Universidad de las Ciencias Informáticas
Facultad 4

Desarrollo de una ontología para el proceso de pruebas de software del Grupo de Calidad del Centro FORTES.

Trabajo de Diploma en opción al título de Ingeniero en Ciencias Informáticas.

Autores:
Yisel Padrón Marín
Leonel Montero Álvarez

Tutores:
Ing. Delvis Echeverría Pérez
Ing. Odelky Betancourt González

Ciudad de La Habana, Junio 2011.
"Año 53 de la Revolución".

DECLARACIÓN DE AUTORÍA

Declaramos ser los únicos autores de este trabajo y autorizamos a la Facultad 4 de la Universidad de las Ciencias Informáticas; así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Yisel Padrón Marín	**Leonel Montero Álvarez**
_____	_____
Firma de la autora.	**Firma del autor.**
Ing. Delvis Echeverría Pérez	**Ing. Odelky Betancourt**
_____	_____
Firma del tutor.	**Firma del tutor.**

OPINIÓN DEL USUARIO DEL TRABAJO DE DIPLOMA

El Trabajo de Diploma, titulado "Desarrollo de una ontología para el proceso de pruebas de software del Grupo de Calidad del Centro FORTES", fue realizado en la Universidad de las Ciencias Informáticas. Esta entidad considera que, en correspondencia con los objetivos trazados, el trabajo realizado le satisface

□ Totalmente

□ Parcialmente en un ____ %

Los resultados de este Trabajo de Diploma le reportan a esta entidad los beneficios siguientes (cuantificar):

_____

_____

_____

_____

Como resultado de la implantación de este trabajo se reportará un efecto económico que asciende a <valor en MN o USD del efecto económico>

Y para que así conste, se firma la presente a los ____ días del mes de _____ del año _____.

_____

Representante de la entidad Cargo

_____

Firma Cuño

Agradecimientos:

A mis padres, por ser las personas que me han apoyado siempre y han confiado en mí a lo largo de toda mi carrera. Gracias por la vida.

A mi familia, por estar presentes en todo momento brindando cariño y amor. Los quiero mucho.

A mi hermana Yipsy, tu apoyo incondicional es y será mi mayor aliento para vivir. Te quiero.

A mis tutores Odelkys y Delvis, por guiarme y apoyarme en toda la realización de mi Trabajo de Diploma, ambos son personas maravillosas las cuales nunca olvidaré.

A mi novio, por estar a mi lado en todo momento, gracias por darme siempre las fuerzas para continuar.

A mi compañero de tesis Leonel, por tolerarme todo este tiempo, apoyarme en toda la carrera y siempre regalarme ese "¡Tú sí puedes!". Gracias Leo.

Gracias a todos los que de una forma u otra han contribuido en mi formación, a la Revolución por esta hermosa oportunidad y a todos mis amigos, siempre los llevaré en mi corazón.

Gracias a todos,

Yisel Padrón Marín.

A mis padres, a mi papá por ser mi paradigma, todo lo que quiero en la vida es ser como tú, a mi mamá por la sonrisa y por guiarme hasta ser hoy lo que soy, Gracias por el apoyo y la confianza, Gracias por la vida. Los quiero mucho.

A mi hermano Victor, gracias por ser mi confidente y mi mejor amigo, te quiero, Gracias por existir.

A mi familia, a mis tíos Ernesto y Carlos, por ser mis otros dos padres, Gracias por todo, a mis tías, Yamila, Gricel y Mariela, Gracias por el cariño y la confianza depositada en mí.

A mis abuelas Fela y Esperanza, gracias por cuidarme y por verme como el hombre que hoy soy. A mis abuelos, donde quiera que estén, esta tesis es para ustