresponde a una de las técnicas categorizadas dentro de la Minería WEB, para realizar la búsqueda de patrones de acceso común que permitan generar modelos de ingreso. Así permiten ubicar los contenidos de una forma más accesible y en lugares más relevantes de acuerdo a la condición de uso. Por otra parte, pretende analizar las tendencias y comportamientos por usuario.

2. **Minería de Contenido.** Es otra de las técnicas categorizada dentro del ámbito de la Minería WEB. Su principal objetivo es la extracción de información del contenido presente documentos, archivos y otros recursos por medio del concepto de indexación. De tal forma

que sea posible organizarla y clasificarla de una mejor manera para que el acceso y recuperación de información se haga de una forma mucho más eficiente.

3. **Minería de Estructura WEB:** Corresponde a la tercera categoría presente en la minería WEB y es utilizada para inferir conocimiento a partir de la organización y estructura de los enlaces contenidos en la WEB. Con esta técnica es posible generar información de similitud y relación a partir del uso de algoritmos como *Page Ranker* y *Hiper link Induced Topic Research* **[BDL98]**.

La figura 2.5 muestra el diagrama esquemático de los dominios de composición de la minería WEB.

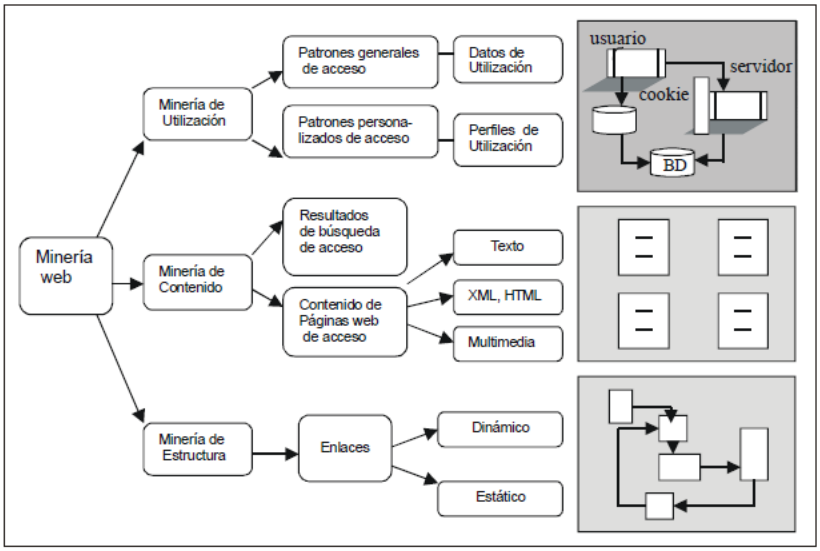


Figura 2.5 Dominios de Extracción de Minería WEB.

2.4.3.1 Procesos de la Minería WEB.

La Minería WEB cuenta con cuatro procedimientos para la selección y transformación de datos:

1. La primera etapa, consisten en la localización y descubrimiento de las fuentes de información sobre la que se aplicará el proceso de minería.
2. La segunda etapa, consiste en el proceso de selección y pre – procesado, en la cual se extrae de forma automática información específica de las fuentes antes mencionadas por medio de procesos de selección y transformación de datos como *Harvest* y *FaoFinders.*
3. La tercera fase, corresponde al reconocimiento de patrones generales encontrados durante la etapa de extracción. Suele utilizarse técnicas basadas en el agrupamiento de clúster y reglas de asociación para diagnosticar la secuencia de patrones.

4. La última etapa corresponde al proceso de análisis, en la cual se desarrollan técnicas y herramientas que permitan modelar la forma en que se utiliza el conocimiento minado.

En los últimos años el uso de estas herramientas ha marcado sin lugar a duda una revolución tecnológica en la forma de búsqueda de nuevos conocimiento y por ende oportunidades de mercado para muchas organizaciones. Sin embargo, los escenarios de la minería WEB podrían ser susceptibles a mejoras a través de una Web más estructurada que basada en componentes semánticos permitan inferir conocimiento. Teniendo en cuenta el alto grado de madurez que ha alcanzado la web Semántica en materia de codificación y representación del conocimiento a través del uso de ontologías, así como del gran numero de interrelaciones semánticas presentes hoy día. Se hace necesario pensar en modelos que perciban una unión entre la WEB semántica y la minería de datos, de tal forma que al realizar adaptaciones sobre algunas de las herramientas de *Data Mining* sea posible representar conocimientos de dominio que incluyan información estructurada y reglas de inferencia (*Semantic Data Mining*).

Dos de los aspectos de mayor relevancia al momento de adoptar estrategias de Semantic Data Mining, son la definición del proceso de anotación semántica y la definición de la extracción de la información, por un lado la anotación semántica se considera como el elemento básico de información para la descripción formal de una fuente semántica **[WEIU].** En tanto la extracción de información se puede denotar como la técnica que haciendo uso de ontologías permite definir un dominio sobre un conjunto de datos y sobre las relaciones y participaciones de estos mismos **[FKAD3]**. Las técnicas de extracción de información pueden estar basadas en reglas, en donde se asume una asignación que parte de un conocimiento ontológico previamente definido o basadas en ontologías que pueden adquirir conocimiento de forma dinámica.

La extracción de datos basadas en ontologías define dos tipos de objetos. Aquellos que corresponde a la categoría de léxicos y contienen la instanciación de objetos concretos. Por ejemplo en el dominio de una ontología de Ideas puede llegarse a encontrarse un objeto léxico llamado “proyecto relacionado”, el cual podría llegar a tener una instancia llamada KDE. Por otro lado, se cuenta con los objetos de tipo no léxico en los cuales la información se representa por medio de conceptos abstractos o aparece en forma implícita. En el caso del dominio de la ontología de ideas podría mencionarse el objeto identificativo de la idea, el cual podría ser por ejemplo idea numero 20.

Una característica de los procesos de extracción basados en ontologías, es el uso de *data frames*, los cuales encapsulan las propiedades de cada de dato en forma de ítem. Esto lo hace por medio de una representación abstracta que no sólo incluye una representación interna de las operaciones

sino también una información de representación contextual, que permite hacer clasificación de objetos de acuerdo a un contexto. Igualmente es oportuno destacar que la gran mayoría de los procesos de extracción basados en ontologías hacen uso de lenguajes como OSML (*Object Oriented System Model Language)* el cual se basa en sistemas de lógica formal(First Order Logic) que permiten especificar las jerarquías de conceptos y sus relaciones.

2.5 Clustering

2.5.1 Introducción Clustering.

El llamado análisis de conglomerado o análisis de Clúster (Inglés), es una técnica que consiste en agrupar un conjunto de observaciones en un número establecido de grupos basados en el criterio de distancia o similitud entre muestras. Es ampliamente usada como técnica de diferenciación o segmentación de productos en líneas de investigación de mercados, dado que mediante esta, es posible establecer comportamientos de uso por identificación de segmentos o grupos. Así mismo, permite identificar nuevas oportunidades de negocio y seleccionar mercados de prueba de acuerdo a ciertos rasgos o comportamientos de agrupación **[AGEST]**.

Hablando en términos de complejidad, tal vez el factor más importante al aplicar este tipo de técnicas tiene que ver con la reducción de las muestras, lo cual facilita el no incremento de complejidad en el manejo de la información. Otro aspecto relevante de esta técnica, son los conceptos de alta homogeneidad interna, el cual se consigue cuando los elementos dentro de cada grupo son similares entre sí. Por otra parte se tiene el concepto de alta heterogeneidad externa, el cual se consigue cuando los elementos de un grupo determinado son diferentes a los elementos de otros conglomerados.

Clustering, también se caracteriza por ser una técnica más de aprendizaje automático en la que el aprendizaje realizado es no supervisado. Esto significa que un algoritmo de *clustering* deberá ser capaz de establecer por sí mismo patrones de comportamiento genérico en la muestra de datos y a su vez categorizar cada instancia de datos en alguno de los grupos formados **[MCAP]**. El porque elegir a Clustering como técnica de representación para la información tratada, tiene un argumento bastante fuerte y está relacionado con el problema que enfrenta actualmente la sociedad de la información, el cual es la gestión óptima y productiva de la información disponible frente al masivo incremento de datos, es decir, se hace necesario la participación de

sistemas de resumen automático que den una correcta organización de la información independiente del tamaño de la muestra.

No obstante, la técnica de Clúster carece de propiedades inferenciales, lo cual hace que la información procesada sólo sirva a manera de muestra de diseños y no halla lugar a contraste ni deducción de hipótesis. Así mismo, en Clúster el criterio de similitud de muestras generalmente suele hacerse por medio de la medida de proximidad entre dos puntos lo cual permite indicar que entre mayor sea la distancia entre dos puntos, estos presentan comportamientos heterogéneos entre sí. El sustento teórico que respalda esta afirmación se basa en la distancia de *Euclidea* al cuadrado, la cual se aprecia en la ecuación 1

$$\text{Para } x = (x_1, \dots, x_r) \text{ e } y = (y_1, \dots, y_r):$$

$$d^2{}_{xy} = \sum_{i=1}^{r} (x_i - y_i)^2$$

Ecuación 1. Distancia de Euclidea.

Tomando como punto de referencia los aspectos mencionados anteriormente es posible definir los algoritmos de Clúster en dos categorías:

Algoritmos Jerárquicos.

Este tipo de algoritmos suelen ser utilizados para muestras de mediano valor, en las que el criterio de agrupamiento se realiza en forma no iterativa, es decir, cuando un dato ha sido clasificado dentro de un grupo en particular, no es posible que a futuro éste salga de la clasificación establecida inicialmente.

Algoritmos No Jerárquicos.

Sobre esta clasificación se aloja el conocido algoritmo de K means, el cual suele utilizarse para muestras de gran valor, en las que generalmente se busca formar clústeres (grupos), los cuales serán representados por K objetos. En términos generales, cada K representado es el valor medio de los objetos que pertenecen al grupo. El apartado 2.5.2 hablara en detalle los mecanismos de diseño e implementación de este algoritmo.

Ahora bien, es necesario mencionar que los resultados de similitud o no similitud de K means se consiguen por medio de una serie de fases que preservan gran similitud con los procesos de análisis de información descritos en el apartado de minería de datos

1. Selección de la muestra de datos.
2. Selección y transformación de variables a utilizar.
3. Selección del concepto de distancia o similitud y medición de las mismas.
4. Selección y aplicación del criterio de agrupación.
5. Determinación de la estructura correcta (Número de grupos).

Finalmente respecto al tema de Clustering, es necesario decir que desde el punto de vista empresarial y organizacional podría resolver problemas relacionados con la percepción de consumo y muy probablemente con la predicción de tendencias en el mercado. Teniendo en cuenta que el desarrollo de este proyecto se enmarca en una línea meramente académica, el tipo de análisis que podría extrapolarse sobre la información de Clúster estaría orientado a tomar métricas relacionadas con las estadísticas de uso y publicación sobre el proyecto llamado Brainstorm Ubuntu.

2.5.2 Algoritmos K-means.

Corresponde a uno de los múltiples algoritmos que pueden ser empleados como condición de agrupamiento dentro de la técnica de Clúster. Pertenece a un método en el que se asigna cada muestra sobre alguno de los centros de grupo construidos de forma aleatoria o definida con anterioridad. El criterio de asignación se realiza en términos de los centroides más próximos a la muestras, para lo cual se utiliza el concepto de distancia Euclidea. Posterior a ello se reubican nuevamente las k-medias, está vez intentando buscar acercarse al centro de cada uno de sus grupos recién definidos. El próximo paso es re calcular el centroide de cada grupo y volver a distribuir todos los objetos según el centroide más cercano **[JBJMC]**. El proceso se repite hasta que ya no hay cambio en los grupos de un paso al siguiente. En general los algoritmos de K means sigue el siguiente flujo de información:

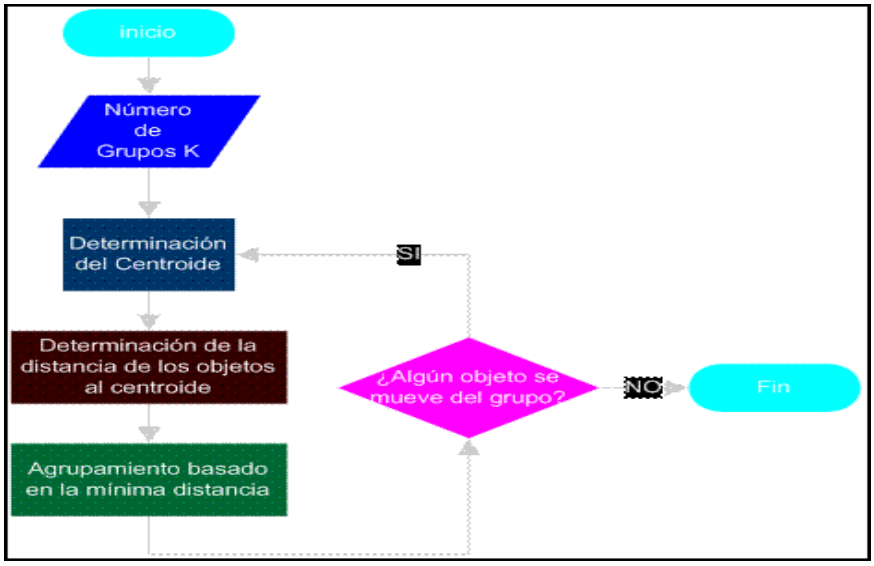


Figura 2.6. Estructura de Ejecución de K means.

No obstante, el empleo de este tipo de algoritmos suele presentar fallas cuando los puntos de una muestra están muy cerca del centroide, así como, también cuando el tamaño y forma de los grupos albergan valores muy disparejos o alejados entre sí **[CEACE]**.

2.7 Análisis de Sentimiento.

Análisis de Sentimiento, también llamado Minería de Opinión, es una rama de la inteligencia artificial centrada en medir la opinión, el sentimiento o la subjetividad de un texto. Esta área combina técnicas de procesado del lenguaje natural, lingüística computacional y minería de textos.

Aunque en la década de los 80 y 90 aparecieron las primeras investigaciones sobre el reconocimiento de la subjetividad de un texto, el progreso de este campo no se produjo hasta la llegada de la Web 2.0 . **[ADWAO]**. La proliferación de servicios como blogs, redes sociales o el comercio electrónico supuso un incremento del número de comentarios, valoraciones, recomendaciones u otras formas de expresar opinión. De este modo, Internet se convertía en un amplio repositorio público de contenido de opinión generado por usuarios. Sin embargo, a pesar del claro beneficio de tener tanta información disponible, un gran porcentaje de los usuarios de Internet (58%) **[OSPI]**. señalaba que la búsqueda de información sobre un tema o producto concreto es una tarea compleja o, por el contrario,

abrumadora por la cantidad de información disponible.

Así pues, herramientas que puedan extraer información de opinión de este tipo de recursos se presentan en una posición atractiva tanto para usuarios como empresas. Los primeros se beneficiarán de esta información para tomar decisiones más acertadas, mientras que las compañías emplearían este conocimiento para describir la posición de mercado de sus productos, conocer su reputación e identificar nuevas oportunidades de negocio. Así, la necesidad de analizar grandes volúmenes de información de opinión, junto con los avances en las áreas de procesado del lenguaje natural y máquinas de aprendizaje constituyen la base del interés depositado en el campo emergente del Análisis de Sentimiento.

Existe cuatro posibles tipos de tratamiento de la información de opinión: análisis de sentimiento a nivel de sentencia u oración, análisis de sentimiento a nivel de documento y análisis de sentimiento basado en características. El primero puede dividirse en dos sub tareas: determinar si la oración es subjetiva **[ERJW]**. y, en caso afirmativo, clasificarla como positiva, negativa o neutral. El siguiente tipo extiende el análisis de sentimiento a nivel de oración a todo el documento, aunque suele reducirse a la obtención de la polaridad del texto. El famoso algoritmo de Turney **[PTTD]**., basado en un motor de búsqueda y dos adjetivos con una polaridad definida ("*poor*" y "*excellent*"), es un ejemplo perfecto de este tipo de tratamiento. Por último, el análisis de sentimiento basado en características es un proceso más detallado y complejo, ya que no sólo tiene como objetivo determinar la polaridad y la subjetividad del texto, sino que también debe extraer las propiedades o características del documento y especificar la información de opinión asociada a cada una de ellas.

Como se ha mencionado anteriormente, el interés de esta área ha sido causado por el crecimiento del contenido Web generado por usuarios. Una de las primeras características de este contenido es su desorden y alta diversidad textual. Las opiniones son expresadas normalmente empleando un lenguaje informal y el estilo varía entre cada sitio web. Así pues, las herramientas destinadas a la extracción de información de opinión tienen que adaptarse al contexto, tipo de evaluación, nivel de interés o detalle y al tipo de vocabulario usado. El contexto hace referencia a la ubicación del contenido, ya sea un portal especializado en realizar evaluaciones de productos o un foro. El tipo de evaluación divide las opiniones en dos grandes grupos: directas y comparaciones. Las opiniones directas normalmente describen las propiedades de un objeto ("La resolución de la pantalla es increíble"), mientras que las comparaciones expresan un contraste entre dos elementos ("El teclado del teléfono X es mejor que el del teléfono Y). Respecto al vocabulario usado, no sólo hace referencia a las expresiones empleadas por la comunidad de usuarios, sino que también hay que considerar otros

aspectos como el uso de frases hechas (“El teléfono encaja perfectamente en mi bolsillo”) o el empleo de emoticonos.

Una vez que se ha definido el campo de estudio, se puede introducir los beneficios del mismo. Las técnicas de Análisis de Sentimiento pueden ser aplicadas en múltiples escenarios:

- **Motores de búsqueda:** es la aplicación más directa de las técnicas de minería de opinión. Gracias a la posibilidad de detectar zonas subjetivas y su orientación, se puede convertir los motores de búsqueda en motores de recomendación. Esta conversión es posible ya que el sistema es capaz de recopilar información de opinión (positiva o negativa) acerca de un determinado tema. De este modo, esta aplicación podrá ofrecer al usuario las ventajas e inconvenientes de un determinado producto o determinar la calidad de la información proporcionada.
- **Control del contenido:** El *spam* en las opiniones es una consecuencia directa del crecimiento de la Web. El contenido generado por los usuario ha ido ganando importancia como referencia a la hora de escoger productos y servicios, y compañías y empresas han tratado de emplear este comportamiento en su beneficio. De este modo, no es sorprendente que aquellos sistemas que permiten la transmisión de opiniones hayan sufrido a menudo un uso abusivo. En pequeña escala el contenido puede ser moderado fácilmente, pero en ambientes colaborativos como foros, grupos de discusión, listas de correo e incluso tiendas en línea pueden ser una tarea difícil. Las técnicas de Análisis de Sentimiento pueden ofrecer una primera aproximación al problema, mejorando la credibilidad del sitio web e, indirectamente, su número de visitas.

Inteligencia de negocio: La clave principal para la venta de productos es responder a la demanda de los usuarios en el momento y lugar adecuado. Las técnicas de minería de opinión pueden ayudar con esta tarea y minimizar costes, ya que sustituirán a los análisis de mercado o a la contratación de consultoras externas especializadas. De este modo, se podrá estimar la tendencia del mercado y conocer la reputación de la empresa y sus productos **[ADKA]**. Además, estas herramientas también pueden suponer una mejora en la organización interna de la compañía. Un ejemplo ilustrativo es la clasificación automática según sus orientación positiva o negativa de las opiniones recibidas por correo electrónico acerca de un servicio. Gracias a este primer análisis se podrá facilitar el reenvío automático de esos mensajes a los departamentos adecuados para que aborden el problema en caso necesario **[DTRA]**.

3. Capítulo III Análisis.

3.1 Dominio del Problema.

Como se ha venido comentando a lo largo del documento, la realización de este proyecto ha llevado al desarrollo de diferentes módulos de software que permiten gestionar y realizar procesos de análisis y búsqueda semántica de información sobre sistemas gestores de Ideas. De esta forma las actividades desarrolladas en este trabajo de fin de máster implican por una parte, la elaboración de un módulo de gestión de contenidos de Drupal, en el que se implementaran capacidades de búsqueda semántica. Por otra parte, se tiene la implementación y adecuación de un aplicativo Java que gestiona el proceso de análisis de información de un sistema manejador de Ideas por medio de representaciones de Clúster utilizando el algoritmo K- means.

El objetivo principal del módulo de búsqueda semántica llamado Gi2se, es introducir precisamente el concepto de búsqueda semántica sobre el gestor de contenido Drupal. Esta labor básicamente implica el desarrollo de un componente de software cuya estructura básica de información está representada por medio de un sistema Gestor de Ideas y cuya capacidad de búsqueda semántica está diseñada por medio de herramientas como SPAQRL, ARC2 y RDF **[AMERD]**. Una característica importante de este módulo es la capacidad de integrar y actualizar de manera periódica repositorios semánticos de información. Esto permitirá que la búsqueda cuente con fuentes dinámicas de información y que no sigan los tradicionales esquemas de búsqueda basados en soluciones SPARQL – RDF.

Tomando nuevamente como punto de referencia el sistema gestor de Ideas, el aplicativo Java mencionado anteriormente tendrá la capacidad de realizar el proceso de análisis de información por medio de datos provenientes del sistema gestor de Ideas. La técnica de representación gráfica utilizada para visualizar la información analizada se hará en base al uso de algoritmos tipo K-means.

Con el fin de hacer una representación mucho más comprensible del modelo propuesto, este trabajo de fin de Máster aborda el concepto de análisis orientado a objetos mediante el uso del lenguaje unificado de modelado (UML). De esta forma será posible conocer las diferentes interacciones que llevan a cabo entre los componentes del sistema y los usuarios participes del modelo de búsqueda semántico propuesto. Haciendo uso de UML no sólo será posible identificar las funcionalidades del modelo propuesto, sino que también será posible definir por medio de representaciones estáticas la relación estructural que hay entre los objetos previstos para el diseño del modelo.

Dadas las condiciones mencionadas anteriormente, es oportuno considerar que el planteamiento de la solución sigue la metodología estándar que se adopta en la gran mayoría de procesos de desarrollo de software, lo cual implica que se den las siguientes etapas en su respectivo orden cronológico:

1. **Fase de Requerimientos.**
2. **Fase de Análisis.**
3. **Fase de Diseño.**
4. **Fase de Codificación y Pruebas.**

De las anteriores fases, es posible mencionar que pese a que llevan un orden cronológico, en algunas ocasiones esto no es lo suficientemente riguroso, pues factores relacionados con alta complejidad del sistema a desarrollar o la introducción de nuevos requerimientos hacen que se pueda adoptar desarrollos de tipo iterativo e incremental.

Con el fin de ofrecer un mayor grado de comprensión al lector acerca de la definición de los requerimientos propuestos para la elaboración del módulo de búsqueda. Sobre la fase de análisis se realizara una sutil modificación respecto al estándar abordado habitualmente por la metodología. De manera puntual se planteará el uso de diagramas de actividades recreados a partir de la información suministrada por los casos de uso, con el fin de observar uno a uno los flujos de trabajo que ocurren en el sistema.

3.2 Definición de Casos de Uso.

Esta sección aborda una descripción general de los casos de uso realizados en la fase de diseño del módulo de búsqueda. Con la elaboración y descripción de estas especificaciones se pretende entregar al lector una visión más detallada de los requerimientos contemplados para la realización del aplicativo. Inicialmente y para efectos de ofrecer un contexto global en el que sea posible observar las diferentes interacciones que se llevan a cabo en el componente. Se presentara al usuario el diagrama de caso de Uso general, posterior a ello se hará la especificación de los casos de uso en formato expandido, en las cuales se harán especificaciones detalladas por cada interacción.

3.2.1 Diccionario de Actores.

El diseño del modelo contempla la participación de múltiples actores que actúan en beneficio del desarrollo de diversas actividades sobre el componente. Puntualmente las interacciones que se llevan a cabo en el sistema se hacen por medio de los siguientes actores:

- **Actor Administrador.** Es quien interactúa con la configuración del plugin, modificando las opciones que este ofrece. Su interacción con el sistema es exclusivamente a través de una sesión de usuario con derechos de administración.

- **Actor Usuario Registrado:** Es quien hace uso de las opciones de búsqueda que ofrece el componente desarrollado, incorporando los criterios de selección necesarios para obtener salidas de acuerdo a configuraciones previamente establecidas. Su interacción con el sistema la hace mediante interfaces gráficas en las que puede indicar los formatos de salida de búsqueda.

- **Actor Analista Información.** Es quien interactúa con la aplicación Java para realizar los procesos de análisis de datos sobre la información colectada en el proceso de extracción. De momento este usuario posee un nivel de interacción predeterminado en los que puede hacer consultas predefinidas. El escenario en que este actor se ubica no contempla manejo de sesiones de usuarios, ya que precisamente es independientemente de acceso al gestor de contenidos Drupal. Un punto interesante a mencionar acerca de este actor, es que su índice y forma de participación no incide sobre los flujos de información presentes en las actividades que se llevan a cabo en el componente, es decir, es un proceso totalmente independiente a las actividades de búsqueda y extracción.

- **Actor Extractor.** Es quien se encarga de realizar el proceso de extracción de datos de forma automática sobre diferentes fuentes de información. Corresponde a un script quien interactúa con diversos sitios a través de la web de acuerdo a parámetros preestablecidos en un archivo de contexto de la herramienta Scrappy. Su participación e intervención son importantes sobre los flujos de información presentes en las actividades ya que este actor, es quien se encarga de extraer los datos que surtirán el gestor de ideas.

3.2.2 Casos de Uso Buscador Semántico Gi2SE.

Gi2se corresponde a una iniciativa del proyecto Gi2mo cuyo propósito general se enmarca en satisfacer las siguientes necesidades:

1. Generar un módulo de búsqueda altamente compatible con el gestor de contenidos Drupal. Dado el alto número de componentes semánticos que en este proyecto se alojan bajo esta misma tecnología.

2. Generar un componente de búsqueda cuyo resultado o salida puede usado para la construcción de nuevos servicios semánticos.

3. Construir un módulo de búsqueda que pueda ser configurado con diversas fuentes de información y que posea la capacidad de generar resultados dinámicos, es decir, proponer un modelo de búsqueda de información que vaya más allá del esquema tradicional de búsqueda estática RDF.

4. Proponer un modelo de búsqueda que basado en modelos ontológicos y herramientas como SPARQL permitan gestionar la relación de conceptos por medio del uso de clases y propiedades, de tal forma que los resultados generados puedan tener un nivel de detalle más amplio y sobre todo generar información relacionada con los tópicos buscados.

La figura 3.1 muestra en detalle las diferentes interacciones que se llevan a cabo entre los diferentes actores participes en el modelo y las diferentes etapas que se conciben en el mismo.

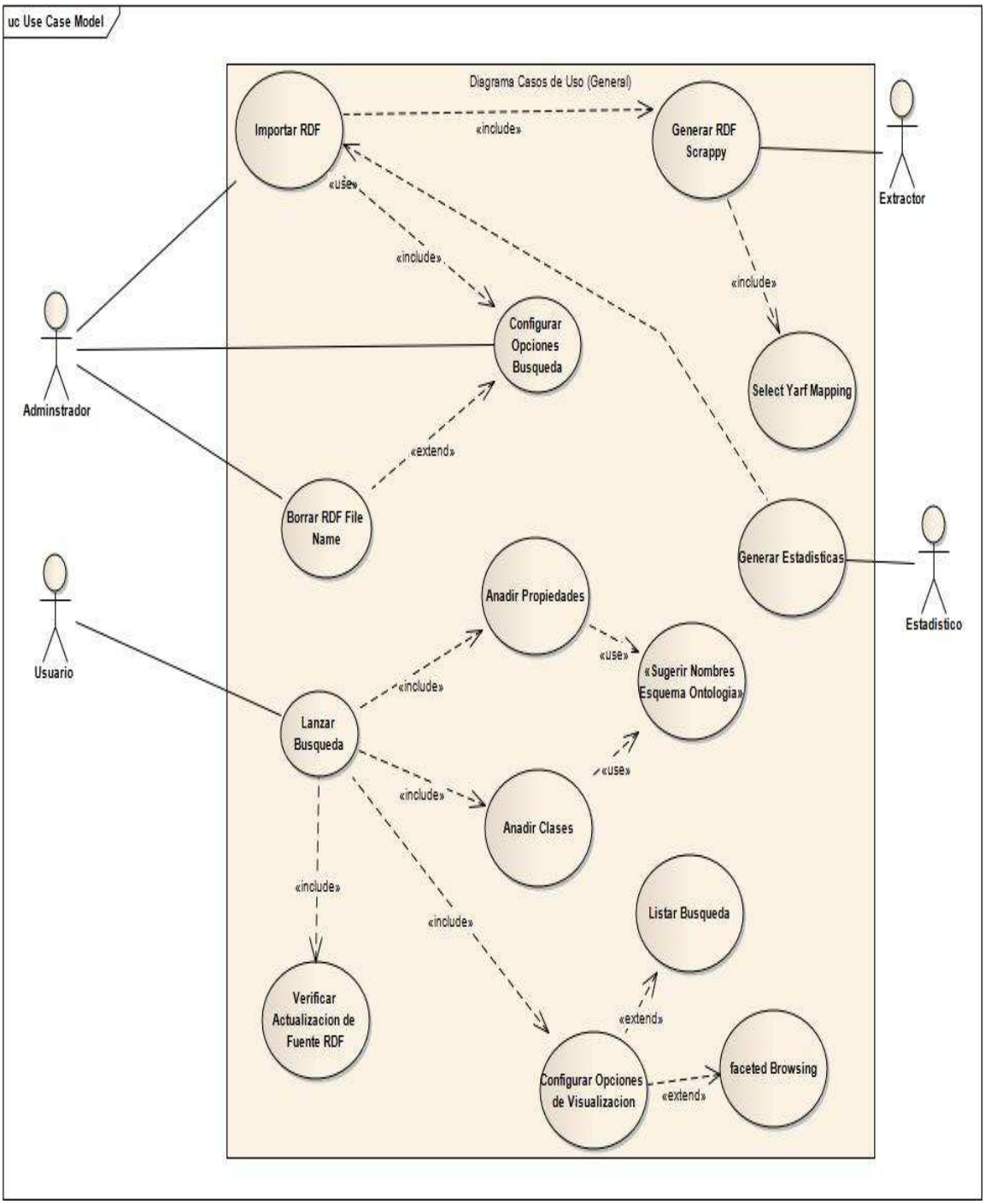


Figura 3.1. Diagrama de Caso de Uso plugin Gi2SE.

3.2.2.1 Caso de Uso I: Configurar Opciones de Búsqueda.

En este caso de uso, los usuarios están en capacidad de establecer los niveles de configuración necesarios que les permitan ejecutar búsquedas sobre dominios específicos de información. Parámetros como la fuente de datos RDF, nombre del grafo y URL END Point son requeridos como elementos obligatorios en la configuración de una fuente de datos.

Caso de uso	**CU1**
Nombre	Configurar Opciones de Búsqueda
Actores	Usuario Administrador
Descripción	Permite crear las instancias básicas de configuración del Plugin
Precondiciones	1. Despliegue satisfactorio del componente 2. Tener usuario con role de administrador en Drupal.
Pos condiciones	Se han configurado satisfactoriamente los valores de RDF, URL End Point y GraphName.
Referencias Cruzadas a casos de uso	NO
	Curso normal de los eventos
	1. [Actor] Ejecuta las opciones de configuración haciendo click sobre el vínculo configurar Gi2SE. **2. [Sistema]** muestra tres cajas de texto para introducir URL End Point, URL de ubicación del archivo RDF y el nombre del grafo a dar de alta. Adicionalmente muestra un botón para guardar los datos ingresados por el usuario. **3. [Actor]** El actor ingresa los datos solicitados en el evento inmediatamente anterior. Realiza el envío de los datos por medio del botón Salvar. **FA-1.** **4. [Sistema]** Comprueba la validez de los datos y envía un mensaje de éxito indicándole al usuario que la carga de archivos ha sido exitosa. Adicional a lo anterior muestra sobre un panel los datos ingresados con opciones de configuración (modificar o eliminar).
Flujos alternos	**FA-1** Si los datos ingresados por el usuario no son correctos el sistema avisara al usuario que la carga de archivo o URL End Point no son validos.

Tabla 3.1. Caso de uso configurar opciones de búsqueda.

3.2.2.2 Caso de Uso 2: Eliminar Atributos Opciones de Búsqueda.

En este caso de uso, los usuarios pueden eliminar la tripla que conforma las opciones de configuración de búsqueda (URL End Point, GraphName, URL RDF)

Caso de uso	**CU2**
Nombre	Eliminar Atributos Opciones de Busqueda
Actores	Usuario Administrador
Descripción	Permite al usuario eliminar los valores de URL End Point y GraphName configurados previamente.
Precondiciones	1. Tener usuario con role de administrador en Drupal. 2. CU1- Configurar 3. CU2-Modificar Opciones de Búsqueda.
Pos condiciones	Los datos han sido eliminados.
Referencias Cruzadas a casos de uso	CU1- Configurar CU2- Modificar Opciones de Búsqueda.
	Curso normal de los eventos
	1. **[Actor]** Selecciona botón eliminar configuración, el cual corresponde al perfil de configuración asociado con la información suministrada en el segundo caso de uso. 2. **[Sistema]**Remueve las entradas seleccionadas de la base de datos que almacena las opciones de configuración. Así mismo, retorna aviso de confirmación de eliminado satisfactoriamente.

Tabla 3.2. Caso de uso eliminar atributos opciones de búsqueda.

3.2.2.3 Caso de Uso 3: Verificar Actualización de RDF.

En este caso de uso, los usuarios tienen la posibilidad de decidir cuáles de las fuentes de datos configuradas pueden ser actualizadas en su contenido con respecto a la información que se almaceno cuando fue dada de alta por primera vez. Básicamente se establece un mecanismo de comparación de los datos contenidos en los archivos.

Caso de uso	**CU3**
Nombre	Verificar Actualización de Fuente RDF.
Actores	Usuario Administrador
Descripción	Permite al usuario seleccionar las fuentes de datos que desea actualizar en contenido.
Precondiciones	1. Tener usuario con role de administrador en Drupal. 2. CU1- Configurar
Pos condiciones	1. La Actualización ha sido satisfactoria.

Referencias Cruzadas a casos de uso	CU1- Configurar
	Curso normal de los eventos
	1. **[Actor]**Selecciona la opción actualizar dando click sobre el vínculo identificando con el nombre del RDF File ubicado en el panel. 2. **[Sistema]**Gestiona internamente mecanismos de comparación basados en tamaño y fecha de actualización de archivos, en caso de que alguno de estos dos criterios presente diferencias, el sistema arroja el mensaje "actualización realizada" y descarga una copia del archivo en el sistema local.FA-3.
Flujos alternos	FA-3 Si el archivo no se ha actualizado en contenido el sistema lanza un mensaje en el que se refiere a la no actualización.

Tabla 3.3. Caso de uso verificar actualización RDF.

3.2.2.4 Caso de Uso 4: Importar RDF.

En este caso de uso, los usuarios tienen la posibilidad de importar el archivo formato RDF. Esta funcionalidad es heredada de una de las opciones que nos ofrece el componente Rdfme. Un punto importante a resaltar acerca de esta actividad, es que si el plugin de búsqueda toma como dominio de datos el gestor de Ideas de Gi2mo, entonces este procedimiento deberá realizarse antes de la etapa de configuración.

Caso de uso	**CU4**
Nombre	Importar RDF
Actores	Usuario Administrador
Descripción	Permite al usuario importar el archivo RDF obtenido durante el proceso de extracción sobre el sistema gestor de Ideas Gi2mo Stream.
Precondiciones	1. Tener usuario con role de administrador en Drupal. 2. Instalar módulos RDFME y GI2MO Ideas Stream sobre el gestor de contenidos DRUPAL. 3. Verificar y configurar la estructura adecuada de los mapeos sobre el componente RDFME.
Pos condiciones	1. Archivo importado satisfactoriamente
Referencias Cruzadas a casos de uso	NO
	Curso normal de los eventos

	1. [Actor] Selecciona la opción importar Archivo RDF sobre sistema gestor de Ideas. **2. [Sistema]**Abre una caja de Diálogo indicando la localización de lugar en el que se encuentra el archivo local a subir. **3. [Actor]** Selecciona el archivo RDF correspondiente y oprime sobre el botón llamado "Cargar Archivo". FA-4. Adicional a lo anterior. **4. [Sistema]** Retorna aviso de confirmación de carga de archivo.
Flujos alternos	FA-4 Si el archivo no tiene una estructura RDF o el sistema de mapeos no reconoce las clases o propiedades presentes en el archivo, el sistema muestra un informe al respecto.

Tabla 3.4. Caso de uso importar RDF.

3.2.2.5 Caso de Uso 5: RDFME Sugerir Name Spaces.

Sobre este caso de uso, los usuarios tienen la posibilidad definir nuevos espacios de nombre, que al ser integrados sobre el sistema relacional de datos permiten que las clases y librerías de ARC2 establezcan la estructura de búsqueda necesaria sobre fuentes de información que utilicen puntos de terminación de remota (URL End Point). Esta funcionalidad nuevamente es heredada de una de las opciones que ofrece el componente Rdfme. Un aspecto adicional a mencionar es que este procedimiento no es estrictamente mandatorio, pero posibilita que los usuarios en el momento en que definen los criterios de búsqueda no deban conocer con exactitud el nombre de clases ni propiedades de la ontología sobre la cual se busca la información.

Caso de uso	**CU5**
Nombre	RDFME Sugerir Name Spaces
Actores	Usuario Administrador
Descripción	Permite al usuario definir nuevos espacios de nombre para búsqueda de información semántica que contenga un punto de terminación remota (URL End Point)
Precondiciones	1. Tener usuario con role de administrador en Drupal. 2. Instalar módulos RDFME y GI2MO Ideas Stream sobre el gestor de contenidos DRUPAL.
Pos condiciones	1. Archivo importado satisfactoriamente
Referencias Cruzadas a casos de uso	CU1- Configurar
	Curso normal de los eventos
	1. [Actor]Sobre las opciones de configuración selecciona la opción importar Name Space. **2. [Sistema]**Muestra dos cajas de texto para introducir el nombre

	del prefijo que identifica la Ontología y su respectiva URL, adicionalmente muestra un botón para guardar los datos ingresados por el usuario. **3. [Actor]**El actor ingresa los datos solicitados en el evento inmediatamente anterior. Realiza el envío de los datos por medio del botón Salvar. **FA-5.** **4. [Sistema]** Retorna aviso de confirmación de la actualización del espacio de nombres o de la inserción, en caso de presentarse uno nuevo.
Flujos alternos	FA-5 Si tanto el nombre del prefijo como la URL que describe el espacio de nombres ya se encuentran en el sistema, este desplegara un aviso diciendo que estos datos ya existen, adicionalmente, esta funcionalidad permite hacer cambios en le URL de descripción pero no en el nombre del prefijo.

Tabla 3.5. Caso de uso Sugerir Name Space.

3.2.2.6 Caso de Uso 6: Lanzar Búsqueda.

Corresponde al caso de uso más importante del componente y sobre él, los usuarios tienen la posibilidad de definir los criterios de búsqueda relacionados con un dominio específico de información. Sobre esta funcionalidad el usuario está en capacidad de crear de forma dinámica las clases y propiedades que considere convenientes para filtrar y relacionar los datos que desee encontrar en un dominio en particular. Un aspecto importante a destacar de este caso de uso tiene que ver con los formatos de salida que actualmente pueden generar los resultados de las búsquedas, siendo éstos basados en métodos y librerías propios de Drupal o con la combinación de librerías externas como SIMILE Exhibit.

Caso de uso	**CU7**
Nombre	Lanzar Búsqueda
Actores	Usuarios Registrados
Descripción	Permite al usuario definir criterios de búsqueda sobre dominios específicos de información por medio de la asignación de clases y propiedades que al relacionarlas generan la información solicitada.
Precondiciones	1. Tener usuario asignado
Pos condiciones	1. El sistema arroja la búsqueda de acuerdo a los criterios ingresados.
Referencias Cruzadas a casos de uso	CU1- Configurar. CU4- Verificar Actualización de RDF CU6- RDFME Import Name Spaces
	Curso normal de los eventos
	1. **[Actor]**Selecciona el vinculo “buscador semántico” sobre las

	opciones del perfil de usuario que contiene el manejador de contenidos DRUPAL en su panel izquierdo. 2. **[Sistema]**Muestra dos cajas de texto en las que son posible introducir / elegir el nombre tanto de la clase como de una propiedad asociada a esta misma. inicialmente esta interfaz es presentada para suplir el nivel de búsqueda más básico, dado que adicionalmente muestra un botón en el que es posible generar tantas clases como propiedades adicionales se quieran vincular.**FA-6** Igualmente muestra un botón para generar la búsqueda de los datos ingresados por el usuario. Por otra parte, sobre el área de búsqueda también es posible identificar la opción que genera el formato de salida de la consulta, el valor relacionado por defecto es aquel que está asociado con el uso de las librerías de DRUPAL. **FA-7.** 3. **[Actor]**El actor ingresa los datos solicitados en el evento inmediatamente anterior. Realiza el envío de los datos por medio del botón Buscar. **FA-8.** 4. **[Sistema]** Retorna la lista de resultados que coinciden con los criterios de búsqueda seleccionados.
Flujos alternos	FA-6 Si el usuario desea vincular un campo adicional que relacione la búsqueda con otra propiedad diferente a la que se muestra por defecto en la interfaz del buscador, puede hacerlo haciendo click sobre el botón añadir propiedad. Es necesario mencionar que esta acción implica que una clase pueda llegar a tener muchas propiedades, lo cual para criterios de selección depende de la naturaleza del dominio de información. Así cómo es posible adicionar nuevas propiedades sobre una clase, sobre esta interfaz grafica también es posible añadir nuevas clases que a su vez contemplen o añadan nuevas propiedades. FA-7 Si el usuario genera un click sobre el botón llamado “formato Exhibit”, el componente genera una nueva vista externa (por fuera del navegador) con los resultados obtenidos pero con un nivel de organización y detalle diferente al que se muestra en el formato tradicional empleado por DRUPAL. FA-8 Si la consulta no contiene resultados, el sistema arroja el mensaje “no fueron encontrados resultados”, en caso contrario despliega los

	resultados en orden de modificación.

Tabla 3.6. Caso de uso lanzar búsqueda.

Teniendo en cuenta las diferentes clasificaciones que pueden llegar a tomar los casos de uso dependiendo de su nivel de complejidad y participación en las actividades del sistema. Bien sea de forma opcional o secundaria **[ESSE]**. Además teniendo en cuenta la complejidad de la interacción que puede llevarse a cabo entre las diferentes actividades contempladas en los casos de uso y el usuario. Muchas veces es necesario adoptar descripciones en forma textual que permitan comprender de forma detallada los procesos y actividades que se llevan en cada uno de estos casos, a continuación se detallan los procesos llevados a cabo durante la extracción de información:

3.2.2.7 Generar RDF Scrappy.

Corresponde a un proceso de extracción de información fundamentado en las técnicas de minería de datos descritas en el apartado 2.4.1. Teniendo en cuenta que el presente capítulo intenta describir los diversos tipos de interacciones que se llevan a cabo entre los usuarios del sistema y las actividades que conforman este mismo. Es necesario resaltar que el primer proceso que deberá realizarse en la generación del archivo en cuestión, es la instalación y configuración del entorno de Desarrollo llamado Ruby on Rails y de su gestor de paquetes Ruby Gems. Este tipo de instalaciones hacen que el proceso de instalación de Scrappy adopte una forma modular y no sujeta a procesos de compilación.

Una vez realizado los pasos mencionados anteriormente, el próximo paso consiste en construir un archivo de contexto tipo Yarf que especifique el nivel y forma de recursividad aplicada para realizar la extracción de información sobre el sitio web previsto para tal propósito. Respecto a este procedimiento debe aclararse que dentro de los esquemas de definición debe tenerse en cuenta los siguientes aspectos:

1. Verificar que el sitio sobre el cual se quiere realizar el proceso de extracción cuente con un dominio de información acorde a la ontología que expone el sistema gestor de ideas. Para ello debe validarse que los datos e información presente en el sitio Web guarden algún tipo de relación con las clases y propiedades expuestas en el sistema.

2. Definir los mapeos que permitan integrar la información que se desea extraer del sitio

previsto sobre las clases y atributos de la ontología que describe el sistema gestor de ideas. Para ello se utilizan las descripciones presentes en el archivo RDF que contiene las clases, subclases y propiedades de la ontología de Scrappy. De esta forma haciendo uso de elementos como CssSelector o XPathSelector asociados con etiquetas de la forma [sc:type], es posible construir la estructura en forma de triplas del archivo tipo RDF que alojará los valores obtenidos del proceso de extracción.

El último pasó contemplado en el proceso de extracción de datos, corresponde a la ejecución de Scrappy ya partir de la estructura obtenida en el paso anterior. Para ello se hace uso de algunas de las opciones que brinda esta herramienta para el modo de ejecución de extracción. Teniendo en cuenta que la complejidad de este proceso aumenta conforme se definen mayor cantidad de triplas en el archivo de contexto. Se hace necesario hacer uso de la opción llamada *level*, por medio de la cual es posible especificar el número de hilos que de forma paralela actuarán durante el proceso de recolección de datos y generación del archivo de salida tipo RDF.

3.2.2.8 Generar Estadísticas.

Corresponde al proceso en el que es posible realizar labores de análisis de datos a partir de la información procesada en el procedimiento descrito anteriormente (Extracción de Datos) y, cuya base de análisis y representación está fundamentada en el uso de técnicas de Clustering, soportadas por la implementación de algoritmos K-means. Teniendo en cuenta que este capítulo intenta hacer la descripción de las distintas interacciones que se producen entre los usuarios y las acciones del componente, a continuación se detallan los procesos que se llevan a cabo durante la generación de estadísticas:

El primer proceso que deberá llevarse a cabo es la especificación de la fuente dinámica de información que deberá ser sometida al proceso de análisis. Para ello deberá modificarse sobre el aplicativo Java previsto para tal labor, el campo llamado Ubicación Fuente RDF, indicando la URL del sitio donde se encuentra alojado el archivo de salida que representa todas las ideas, comentarios, soluciones y valoraciones contenidas en el sistema gestor. Un punto importante a mencionar en este proceso, es que el archivo fuente RDF al cual se hace referencia no corresponde al archivo que se ha obtenido como resultado del proceso de extracción. Realmente al que se hace referencia es a un archivo que se puede obtener gracias a una funcionalidad del componente RDFME, en el cual bajo el formato RDF es posible obtener de forma dinámica todas las ideas del sistema, incluyendo las que han sido importadas como aquellas que han sido creadas de forma manual.

De momento y contemplando algunas mejoras al futuro que permitan que esta aplicación se integre de forma automática con el componente Gi2SE, actualmente sobre esta aplicación debe modificarse de manera manual las consultas tipos SPARQL basadas en el Framework Jena. Para esto se requiere que los usuarios destinados a la realización de esta labor, tengan algún nivel de dominio con los esquemas de búsqueda basados en SPARQL, de tal forma que pueda inferir cierto tipo de análisis estadístico a partir de la información consultada.

Otro aspecto importante que debe tenerse en cuenta respecto a esta funcionalidad tiene que ver precisamente con la forma en que ha sido concebido el análisis de información, la cual está fundamentada en el proceso de encontrar y agrupar datos que presenten características de similitud. La forma en que realiza este procedimiento se basa en el número de Clusters definidos en la aplicación, los cuales han sido configurados en un valor de 3, la razón de elección de este valor se fundamenta en los resultados obtenidos sobre cerca de 15 pruebas elaboradas para 3500 Ideas del sistema.

El último pasó contemplado en el proceso de generación de estadísticas, corresponde a la ejecución del aplicativo Java a partir del modelo de búsqueda definido en la consulta SPARQL y la especificación del archivo que representa la fuente dinámica de información (utilidad *export plugin rdfme*). Para ello se hace uso de las opciones de compilación presentes en Frameworks como Eclipse o NetBeans, los resultados obtenidos deberán corresponder a graficas de la forma de agrupamiento y de área.

3.2.3 Diagramas de Actividades Buscador Semántico Gi2SE.

Tal como se menciono al inicio de este capítulo. Esta sección pretende abordar de manera general la descripción en forma de diagrama de actividades de los procesos que se llevan a cabo durante el uso y despliegue del componente de búsqueda semántica llamado Gi2SE. Teniendo en cuenta que la descripción de las interacciones que se producen entre el sistema y los procesos de extracción y análisis de datos fueron abordados en el anterior capitulo. A continuación se detallan las acciones que tendrán que realizar tanto el actor administrador como el actor consultor en las etapas de configuración y uso del plugin:

3.2.3.1 Área funcional de Administración (Administrador).

Las interacciones que se llevan a cabo por parte del administrador, contemplan el uso de 5 actividades básicas que el usuario deberá realizar como etapa previa al proceso de búsqueda semántica. La gráfica 3.3 muestra en detalle la forma en que se ejecutan las actividades en mención. Tal vez la actividad de mayor relevancia en esta descripción corresponde al proceso de configuración de opciones, en el cual se llevan a cabo labores de registro de información sobre el sistema relacional que adopta el componente.

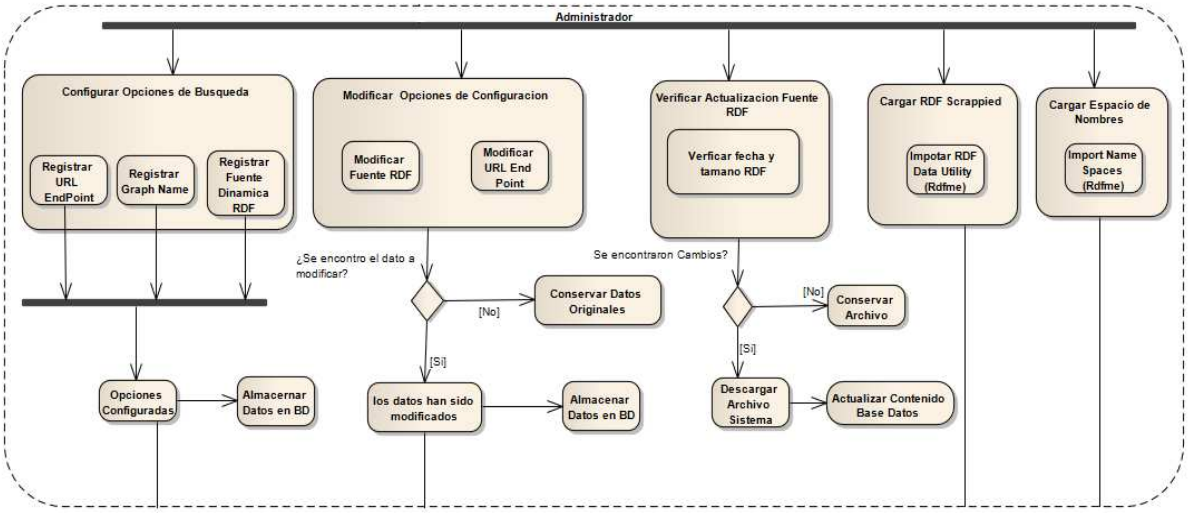


Figura 3.3. Diagrama de Actividades Área Funcional Administración.

3.2.3.1.1 Registrar URL End Point.

Con esta funcionalidad es posible registrar puntos de acceso y consulta remota SPARQL que permitan obtener resultados de búsquedas especificadas de forma local y ejecutada de forma remota. El motivo de implementar funcionalidades de estas características se justifica en el hecho de poder obtener y gestionar información que se encuentre fuera del alcance del dominio de información manejado en este proyecto (Ideas).

Otra característica asociada a esta funcionalidad, es que el componente deberá estar en capacidad de establecer un valor por defecto para el campo URL End Point aun cuando el usuario no lo haya especificado, el cual deberá corresponder a un valor establecido de la forma

[http://localhost/arc2/urlendpoint.php]. Igualmente debe tenerse en cuenta que cuando el URL End Point es usado de forma remota, las clases y métodos utilizados de la librería ARC2 no deberán contemplar almacenamiento de información de forma local.

3.2.3.1.2 Registrar GraphName.

Comúnmente es normal asociar el concepto de GraphName con el conjunto de declaraciones tipo RDF que son identificadas usando un URI **[GI2IS]**. Sin embargo, cuando se habla de este término desde una perspectiva de almacenamiento como los RDF *store* y SPARQL, su entorno de aplicación y uso varían un poco. Con la implementación de esta funcionalidad en el componente Gi2SE se intenta asociar un dominio específico de información con un elemento de descripción única representado en forma de URI con el cual sea posible identificar, referenciar y utilizar de forma más abreviada los mecanismos básicos de consulta previsto por SPARQL. No obstante y, en vista a los recientes avances alcanzados en los últimos meses, respecto a la formas de acceso y uso de las colecciones de grafos en las consultas, como el caso de SPARQL1.1 Graph Store **[APPSW]**, esta solución empieza ser susceptible de cambios, algunos detalles respecto a cambios y mejoras en la definición de los grafos podrá ser consultado en el capítulo 5 de esta memoria.

3.2.3.1.3 Registrar Fuente Dinámica RDF.

Con esta funcionalidad es posible establecer la fuente dinámica de información que será usada para ejecutar los procesos de búsqueda que el usuario llevara a cabo. Un punto importante aclarar sobre este procedimiento es que al momento de declarar el URL de ubicación del archivo RDF, no se intenta decir que la consulta se realizará de forma directa contra este fichero, pues este tipo de enfoque sería una normal y simple consulta SPARQL, lo cual no añadiría ningún tipo de novedad en la realización de este proyecto. Realmente cuando se especifica la fuente RDF lo que se pretende es integrar y manejar dicha información de forma persistente por medio de una serie funcionalidades que ofrece la librería ARC2, el objetivo entonces, es formar una arquitectura en la que los niveles de interoperabilidad entre las tecnologías de la Web Semántica y la Web convencional encuentren un punto de común acuerdo. La figura 3.4 muestra una aproximación al esquema que se desea implementar.

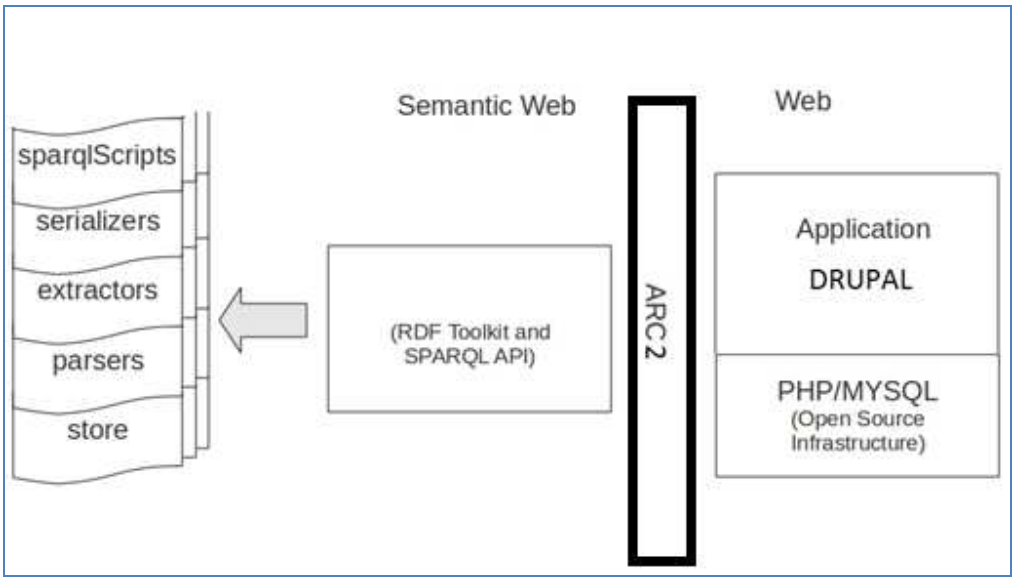


Figura 3.4 Estructura de intermediación de componentes semánticos y convencionales.

De esta manera no sólo se pretende obtener resultados de tipo semántico de forma más adaptable y comprensible a los estándares que hoy por hoy propone la WEB 2.0, sino también proponer cierto tipo de estrategias que permitan hacer que estos servicios sean fácilmente integrables sobre muchas más herramientas y Frameworks. La idea entonces se centra en poder hacer reutilización de servicios que nos permitan acércanos cada vez más aquellos preceptos que propone la WEB 3.0. En el contexto de esta idea el desarrollo de este componente también propone el uso e implementación de la librería Exhibit.

Haciendo uso de Exhibit es posible lograr escenarios en los que la información obtenida no sólo esté condicionada a presentar los resultados de manera tradicional, sino que también, pueda ser usada en otro tipo de aplicaciones como los *Mashups* en donde fácilmente podría representarse información relacionada con algún tipo de propiedad presente en el nivel descriptivo de la ontología de Ideas, por ejemplo haciendo Uso de *Friend of a Friend* (foaf) es posible presentar algún otro tipo de información que se relacione con el autor de una Idea. Un punto importante a mencionar, es que el uso de Exhibit implica adaptar mecanismos de conversión que permitan transformarlos datos obtenidos en el proceso de búsqueda realizado por SPARQL al formato Java Script *Object Notation* (JSON), para ello deberá implementarse una funcionalidad que haciendo uso de la librería ARC2 permita no sólo generar de manera paralela salidas en formato JSON sino también integrar de forma automática dichas salidas sobre el API de Exhibit.

3.2.3.1.4 Verificar Actualización Fuente RDF.

Con esta funcionalidad, es posible actualizar las fuentes de información que se consideren necesarias para realizar procesos de búsqueda de información que presenten algún tipo de modificación con respecto a un estado de búsqueda inicial. Teniendo en cuenta que el componente Gi2SE permite no sólo alojar fuentes de información relacionadas con el sistema gestor de ideas, Gi2mo Ideas Stream, sino que también permite integrar otro tipo de fuentes de información en las cuales los procesos de actualización son gestionados por mecanismos propios del propietario de la fuente RDF. Esta labor implica que en el componente Gi2SE se aborden mecanismos de verificación basados en técnicas de comparación por tamaño y fecha de actualización de archivos. De tal forma que cuando un usuario selecciona esta opción, el sistema deberá inicialmente realizar una comparación de la fecha de actualización del archivo RDF recientemente modificado (expuesto en la WEB) y el archivo RDF local, como local debe entenderse el fichero que se almacena cada vez que una fuente de información es registrada en el sistema (apartado 3.2.3.1.3).

Para el caso de las fuentes de información basadas en el sistema gestor de Ideas Gi2mo Stream, esta estrategia no deberá ser aplicable, dado que al implementar un servicio WEB basado en RESTFUL, el modelo de comunicación se produce sin almacenar estados. De tal forma que la fecha de actualización cambiará conforme se realicen consultas que consuman este servicio. Para ello deberá entonces hacerse uso del criterio de comparación basado en el tamaño de los archivos, donde básicamente por medio de funcionalidades como CURL, propias de PHP es posible obtener un criterio de desigualdad. Cuando el criterio de desigualdad es acertado entonces, la librería ARC2 deberá descargar el nuevo contenido asociado al archivo tipo RDF y hacerlo persistente en las respectivas tablas de la base de datos. La figura 3.5 muestra en detalle la forma en que se lleva a cabo esta funcionalidad.

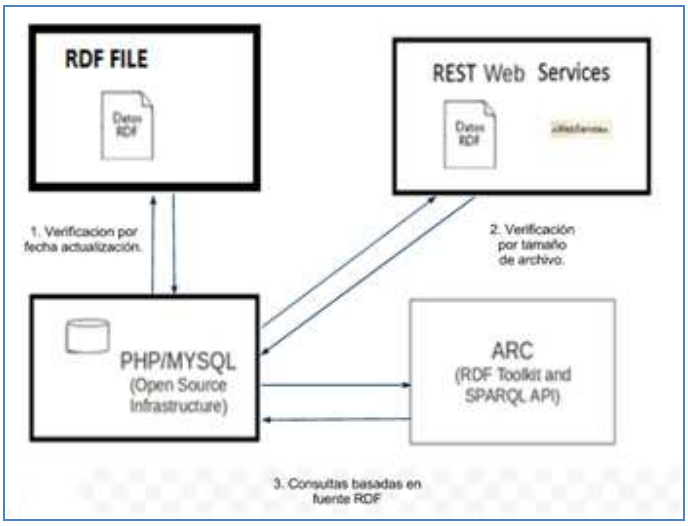


Figura 3.5 Verificación de Fichero RDF.

3.2.3.1.5 Cargar espacios de Nombre.

Haciendo uso de la utilidad de especificación de espacios de nombres ofrecida por el módulo RDFME deberá ser posible ingresar espacios de nombres sobre el sistema relacional que se asocia al componente Gi2SE. De tal forma que cuando los algoritmos que gestionan los procesos de búsqueda intenten asociar las clases y propiedades de un dominio de datos que se encuentre relacionado con un punto de terminación remota, inmediatamente vincularan este valores con los espacios de nombres registrados directamente por el usuario.

Un aspecto interesante a comentar respecto a esta utilidad tiene que ver con las funcionalidades adicionales que éste implementa, pues no simplemente vincula un espacio de nombres sobre un sistema relacional de datos sino que también implementa mecanismos en los cuales con solo dar el URI asociado al Name space de una ontología, obtiene de forma automática el nombre de clases y propiedades que definen dicho sistema. El obtener el nombre de las clases y atributos representa un punto importante en los esquemas de funcionalidad del componente, dado que mediante la implementación de este tipo de utilidades es posible que el usuario tenga una forma de búsqueda más comprensible ya que eventualmente no necesitaría conocer a fondo las propiedades y clases del dominio de información cuando trata de hacer la búsqueda.

3.2.3.2 Área funcional de Búsqueda (Usuario Sistema).

Las interacciones que se llevan a cabo en el contexto del usuario final contemplan el uso de dos actividades básicas que deberán realizarse como etapa previa al proceso de búsqueda. La gráfica 3.6 muestra en detalle la forma en que se ejecutan las actividades en mención. Tal vez la actividad de mayor relevancia en esta descripción corresponde al proceso llamado especificar número de clases y número de propiedades utilizadas en las opciones de búsqueda.

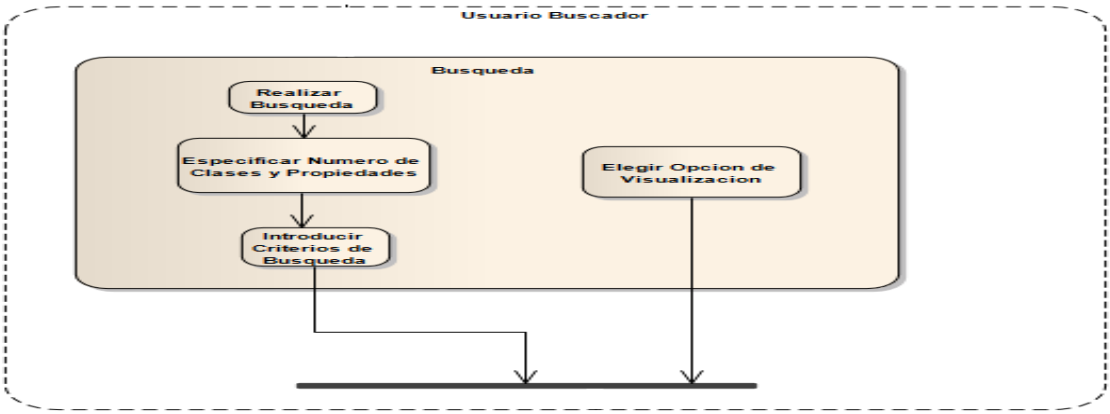


Figura 3.6. Diagrama de Actividades Área Funcional Usuario

3.2.3.2.1 Especificar número de Clases y Propiedades.

Con esta funcionalidad es posible especificar el número de clases y propiedades que pueden ser utilizadas durante el proceso de búsqueda de información, el motivo de hacer uso de funcionalidades de este tipo, se fundamenta básicamente en el concepto de adaptación / aproximación, que deben contemplar actualmente los sistemas gestores de búsqueda semántica respecto a la forma de implementar estrategias en las que el usuario pueda fácilmente relacionar los conceptos de forma sencilla, es decir, ante el marcado nivel de des ambigüedad que suele presentarse cuando un usuario introduce un cierto número de palabras sobre una caja de texto, se requiere implementar algún tipo de solución que temporalmente pueda solucionar este problema.

La solución planteada deberá comprender una estrategia en la cual sea posible crear de forma dinámica objetos representados en forma de clases y atributos, mediante los cuales el usuario pueda intuitivamente crear y extender relaciones. Para cumplir con este objetivo, este trabajo fin de Máster propone el uso de herramientas y librerías que basadas en funcionalidades como Evoc

y AhAh permitan que un usuario pueda hacer uso de las opciones de auto completado de texto. Esto facilita el proceso de conocimiento de información de un sistema ontológico sin previo estudio. Además de ello, la creación de clases de forma dinámica propone una condición de uso bastante favorable para la relación de propiedades y atributos de diferentes dominios de conocimiento, ya que con sólo dar un click es posible formar cadenas de la forma:

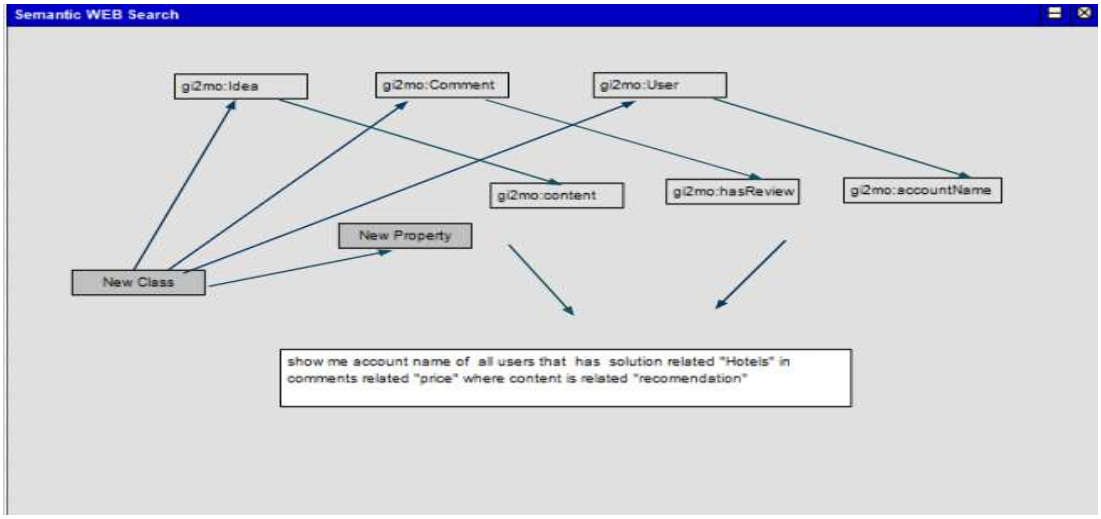


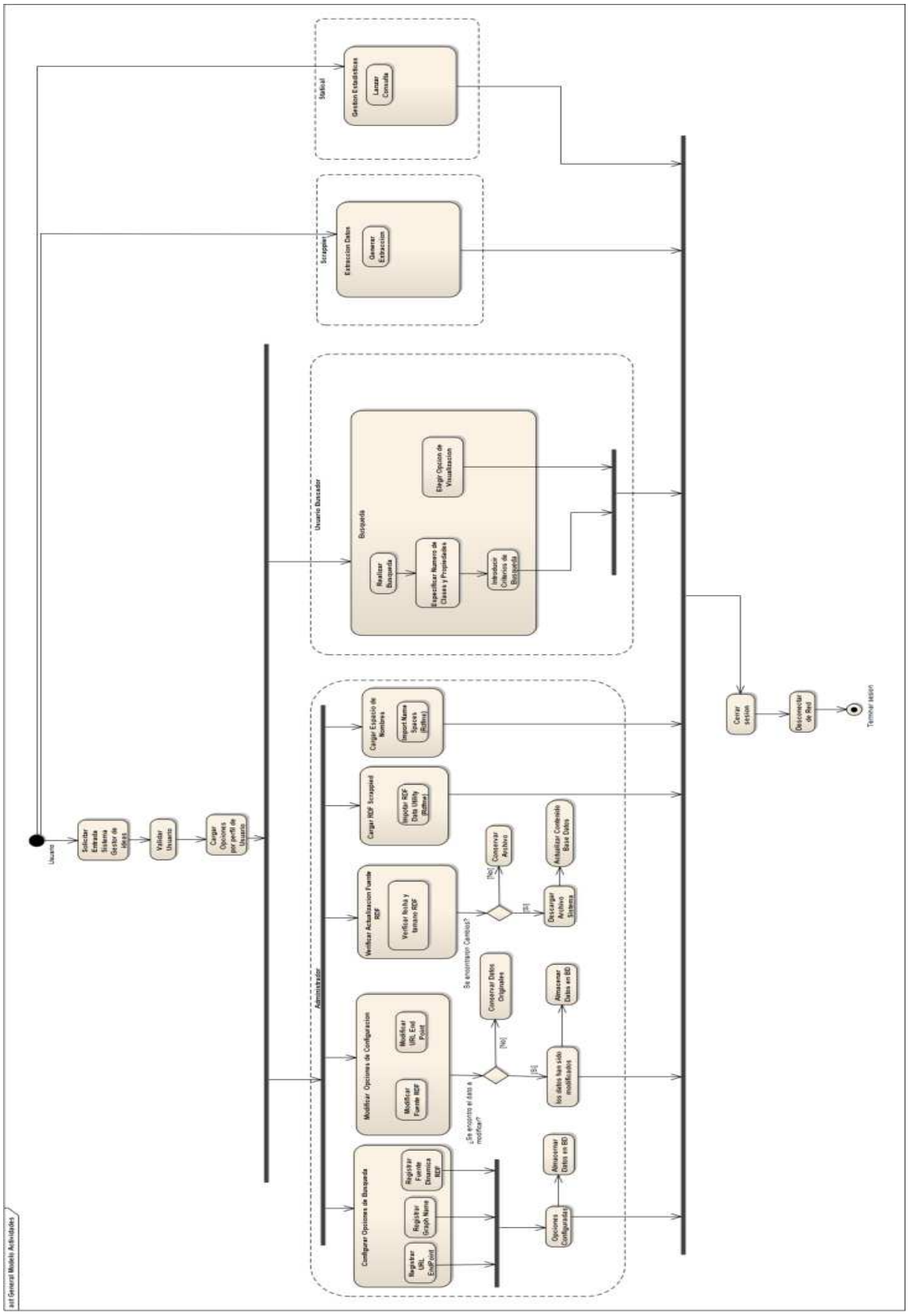
Figura 3.7. Asociación dinámica de clases y objetos.

3.2.3.2.2 Elegir Opciones de Visualización.

Con esta funcionalidad es posible especificar la forma de visualizar los resultados obtenidos durante el proceso de búsqueda de información. Sin duda este proceso representa uno de los mayores atractivos en la realización de este proyecto, ya que mediante el, no sólo es posible establecer modelos gráficos de representación de la información (GUI), sino que también es posible tomar esta actividad como punto de partida para procesos de comparación relacionados con el uso de técnicas de *Faceted Browsing* (Exhibit) y técnicas de búsqueda semántica tradicionales. Además de lo anterior, es importante resaltar que la aplicación de este tipo de técnicas puede llevar al estudio de una nueva gama de servicios que basados en el concepto de *Mashup*, eventualmente puedan formar funcionalidades adicionales a partir del uso de los resultados obtenidos en la búsqueda de información.

3.3 Conclusiones.

La fase de análisis de requerimientos ha permitido identificar la línea base del proyecto, logrando establecer las interacciones y, acciones que se presentan entre las distintas actividades que se llevan a cabo en el componente Gi2SE. La fase de análisis realizada en este proyecto no sólo ha permitido determinar las necesidades básicas que deberá suplir el sistema a través del desarrollo del componente de software en estudio, sino que también ha permitido establecer los modelos y especificaciones necesarias para abordar la etapa de diseño. Sin duda, la técnica de análisis estructurado ha representado una muy buena alternativa para la construcción de modelos a partir de procesos y / o actividades, ya que el comportamiento que estos adoptan puede ser fácilmente usado y extendido para determinar todas y cada una de las relaciones estructurales del sistema, por medio de diseños arquitectónicos.


Usuario
Solicitar Entrada Sistema Gestor de Ideas
Validar Usuario
Cargar Opciones por perfil de Usuario
Estadistical
Gestion Estadisticas
Lanzar Consulta
Scrapper
Extracion Datos
Generar Extracion
Usuario Buscador
Busqueda
Realizar Busqueda
Especificar Numero de Clases y Propiedades
Introducir Criterios de Busqueda
Elegir Opcion de Visualizacion
Administrador
Cargar Espacio de Nombres
Import Name Spaces (Rdfxx)
Cargar RDF Scrapped
Importar RDF Data Utility (Rdfxx)
Verificar Actualizacion Fuente RDF
Verificar fecha y tamano RDF
Se encontraron Cambios?
[No]
Conservar Archivo
[Si]
Descargar Archivo Sistema
Actualizar Contenido Base Datos
Modificar Opciones de Configuracion
Modificar Fuente RDF
Modificar URL End Point
¿Se encontro un dato a modificar?
[No]
Conservar Datos Originales
[Si]
los datos han sido modificados
Almacenar Datos en BD
Configurar Opciones de Busqueda
Registar URL EndPoint
Registar Graph Name
Registar Fuente Dinamica RDF
Opciones Configuradas
Almacenar Datos en BD
Cerrar sesion
Desconectar de Red
Terminar sesion
act General Models Actividades

4. Capítulo IV Arquitectura.

4.1 Introducción.

El presente capítulo introduce el diseño arquitectónico que describe las diferentes relaciones e interacciones que se llevan a cabo entre las actividades y elementos que conforman el sistema y que fueron descritas en el capítulo 3. Siguiendo algunos de los principios de Ingeniería de Software expuestos en el Modelo Unificado de Lenguaje (UML). Este proyecto fin de Máster pretende desarrollar una componente de software en el cual el concepto de modularidad sea extendido y caracterizado sobre la herramienta gestora de contenidos Drupal. Para lograr este objetivo se seguirán técnicas de programación en las que sea posible utilizar el concepto de independencia funcional estableciendo métodos y clases que cuenten con definiciones y descripciones lo suficientemente claras que eviten una excesiva dependencia entre funcionalidades.

La correcta aplicación de estas técnicas deberá conducirnos a escenarios en los cuales sea posible no sólo obtener niveles de composición y descomposición lo suficientemente claros, sino también a establecer modelos de comunicación en los cuales los componentes adopten las interacciones estrictamente necesarias. En términos de programación de software este tipo de actividades se consiguen reduciendo la cantidad y complejidad de los parámetros presente en las operaciones. Además de ello, también será posible conseguir escenarios en los cuales se garantice un nivel de cohesión lo suficientemente alto entre los elementos internos de un componente.

Con el fin de mostrar la relación estructural y las interacciones que se llevan a cabo entre los diferentes elementos que conforman la arquitectura del sistema. Este trabajo fin de Máster hará las especificaciones necesarias que permitan conocer la relación dinámica que se establece entre cada uno de estas entidades a través de diagramas de secuencia. Igualmente se especificara el comportamiento que adopta el sistema en función del tiempo, representando diagramas de actividades que guíen paso a paso los flujos de trabajo presentes en el sistema. Otro importante aspecto que contempla la etapa de diseño, es la realización del modelo arquitectural representado en forma componentes. Allí se especificara la forma en la cual interactúan las interfaces tipo Hook del gestor de contenidos y los módulos propios del proyecto Gi2mo, como es el caso de

Rdfme y Gi2SE.

La etapa de diseño también contempla el uso del modelo conceptual de datos representado en forma de Modelo Entidad – Relación. Lo anterior con el fin de dar una visión global respecto a la forma en que se maneja la persistencia en el componente.

4.2 Arquitectura Software – Patrones de Diseño

La arquitectura propuesta en este trabajo fin de Máster, adopta una serie de principios y conceptos que tienen como objetivo introducir elementos que permitan alcanzar niveles de calidad de software lo suficientemente estableces en términos de robustez, flexibilidad y escalabilidad. En esta sección se detallaran los aspectos más relevantes acerca del concepto patrón de diseño como principal componente de solución común a los problemas de calidad presentes durante las fases de diseño y codificación de las aplicaciones **[GI2IS]**.

4.2.1 Patrón de Software Plugin.

Teniendo en cuenta la importancia que representa el uso de patrones de diseño dentro de la etapa de construcción y codificación de componentes de software. Es importante mencionar que durante la realización de este proyecto se hizo necesaria la utilización de un patrón diseño clasificado en la categoría de comportamiento. Puntualmente se está haciendo referencia al patrón llamado *Plugin*, el cual permite vincular y crear objetos sobre un componente en ejecución, por medio de la exposición de diferentes tipos de interfaces asociadas a un servicio en particular.

Haciendo uso de este tipo de patrones, básicamente se aboca al concepto de modularidad expuesto en el apartado anterior. Dado que el comportamiento que desea implementar propone "el enchufe" de nuevas clases u objetos sobre un componente principal sin necesidad de modificar o re desplegar este artefacto. Sin duda la adopción de este tipo de patrones favorece sustancialmente a grandes proyectos de software libre como Drupal, ya que ofrece la posibilidad de que un tercero añada nuevas funcionalidades sobre su gestor de contenidos. Así, la fase construcción del módulo Gi2SE ha seguido este patrón de diseño con el fin de implementar funcionalidades que permitan realizar procesos de búsqueda semántica sobre dicho gestor de contenidos. La figura 4.1 representa la forma en que interactúan el componente Gi2SE y el núcleo de funcionalidades presentes en el gestor de contenidos Drupal.

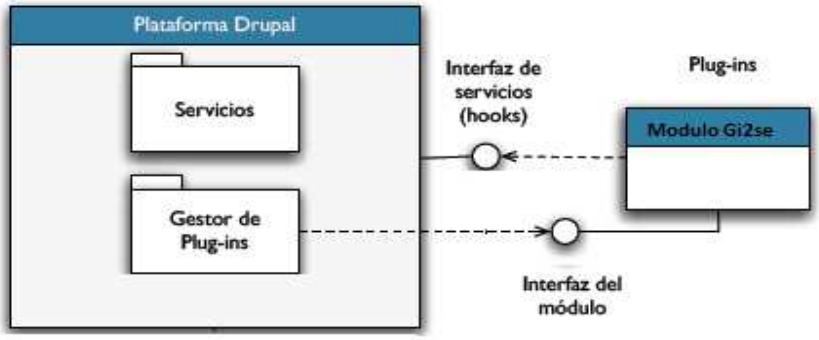


Figura 4.1. Patrón de Diseño Plugin aplicado sobre Gi2se.

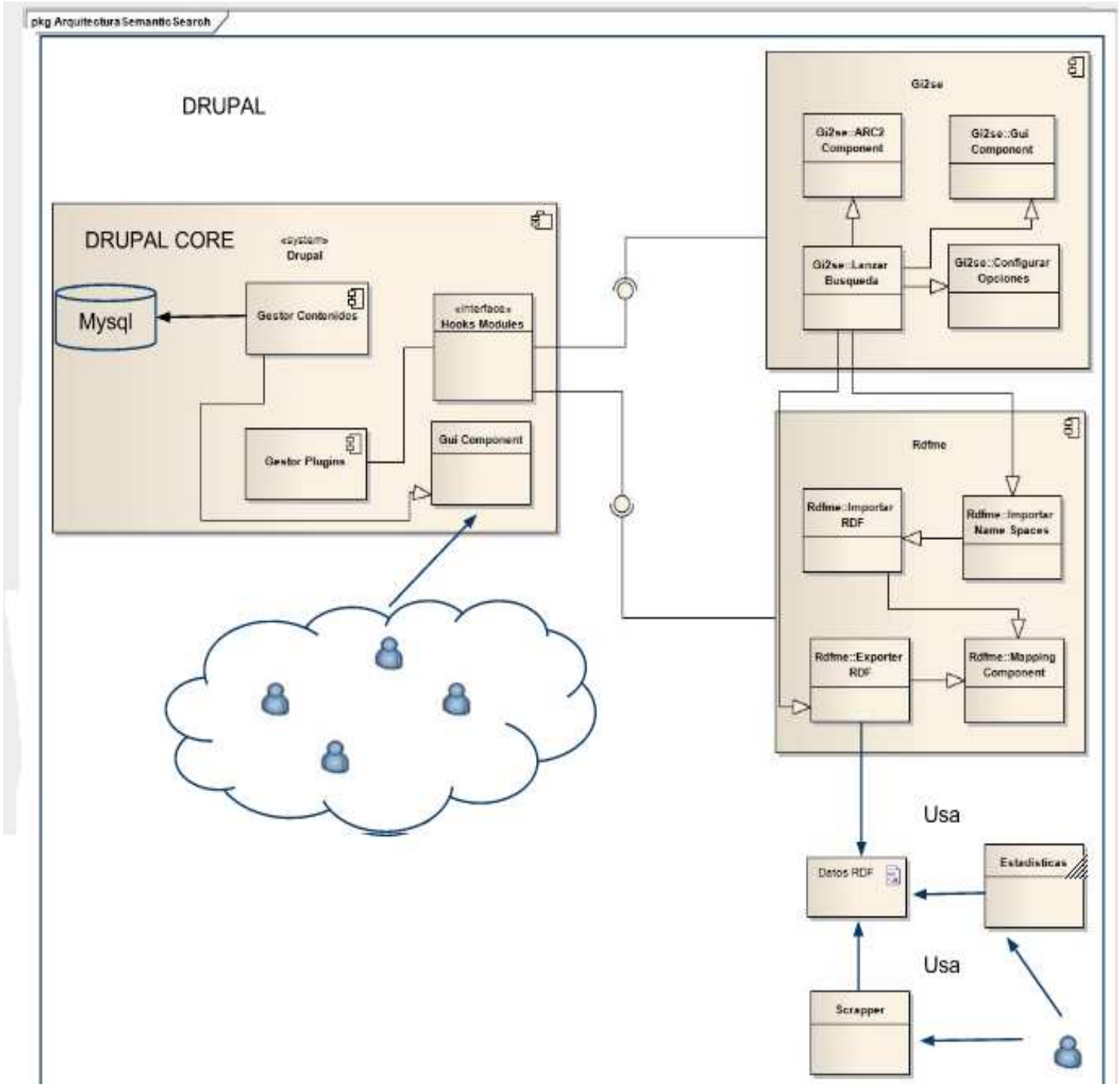
A continuación se detallara la forma y características más importantes de funcionamiento del patrón de diseño llamado Plugin sobre el sistema gestor de contenidos de Drupal:

Inicialmente la aplicación principal (núcleo Drupal) proporciona una serie de funcionalidades expuestas en forma de descripciones a través de interfaces tipo Hooks, en las cuales se exhiben los servicios que los complementos pueden utilizar. Estas descripciones además de incluir un método para el registro de las aplicaciones también poseen un protocolo para habilitar el intercambio de datos e información. La forma en la cual se consumen estas interfaces es a través del API de métodos y clases que provee la interfaz de programación de Drupal (*http:// api.drupal.org/*).

Las actividades que permite realizar este patrón de diseño sobre el gestor de contenidos Drupal se pueden resumir haciendo referencia a las siguientes características:

- Permite que los desarrolladores externos colaboren de manera activa con el núcleo de aplicaciones de Drupal, incorporando nuevas funcionalidades adaptadas en forma de plugin.
- Permite construir modelos de software en los cuales el concepto de modularidad puede aplicarse eficientemente, pues la fácil composición y descomposición de componentes hace que no allá una excesiva dependencia funcional.
- Permite que los módulos preserven estados de configuración, de tal forma que sea posible que un módulo no necesariamente deba ser des instalado aun cuando no se desee usar, sino que pueda tomar el estado des habilitado (hook_disable).

Con base a las descripciones realizadas anteriormente, este proyecto busca proponer un modelo arquitectónico en el que sea posible reducir al máximo el acoplamiento entre componentes y aumentar la cohesión interna de cada uno de ellos, el objetivo se centra en alcanzar un escenario como el de la figura 4.2.


pkg Arquitectura Semantic Search
DRUPAL
DRUPAL CORE
«system»
Drupal
Mysql
Gestor Contenidos
«interface»
Hooks Modules
Gestor Plugins
Gui Component
Gi2se
Gi2se::ARC2
Component
Gi2se::Gui
Component
Gi2se::Lanzar
Busqueda
Gi2se::Configurar
Opciones
Rdfme
Rdfme::Importar
RDF
Rdfme::Importar
Name Spaces
Rdfme::Exporter
RDF
Rdfme::Mapping
Component
Usa
Datos RDF
Estadisticas
Usa
Scrapper

4.3. Diseño Módulo Gi2se.

La fase de diseño del módulo propuesto para este trabajo fin de Máster (Gi2se), contempla el uso de tres (3) actividades de software que permitirán generar flujos de trabajo sobre 4 procesos que se consideran de nuevo uso y sobre 4 procesos que hacen uso de funcionalidades extendidas de una serie de complementos que se desarrollaron en etapas anteriores en el proyecto Gi2mo. A partir de los casos de uso expuestos en el capítulo de análisis y empleando la arquitectura presente en la figura 4.2, a continuación se presentará la implementación de las diferentes funcionalidades que hacen parte del módulo Gi2se.

4.3.1 Configuración de Opciones de Búsqueda

Corresponde al conglomerado de opciones que permiten realizar los procesos de configuración necesarios para la búsqueda semántica de información. Esta actividad adopta una serie de funcionalidades que permiten establecer parámetros como la fuente dinámica de información, nombre del grafo que representara las triplas de información y valor del Url End Point que se desea configurar.

4.3.1.1 Configuración RDF File.

4.3.1.1.1 Descripción

Esta funcionalidad permite establecer un valor URL para especificar el lugar de ubicación de la fuente RDF que se desea utilizar como fuente de datos en el proceso de búsqueda. A diferencia de los tradicionales esquemas de búsqueda basados en tecnologías SPARQL y RDF, esta utilidad permite actualizar el contenido presente en los archivos de forma dinámica.

4.3.1.1.2 Diseño e Implementación.

La figura 4.3 muestra en detalle las interacciones que se llevan a cabo entre los diferentes objetos que componen la actividad de configuración de la fuente dinámica de información. El primer paso que se lleva a cabo durante este proceso, es la verificación del contenido del archivo tipo RDF, el cual en caso de ser especificado por primera vez deberá ser descargado sobre la carpeta

llamada *Download* ubicada en la ruta donde residen los archivos de configuración e instalación de Gi2SE.

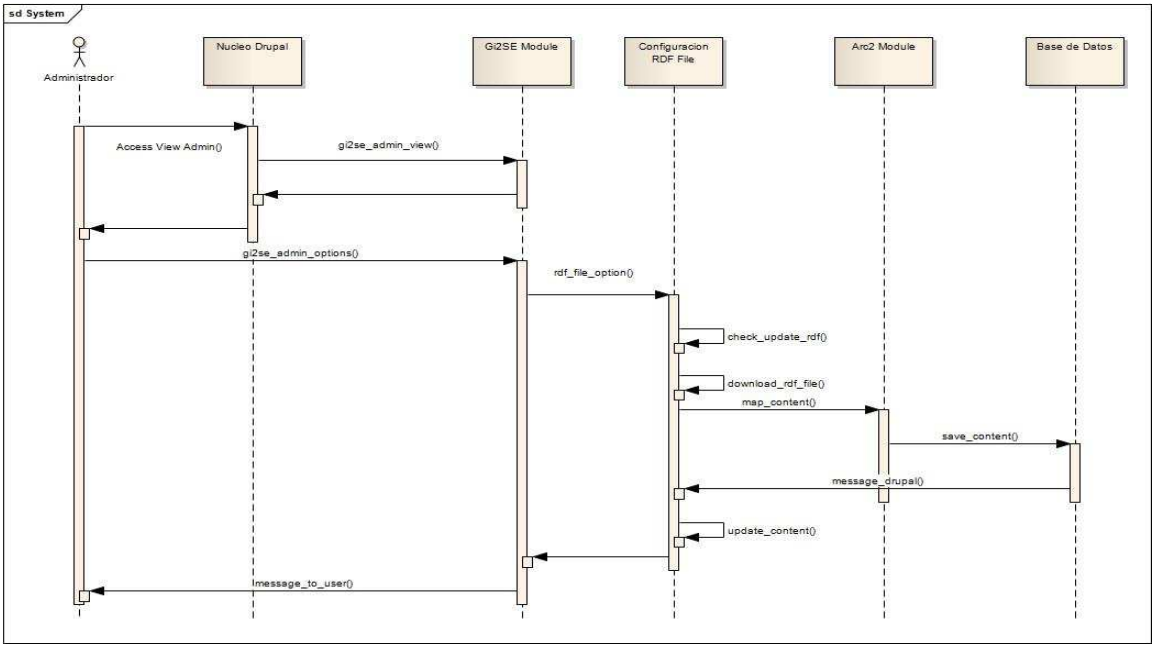


Figura 4.3 Diagrama de Secuencia de la actividad de configuración de Archivo RDF.

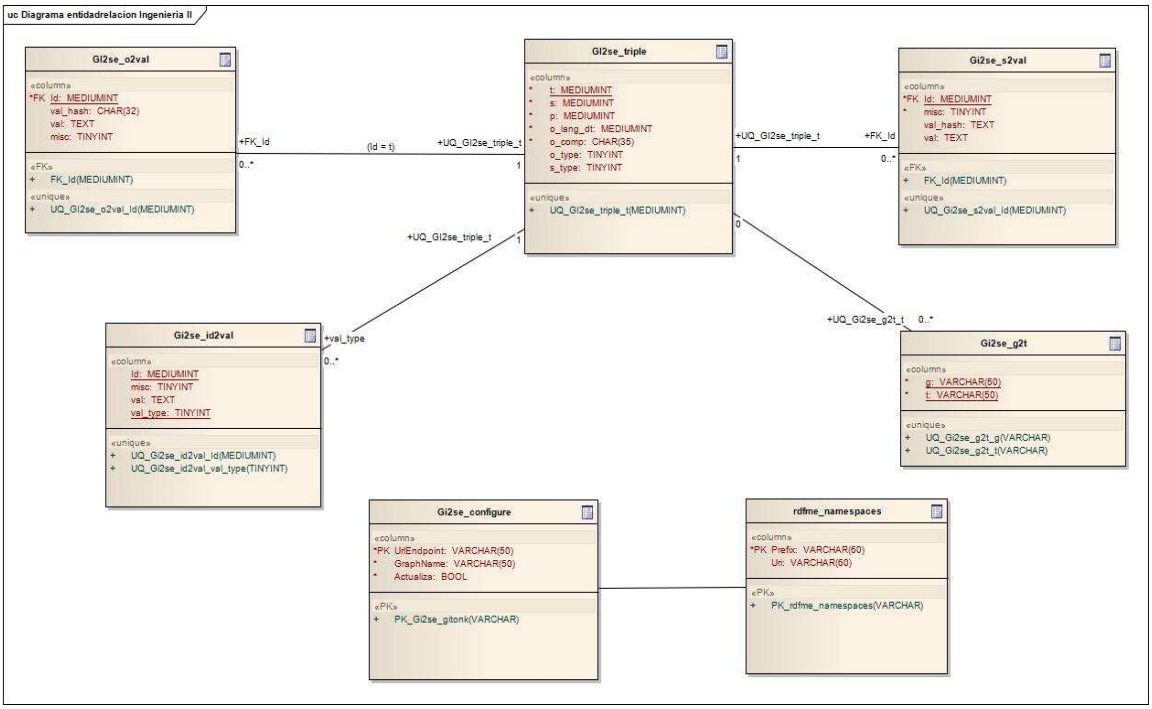
En caso que la fuente RDF haya sido especificada con antelación, la actividad deberá plantear un mecanismo de verificación en el cual dependiendo de la forma en que se encuentre publicado el archivo tipo RDF (refiérase a la sección 3.2.3.1.4), hará la comparación de contenidos con base a la fecha de actualización o tamaño. Es importante mencionar que estos mecanismos adoptan el uso de funcionalidades extendidas de librerías como Curl de PHP.

Cuando efectivamente se comprueba que la fuente de información ha sufrido cambios con respecto al contenido de la primera versión almacenada, este procedimiento descarga el archivo más actualizado y lo pone a disposición de la actividad de mapeo y almacenamiento que ofrece la herramienta Arc2. Respecto a la actividad de persistencia, el apartado 4.3.1.1.3 detalla un poco más al respecto. El tema referente a los mapeos será tratado en el apartado 4.3.3.

El último paso que se lleva a cabo durante la ejecución de esta actividad, es la actualización de la vista de configuración del usuario. Puntualmente del contenido ingresado como fuente RDF, ya que a medida que se hace un nuevo ingreso de fuente de información, una tabla ubicada en la parte inferior de la vista del usuario deberá mostrar este valor. Además deberá asociar este nombre con una serie de opciones que se puedan realizar sobre él, como es el caso de borrar y actualizar.

4.3.1.1.3 Modelo Relacional.

El presente apartado hará referencia al modelo relacional que ha sido adaptado en la fase de diseño del componente Gi2se. La figura 4.4 muestra la estructura. Uno de las decisiones de diseño adoptadas buscando sencillez y claridad, ha sido comenzar los nombres de todas las tablas, que pertenecen directamente al módulo, con el prefijo "*gi2se_*".



4.3.1.1.3 Tabla Gi2se_Configuracion.

Respecto a la figura 4.4, es importante decir que este modelo no sólo sirve como referente a la actividad relacionada con la configuración de la fuente RDF. Este representa en general la totalidad de las actividades de tipo persistente que se llevan a cabo en el componente. La primera tabla analizar corresponde a Gi2se_Configuracion, esta tabla tiene como función almacenar todos los valores de Url donde se ubican las fuentes de información representadas en forma de RDF. Así mismo, esta tabla posee un par de campos adicionales llamados GraphName y bandera cuyas funciones básicas son almacenar el nombre del grafo que será vinculado directamente con el archivo RDF y servir como elemento de referencia para la selección de fuentes a actualizar respectivamente.

4.3.1.1.3 Tabla Gi2se_g2t.

Por otra parte, se tiene la tabla Gi2se_g2t, la cual contiene los campos llamados numberGraph e ítems. Básicamente la labor de la columna numberGraph se centra en almacenar por cada ítem procesado en la actividad de mapeo, el número del grafo bajo el cual será almacenado cada uno de los elementos tratados. Acerca de la columna ítem, sobre este campo se almacena en forma auto incremental el número total de elementos que han sido introducidos a la base de datos, bien sea en forma de sujeto, predicado u objeto.

4.3.1.1.3 Tabla Gi2se_triple.

Siendo la tabla de mayor relevancia dentro del modelo relacional, Gi2se_triple define la estructura de organización adoptada para la búsqueda de información sobre los nodos y arcos almacenados de forma persistente. Básicamente cuenta con 7 campos en los que aloja la relación entre los nodos y arcos de las triplas, los campos de mayor relevancia corresponden a t (item), s (sujeto), p (predicado) y o (objeto). La forma en la cual se estructura y organiza la información proveniente de la etapa del mapeo, se centra básicamente en el uso de un campo tipo UNIQUE, el cual sirve como elemento referente de asociación y permite relacionar todos los valores correspondientes a una clase en particular. Así por ejemplo, sobre esta tabla será posible observar información como la observada en la tabla 4.1

i (ítem)	s (sujeto)	p (predicado)	o (objeto)
1	2	3	4
2	2	6	7
3	2	8	9

Tabla 4.1. Campos de la tabla Gi2se_triple

Para el caso de la primera columna el valor de t corresponde a 1, dado que ha sido procesado como el primer elemento del mapeo. El valor 4 corresponde al valor de la clase que representa dicho elemento, gi2mo: User, por ejemplo. El valor p =3 corresponde al tipo de atributo que representa, rdf: about en este caso. En tanto el valor s = 2 corresponde al valor de la clase. De esta forma la estructura tipo Rdf desde la cual se han obtenidos los datos, seria de la siguiente forma:

```
<gi2mo:User rdf:about="http://brainstorm.ubuntu.com/contributor/joel_gabor/tordf">
```

Ahora bien, con base en la representación hecha sobre la primera estructura, los demás elementos que conforman la tabla siguen un patrón de comportamiento similar sobre las propiedades y valores presentes en la clase gi2mo:User. Es de aclarar qué la forma en la cual se asocian los valores presentes en un cualquiera de las clases, es a través de la información que se encuentra en el atributo que las distingue. En este caso en particular, será rdf:about="http://brainstorm.ubuntu.com/contributor/joel_gabor/tordf">. Así, es posible decir que el segundo y tercer ítem de la tabla han sido obtenidos a partir de una estructura tipo RDF como la observada en la figura 4.5:

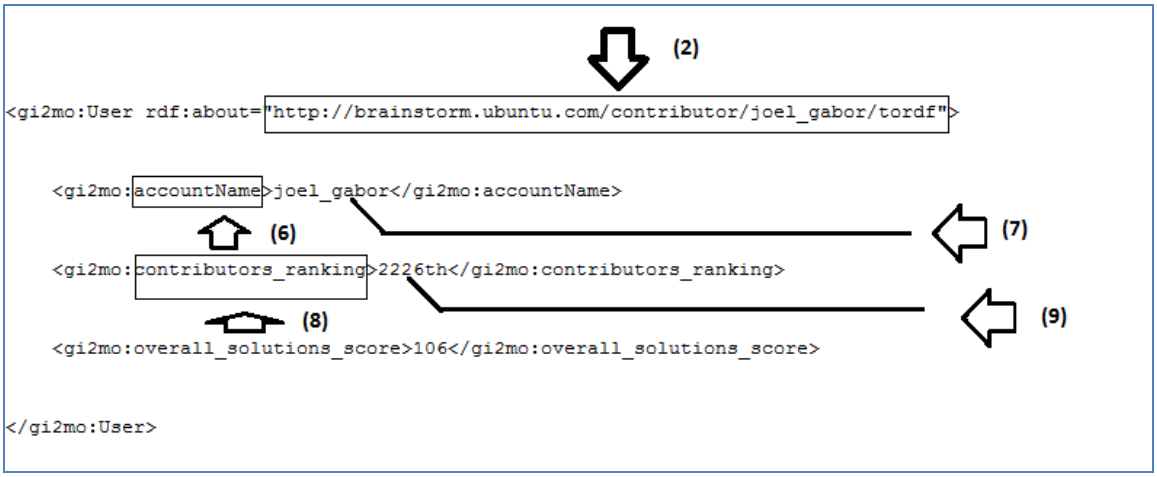


Figura 4.5 Modelo de datos almacenado en tabla Gi2se_triple.

4.3.1.1.3 Tablas Gi2se_o2val – Gi2se_s2val.

Igualmente el modelo relacional propone el uso de un par de tablas adicionales llamadas gi2se_o2val y gi2se_s2val, en las cuales sólo se almacena información relacionada con los nodos (Sujeto y Objeto) obtenidos del proceso de mapeo. Estas tablas cuentan una estructura de organización similar a la tabla gi2se_triple pero con campos de mayor simplicidad. Cuentan con 1 columna llamada “val” en la que por medio de un “id” alojan el valor del atributo que identifica y distingue una clase de las demás y por el cual se establecen relaciones con otras clases. Igualmente cuenta con un campo llamado “val_hash”, en el cual se almacena un valor único que distingue el atributo de una propiedad en caso de que el nombre de estas se repitan en ocasiones futuras.

Así por ejemplo, sobre estas tablas será posible encontrar información como la observada en la tabla 4.2 La idea de estas tablas es referenciar los valores de los elementos que caracterizan una clase, con el fin que las consultas a ser realizadas puedan usar los valores de las subclases y propiedades, es decir, esta tabla sirve como puente para poder entregar los resultados de las

consultas que exigen el uso de múltiples atributos sobre una clase.

Id	val_hash	Val
192	1010073298	http://brainstorm.ubuntu.com/idea/27296/coment/1

Tabla 4.2. Ejemplo de estructura de las tablas gi2se_o2val y gi2se_s2val

4.3.1.1.4 Algoritmo Principal.

Como se menciono en el apartado 4.3.1.1.2, el algoritmo utilizado en la funcionalidad de configuración de la fuente dinámica de información, hace uso de la librería Curl para el proceso de verificación de contenidos de la fuente RDF. Este algoritmo básicamente propone la comprobación con base en dos criterios. El primero de ellos hace referencia a la fecha de modificación del archivo, en tanto el segundo criterio toma como referencia el tamaño del mismo. No obstante, puede surgir la pregunta respecto al por qué o cuándo se hace uso de alguno de estos criterios. La respuesta a este interrogante se justifica en la forma en que la fuente de información tipo RDF ha sido expuesta para su actualización, es decir, cuando el fichero que se requiere verificar corresponde a un archivo que no sufre cambios en el tiempo bien sea por que es el resultado final de un proceso de extracción o ha sido generado manualmente, la mejor técnica a considerar tiene que ver con comprobación de la fecha de actualización. La figura 4.6 muestra en detalle la abstracción utilizada en este algoritmo.

```
$fp = @fsockopen($host, '80', $errno, $errstr, $connTimeout );
     // Header Info
     $header = "GET $path HTTP/1.0\r\n";
     $header .= "Host: $host\r\n";
     $header .= "User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.6) Gecko/20070725 Firefox/2.0.0.6\r\n";
     $header .= "Accept: */*\r\n";
     $header .= "Accept-Language: en-us,en;q=0.5\r\n";
     $header .= "Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n";
     $header .= "Keep-Alive: 300\r\n";
     $header .= "Connection: keep-alive\r\n";
     $header .= "Referer: http://$host\r\n\r\n";
     $response = '';
     fputs($fp, $header);
     // Get the file content
     while($line = fread($fp, 4096)){
          $response .= $line;
     }
     fclose( $fp );

    if (file_exists($localFile))
    {

       // check the updated and remote date may 08 2011
       // date last modified local file system

      $timetemp=filemtime($localFile);
      $timeLocal= date ('d/m/y H:i:s', $timetemp);

       // date last modified  remote  file system

      $curl = curl_init();
      curl_setopt($curl, CURLOPT_URL,$remoteFile);
      curl_setopt($curl, CURLOPT_NOBODY, 1);
      curl_setopt($curl, CURLOPT_FILETIME, TRUE );
      $result = curl_exec ($curl);
      $time = curl_getinfo($curl, CURLINFO_FILETIME);
      $remoteTime = date('d/m/y H:i:s', $time);
      curl_close ($curl);
```

Figura 4.6 Algoritmo comprobación fecha actualización Configuración RDF.

Por otra parte, cuando el contenido del archivo que se requiere verificar es susceptible a cambios en el tiempo, bien sea por que es expuesto en forma de servicio Web o porque tiene procesos que de forma automática actualizan el contenido del archivo, la mejor técnica a considerar es la verificación por comparación en el tamaño del archivo. La figura 4.7 muestra en detalle la abstracción utilizada en este algoritmo.

```
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $remoteFile);
curl_setopt($ch, CURLOPT_HEADER, true);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$site = curl_exec($ch);
curl_close($ch);
if (preg_match('/Content-Length: (\d+)/', $site, $matches))
  {
   $contentLength = (int)$matches[1];
  }
// Ckeck  Local Size File then
$filename = $localFile;
$sizefilename = filesize($filename);


if ( $contentLength != $sizefilename)
      {
        if(is_writable($localFile))
             {
              if($fp = fopen($localFile, 'w'))
                {
                 fwrite($fp, $response);
                 fclose($fp);
                 return $back=1;
                }
             }
      }
```

Figura 4.7 Algoritmo comprobación tamaño Configuración RDF.

Un aspecto adicional a mencionar respecto al proceso de verificación de cambios en el contenido de los archivos, es que esta funcionalidad ha sido adaptada para ofrecer cierto nivel de interoperabilidad con fuentes de información que cambien de forma dinámica en el tiempo. De manera particular debe hacerse referencia al sistema gestor de Ideas llamado Gi2mo Ideas Stream, el cual corresponde a un componente de software desarrollado en forma de plugin sobre el gestor de contenidos Drupal y que tiene la funcionalidad de crear y publicar contenidos de ideas de forma semántica.

El porque utilizar este componente dentro de la actividad de búsqueda de Gi2se tiene básicamente dos razones, la primera de ellas es que este componente cuenta con características de importación de archivos tipo RDF en forma de Web Services (obsérvese figura 4.2 componente Rdfme). Esto hace que parte de los contenidos extraídos por minería de datos puedan ser fácilmente volcados sobre el sistema gestor. La temática relacionada con los procesos de extracción de datos será abordada en el apartado 4.4. La segunda razón, es que este sistema tiene la capacidad de importar todo el contenido de las ideas en archivos tipo RDF en forma de Web Services (obsérvese figura 4.2 componente Rdfme). Esto hace que la información contenida en este repositorio de datos sirva como elemento de entrada para el proceso de búsqueda planteado por Gi2se.

De esta forma la fuente de información podrá ser especificada en forma de Uri y ser fácilmente integrable sobre la librería Arc2, quien en su esquema de funcionamiento requiere de un Url para poder hacer el proceso relacionado con los mapeos. Esta temática será explicada en el capítulo 4.3.2. La idea se centra en hacer uso de herramientas que puedan ser lo suficiente interoperables. Lo anterior, con el fin que los resultados ofrecidos por la búsqueda puedan ser utilizados a futuro por otros servicios.

4.3.1.1.5 Interfaz Gráfica.

El diseño de la interfaz de configuración ha adaptado mecanismos en los que la sencillez y claridad son claves en la forma en que el usuario interactúa con el sistema. En la etapa de configuración de los parámetros de búsqueda, básicamente se ha propuesto el uso de elementos gráficos basados en la interfaz de aplicaciones de Drupal (API). Haciendo uso de funcionalidades previstas en la función Theme de Drupal, ha sido posible no sólo utilizar los formularios que permiten el ingreso de información, sino también gestionar de forma dinámica los datos ingresados por el usuario. De esta forma el contenido que se obtiene como resultado del ingreso de información, puede ser manipulado sin necesidad de hacer uso de páginas externas o de actividades de recarga sobre las páginas. La figura 4.8 muestra más al respecto.

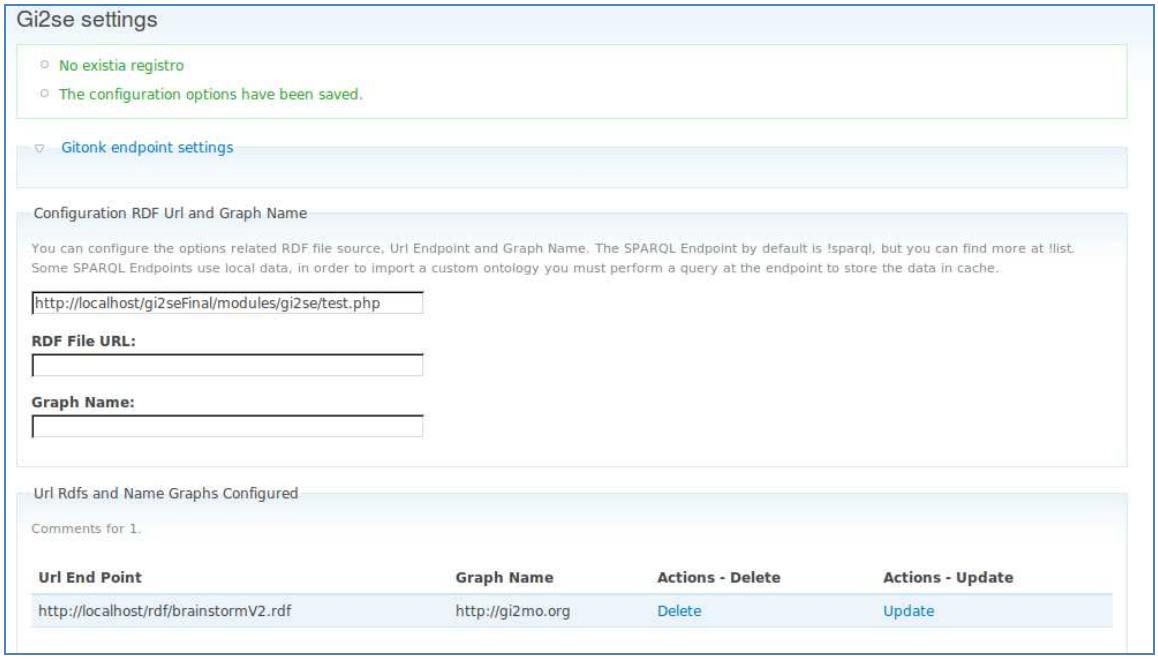


Figura 4.8 Interfaz gráfica opciones configuración Gi2se.

4.3.1.2 Configuración SPARQL EndPoint.

4.3.1.2.1 Descripción

Esta funcionalidad permite establecer el valor del Url por medio del cual es posible especificar el lugar donde deberán ejecutarse las búsquedas de tipo semántico. La razón por la cual es posible habilitar una fuente de consulta remota de información, es debido a que se quiere implementar mecanismos con los cuales el buscador semántico tenga la habilidad de ofrecer servicios de búsqueda genéricos, en los cuales las consultas no estén condicionadas a dominios locales de información, es decir, se pretenden construir escenarios de búsqueda que no sólo permitan ejecutar consultas de forma local sino también de forma de remota. Igualmente se pretende habilitar un escenario que haciendo uso de parámetros como el Url End Point permita consumir de forma remota el servicio de búsqueda que propone Gi2se.

4.3.1.2.2 Diseño e Implementación.

La figura 4.9 muestra en detalle las interacciones que se llevan a cabo entre los diferentes objetos que componen la actividad de configuración del punto remoto de búsqueda Sparql. La actividad de configuración que se lleva a cabo durante este procedimiento, simplemente corresponde a un método de inserción en el que por medio de un sistema de referencia basado en el concepto de "banderas", es posible otorgar un orden de prioridad en el uso del Url EndPoint. Un punto importante a resaltar, es que durante el proceso de configuración se establece un mecanismo en el que se le asigna al usuario un valor por defecto de Url EndPoint, aun cuando este no digita ninguna información.

La razón de especificar un valor de Url En dPoint por defecto, es debido a que el componente tanto ofrece como consume puntos remotos de consulta Sparql, es decir, aun cuando la consulta adopte mecanismos gestionados y procesados de forma local, realmente los procesos asociados a la búsqueda están haciendo consumo del servicio Url EndPoint que corre localmente. La especificación que se hace del Url EndPoint es de la forma *http://localhost/endpoint/clase.php*. El apartado 4.3.1.2.3 dará una breve explicación respecto a la forma en la cual se gestiona la asignación de este valor por defecto.

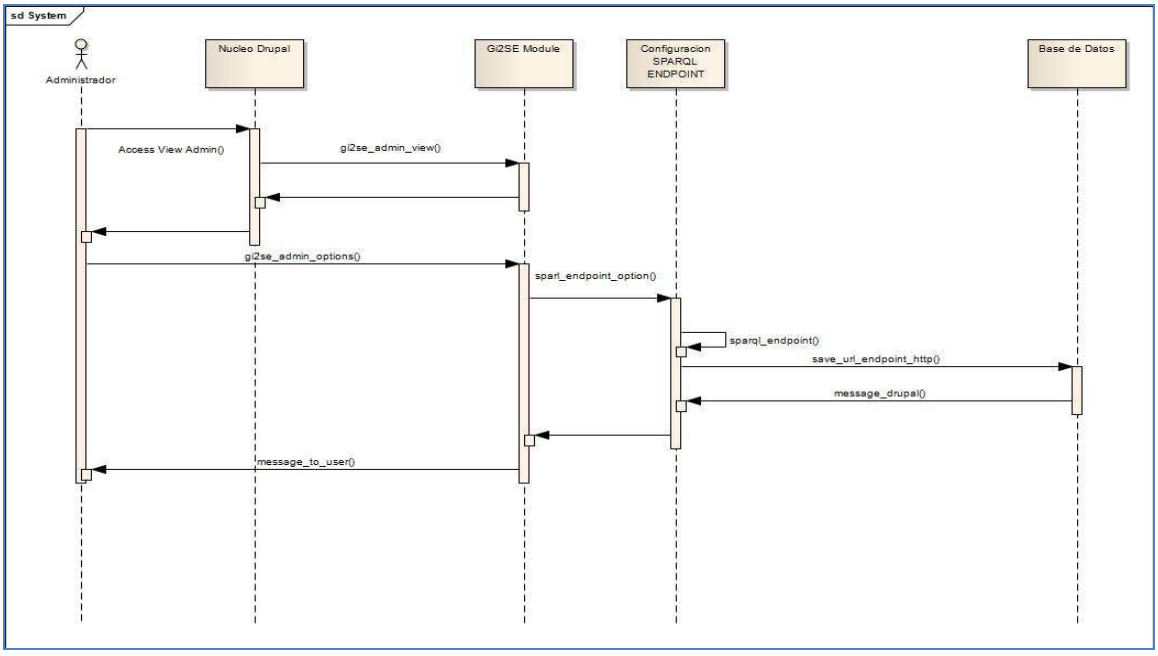


Figura 4.9 Diagrama de Secuencia de la actividad de configuración de Url End Point.

Es de recordar que este apartado trata acerca del proceso de asignación y configuración del Url EndPoint. La forma, tecnología y mecanismos que se emplean para exponer de manera remota el servicio de Url End Point serán abordados en el apartado 4.3.2.

4.3.1.2.3 Algoritmo Principal.

Como se menciono en el apartado 4.3.1.2.2, el algoritmo utilizado en esta funcionalidad hace uso de un mecanismo bastante sencillo, por medio del cual es posible especificar el lugar destinado para el Url EndPoint. Básicamente se ha hecho uso de la función llamada sparql_endpoint (obsérvese figura 4.9), en la cual se asigna una ruta base de localización para la clase Php que alberga las subclases y métodos que exponen el Url End Point.

La forma en la cual se asigna esta ruta, está directamente relacionada con el proceso de instalación del módulo, el cual será detallado en el apartado de anexos. Para efectos de poder comprender la forma en que funciona el algoritmo de asignación, es importante decir que cuando el plugin se instala sobre el núcleo de aplicaciones del sistema gestor de contenidos, este siempre se ubica en una ruta que sigue el estándar y la convención ofrecida en su API. Así por ejemplo, es fácil identificar rutas que se caractericen por tener el formato que se muestra en la tabla 4.3.

http://$Web_Directory_Server/$Root_Directory/$Modules/$Plugin_Name

Tabla 4.3. Ejemplo de estructura de ruta de instalación de plugin Drupal.

En donde:

- $Web_Directory, corresponde al espacio asignado por el servidor HTTP para poder interpretar los archivos que desean ser visible en la internet. El valor sugerido de configuración es “/var/www”.
- $Root_Directory, corresponde al nombre de la carpeta en el cual ha sido alojado el sistema gestor de contenidos Drupal, este nombre puede cambiar de acuerdo a las necesidades del administrador y no tiene un valor sugerido.
- $Modules, corresponde al espacio asignado por el gestor de contenidos Drupal para poder alojar todos los módulos instalados y desplegados por terceros sobre el sistema (observese apartado 4.2.1). El nombre que ofrece el estándar para este directorio es “modules”.
- $Plugin_Name, corresponde al nombre con el cual se ha identificado el plugin durante el proceso de desarrollo y despliegue, este nombre cambia con las necesidades del administrador.

Un ejemplo en concreto de las anteriores descripciones, es el nombre del plugin de este proyecto:

/var/www/gi2seFinal/modules/gi2se

Tabla 4.4. Ruta de instalación por defecto del componente Gi2SE.

Un punto interesante a destacar acerca de la característica del nombre del plugin, es que la gran mayoría de los complementos desarrollados para Drupal, proponen esquemas de configuración en la cual los usuarios deben modificar de forma plana el nombre y ruta de los archivos. En el caso de Gi2se, se ha decidido elaborar un algoritmo que ubique y asigne automáticamente estos nombres. De tal forma que independientemente de donde se instale el plugin, se pueda garantizar que el Url End Point preste las funciones para las cuales ha sido diseñado.

Parte del algoritmo usado para la funcionalidad de configuración de Url EndPoint se aprecia en la figura 4.10.

```php
<?php

   function sparql_admin_settings()
{

 $php_self_database = $_SERVER['DOCUMENT_ROOT'].$_SERVER['PHP_SELF'];
 $filename_database = explode("/", $php_self_database);
   for( $i = 0; $i < (count($filename_database) - 1); ++$i )
        {
         $filename2 .= $filename_database[$i].'/';
         $gi2se_path_modules = $filename2;
        }
 $form = array();
 global $base_path;
 $installation_drupal=$base_path;
 $pasar= $_SERVER['DOCUMENT_ROOT'].$installation_drupal;
 $host='http://'.$_SERVER['SERVER_NAME'].$installation_drupal.$$gi2se_path_modules = $filename2.'/UrlEndPoint.php';
 $endpointgpoveda = $host;

}

 ?>
```

Figura 4.10 Algoritmo asignación dinámica de ruta instalación Gi2se.

4.3.1.2.5 Interfaz Gráfica.

El diseño de la interfaz de configuración ha adoptado mecanismos que extienden de las

funcionalidades mencionadas y detalladas en el apartado 4.3.1.1.5. Obsérvese la figura 4.8. Nuevamente se ha hecho uso de elementos gráficos basados en la interfaz de aplicaciones de Drupal (API). A diferencia de los mecanismos de diseño usados en la sección de configuración de la fuente RDF, sobre esta actividad no se requiere actualización de contenido de forma dinámica. Así, la interacción gráfica que tiene el usuario con el sistema durante este proceso se limita a diligenciar los campos de información correspondientes.

4.3.2 Actividad de Búsqueda

Corresponde a la funcionalidad de mayor relevancia e importancia en el desarrollo de este proyecto fin de Máster, ya que sobre esta fase se propone la realización de dos importantes actividades que permitirán cumplir con los objetivos trazados en la realización de este proyecto. Estos objetivos corresponden al diseño de una interfaz gráfica y a la implementación y configuración de la librería Arc2 como principal herramienta de apoyo en los procesos de búsqueda semántica sobre las ambientes Web convencionales. Haciendo uso de Arc2 es posible alcanzar escenarios de búsqueda que tengan la capacidad de brindar / exponer servicios de consulta de forma remota.

Por otro parte, con la implementación de esta actividad se pretende alcanzar escenarios de consulta similares a los que se encuentran hoy día en muchos otros sitios como la DBPEDIA, pero con el valor adicional de poder realizar procesos de consulta auto-asistidos. De esta forma el usuario no deberá ingresar estructuras de consulta SPARQL directamente sobre cajas de texto, sino que podrá hacer uso de interfaces gráficas que lo guíen acerca de los elementos que el deberá incluir en las búsquedas. Además de lo anterior, también se procura que sea el usuario quien configure las fuentes de información sobre las cuales desea ejecutar los procesos de búsqueda (Este tema fue tratado en el apartado 4.3.1.1.1).

Dado que el desarrollo de estas actividades se concibe de forma secuencial, es decir, la actividad de diseño de la interfaz gráfica depende de los mecanismos propuestos en la actividad de integración y configuración de la librería Arc2. Es necesario mostrar la forma en la cual se llevan a cabo las interacciones entre los distintos objetos que componen cada una de estas actividades y el punto en donde converge la participación de cada una de ellas. La figura 4.11 muestra en detalle el diagrama de secuencia realizado para la actividad de búsqueda.

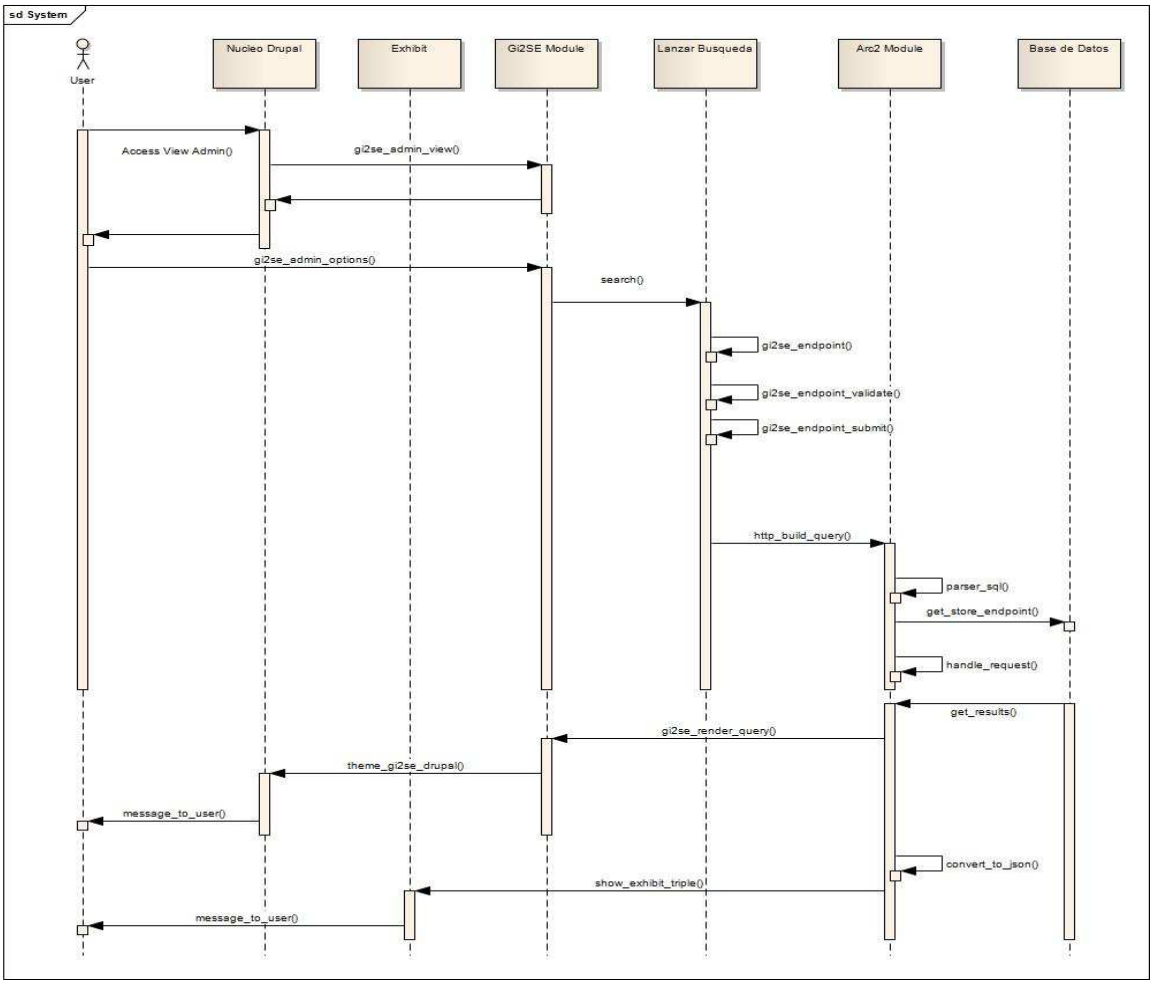


Figura 4.11 Diagrama de Secuencia de la actividad de búsqueda.

4.3.2.1 Implementación Framework ARC2.

4.3.2.1.1 Descripción

Arc2 es una librería que ayuda a implementar parte de las funcionalidades ofrecidas en el componente de búsqueda Gi2se. Básicamente corresponde a un Framework que sirve como elemento de intermediación entre el mundo de la Web Semántica representado en forma de triplas RDFs y tecnologías de uso convencional en la Web como Php. Este Framework tiene la característica de implementar funcionalidades como analizadores sintácticos y serializadores para el manejo, interpretación y transformación de la información semántica en el ambiente Web. Otras características que posee esta herramienta y que son utilizadas durante la fase de diseño de

este proyecto son:

- Soporte para manejar fuentes de almacenamiento RDF haciendo uso de Mysql.
- Soporte para configurar puntos de terminación Sparql.
- Soporte para usar plugins con diversos Frameworks como Exhibit.

La razón de uso de Arc2 sobre este proyecto, se justifica en la gran cantidad de funcionalidades que ella incorpora como herramienta de intermediación entre las tecnologías web convencionales y las tecnologías semánticas. De esta forma este proyecto propone hacer adaptaciones sobre los métodos y procedimientos presentes en esta librería, de tal forma que sea posible poder vincularlos estos cambios con parte de las funcionalidades presentes en el gestor de contenidos Drupal. En este sentido será posible brindar procesos de búsqueda que cumplan con los objetivos trazados en este proyecto.

4.3.2.1.2 Diseño e Implementación.

Al observar la figura 4.11, es posible notar que las dos primeras actividades que se llevan a cabo durante esta interacción, corresponden a procedimientos que se enmarcan dentro del ámbito del diseño de interfaces gráficas. Sin embargo y pese a que mantienen una relación directa con el uso de la librería Arc2, este apartado intenta explicar los procedimientos de tipo lógico que se llevan a cabo durante la construcción del modelo. Por esta razón los temas de diseño de interfaces gráficas serán detallados en el apartado 4.3.1.4.

En esta forma, es necesario mencionar que la primera actividad de tipo lógico que se produce cuando el usuario interactúa con la interfaz de búsqueda, es la funcionalidad asociada a la creación dinámica de objetos. Esta actividad básicamente permite relacionar la cantidad de clases y propiedades que el usuario necesita incorporar para ejecutar un proceso de búsqueda determinado, es decir, se propone el uso de un mecanismo que permita construir de forma dinámica las consultas tipo SPARQL con base a la información que el usuario ingresa o escoge de un dominio específico de datos.

Haciendo uso de tecnologías como Ajax, Xml y una serie de funcionalidades previstas en el Api de Drupal (formularios). Estos mecanismos no sólo permiten ofrecer al usuario final, un nivel de interpretación lo suficientemente óptimo en términos de presentación de la información, sino que también permiten adaptar mecanismos dinámicos de control sobre las estructuras de información, es decir, este mecanismo paralelamente ofrece la posibilidad de identificar la

información que se obtiene de forma dinámica sobre los formularios, para posteriormente procesarla e interprétala con SPARQL por medio de algunas de las funcionalidades que ofrece la librería ARC2.

Un par de aspectos que son relevantes dentro de este proceso y que hacen parte de los requerimientos de la actividad de búsqueda, son el proceso de importación de espacios de nombre y el proceso de registro de las triplas sobre el motor de persistencia. En este caso se trata del motor de búsqueda Mysql. En lo referente al proceso de importación de nombres, es necesario especificar que esta actividad corresponde a un procedimiento gestionado por una de las utilidades que ofrece el componente llamado Rdfme. La forma en la cual se hace uso de la funcionalidad expuesta por Rdfme, es a través de la invocación de un servicio Web que es consumido solicitando un Uri similar al que se aprecia en la tabla 4.5

http://localhost/gi2mo/?q=admin/settings/rdfme/rest/import

Tabla 4.5. Forma de Invocación Web Service Rdfme.

Cuando este servicio es consumido, los algoritmos que implementa Rdfme hacen que las triplas contenidas en las fuentes de información RDF, sean identificadas, procesadas y almacenadas de forma persistente. Es importante resaltar que este procedimiento corresponde a una actividad en la cual se aplican técnicas de relación e identificación semántica, lo cual hace que la información contenida en la base de datos preserve los diferentes grados de relación que se producen entre las clases y objetos. Es de aclarar que esta actividad sólo contempla el almacenamiento de los nombres de las clases y propiedades y en ningún momento almacena los valores precedentes de cada tripla. Un ejemplo de la anterior situación, se presenta en la tabla 4.6

Clases		
Prefix	Id	Comentario
Foaf	Person	The subject Tagging occurred a the subject
Propiedades		
Foaf	accountName	The subject Tagging occurred a the subject time a...

Tabla 4.6. Campos de información creados en Rdfme.

La importancia que tiene este proceso durante la actividad de búsqueda, se relaciona

directamente con la creación dinámica de objetos. En especial con aquellas actividades que gestionan y asisten al usuario respecto al nombre de las clases y propiedades que pueden ser utilizadas en un dominio especifico de información. De manera puntual, este modelo propone el diseño de una funcionalidad que basada en utilidades de auto completacion de texto, permite al usuario obtener información respecto a las clases y propiedades que deberá referenciar sobre el proceso de búsqueda. La forma en la cual opera esta funcionalidad es bastante simple, dado que hace uso de un método que implementa una tarea de sincronización entre una librería Ajax y los datos almacenados de forma persistente (obsérvese tabla 4.6). El aparatado 4.3.2.1.3 ofrecerá más información de las interfaces gráficas relacionadas con esta actividad.

Con respecto al proceso de registro de triplas, es necesario especificar que esta actividad corresponde a la acción de mayor relevancia e importancia de este proyecto fin de Máster. Sin duda, la inclusión de escenarios persistentes en el mundo de la Web semántica representa un importante avance. Con este tipo de conceptos es posible no sólo obtener niveles adecuados de concurrencia, sino también introducir modelos de búsqueda que se caractericen por ir más allá de una simple y plana consulta Rdf enmarcada en un contexto local. Además de lo anterior, también es posible obtener escenarios de consulta que se caractericen por ser mas óptimos y efectivo con respecto a los esquemas tradicionales.

En esta forma, es posible establecer que el modelo propuesto aboca a la construcción de escenarios de consulta que puedan ser usados de manera remota y de forma concurrente. Cuando se hace referencia al término remoto, se intenta decir que los usuarios podrán ejecutar procesos de búsqueda que sean interpretados por puntos de terminación SPARQL que no se encuentren necesariamente dentro del dominio de despliegue del componente. Igualmente, se propone generar servicios de búsqueda que puedan ser consultados de forma remota haciendo uso del punto de terminación SPARQL configurado localmente. El concepto usado para abordar los planteamientos expresados anteriormente, corresponde a una utilidad ofrecida por la librería ARC2 y cuyo nombre es SPARQL Url EndPoint.

De esta forma es posible hacer referencia ARC2 como la librería que permite conseguir gran parte de las funcionalidades mencionadas anteriormente. Básicamente corresponde a una clase desarrollada en lenguaje Php que ofrece los métodos necesarios para la realización de 4 funcionalidades básicas sobre la información de tipo semántico:

- *Store.*
- *Serializer.*
- *Parsing.*
- *Extractor.*

Aunque parte de los mecanismos de implementación de cada una de estas funcionalidades serán abordados en el apartado llamado *Algoritmo principal*, a continuación se ofrece una corta descripción respecto a labores que algunos de ellos hacen:

- *Store:* Corresponde a la utilidad encargada de manejar la persistencia de las fuentes de datos del componente diseñado. Propone el uso de mecanismos de consulta e inserción por medio de métodos que crean y verifican la información sobre el dominio de datos a utilizar. Es de resaltar que los mecanismos de inserción se ejecutan durante el proceso de configuración de la fuente RDF, específicamente cuando el usuario ingresa por primera vez la fuente de datos que desea utilizar para los procesos de búsqueda (obsérvese aparatado 4.3.1.1.1).

  Con el fin de alcanzar integridad referencial, esta utilidad hace uso de técnicas de transformación de datos basadas en funcionalidades como *Serializer* y *Parsing*. Otra funcionalidad que extiende de esta utilidad, son los llamados Almacenes remotos "*Remote Store*". Los cuales permiten trabajar con consultas SPARQL de forma remota.

- *Serializer:* Es la utilidad encargada de hacer los procesos de transformación entre los datos que tiene formatos asociados a la Web semántica y los datos que desean ser almacenados de forma persistente. Acerca de esta utilidad es importante mencionar un par de generalidades. La primera de ellas tiene que ver con los distintos mecanismos de *serializacion* que pueden ser usados para dichas transformaciones, siendo estos Rdf/Xml, Rdf/Json y N-triples. Por otra parte, esta utilidad de *serializacion*, es usada como técnica de mapeo durante el proceso de transformación que ocurre cuando los espacios de nombres son almacenados con los prefijos y propiedades en el sistema de almacenamiento.

- *Parsing:* Es la utilidad encargada de hacer los procesos de transformación entre los datos que se obtiene de las consultas realizadas y los formatos de salida que se deseen obtener. Esta utilidad cuenta con dos tipos. El primero de ellos se ejecutan de forma remota, para lo cual se cuenta con una funcionalidad que recibe como parámetro el valor donde se encuentra el fichero RDF. En tanto el segundo está relacionado con el *parsing* de datos local.

Una vez descritas algunas de las funcionalidades más importantes que se llevan a cabo durante este proceso y teniendo en cuenta el desarrollo de la primera actividad (creación dinámica de objetos). La actividad llamada implementación de Framework Arc2, continúa con la validación y verificación de la información que se ha obtenido de los objetos creados de forma dinámica y

quienes han recogido la información base de la consulta.

Para manejar la información almacenada sobre las estructuras dinámicas de información, el diseño de esta funcionalidad propone la creación de un método que construye de forma automática las consultas SPARQL con base en la información contenida en cada una de las estructuras de información anteriormente mencionadas. Este método corresponde a una funcionalidad que al tomar el nombre de una propiedad o clase, identifica el espacio de nombres asociado y lo relaciona con una operación de selección y/o filtrado. En caso de estar presentes dos o más clases, el algoritmo deberá estar en capacidad de vincular las operaciones por medio del operador unión.

En lo que respecta a los procesos de validación y verificación, este modelo propone el uso de una funcionalidad que permite definir la utilidad llamada Url EndPoint como principal mecanismo de búsqueda y como herramienta de intermediación entre Php - SPARQL y Mysql. Básicamente corresponde a una clase en Php que es invocada desde la función encargada de manejar la creación dinámica de objetos y quien a su vez hereda parte de las funcionalidades expuestas en la librería ARC2. Algunas de las funcionalidades ofrecidas por esta clase son:

- Gestión en el proceso de inserción y selección de datos semánticos.
- Gestión en la Conexión y Obtención de los almacenes remotos “Remote Store”.
- Verificación del estado actualización de la fuente de información RDF.

La figura 4.12 pretende mostrar el escenario en el que se producen las interacciones entre cada uno de los elementos que participan en la actividad.

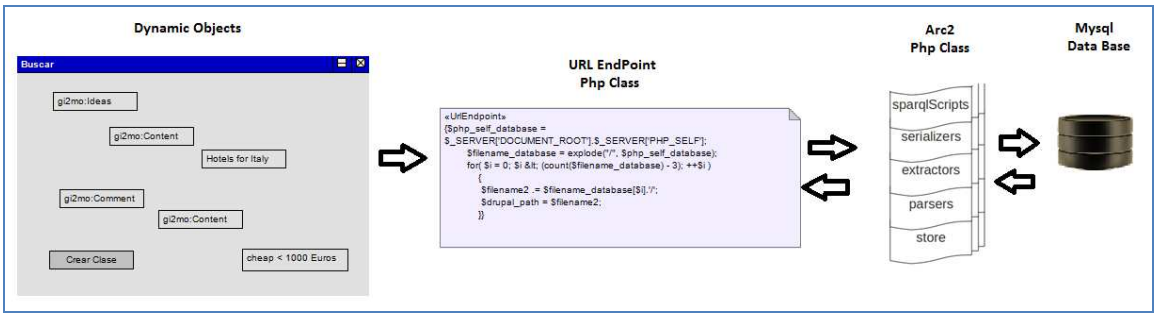


Figura 4.12 Interacciones producidas entre el Url End Point – Php y ARC2.

Un procedimiento que es importante dentro de esta funcionalidad, es el relacionado con la verificación del Url End Point. La actividad de verificación corresponde a una función que basada en una subrutina de la clase que controla la creación dinámica de objetos, permite saber

si el Url introducido efectivamente corresponde a una clase Php que extiende o hereda comportamientos de la librería ARC2.

Otro procedimiento que es relevante sobre la etapa de diseño, es aquel que se relaciona con el proceso de transformación que ocurre entre los datos que construyen la consulta tipo SPARQL y los datos que deben ser interpretados en lenguaje SQL. Esta rutina básicamente hace uso de una función que mapea sentencias como Select, Describe, Filter y Load en operaciones que son propias de SQL. Un aspecto que facilita considerablemente este tipo de actividades, es la estructura interna de vínculos que posee RDF. Considerando que RDF incorpora implícitamente las relaciones como atributos no coincidentes dentro de su modelo de datos, es posible aplicar técnicas convencionales de mapeos que nos permitan obtener fácilmente el modelo relacional de datos.

Un ejemplo de esta situación se presenta especialmente sobre el operador Join. En SPARQL esta relación suele presentarse de forma implícita, en tanto sobre SQL esta relación debe ser expresada de manera explícita. Las tablas 4 .7 y 4.8 ilustran la forma en la cual es usado este operador tanto en SPARQL como en SQL.

```
SELECT ?d ?apt WHERE { <Orders.id=3183> <Orders.shippingAddress> ?d .
?d <Addresses.apt> ?apt }
```

Tabla 4.7. Ejemplo uso operador Join SPARQL.

```
SELECT A.id AS d, A.apt AS apt
  FROM Orders AS O_0
 INNER JOIN Addresses AS A ON O.shippingAddress=A.id
 WHERE A.id=3183
```

Tabla 4.8. Ejemplo uso operador Join SQL.

El procedimiento de entrega de resultados es otra importante actividad que se desarrolla en la etapa de diseño de este componente. Esta rutina hace uso de una función desde la cual es posible invocar el formato de salida que se desee utilizar para la gestión y representación de los resultados obtenidos. Sin duda, es la etapa de mayor importancia dentro del proceso de representación de la información (construcción de interfaz gráfica), dado que a partir de los múltiples formatos que se pueden obtener, es posible gestionar diferentes tipos de interfaces gráficas. El porque es importante generar utilidades de estas características no sólo centra en

aspectos de tipo gráfico sino que también obedece a temas de re uso y reutilización de la información.

Por un lado se pretende que la información obtenida en formatos como Json, pueda ser utilizada en procesos de representación gráfica como Widgets o Mashups. Por otra parte, la generación de archivos en formato RDF, permite utilizar la información obtenida como entrada de otros procesos que se desarrollen en el ambiente de la Web semántica. Un aspecto importante a considerar, es que independientemente del tipo de formato que se genere en la información de salida, el proceso de conversión de datos siempre sigue un estándar. En este estándar se asume la creación de un método que mapea el resultado de la consulta generada en archivos de tipo Xml.

Es necesario establecer que aunque ARC2 brinda la posibilidad de generar múltiples formatos de salida como Json, Xml y Rdf, la fase de construcción de la interfaz gráfica de este proyecto ha contemplado dos pequeñas variantes. La primera de ellas hace referencia a la adaptación realizada sobre el método get_results(). Básicamente se ha propuesto la realización de un mecanismo de serializacion que permita integrar los resultados obtenidos de la búsqueda con instrucciones y métodos del lenguaje Php.

Por otra parte y, teniendo en cuenta que uno de los objetivos que se plantea en este proyecto tiene que ver con la búsqueda de nuevas forma de representación visual en las consultas semánticas. Se ha propuesto el uso de una funcionalidad que al tomar los resultados generados en Xml permita procesar la información y generar un formato de salida tipo Json.

Los criterios de diseño e implementación de esta funcionalidad serán detallados en el apartado 4.3.2.1.4.

Como se menciono anteriormente, ARC2 dispone de mecanismos que permiten generar de forma automática archivos tipo Json como formato de salida de las consultas realizadas. Sin embargo, la forma en como se representan estas estructuras no son acordes a los requerimientos que propone la herramienta Exhibit para el procesamiento gráfico de información. Es importante recordar que Exhibit, propone el diseño de interfaces gráficas haciendo uso de librerías Ajax que requieren como parámetro de entrada archivos tipo Json que tengan en su estructura una etiqueta identificativa llamada “Ítem”.

Teniendo en cuenta que el formato Json generado por ARC2 crea etiquetas que usan como elemento raíz “bindings” y no “ítems”. Se hace necesario diseñar algunas funcionalidades adicionales que estén en capacidad de hacer este tipo de transformaciones. Es por esta razón que se propone el desarrollo de una funcionalidad adicional para hacer los procesos de

transformación necesarios.

4.3.2.1.2.1 Modelo de Despliegue

Con el fin de ofrecer un nivel de comprensión mucho más detallado respecto a la forma en que se ofrecen algunas de las funcionalidades realizadas en la fase de diseño de esta actividad. Es necesario mencionar que el diseño y desarrollo de este plugin se ha realizado bajo el escenario del proyecto semántico de Ideas Gi2mo. Este proyecto implementa una serie de funcionalidades que se han realizado usando utilidades del manejador de contenidos Drupal. Haciendo uso del concepto de plugin ha sido posible construir herramientas de inter operabilidad semántica. Una de estas actividades precisamente corresponde a un plugin comentado anteriormente y cuyo nombre corresponde a Rdfme. Teniendo en cuenta que este componente ha contemplado una fase de diseño y desarrollo diferente a la que actualmente se propone. No se considera necesario evaluar los detalles técnicos de codificación, dado que sólo se hace uso de algunos de los métodos y procedimientos que este ofrece. Además de este componente, Gi2SE hace uso de cerca de 4 componentes de características similares, entre ellos es posible mencionar a SPARQL, RDF y Gi2mo Ideas Stream.

Respecto a la forma en que se integran cada uno de estos componentes, es preciso mencionar que cada una de ellos se acopla sobre el núcleo de operaciones de Drupal de acuerdo al modelo de plugin explicado en el apartado 4.2.1. La figura 4.13 muestra en detalla la forma de integración y despliegue de cada uno de los componentes asociados y utilizados en este proyecto.

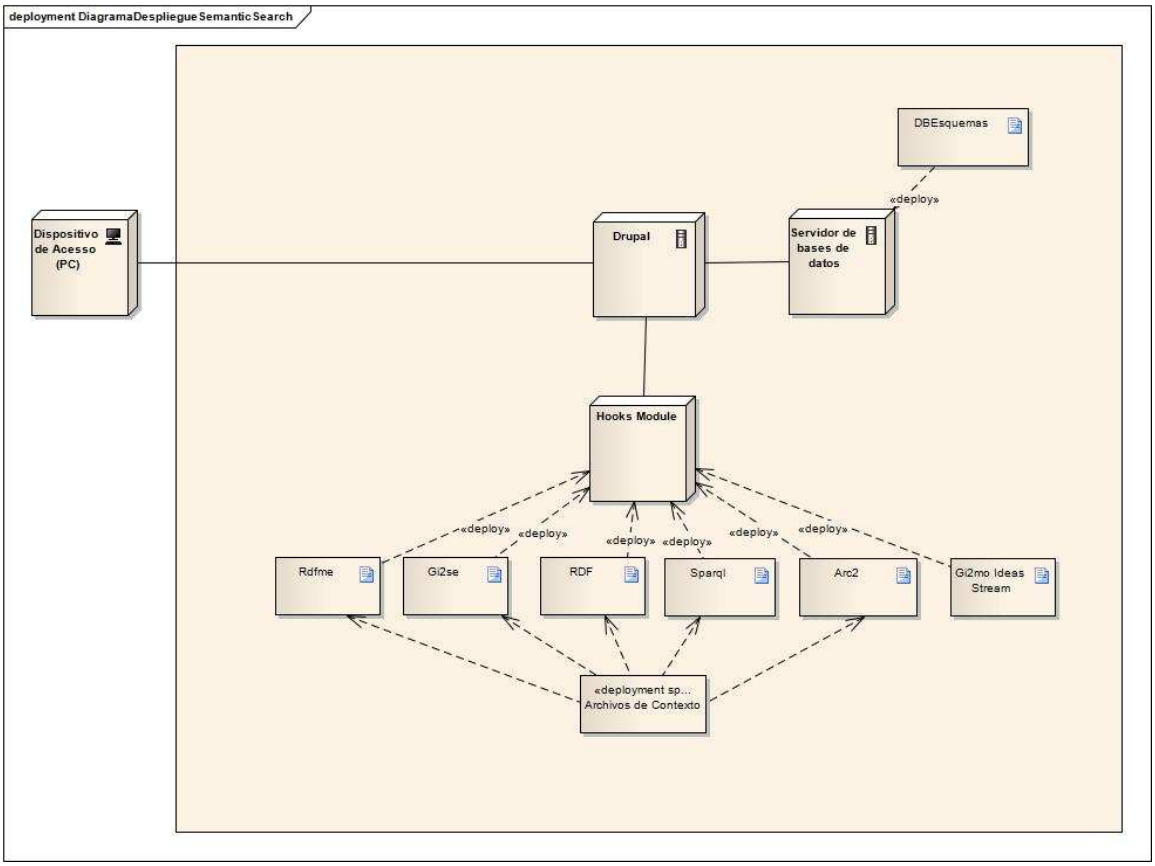


Figura 4.13 Modelo de Despliegue del componente de Búsqueda Gi2se.

Un aspecto importante dentro del proceso de diseño y despliegue de este componente, tiene que ver con la declaración de los métodos que permiten implementar funciones en forma de Hooks (Obsérvese apartado 4.2.1). En el caso del módulo Gi2se, se está haciendo referencia a los ficheros llamados Gi2se.module y Gi2se.install. Respecto a Gi2se.install es necesario comentar que corresponde al archivo que gestiona el proceso de instalación del componente dentro del gestor de contenidos Drupal, es decir, este fichero alberga los métodos e instrucciones que deberán ejecutarse cuando el proceso de instalación es iniciado. En el caso de Gi2se se cuenta con cuatro operaciones básicas, siendo estas: gi2se_install, gi2se_uninstall, gi2se_schema y gi2se_enable.

La funcionalidad de mayor importancia dentro de este fichero corresponde a gi2se_schema, ya que mediante esta, es posible crear las tablas que manejan parte de la lógica del componente. Teniendo en cuenta las descripciones realizadas en este capítulo acerca del modelo persistente, es necesario decir que sobre este fichero se crean sentencias que heredan de la interfaz de aplicaciones de Drupal, las cuales permiten ejecutar las largas y tradicionales sentencias de SQL de una forma mas simple y ágil. Tal es el caso de la instrucción db_query (). De otra parte se cuenta con la funcionalidad relacionada con la activación y desactivación del módulo sobre el gestor de

contenidos, básicamente lo que permite esta funcionalidad es poder bridar al usuario la opción de habilitar y des habilitar el funcionamiento de Gi2se sin necesidad de hacer todo el proceso des instalación correspondiente. Al igual que el Hook mencionado anteriormente, esta función hereda del Api de Drupal.

En lo que respecta al fichero llamado Gi2se.module, es importante indicar que este corresponde a un conjunto de funcionalidades en las cuales es posible usar métodos tipo Hook para atender el llamado del sistema gestor de contenidos en tiempos predeterminados. Este fichero se caracteriza por cumplir dos funciones básicas, la primera de ellas está relacionada con el manejo dinámico de parámetros por medio de vínculos, es decir, se intenta aplicar un concepto similar al de sobre carga de métodos, pero haciendo uso de estructuras dinámicas que a través del operador "%" permiten dar flexibilidad en el paso de parámetros. Por otra parte, algunas de las funciones que se desarrollan en gi2se.module permiten gestionar el manejo de vínculos de cierto tipo de funcionalidades por medio de la declaración de una función asociada a un archivo.

Los 4 módulos restantes, presentan características similares en la declaración de los métodos de los ficheros module e *install*.

4.3.2.1.3 Algoritmo Principal.

Como se pudo observar en el apartado 4.3.2.1.2, esta actividad hace uso de múltiples funcionalidades que ejecutadas de forma individual y secuencial permiten cumplir con los objetivos trazados en el diseño de este componente. Teniendo en cuenta que la primera actividad de tipo lógico que ocurre en el sistema, es la creación dinámica de los objetos encargados de almacenar la información de las clases y propiedades. La primer labor que se ha realizado sobre esta actividad, ha sido la declaración de la librería que permitirá representar de forma gráfica las acciones de código realizadas.

```
drupal_add_js(drupal_get_path('module', 'gi2se') . '/js/ahah.js', 'module', 'footer');
```

Tabla 4.9. Declaración librería Ahah.

Para esta actividad, se ha hecho uso de una función llamada Gi2se_ dynamic_class, desde la cual es posible instanciar nuevos objetos de tipo "clase". Respecto a esta actividad, es importante destacar la importancia que tiene la variable $form_state como elemento encargado de mantener actualizado todos los cambios producidos entre los elementos de tipo gráfico y lógico asociados a la aplicación. Es importante destacar que esta funcionalidad efectúa el llamado de un objeto externo, a través de la declaración de una función tipo Hook contenida en el fichero de extensión

".module" del componente. La figura 4.14 muestra en detalle la forma en que se realiza este procedimiento.

```
$form[$type]['gi2se_add_class_'] = array(
    '#type' => 'submit',
    '#value' => t('Add new  class', array('@name' => $name)),
    '#tree' => FALSE,
    '#submit' => array('gi2se_ahah_class_handler'),
    '#ahah' => array(
    'path' => 'admin/settings/gi2se/ahah/class/' . $type,
    'wrapper' => 'new-class-' . str_replace('_', '-', $type),
    'method' => 'append',
    effect' => 'fade',
```

Figura 4.14 Función que invoca la creación de un nuevo objeto tipo clase.

La función referenciada no sólo permite llevar un control del número de objetos creados, sino que también permite asociar las propiedades que corresponden a cada una de las clases instanciadas. Esto quiere decir que en el momento que una clase es creada, los elementos que sobre ella se aniden (propiedades) deberán corresponder única y exclusivamente al objeto en mención. Por una parte, se evita una carga excesiva de elementos en la lista de propiedades sugeridas, así mismo se reduce el posible número de equivocaciones que pueda llegar a tener un usuario al momento de introducir información. En lo que se refiere al mecanismo de auto "completacion" de texto, esta funcionalidad hace uso de un método llamado *implode* para poder concatenar todas las clases y propiedades pertenecientes a un prefijo. Los elementos se sugieren con base a comparaciones hechas en la información que hay en los campos de las tablas, con la información introducida por el usuario por medio de sentencias como "Like %%%s%%%".

Otra importante actividad que se contempla dentro de esta fase, es el diseño y codificación de la clase Php que permitirá definir la utilidad llamada Url End Point y algunos de los procedimientos que se llevan a cabo durante el proceso de registro de triplas. Puntualmente se está haciendo referencia a la clase llamada UrlEndPoint.php y cuya estructura contempla la declaración de una serie de métodos que permiten crear acciones intermediarias entre Php y SPARQL. Uno de estos métodos es precisamente aquel que permite realizar el proceso de inserción de datos (RDF) sobre las tablas previstas para tal propósito.

Es necesario mencionar que esta actividad es antecedida por un proceso de conexión y autenticación similar al que se efectúa cuando desde clases tipo Php se hace uso de funciones Mysql (ARC2::getStoreEndpoint). Teniendo en cuenta que UrlEndPoint.php crea instancias de algunos de los objetos presentes en Arc2.php, el siguiente procedimiento que se realiza durante esta

actividad, es la creación del objeto encargado de generar las estructuras RDF correspondientes. Es necesario mencionar que este procedimiento corresponde a un proceso de transformación efectuado sobre un documento de entrada. Una vez el documento ha pasado por este procedimiento, la actividad continua con la obtención de las triplas presentes en el fichero transformado. De eta forma haciendo uso de funciones como getStoreEndpoint y métodos como Insert, es posible salvar la información contenida en la triplas sobre el sistema gestor de datos. La tabla 4.10 muestra un fragmento de código que permite realizar parte de la funcionalidad descrita anteriormente.

```
$parser = ARC2::getRDFParser();
$parser->parse($doc);
$triples = $parser->getTriples();
$store->insert($triples, $doc);
```

Tabla 4.10 Proceso de parsing y obtención de triplas RDF.

Teniendo en cuenta que la información contenida en la tabla 4.11 intenta explicar de forma detallada los procedimientos de codificación realizados durante esta etapa, por efectos de simplicidad, rendimiento y menor consumo de memoria, la librería Arc2 permite crear sentencias de la forma:

```
$store->query("LOAD <$url>");
```

Tabla 4.11 Uso de función load para carga de triplas.

Con la especificación de este tipo de sentencias, la función Load no sólo deberá contemplar funcionalidades de lectura remota de archivos sino que también deberá estar en capacidad de decidir el criterio de uso de un *parser* en particular. Esto en base a los resultados entregados por un detector de formato.

En lo que respecta al proceso de inserción de triplas, se ha propuesto un algoritmo que haciendo uso de los ya mencionados procesos de *mapping*, permite identificar las jerarquías presentes en cada una de las triplas, manteniendo y preservando las relaciones de cada uno de estos elementos por medio de estructuras dinámicas de datos. La forma en la cual se han preservado las relaciones ha sido en base al modelo relacional de datos propuesto en el apartado 4.3.1.1.3. Así como también, con base a conceptos como objeto, sujeto y predicado. Básicamente, se plantea la creación de nodos de información que contemplen un campo “Id” como identificativo y una serie de índices que relacionen las propiedades y atributos. Cuando estas operaciones son efectuadas, se procede con la ejecución de un bucle que en base al número de triplas presentes, realiza un proceso temporal de almacenamiento de información (o,s,p) sobre los arreglos correspondientes.

De esta forma, este algoritmo propone la inserción de información con base a funciones como ARC2: getStoreEndpoint () y ARC2_Store:setSettings (), en donde básicamente se toma la información de los arreglos mencionados anteriormente y se asocian a los prefijos que fueron asignados a las tablas durante el proceso de creación de las mismas. De esta forma se plantean mecanismos de verificación orientados a comprobar si el valor serializado corresponde a información nueva o información actualizada. Un aspecto que es importante mencionar y que es posible observar en la tabla 4.12 es que en la función setSettings () asume un proceso de serializacion adicional como parte de la estructura SQL. Este estilo de codificación hace pensar que cuando este tipo de operaciones se ejecutan sobre la operación queryDB del Api de Drupal, los procesos de consulta e inserción de datos son realizados de forma eficaz.

```
function setSetting($k, $v)
 {
  $con = $this->getDBCon();
  $tbl = $this->getTablePrefix() . 'setting';
  if ($this->hasSetting($k))
  {
   $sql = "UPDATE " .$tbl . " SET val = '" . mysql_real_escape_string(serialize($v),
$con) . "' WHERE k = '" . md5($k) . "'";
  }
  else
  {
   $sql = "INSERT INTO " . $tbl . " (k, val) VALUES ('" . md5($k) . "', '" .
mysql_real_escape_string(serialize($v), $con) . "')";
  }
  return $this->queryDB($sql, $con);
 }
```

Tabla 4.12 Fragmento de Algoritmo de Inserción.

Por otra parte y, continuando con la descripción del proceso de construcción automática de consultas tipo Sparql, es preciso mencionar que este procedimiento hace uso de estructuras dinámicas de datos (Arrays) para el manejo de la información presente en los elementos de tipo gráfico. En esta forma, se crea una variable global llamada $cadenaSparql que se usa para

actualizar constantemente el valor que almacena la variable $form_state. Es importante recordar que esta variable conserva los valores temporales e instantáneos presentes en cada unas de las clases, propiedades y atributos, es decir, teniendo en cuenta que esta aplicación tiene la capacidad de crear múltiples atributos por múltiples clases, es posible obtener en un espacio de tiempo determinado arreglos de la forma: $form_state['values']['Class'][n], form_state['values']['Property'][n], form_state['values']['Attribute'] [1] representados a través de la variable $cadenaSparql.

La forma en la cual esta variable crea las consultas tipo SPARQL tiene dos criterios. El primero de ellos es la asociación automática de los prefijos dependiendo del tipo de clases utilizadas. Para este procedimiento básicamente se hace una asociación entre el prefijo y el Uri de un espacio de nombres por medio de consulta sobre el sistema de datos. El segundo criterio tiene que ver con la forma en la cual relaciona y se filtra de manera automática las clases y propiedades escogidas como criterio de búsqueda. Para el procedimiento de filtrado, se realiza una operación bastante sencilla, en la que se toma el valor presente en el atributo y se vincula de forma directa con el valor de la propiedad por medio de sentencias de la forma:

```
FILTER regex(?'.$valuesProperty[0].','.'"'.$valuesAttribute[0].'"'.')
```

Tabla 4.13 Fragmento de código para la construcción de sentencia Filter.

El procedimiento que relaciona los conceptos introducidos sobre una misma clase, bien sea en forma propiedades o atributos, hace uso de funcionalidades de concatenación sobre cada una de las sentencias extendidas y construidas desde la información presente en la variable form_state. Un punto interesante a mencionar, es el uso del operador Unión como elemento de relación entre las propiedades y atributos de dos o más clases. En este caso, se hace uso de una operación que reconociendo las propiedades asociadas a cada una de las clases, permite diagnosticar la correspondencia de cada uno de estos términos por medio de las relaciones establecidas entre el Id de la clase y los índices de las propiedades. La tabla 4.14 muestra un fragmento de código de este procedimiento.

```
$cadenaSparql .=  '?x rdf:type'.' '.$valuesClase[0].' '.'.'.' ';
$cadenaSparql .= '?x'.' '.$valuesProperty[0].' '.'?'.$valuesAttribute[0].' '.'.'.' ';
$cadenaSparql .= 'FILTER regex(?'.$valuesProperty[0].','.'"'.$valuesAttribute[0].'"'.') '.'';
```

Tabla 4.14 Fragmento de código de concatenación de consulta.

Una vez la variable $cadenaSparql contiene la información que será interpretada como sentencia de búsqueda Sparql, la siguiente operación que se lleva a cabo es la vinculación de este valor con

los valores correspondientes de grafo y SPARQL EndPoint. Para ello, la variable adiciona un par de campos en forma de *array* asociativo sobre su estructura, de tal forma que la sentencia pueda ser usada como parámetro de entrada sobre la operación que tiene la función de retornar y mostrar los resultados de la búsqueda (sparql_render_query_results).

```
1.
$cadenaSparql = array_filter(array('endpoint' => URL_ENDPOINT, 'default-graph-uri'
=> $graph, 'query' => $ cadenaSparql), 'strlen');
2.
function sparql_render_query_results($cadenaSparql, $options = NULL)
```

Tabla 4.14 Fragmento de código función de búsqueda.

Con respecto a esta operación, es importante mencionar que parte de su funcionamiento se basa en dos (2) procesos. El primero de ellos tiene que ver con la forma en la cual interpreta los resultados que recibe en forma de parámetro. Para esto, hace uso de una operación llamada sparql_Query: prefix, en donde básicamente se verifica la integridad de la sentencia formada en la variable $cadenaSparql y se envía la consulta sobre el Url EndPoint configurado. Tal como se menciono anteriormente, el concepto de Url EndPoint se abordo con la implementación de una clase Php, en la cual además de hacer los procesos de inserción de las triplas presentes en la fuente de información. Esta clase también sirve para definir los esquemas de consulta que estructurados en forma de sentencias Sparql deberán ser transformados a consultas tipo SQL.

En este sentido, la forma en que se invoca la clase UrlEndPoint.php es través del método http_build_query(), en donde básicamente se pasa como parámetro la variable $cadenaSparql. De esta forma haciendo uso de la función (drupal_http_request) de Drupal, es posible diagnosticar la integridad del Url EndPoint en cuestión.

Ahora bien, cuando efectivamente el Url invocado pertenece a un punto de terminación Sparql. Url EndPoint.php, es quien toma el control del flujo del proceso. Al heredar funciones y comportamientos de la librería Arc2, esta clase instancia un objeto de ARC2_StoreEndPoint, puntualmente el método de nombre handlyQueryRequest, quien implementa funcionalidades relacionadas con la transformación de sentencias tipo SPARQL a SQL por medio de la función (ARC2_SPARQLPlusParser). Además de ello, implementa mecanismos en los que verifica la forma y el tipo de operación que se está invocado.

Cuando estas operaciones toman lugar, el siguiente paso hace uso de un método que obtiene los resultados en forma de XML Bindings. Puntualmente, se genera una estructura Xml, con la que es

posible representar los datos obtenidos en etiquetas tipo HTML. La función que gestiona esta operación, getSparQlSelectResults(), adopta la codificación mostrada en la tabla 4.15

```
$vars = $r['result']['variables'];
$rows = $r['result']['rows'];
$r = '' .
    '<?xml version="1.0"?>' .
  $r .= $nl . '  <results>';
  if (is_array($rows)) {
        $r .= $nl . '     <binding name="' .$var. '">';
        if ($row[$var . ' type'] == 'uri') {
          $r .= $nl . '       <uri>' .htmlspecialchars($row[$var]). '</uri>';
        }
        elseif ($row[$var . ' type'] == 'bnode') {
          $r .= $nl . '       <bnode>' .substr($row[$var], 2). '</bnode>';
        }else {
          $dt = isset($row[$var . ' datatype']) ? ' datatype="' .htmlspecialchars($row[$var . '
datatype']). '"' : '';
          $lang = isset($row[$var . ' lang']) ? ' xml:lang="' .htmlspecialchars($row[$var . ' lang']).
'"' : '';
          $r .= $nl . '       <literal' . $dt . $lang . '>' .htmlspecialchars($row[$var]). '</literal>';
        }
        $r .= $nl . '     </binding>';}}$r .= $nl . '    </result>';
```

Tabla 4.15 Fragmento de código función de búsqueda.

Una vez se han producido todas las interacciones anteriormente mencionadas, y tomando nuevamente como punto de referencia el método llamado sparql_render_query_results () como segundo proceso, es importante hacer referencia a los procedimientos que permiten visualizar los resultados obtenidos durante el proceso de consulta. Como se menciono en el apartado 4.3.2.1.2, la fase de diseño de este proyecto permite el uso de interfaces basadas en las funcionalidades que ofrece las librerías gráficas del Api de Drupal, así como también las librerías del Framework Exhbit. Teniendo en cuenta que este apartado pretende explicar las abstracciones y modelos que se llevan a cabo durante la construcción de estas interfaces, es necesario especificar las interacciones que se producen por cada una de estas alternativas.

La primera de estas alternativas corresponde al uso de Exhibit como herramienta de Faceted Browsing, en la cual haciendo uso de un archivo Java Script (exhibit-api.js), es posible obtener

una representación categórica de las triplas contenidas en un archivo formato .Json. Una de las razones por las cuales se habla explícitamente de un archivo tipo Json, es porque la representación gráfica y estándar de Exhibit no contempla el uso de datos almacenados de forma persistente, pero sí de información representada en formato Json.

La forma en la cual se aborda este proceso es inicialmente realizando un proceso de conversión de la fuente de datos Rdf a un fichero tipo Json. Cabe resaltar que esta es una actividad bastante sencilla, dada la gran cantidad de herramientas que existen en la Internet para efectuar este tipo de labores. Una vez es realizada esta actividad, el siguiente paso consiste en generar un archivo tipo html, en el cual se referencie tanto el archivo Java script llamado exhibit-api.js, como el archivo que tiene formato Json y que ha sido generado a partir de la labor de transformación. La Tabla 4.16 muestra un ejemplo de la forma en que se hace uso de Exhibit.

```
<html><head>
<title>Prueba de Exhibit Geovanny 2011 GSI-UPM</title>
<link href="nombreDeArchivoJson.js" type="application/json" rel="exhibit/data" />
   <script src="http://static.simile.mit.edu/exhibit/api-2.0/exhibit-api.js"
         type="text/javascript"></script>
   <style>
   </style>
 </head>
 <body></body>
 </html>
```

Tabla 4.16 Fragmento de código usando exhibit stand alone faceted browsing

No obstante, debe tenerse en cuenta que las actividades mencionadas anteriormente corresponden a procesos que se desarrollan de forma local “stand alone”. Dado que este proyecto se caracteriza por promover el uso de herramientas que permitan formar modelos dinámicos de búsqueda, a continuación se detalla la segunda alternativa para el uso de Exhibit como herramienta de Faceted Browsing

Esta alternativa contempla el uso de funcionalidades de Faceted Browsing directamente desde los métodos y funciones que se han construido para este módulo. Puntualmente desde la función llamada sparql_render_query_results, la cual al basarse en métodos heredados directamente de Drupal como drupal_render(), permite mostrar cambios en el contenido de forma dinámica. De esta forma, sobre este método se crea un elemento gráfico que permita definir las opciones de salida de búsqueda que el usuario desea escoger, siendo estas: Exhibit y Drupal Native.

En lo que respecta a Exhibit, se crea una variable que permita interpretar estilos y formas html en forma de tags. De tal manera que al momento de codificar funcionalidades sobre estos *tags* sea posible observar contenido de scripts invocados remotamente. Sobre esta opción se hace el consumo de un servicio web ofrecido por el proyecto Smile, creador de la herreamienta Exhibit. Este servicio permite efectuar los procesos de transformación y generación de contenido únicamente con la especificación de una fuente dinámica de información RDF y la configuración de algunos parámetros sobre la consulta. La tabla 4.17, ofrece más información al respecto.

```
$output .= '<div style="display: table-cell; padding-right: 15px">';
    $output .= drupal_render($form['exhibit']);
    http://simile.mit.edu/babel/translator?reader=rdf-xml&
     writer=exhibit-json&mimetype=output-mime-type&url=http://localhost/rdf/fuente.rdf
    $output .= drupal_render($form['native']);
$output .= '</div>';
```

Tabla 4.17 Fragmento de código usando exhibit web service faceted browsing

Respecto a los datos representados en la tabla 4.17, es posible especificar:

- reader=rdf-xml : Corresponde al formato original en el que se encuentra la fuente de información Rdf.
- writer=exhibit-json : Corresponde al formato de salida en que se quieren estructurar las triplas presentes en el Rdf original.
- url=http://localhost/rdf/fuente.rdf : Corresponde al sitio en el cual se encuentra ubicada la fuente Rdf que sufrirá la transformación y representación.

Por otra parte, se cuenta con la opción de representación basada en las funcionalidades que ofrece las librerías gráficas del Api de Drupal. En este caso la función sparql_render_query_results() deberá hacer uso de la estructura Xml que se ha obtenido como resultado del proceso de consulta Sparql por medio de la función getSparQlSelectResults(). No obstante, sobre esta actividad deberá efectuarse una ligera modificación respecto a la forma en la que se estructura y se almacena la información sobre el fichero Xml. De manera puntual deberá realizarse una funcionalidad que vincule cada uno de los nodos presentes en forma de binding sobre una estructura dinámica (Array). Haciendo uso de funciones como array_keys y array_maps sera posible consultar toda la información que se almacene en los arreglos propuestos. De esta forma, deberá crearse una variable que permita interpretar estilos y formas html en forma de tags. De tal manera que al momento de utilizar la información presente en los arreglos, esta pueda ser visualizada fácilmente. La tabla 4.18, ofrece un fragmento de código que implementa parte de la funcionalidad.

```
$output = $bnodes = array();
 foreach ($input->results->result as $result) {
   $item = array();
   foreach ($result->binding as $binding) {
     if (isset($binding->uri)) {
       $value    = rdf_uriref((string)$binding->uri);
     }
     else (isset($binding->bnode)) {
       $bnode_id = (string)$binding->bnode;
       $value    = isset($bnodes[$bnode_id]) ? $bnodes[$bnode_id] : ($bnodes[$bnode_id] =
rdf_bnode());
     }
$vars = array_keys($result[0]);
$rows = array_map('array_values', $result);
```

Tabla 4.18 Fragmento de código que almacena la estructura Xml sobre arreglos temporales.

Rdf Scrapping ha sido el nombre que se le ha otorgado a la actividad que permite extraer información de sistemas de gestores de Ideas externos. Su objetivo principal es proporcionar un conjunto de datos organizados en forma de triplas que sirvan como información base para el proceso de búsqueda planteado en este proyecto. La estructura de este elemento, como muestra la figura 4.15, consta de dos partes:

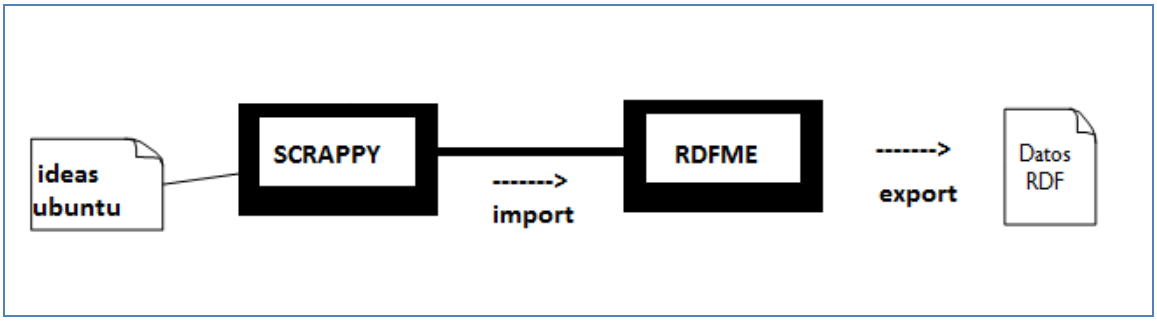


Figura 4.15 Arquitectura del modelo de extracción de datos

La primera de ellas, corresponde a un proceso de extracción de datos usando la herramienta llamada Scrappy. Esta herramienta corresponde a una aplicación desarrollada en el grupo de Sistemas Inteligente de la Universidad Politécnica de Madrid, la cual implementa técnicas de extracción basadas en reglas que asumen un conocimiento ontológico previo. Haciendo uso de

una ontología, provee clases y propiedades por medio de las cuales es posible realizar procesos de extracción con base a criterios de selección. Algunos de los criterios más importantes de selección son:

- Diseño (sc: CssSelector): Corresponde al criterio por medio del cual es posible realizar procesos de extracción con base en las etiquetas presentes en las hojas de estilo de los ficheros HTML. La idea de este criterio es poder ubicar y extraer el contenido presente sobre un elemento de diseño en particular.
- Localización (sc: XPathSelector): A través de este criterio, es posible realizar la extracción de información con base en la ubicación de un elemento. Este criterio de selección es ampliamente usado cuando se requiere extraer información que se deriva y asocia de un nodo común.
- Coincidencia (rdf: type: sc: SliceSelector): Corresponde al criterio más elemental de extracción de datos, ya que mediante el es posible obtener el contenido de una estructura que se identifique con un nombre.

Un punto interesante a destacar acerca de Scrappy, es que esta herramienta permite vincular la información que ha sido extraída con el modelo de datos ontológicos para el que será utilizado. Esta actividad es realizada haciendo uso de una serie de definiciones sobre un archivo de contexto tipo yarf. Cuando los procesos de extracción de datos se han llevado a cabo, esta herramienta tiene la posibilidad de ofrecer múltiples formatos de salida, entre ellos ficheros de la forma Rdf, Json y Jpg.

La segunda actividad que se lleva a cabo durante este proceso, corresponde al uso de la información que ha sido recolectada en el proceso inmediatamente anterior. Tal como se expreso al inicio de este apartado, estos datos serán la fuente de información base para el proceso de búsqueda. La forma en la cual se vinculan estos datos sobre el sistema, es haciendo uso de una utilidad de importación presente en el modulo llamado Rdfme.

4.3.2.1.3 Interfaz Gráfica.

Al observar la figura 4.3, es posible notar que la primera actividad que se lleva a cabo durante este proceso, es la invocación del elemento gráfico que permitirá asistir al usuario durante el proceso de búsqueda de información. Básicamente corresponde a una actividad que tiene como objetivo ofrecer la vista de búsqueda al usuario final y vincularla con las actividades propuestas y

desarrolladas en el componente Gi2se. El desarrollo de la interfaz de usuario de esta actividad, ha propuesto el uso de elementos gráficos presentes en algunas de las funcionalidades de la interfaz de aplicaciones de Drupal (formularios). Así mismo, ha incorporado un par de utilidades en forma de plugin. Una de estas utilidades es precisamente la librería *ahah Frameworks Form*, quien es elemento encargado de crear gráfica y dinámicamente los formularios necesarios para introducir la información que las búsquedas requieran. Es importante destacar que esta utilidad hace uso de un mecanismo de Java script por medio del cual es posible crear tantos elementos gráficos se requieran si necesidad de "renderizar" la totalidad de elementos presentes en una página. La figura 4.19 muestra en detalle la forma en que presenta visualmente estos elementos.

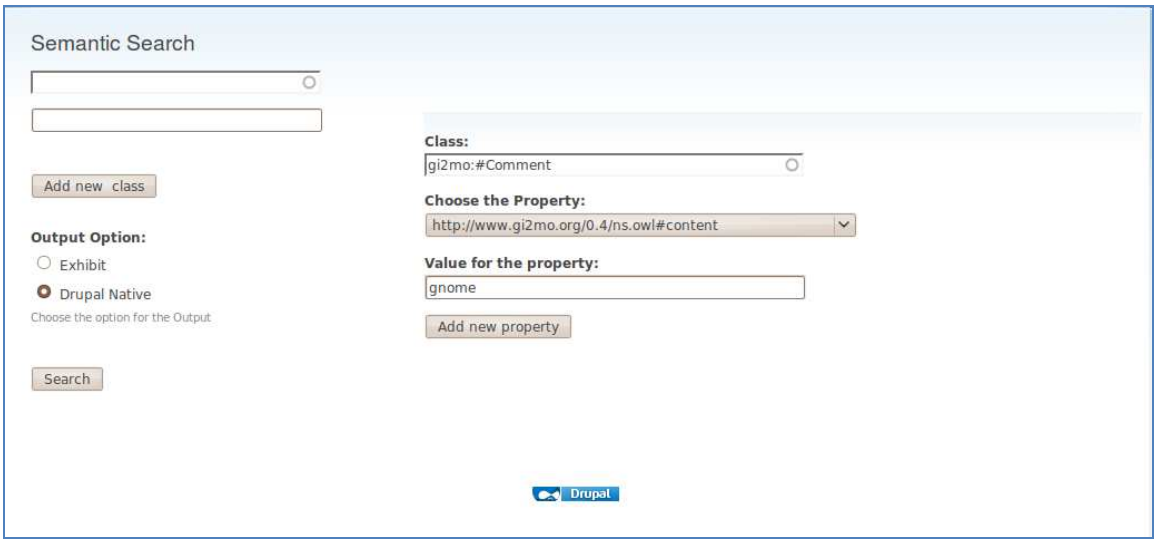


Figura 4.19 Generación Dinámica de Formularios Gi2SE.

Un aspecto interesante a mencionar respecto a esta utilidad, tiene que ver con el nivel de jerarquías presente en los elementos gráficos de este componente. Este tipo de características garantizan la integridad referencial de clases y propiedades, lo cual hace posible construir escenarios de búsqueda que respecten el nivel de relación y herencia establecida. Por otra parte, se cuenta con una segunda utilidad llamado Evoc y cuyo propósito general, es servir como herramienta de importación de vocabulario sobre gestores de contenido. No obstante y teniendo en cuenta que este apartado hace referencia al diseño de las interfaces gráficas de usuario, es preciso mencionar que esta utilidad posee una funcionalidad de tipo gráfica, por medio de la cual es posible establecer sugerencias de nombres conforme un usuario digita información sobre una caja de texto. Básicamente corresponde a una función Java Script que realiza mecanismo de sincronización entre los elementos almacenados en una base de datos y los espacios de nombres sugeridos en una clase. La figura 4.20 ofrece detalles respecto a esta funcionalidad.

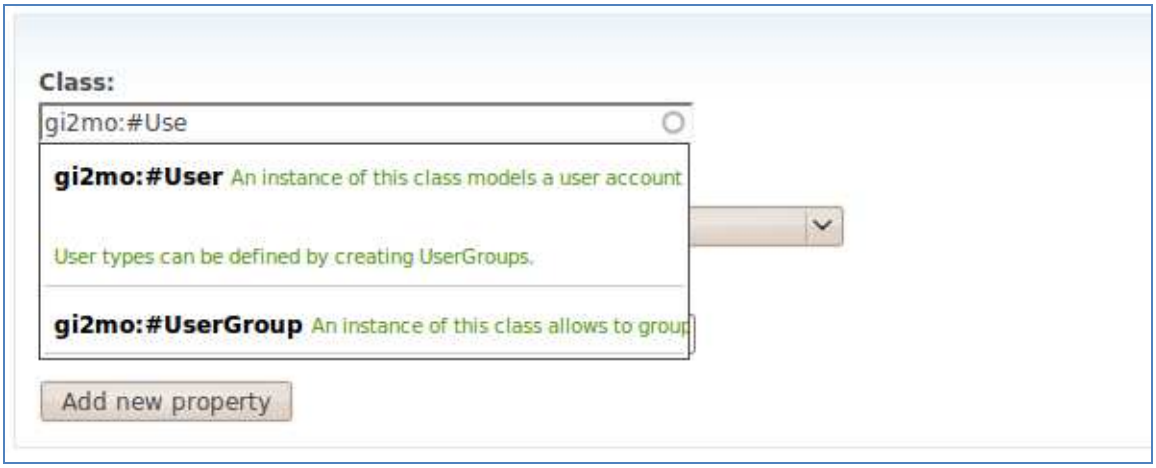


Figura 4.20 Implementación de Librería Evoc como herramienta de auto completacion.

El último procedimiento que contempla la etapa de diseño de la interfaz gráfica, corresponde a la generación de una vista que muestre los resultados obtenidos del proceso de búsqueda. Dicho procedimiento, es posible realizarlo haciendo uso de dos (2) técnicas.

La primera de ellas es efectuando un mecanismo de serializacion que permita que los datos obtenidos en forma de archivo Xml puedan ser integrados sobre estructuras de datos dinámicas (arreglos) en Php. De esta forma, por medio de tablas y elementos propios del API de Drupal, es posible predefinir los formatos de presentación de las búsquedas. La figura 4.21 ofrece detalles de un posible formato de presentación.

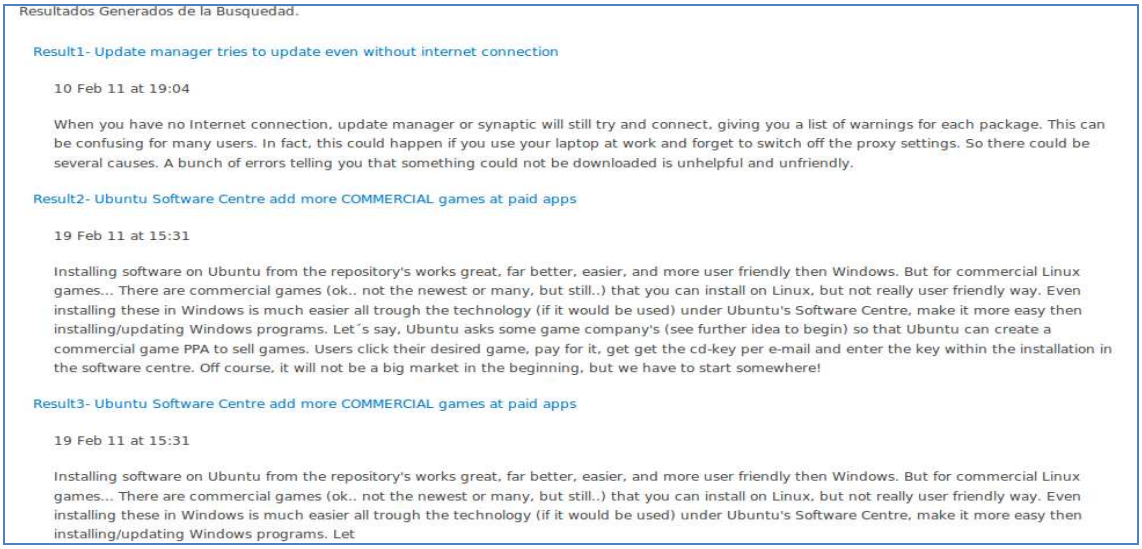


Figura 4.21 Formato de presentación serializacion Php.

En este punto, es importante hacer una clara distinción acerca de las formas en que se pueden abordar el uso de técnicas basadas en Exhibit. La primera de ellas hace referencia a la técnica de Faceted Browsing, en la cual es posible generar procesos de búsqueda que se caractericen por tener componentes gráficos más dinámicos e interactivos. En general Exhibit es una utilidad que haciendo uso de un archivo Java Script, permite organizar y configurar los resultados de una búsqueda de acuerdo a patrones de similitud presente en la información procesada.

Para ello, Exhibit requiere de archivos cuyo contenido posea una estructura organizada bajo la notación literal de objetos de Java Script (Json). Es de aclarar que este tipo de técnica no requiere nivel de persistencia alguno, dado que la forma en la cual se organiza la información puede ser predeterminada u ordenada taxonómicamente. Para este tipo de visualización suele usarse procesos de conversión de archivos RDF a Json, ya que RDF se presenta como una buena alternativa para estructurar la información de dominios de conocimiento en forma de triplas, lo cual es precisamente una de las bases funcionales de Exhibit. La figura 4.22 muestra el uso de Exhibit como herramienta Faceted Browsing.

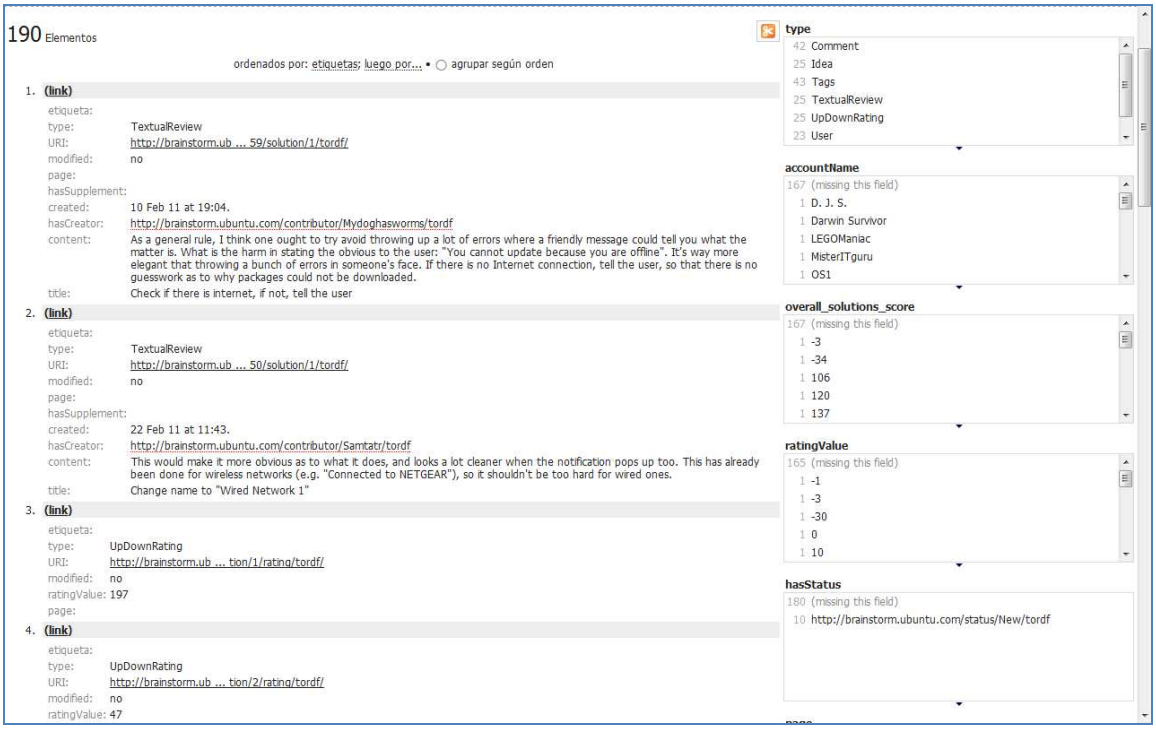


Figura 4.22 Exhibit como herramienta Faceted Browsing.

Por otra parte y como segunda opción de visualización, se tiene el uso de Exhibit como herramienta de representación gráfica sobre los datos e información obtenida de forma persistente. Básicamente se ha propuesto el desarrollo de una actividad a través de la cual sea

posible realizar procesos de serializacion sobre los datos obtenidos de la consulta.

Teniendo en cuenta que el proceso de consulta realiza una actividad de representación en forma de Xml Bindings, se hace necesario realizar una modificación sobre uno de los métodos que incluye la librería Arc2, especialmente del método llamado GetRdfJSonSerializer(), de tal forma que el formato de salida que se obtenga incluya ficheros tipo Json con etiquetas tipo "ítem" y no etiquetas tipo "binding". Una vez estos archivos han sido generados, la opción de visualización contempla el uso de un archivo Java Script para la representación gráfica del fichero Json. El archivo que contiene las funcionalidades Ajax, puede usarse remotamente invocando el Url que a continuación se detalla (http://static.simile.mit.edu/ajax/api-2.0/simile-ajax-api.js.). La figura 4.23 representa los datos generados de un proceso de consulta haciendo uso de Exhibit.

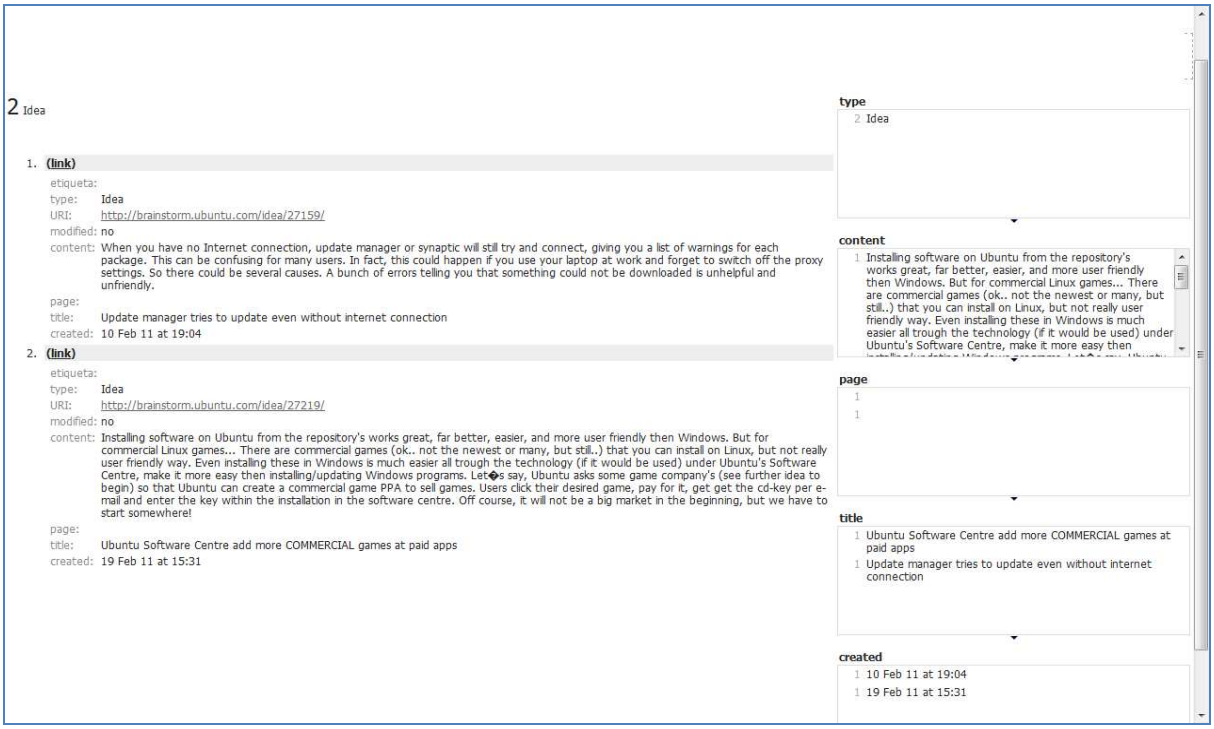


Figura 4.23 Exhibit como formato de salida de procesos persistentes.

4.4 Conclusiones

El estudio de la arquitectura ha definido los pasos a seguir para alcanzar el proceso de implementación. Igualmente ha destacado el patrón de software plug-in como una de las técnicas de más amplia aceptación para añadir nuevas funciones a aplicaciones de

mayor envergadura, en este caso el sistema de gestión de contenidos Drupal. Esta plataforma presenta una API extensa y muy bien documentada que contrasta con la dificultad que presenta su aprendizaje inicial. Los objetivos marcados inicialmente en la descripción de la arquitectura han sido cubiertos durante el desarrollo de los componentes. Además, se han implementado con éxito los casos de uso descritos en el capítulo anterior.

5. Capítulo V Conclusiones y Trabajos Futuros.

5.1 Introducción

Una vez concluidas las fases de diseño, análisis e implementación, es posible analizar el alcance obtenido sobre los objetivos planteados inicialmente. Este tipo de análisis, no sólo pretende emitir las conclusiones obtenidas y las valoraciones realizadas en este trabajo. También, pretende ofrecer una visión crítica respecto a las múltiples actividades y procedimientos desarrollados durante las diferentes fases de este proyecto, especialmente sobre aquellas que se han desarrollado en el capítulo IV. De esta forma, se busca ofrecer una serie de pautas y recomendaciones que permitan garantizar la continuidad del proyecto con posibles líneas de investigación a futuro.

5.2 Conclusiones

5.2.1 Gi2se Vs Exhibit.

Los tiempos de respuesta obtenidos como resultado de las pruebas efectuadas en la medición de los tiempos de consulta, hacen pensar que el concepto de Faceted Browsing para ser una muy buena alternativa para abordar los tema de consulta sobre fuentes semánticas de información. Además de ello, los componentes gráficos que usa Exhibit parecen tener una muy buena aproximación con las tecnologías que hoy día proponen conceptos como el de *Mashup*. Sin embargo y pese a todas las bondades que parece tener esta utilidad, el uso de arquitecturas basadas en tecnologías como ARC2 y SPARQL, sin lugar a dudas, representan la mejor opción en temas de consulta semántica de información. Algunos de los argumentos que permiten sustentar esta afirmación, se mencionan a continuación:

En primer lugar, el tema de actualización dinámica de contenidos representa uno de los tópicos más importante del escenario propuesto, dado que por medio de este concepto ha sido posible conseguir modelos de búsqueda que se caractericen por obtener resultados mucho más

actualizados de los que en un momento dado puede llegar a tener Exhibit como técnica de *Faceted Browsing*.

Detrás de todo el esquema de actualización mencionado, se encuentra el elemento de mayor peso en la arquitectura propuesta, el cual corresponde al gestor semántico de Ideas. Gi2mo Ideas ha representado una muy buena opción para plantear escenarios de prueba de búsqueda de información, dado la gran cantidad de relaciones semánticas que proporciona su ontología. Además de ello, porque hace uso de utilidades de exportación e importación de datos, lo cual facilita el nivel de interoperabilidad de esta herramienta con otro tipo de sistemas.

Sin embargo las características que realmente marcan la diferencia con Exhibit , son aquellas que están vinculadas con el esquema de búsqueda relacional que propone Gi2SE y con la forma en que asiste al usuario respecto a los procesos de búsqueda que impliquen dos o más clases sobre un mismo dominio de información. Con la característica de búsqueda relacional, Gi2SE tiene la posibilidad de realizar consultas que contemplen operaciones lógicas. Labor que no es posible realizar en Exhibit, ya que este Framework utiliza el concepto de similitud para ofrecer los resultados de sus búsquedas. En este sentido, será imposible realizar procesos de búsqueda en Exhibit que contemple el uso de operadores como mayor o menor. Además de ello, Gi2SE también ofrece la posibilidad de relacionar dos o más propiedades que pertenezcan a diferentes clases, lo cual hace que las búsquedas presente un nivel de detalle más alto.

No obstante, las interfaces gráficas de Exhibit plantean una muy buena propuesta para el manejo y visualización de la información obtenida durante los procesos de búsqueda. Hasta tal punto que sea posible pensar en escenarios que además de contemplar SPARQL + ARC2, tengan la capacidad de usar Exhibit como Framework de diseño para la representación gráfica de elementos. Este y otros temas serán tratados en el apartado de trabajos futuros.

5.2.2 Acerca de Drupal.

Las fases de análisis y desarrollo han confirmado a Drupal como el gestor de contenidos de software libre con el más amplio catalogo de funcionalidades semánticas. Temas como, el soporte de formatos RDF y el soporte de consultas semánticas basadas en SPARQL, han demostrado la gran estabilidad que tiene esta aplicación para el desarrollo e incorporación de servicios relacionados con el área de la WEB semántica. Sin embargo el punto más interesante a analizar de este gestor, tiene que ver con las interfaces de servicios que ofrece para el desarrollo e incorporación de nuevas tecnologías semánticas sobre su núcleo. Haciendo uso de interfaces tipo Hook propone un modelo en el que es posible conseguir un nivel de acoplamiento lo

suficientemente eficiente entre los componentes. Un ejemplo que demuestra lo estable que es la capacidad semántica de Drupal, es el proyecto Gi2MO, el cual ha desarrollado más de 7 actividades en forma de componentes y ha permitido sincronizar cada una de estas en función de un módulo llamado Gi2MO Ideas Stream. En este sentido se ha formado toda una gama de servicios alrededor de un sólo producto. En el caso de Gi2SE se ha creado un asistente de búsqueda semántica usando parte de las funcionalidades presentes en los módulos del proyecto Gi2mo.

5.2.3 Acerca de ARC2.

El uso de ARC2, ha marcado sin lugar a duda una muy buena opción en el manejo de funcionalidades semánticas como SPARQL sobre ambiente WEB que implementan métodos y funciones de PHP, sin embargo y en vista a las actuales condiciones que impone la WEB, es interesante proponer modelos de consulta que haciendo uso de estas funcionalidades permitan generar procesos de búsqueda más activos y dinámicos, en lo que el re uso de información, de utilidades y de componente se conviertan en temas clave para la construcción de nuevos servicios . Este proyecto fin de máster precisamente coincidió con una de estos objetivos y propuso el desarrollo de una herramienta gráfica que permitiera mejorar el nivel de interacción entre el usuario y el sistema. De tal forma que haciendo uso de otras funcionalidades asociadas al núcleo de aplicaciones de Drupal, es posible que los usuarios puedan realizar procesos de búsqueda autos asistidos.

Una importante funcionalidad que implementa la librería ARC2, son los llamados puntos de terminación SPARQL (URL END POINT), en los cuales, es posible consumir servicios de búsqueda SPARQL de forma remota. No obstante en su forma original, estos puntos no ofrece la posibilidad de configurar fuentes de información semántica para los procesos de búsqueda. Por esta razón, este proyecto fin de Master propuso el desarrollo de una actividad en la que fue posible configurar de forma dinámica las fuentes de información semántica que el usuario desee usar para los procesos de búsqueda.

Un punto cuestionable y debatible respecto al uso de los puntos de terminación SPARQL, es la forma en que se exponen para su invocación y la forma en que retornan los datos de los resultados obtenidos. En lo referente a los mecanismos de invocación, esta herramienta hace uso de antiguos mecanismos de paso por parámetros sobre las operaciones HTTP. Por otra parte, la

forma en que se retornan las respuestas corresponde a mecanismos de auto descarga gestionado en las secciones del usuario. Las ideas descritas anteriormente nos hacen pensar en la necesidad de construir puntos remotos de consulta SPARQL consumidos probablemente en forma de Web Services y obteniendo respuestas en archivos de formato XML. Este y otros temas serán tratados en el apartado de trabajos futuros.

5.2.4 Generalidades.

El desarrollo de este proyecto, ha exigido un elevado nivel de dedicación y un alto nivel de aprendizaje respecto a la forma de desarrollar componentes de tipo semántico sobre ambientes WEB, especialmente en la forma de comprender los método y algoritmos que utiliza la librería ARC2, para realizar los procesos de mapeos y transformación de información entre SPARQL y SQL propiamente. El uso de un motor de base de datos relacional dentro del modelo de búsqueda semántica, representa una muy buena opción para los temas de integridad referencial, los cuales de alguna forma pretenden implementar principios y funcionalidades asociadas con la inferencia y razonamiento de información. Es necesario recordar que ARC2 carece de estas propiedades.

Finalmente, el sistema desarrollado pone de manifiesto la importancia de la Web Semántica como futuro de Internet, demostrando la capacidad del modelo RDF para la comunicación de conocimiento y de las ontologías para expresarlo.

5.3 Trabajos futuros

Una vez que ha finalizado el desarrollo del proyecto, es conveniente analizar el sistema en busca de nuevas características y servicios que puedan mejorar cada uno de los componentes. Las siguientes secciones están dedicadas a describir brevemente propuestas para la futura orientación del proyecto.

5.3.1 Concepto de Web Services End Point SPARQL

Como se menciono en el apartado 5.2.3, los puntos de terminación SPARQL representan una muy buena opción en el tema de consultas semánticas de forma remota. De hecho, parte de esta funcionalidad ha sido adaptada y configurada en el módulo de búsqueda semántica Gi2SE. Es necesario destacar, que cuando un usuario ejecuta un proceso de búsqueda local sobre el componente en mención, realmente está haciendo uso de un Url End Point que ha sido

configurado y desplegado de forma local. Sin embargo y, teniendo en cuenta la forma redundante en cómo se producen las diferentes interacciones y conversiones entre los elementos de cada uno de los escenarios partícipes en esta actividad, es necesario proponer una serie de alternativas que permitan reducir la complejidad y tiempos de respuesta de las operaciones. Un punto interesante a mencionar antes de entrar en los detalles de esta propuesta, es aquel que hace referencia a los escenarios partícipes, de manera general, cuando un usuario ejecuta un proceso de búsqueda deberá producirse un modelo de interacción semejante al que se aprecia en la figura 5.1

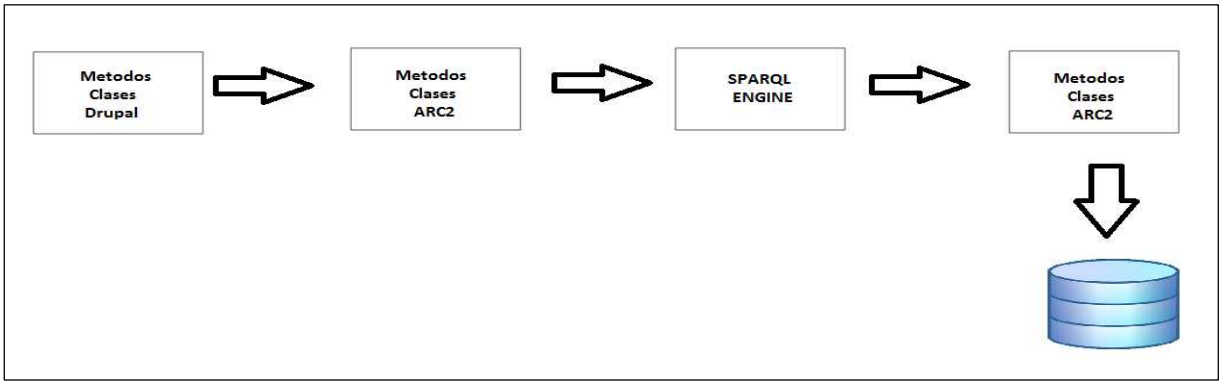


Figura 5.1 Modelo de interacción entre los escenarios de búsqueda.

El modelo o propuesta a considerar, tomaría como punto central, el desarrollo de un servicio Web generado y desplegado en tecnología REST, el cual a manera de ejemplo debería invocarse de la siguiente forma:

```
http://gsi.dit.upm.es/ws/sparql/?graph=param1&query=param3
```

Tabla 5.1 Forma de consumo del servicio Web.

Cuando este servicio es invocado deberá producirse un formato de salida similar al que se aprecia en la figura 5.2

```
<table class="sparql" border="1">
  <tr>
    <th>s</th>
    <th>p</th>
    <th>o</th>
  </tr>
  <tr>
    <td>http://gsi.dit.upm.es/AbductiveInferenceProcess</td>
    <td>http://www.w3.org/1999/02/22-rdf-syntax-ns#type</td>
    <td>http://gsi.dit.upm.es/umbel#AbstractConcept</td>
  </tr>
</table
```

Tabla 5.2 Ejemplo de posible respuesta de Web Service End Point.

Con la utilización de un servicio web, la complejidad de las interacciones podría ser reducida y por ende los tiempos de resolución de las consultas. Un punto importante a destacar, sería la ausencia de transformaciones entre tipos de formatos del lado del escenario desde donde se lanza la búsqueda.

A. Manual de instalación de Módulo de Búsqueda Semántica (Gi2se).

A.1 Activación del módulo

La instalación y activación del módulo de asistencia de búsqueda semántica (Gi2se) deberá seguir los procedimientos que a continuación se mencionan:

Descarga del módulo. El usuario deberá comprobar que la versión del módulo es compatible con la versión del gestor de contenidos sobre el cual desea realizar el proceso de instalación.

Descompresión del archivo. Haciendo uso de una herramienta de descompresión de archivos, el usuario deberá des empaquetar el contenido del fichero llamado Gi2se. De esta forma sobre el directorio en que se ha llevado a cabo la actividad de descompresión, deberá aparecer una carpeta denominada *gi2se.* Algunos de los elementos que contiene este directorio son las carpetas *include, store, extractors, parser y serializer*, así como también los archivos *gi2se.module, gi2se.install* y *gi2se.info.*

Ubicación de la carpeta en el núcleo Drupal. El siguiente paso, consiste en transferir la carpeta *gi2se* sobre la carpeta modules, la cual se encuentra ubicada en el directorio de instalación de Drupal. Es importante mencionar que esta carpeta esta estrictamente reservada para desplegar los módulos del nucleó de Drupal, así como también para los módulos desarrollados por terceros.

Una vez se han desarrollado estas actividades, el siguiente paso consiste en realizar el proceso de instalación como tal. Para ello se hace necesario que los usuarios se autentiquen sobre el sistema gestor de contenidos con privilegios de administrador.

Comprobación de Dependencias. A fin de comprobar las dependencias que requiere el componente. El usuario deberá seguir la ruta *Administer > Site building > Modules.* Para activar el componente *gi2se,* el sistema deberá comprobar que se encuentre activados los módulos llamados *rdf* y *sparql.*

Activación del Modulo. Seleccionando la opción "*Enabled*". Ubicada al lado izquierdo de la descripción del módulo y haciendo click sobre el botón llamado "Save Configuration". Ubicado en

la parte inferior de la pantalla. El usuario deberá estar en capacidad activar el modulo. La figura A.1 muestra en detalle el desarrollo de esta actividad.

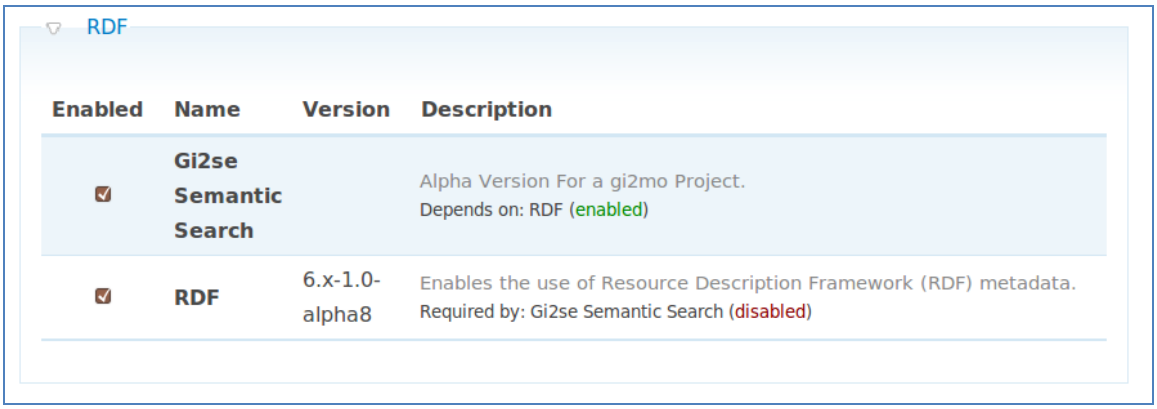


Figura A.1. Interfaz de usuario: activación del módulo Búsqueda Semántica (Gi2se)

Cuando el proceso de instalación se ha llevado de forma satisfactoria, el gestor de contenidos deberá mostrar un aviso en el que indique el siguiente procedimiento a realizar. El cual se trata precisamente de la etapa de configuración. La figura A.2 muestra la información mencionada.

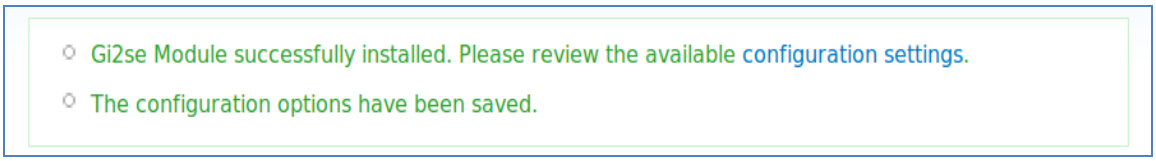


Figura A.2. Aviso de opciones de configuración del módulo Búsqueda Semántica (Gi2se)

B. Manual de usuario de Módulo de Búsqueda Semántica (Gi2se).

B.1 Introducción

Gi2mo Semantic (Gi2se) es una extensión del gestor de contenidos de Drupal que permite realizar procesos de búsqueda semántica de forma auto asistida. El objetivo de este módulo es proponer escenarios de búsqueda que se caractericen por tener un nivel de dinamismo mayor al que presentan muchos de los actuales sistemas de búsqueda semántica. Uno de los mecanismos que precisamente impulsa este tipo de propuestas, es la posibilidad de configurar múltiples fuentes de información sobre el componente de búsqueda. Al mismo tiempo, se pretende alcanzar escenarios de búsqueda en los que el usuario se tenga la posibilidad de gestionar de forma dinámica el control de cambios asociados a las fuentes de información.

Al igual que los temas configuración, la forma en que interactúan los usuarios frente al sistema, también representan un importante tema para la búsqueda de escenarios agiles y dinámicos.

B.2 Opciones de Configuración.

Sobre esta etapa es posible introducir los parámetros básicos de configuración que se utilizarán sobre los procesos de búsqueda. Como se aprecia en la figura B.1, esta opción cuenta con tres alternativas (Sparql UrlEndPoint, GraphName y UrlRdfFile):

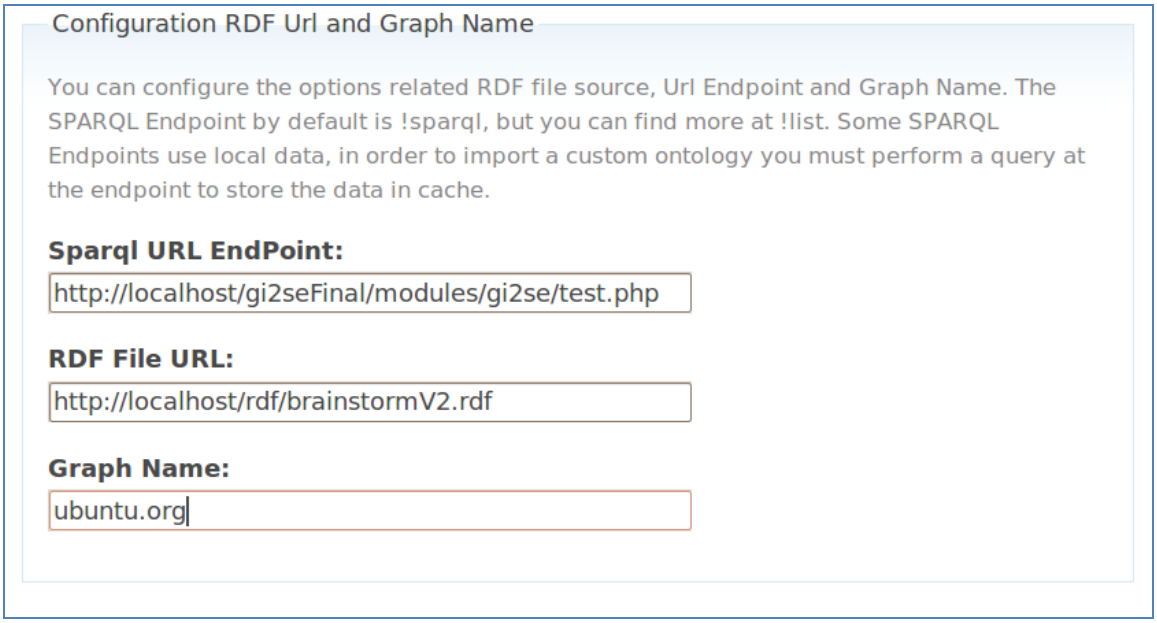


Figura B.1. Opciones de Configuración del módulo Búsqueda Semántica (Gi2se).

Sparql URL EndPoint: Sobre este campo deberá introducirse el Url desde el cual se ejecutará la búsqueda Sparql. Por defecto, este campo asocia un valor de un Url EndPoint que se ha configurado y gestionado localmente.

RDF File URL: Sobre este campo deberá introducirse el Url donde se encuentra ubicada la fuente de información que será usada como base de los procesos de búsqueda.

Graph Name: Sobre este campo deberá colocarse el nombre del grafo que se asociara con la fuente de información configurada en el campo inmediatamente anterior.

B.3 Búsqueda de información.

Sobre esta etapa es posible hacer uso de los elementos de tipo gráfico que guiaran al usuario durante los procesos de búsqueda. Cuenta con una interfaz gráfica prevista de una serie de elementos a través de los cuales es posible introducir información de forma auto asistida. El primer punto de contacto que deberá tener el usuario con este proceso, corresponde a un botón de nombre “New Class”, sobre el cual el usuario deberá click para crear los campos de búsqueda asociados a la primera clase instanciada de búsqueda. La forma en la cual es posible acceder a esta configuración, es ingresando a un vinculo ubicado en el panel izquierdo del gestor. Este enlace es de nombre “Semantic Search”.

Una vez producida esta acción, el usuario deberá observar la aparición de tres campos creados de

forma dinámica, los cuales corresponden a los siguiente nombres: “Class”, “Choose the Property” y “Value for Property”. La figura B.2 muestra la información que se despliega una vez el usuario ha presionado el botón en mención.

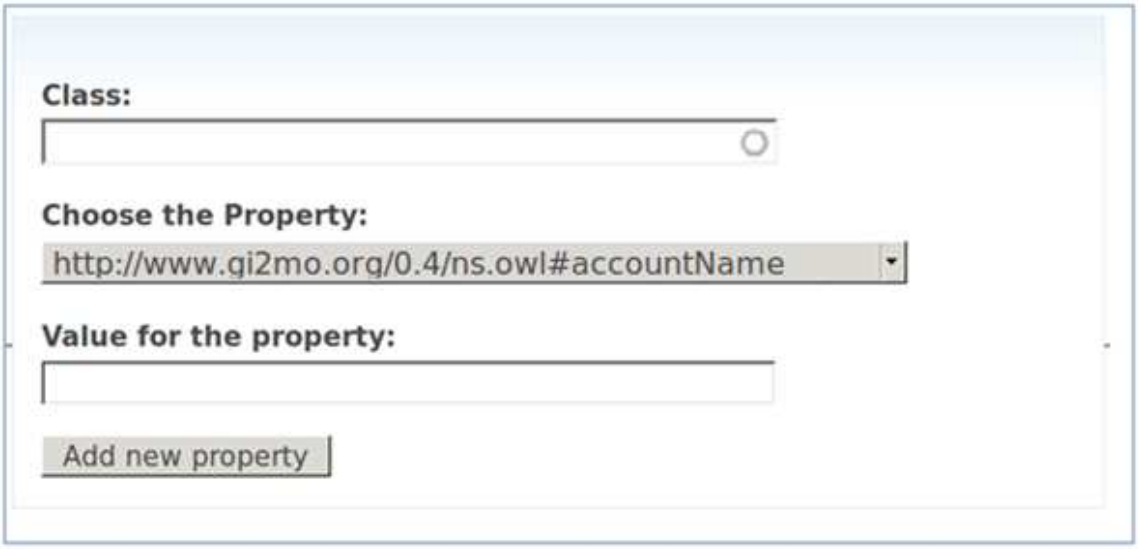


Figura B.2. Campos creados después de hacer uso del botón “New Class”.

La opción llamada “Class”, permite al usuario introducir el nombre de la clase que deberá usar como elemento base de la búsqueda. Esta opción dispone de una funcionalidad que permite emitir / sugerir de forma dinámica el nombre de las clases que el usuario podrá usar para realizar los procesos de búsqueda propuestos. La figura B.3 muestra un ejemplo de esta funcionalidad.

La opción llamada “Choose Property”, permite al usuario escoger las propiedades que deberá asociar a la clase anteriormente escogida. Los nombres de las propiedades desplegadas en la lista, se vinculan de forma directa con la clase en mención, es decir en el momento en que un usuario desee seleccionar algunas de las opciones presentes en la lista, sólo tendrá la posibilidad de escoger un número reducido de opciones ya que el sistema ha gestionado un procedimiento de filtrado en el que sólo muestra los términos que se encuentran directamente relacionados con la clase escogida. Un ejemplo de esta situación es posible observarlo en la figura B.3 en donde la lista despliega solo las propiedades relacionadas con la clase llamada Usuario.

La opción “Value for the Property”, permite ingresar y al tiempo vincular un valor sobre la propiedad escogida anteriormente. Esta opción no propone el uso de herramientas de auto sugerencia, dado que los datos introducidos sobre esta forma no son conocidos en principio por el sistema.

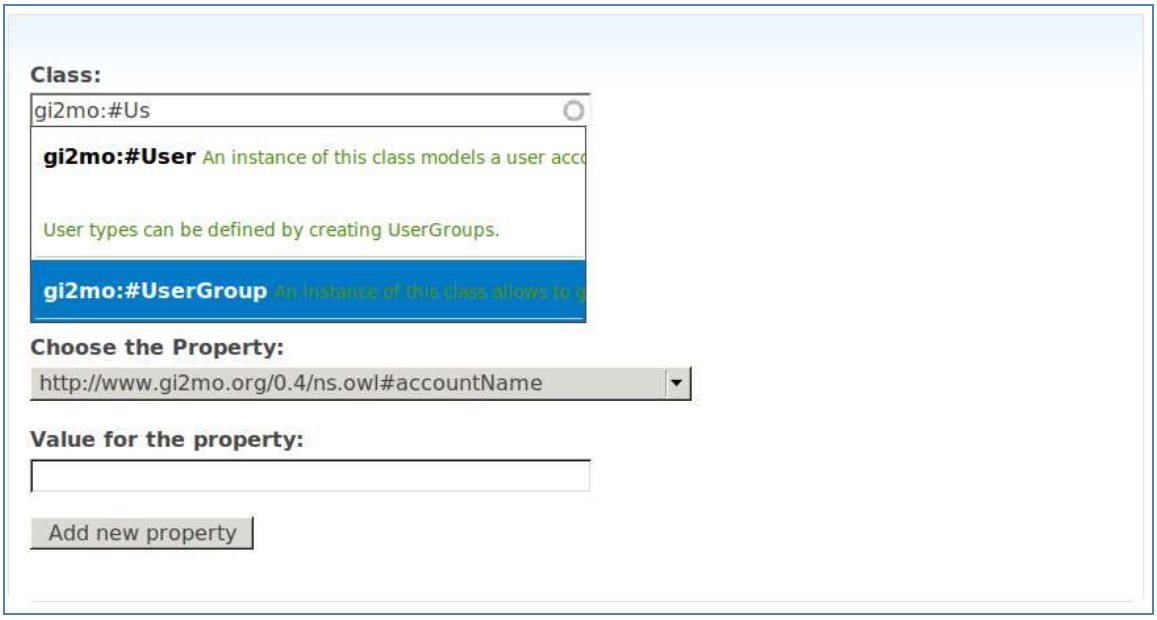


Figura B.3. Funcionalidad de auto sugerencia en Gi2se.

Como se aprecia en la figura B.3, En esta opción es posible vincular propiedades adicionales sobre las clases creadas. En esta forma, cuando el usuario hace uso del botón llamado "Add new Property", tendrá la posibilidad de crear campos adicionales que se vinculen como una propiedad más sobre la clase que ese momento este siendo utilizada. Este tipo de funcionalidades permite al usuario especificar múltiples criterios de búsqueda por clase instanciada. La figura B.4 muestra un ejemplo de dos propiedades asociadas a una misma clase.

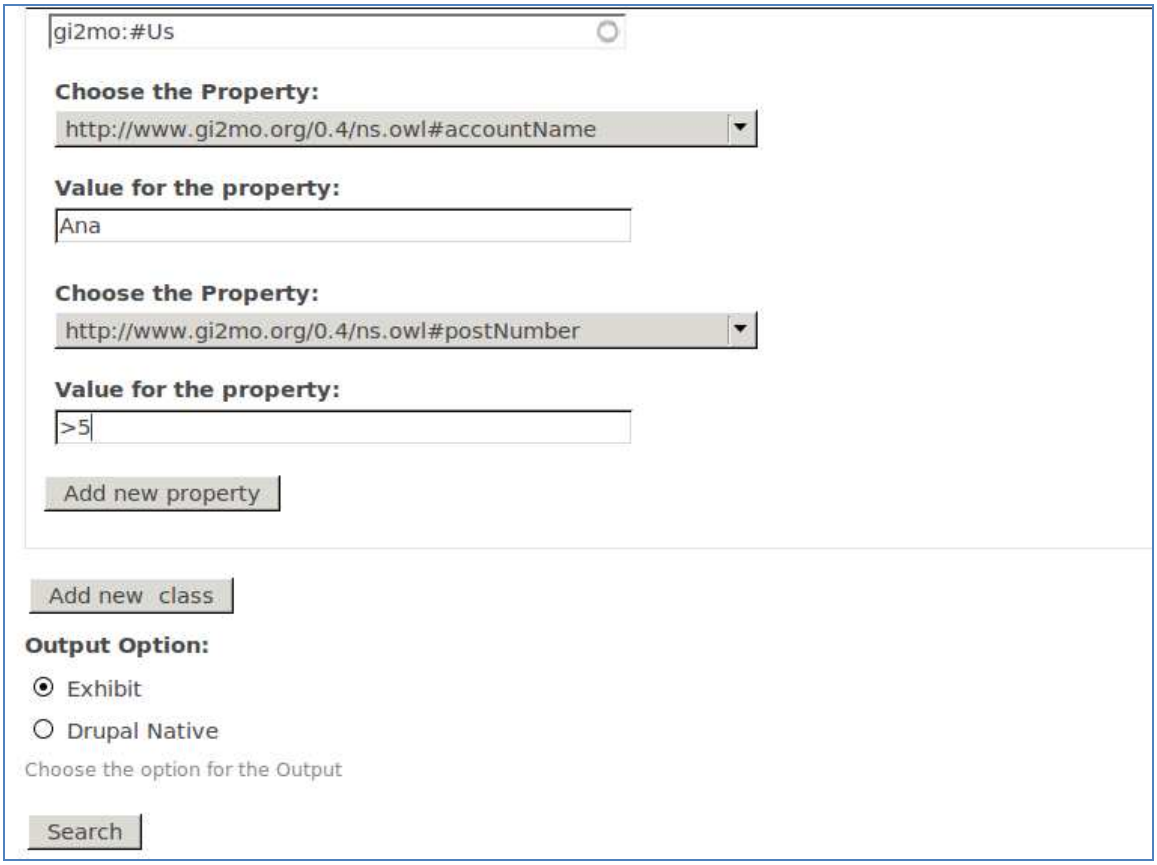


Figura B.4. Asociación de propiedades por clase instanciada.

Así como es posible generar múltiples propiedades por clase, en esta actividad también es posible generar múltiples clases sobre el proceso de búsqueda. Este tipo de funcionalidades ofrece al usuario la posibilidad no sólo de obtener información referente a una sola clase del dominio de información, sino que eventualmente el usuario puede ejecutar procesos de búsqueda que relacionen dos o más clases que contenga a su vez dos o más propiedades. De esta forma, se pretende que el usuario pueda obtener información más extendida y precisa en sus procesos de búsqueda.

Para concluir el manual de usuario, es necesario mencionar que la actividad de búsqueda ofrece la posibilidad de escoger dos formatos de salida para la entrega de resultados. La primera opción se relaciona con el concepto de faceted Browsing, gestionada por medio del Framework Exhibit. Por otra parte la segunda opción, "Drupal Native" permite mostrar los resultados en forma de listas, semejante al estilo en que lo hacen la gran mayoría de buscadores no semánticos.

6. Referencias

[GI2IS] Sistema de gestión de ideas: Gi2mo
http://ideas.gi2mo.org/

[XEHBI] Framework Exhibit
http://simile-widgets.org/wiki/Reference_Documentation_for_Exhibit
.

[COLIDA] CODINA, L. (2009): "¿Web 2.0, Web 3.0 o Web Semántica?: El impacto en los Sistemas de información de la Web". En I Congreso Internacional de Ciberperiodismo y Web2.0. Bilbao: Noviembre 2009.
<http://upf.academia.edu/lluiscodina/Papers/119339/%C2%BFWeb-2-0--Web-3-0-o-WebSem%C3%A1ntica---El-impacto-en-los-sistemas-de-informaci%C3%B3n-de-la-Web> [Consulta: 15/05/2011].

[MOOBR] F. Ruiz, J. Dolado. Una Ontología para la Gestión del Conocimiento de Proyectos Software. REICIS Revista Española de Innovación, Calidad e Ingeniería del Software, 2008.

[REAW] K. Finley, "3 Trends in Idea Management", Read Write Web, 2011

[TRA1] M. Turrell, "Forrester Update: Poor Use of Idea Management Systems",
<http://markturrell.wordpress.com/2010/05/24/forresterupdate-poor-use-of-idea-management-systems/> [Consulta: 15/05/2011].

[DRUPAL] Proyecto Drupal: Sistema Gestor de Contenidos.
http://drupal.org/press/drupal-6.0

[BGLI] J. P. Riederer, E. C. M. Baier, G. Graefe, *"Innovation Management – An Overview and some Best Practices"*, C-LAB Report Vol. 4, No. 3 Cooperative Computing & Communication Laboratory, 2005.

[GDES] J. Liedo, "Gestión de Innovación de Ideas: Entorno del Social Media",
<http://markturrell.wordpress.com/2010/05/24/forresterupdate-Poor-use-of-idea-management-systems/>[Consulta: 22/05/2011].

[GAYAL] G. Ayala, "Análisis de Clúster",
<http://www.uv.es/ayala/docencia/tami/t5_Cluster.pdf >[Consulta: 22/05/2011].

[TIMB2] Andy Carvin, "Tim Berners-Lee: Weaving a Semantic Web", Digital divide network, 2005.

[DBRV] D. Brickley, R.V. Guha, B. McBride, “RDF Vocabulary Description Language 1.0: RDF Schema”, *http://www.w3.org/TR/rdf-schema,* W3 Recommendation, 2004.

[SBFHJ] S. Bechhofer, F. Harmelen, J. Hendler, I. Horrocks, D. L. McGuiness, P. F. Patel-Schneider, L. A. Stein, “OWL Web Ontology Language Reference”, *http://www.w3.org/TR/owl-ref*, W3C Recommendation, 2004

[SPARL] M. Shaw, “SPARQL: SPARQL Protocol and RDF Query Language”, *http://students.washington.edu/mgaldzic/public_files/shaw_mebi_sparql20100126.pdf [Consulta: 24/05/2011].*

[TL03] M. Turrell, Y. Lindow, “The Innovation Pipeline”, Imaginatik Research White Paper, March 2003

[TAYLC] W. C. Taylor, "Here's an Idea: Let Everyone Have Ideas", The New York Times, 2006.

[ZBSA] K. A. Zien, S. A. Buckler, “From experience: Dreams to market: Crafting a culture of innovation”, Journal of Product Innovation Management, 14(4), 1997, p.274-287

[DSEPA] S. Corlosquet, “The RDFa initiative in Drupal , and how it will impact the Semantic Web”, *<http://semanticweb.com/files/SU/RDFa-in-Drupal-7-Slides.pdf> [Consulta: 02/06/2011].*

[DGHP] Página de referencia de la API de Drupal: hook *http://api.drupal.org/api/group/hooks*

[GSI2] A. Westerski, “What is Ideas Stream“ *<http://www.gi2mo.org/apps/ideastream/> [Consulta: 02/06/2011].*

[IDTP] Proyecto Idea Torrent: MySQL *<https://launchpad.net/ideatorrent-mysql>*

[RKAW] *Agrawal, R. and R. Srikant. Fast Algorithms for Mining Association Rules. in 20th Int. \ Conf. Very Large Data Bases, {VLDB}. 1994: Morgan Kaufmann.*

[CWS9] P. Castell, “La Web Semántica“ *< www.ii.uam.es/~castells/publications/castells-uclm03.pdf > [Consulta: 09/06/2011].*

[GPIAB] A. Grau, “Google puede conocer el estado de animo de su plantilla”, *<http://semanticweb.com/files/SU/RDFa-in-Drupal-7-Slides.pdf> [Consulta: 12/06/2011].*

[DTMT] G. Willians, “A Data Mining Tutorial”, *<http://maths.anu.edu.au/~steve/pdcn.pdf >[Consulta: 12/06/2011].*

[CJOFD] M. S. Chen, J. Han, and P. S. Yu. Data mining: An overview from a database perspective. IEEE Trans. Knowledge and Data Engineering, 8:866-883, 1996

[BDL98] BOLEY, D.L. 1998. Principal direction divisive partitioning. Data Mining and Knowledge Discovery, 2, 4, 325-344.

[WEIU] M. Hu, B. Liu, "Mining and summarizing customer reviews", KDD'04, 2004

[FCAD3] F. Ciravegna, A. Dingli, D. Guthrie and Y. Wilks. Mining Web Sites Using Unsupervised Adaptive Information Extraction. In Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03), Budapest, Hungary,April 2003.

[AGEST] AGESTIC, "Informe de los Clustering de España", < *www.tfinnova.es/userfiles/file/R02740_cluster%5B1%5D.pdf* > *[Consulta: 14/06/2011].*

[JBJMC] J. Burkardt, "K-Means Clustering", < *http://people.sc.fsu.edu/~jburkardt/presentations/clustering_kmeans.pdf*> *[Consulta: 17/06/2011].*

[CEACE] Proyecto CEACES, "Metodos de Analisis de Cluster", <*http://www.uv.es/ceaces/multivari/cluster/metodos.htm*> *[Consulta: 17/06/2011].*

[MCAP] M. ANDERBERG, "Cluster Analisys for applications" Ed.Academic Press,New York 1993.

[ADWAO] A. Westerski, "Sentiment Analysis: Introduction and the State of the Art overview", Universidad Politecnica de Madrid, Spain, 2010.

[OSPI] J. Horrigan, "Online Shopping", Pew Internet and American Life Project – Research Report,2008.

[ERJW] E. Riloff., J. Wiebe, "Learning Extraction Patterns for Subjective Expressions", Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP), Japan, Sapporo, 2003

[PTTD] P. Turney, "Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews", in Proceedings of the Association for Computational Linguistics (ACL), pp. 417–424,2002.

[ADKA] A. Devitt, K. Ahmad, "Sentiment Polarity Identification in Financial News: A Cohesion

Based Approach", Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 984–991, Prague, Czech Republic, 2007.

[AMERD] A. Mcintyre, "Easy RDF and SPARQL for LAMP SYSTEMS", < *http://tinman.cs.gsu.edu/~raj/ARC_RDF_Store_Presentation.pdf*> *[Consulta: 30/06/2011].*

[APPSW] A. Polleres, "Programming in Semantic WEB", <*http://axel.deri.ie/presentations/20100707_CILC_Tutorial_OWL2_RIF_SPARQL11.pdf*> *[Consulta: 30/06/2011].*

7. Glosario

AJAX **A**synchronous **J**avaScript **A**nd **X**ML

API **A**pplication **P**rogramming **I**nterface

CCS **C**ascading **S**tyle **S**heet

CMS **C**ontent **M**anagement **S**ystem

FOAF **F**riend **o**f **a** **F**riend

Gi2MO **G**eneric **I**dea and **I**nnovation **M**anagement **O**ntology

GNOME **GN**U **O**bject **M**odel **E**nvironment

HTML **H**yper**t**ext **M**arkup **L**anguage

HTTP **H**yper**t**ext **T**ransfer **P**rotocol

IDE **I**ntegrated **D**evelopment **E**nvironment

JSON **J**ava**S**cript **O**bject **N**otation

MVC **M**odel-**V**iew-**C**ontroller

PNG **P**ortable **N**etwork **G**raphics

RDF **R**esource **D**escription **F**ramework

RDFme **RDF m**apper and **e**xporter/importer

REST **Re**presentational **S**tate **T**ransfer

RESULTA **Re**d de consultoría para la gestión de procesos y relaciones

SPARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage

UML **U**nified **M**odeling **L**anguage

UPM **U**niversidad **P**olitécnica de **M**adrid

URI **U**niform **R**esource **I**dentifier

URL **U**niform **R**esource **L**ocator

W3C **W**orld **W**ide **W**eb **C**onsortium

WWW **W**orld **W**ide **W**eb

SIOC **S**emantically **I**nterlinked **O**nline **C**ommunities

KDD **Knowledge Discovery from Databases**

Contenido

1. I INTRODUCCIÓN. 5

I.I MOTIVACIÓN. 5
I.2 ESTRUCTURA DE LA MEMORIA. 7

2. CAPÍTULO II ESTADO DEL ARTE 9

2.1 INTRODUCCIÓN. 9
2.2 WEB SEMÁNTICA 10
2.2.1 INTRODUCCIÓN 10
2.2.2 RESOURCE DESCRIPTION FRAMEWORK. 13
2.3 SISTEMAS DE GESTIÓN DE IDEAS 14
2.3.1 INTRODUCCIÓN. 14
2.3.2 DRUPAL 15
2.3.3 GI2MO IDEAS STREAM. 16
2.3.4 IDEAS BRAINSTORM UBUNTU. – IDEAS TORRENT 18
2.4 MINERÍA DE DATOS. 20
2.4.1 INTRODUCCIÓN. 20
2.4.2 DESCRIPCIÓN DE ETAPAS DE MINERÍA DE DATOS. 21
2.4.3 MINERÍA WEB. 23
2.4.3.1 PROCESOS DE LA MINERÍA WEB. 24
2.5 CLUSTERING 26
2.5.1 INTRODUCCIÓN CLUSTERING. 26
2.5.2 ALGORITMOS K-MEANS. 28
2.7 ANÁLISIS DE SENTIMIENTO. 29

3. CAPÍTULO III ANÁLISIS. 32

3.1 DOMINIO DEL PROBLEMA. 32
3.2 DEFINICIÓN DE CASOS DE USO. 33
3.2.1 DICCIONARIO DE ACTORES. 34
3.2.2 CASOS DE USO BUSCADOR SEMÁNTICO GI2SE. 35
3.2.2.1 CASO DE USO I: CONFIGURAR OPCIONES DE BÚSQUEDA. 37
3.2.2.2 CASO DE USO 2: MODIFICAR OPCIONES DE BÚSQUEDA. **¡ERROR! MARCADOR NO DEFINIDO.**
3.2.2.3 CASO DE USO 3: ELIMINAR ATRIBUTOS OPCIONES DE BÚSQUEDA. 38
3.2.2.4 CASO DE USO 4: VERIFICAR ACTUALIZACIÓN DE RDF. 38
3.2.2.5 CASO DE USO 5: IMPORTAR RDF SCRAPPY. 39
3.2.2.6 CASO DE USO 6: RDFME IMPORT NAME SPACES. 40
3.2.2.7 CASO DE USO 7: LANZAR BÚSQUEDA. 41
3.2.2.8 GENERAR RDF SCRAPPY. 43
3.2.2.9 GENERAR ESTADÍSTICAS. 44
3.2.3 DIAGRAMAS DE ACTIVIDADES BUSCADOR SEMÁNTICO GI2SE. 45
3.2.3.1 ÁREA FUNCIONAL DE ADMINISTRACIÓN (ADMINISTRADOR). 46
3.2.3.1.1 REGISTRAR URL END POINT. 46
3.2.3.1.2 REGISTRAR GRAPHNAME. 47
3.2.3.1.3 REGISTRAR FUENTE DINÁMICA RDF. 47
3.2.3.1.4 VERIFICAR ACTUALIZACIÓN FUENTE RDF. 49
3.2.3.1.5 CARGAR ESPACIOS DE NOMBRE. 50
3.2.3.2 ÁREA FUNCIONAL DE BÚSQUEDA (USUARIO SISTEMA). 51
3.2.3.2.1 ESPECIFICAR NÚMERO DE CLASES Y PROPIEDADES. 51

3.2.3.2.2 Elegir Opciones de Visualización. 52
3.3 Conclusiones. 53

4. CAPÍTULO IV ARQUITECTURA. 55

4.1 Introducción. 55
4.2 Arquitectura Software – Patrones de Diseño 56
4.2.1 Patrón de Software Plugin. 56
4.3. Diseño Módulo Gi2se. 60
4.3.1 Configuración de Opciones de Búsqueda 60
4.3.1.1 Configuración RDF File. 60
4.3.1.1.1 Descripción 60
4.3.1.1.2 Diseño e Implementación. 60
4.3.1.1.3 Modelo Relacional. 62
4.3.1.1.3 Tabla Gi2se_Configuracion. 62
4.3.1.1.3 Tabla Gi2se_g2t. 63
4.3.1.1.3 Tabla Gi2se_triple. 63
4.3.1.1.3 Tablas Gi2se_o2val – Gi2se_s2val. 64
4.3.1.1.4 Algoritmo Principal. 65
4.3.1.1.5 Interfaz Gráfica. 67
4.3.1.2 Configuración SPARQL EndPoint. 68
4.3.1.2.1 Descripción 68
4.3.1.2.2 Diseño e Implementación. 69
4.3.1.2.3 Algoritmo Principal. 70
4.3.1.2.5 Interfaz Gráfica. 71
4.3.2 Actividad de Búsqueda 72
4.3.2.1 Implementación Framework ARC2. 73
4.3.2.1.1 Descripción 73
4.3.2.1.2 Diseño e Implementación. 74
4.3.2.1.2.1 Modelo de Despliegue 81
4.3.2.1.3 Algoritmo Principal. 83

4.3.2.1.3 Interfaz Gráfica. 93
4.4 Conclusiones 97

5. CAPÍTULO V CONCLUSIONES Y TRABAJOS FUTUROS. 99

5.1 Introducción 99
5.2 Conclusiones. 99
5.2.1 Gi2se Vs Exhibit. 99
5.2.2 Acerca de Drupal. 100
5.2.3 Acerca de ARC2. 101
5.2.4 Generalidades. 102
5.3 Trabajos futuros 102
5.3.1 Concepto de Web Services End Point SPARQL 102
A. Manual de instalación de Módulo de Búsqueda Semántica (Gi2se). 105
A.1 Activación del módulo 105
B. Manual de usuario de Módulo de Búsqueda Semántica (Gi2se). 107
B.1 Introducción 107
B.2 Opciones de Configuración. 107
B.3 Búsqueda de información. 108

6. REFERENCIAS 112

7. GLOSARIO 116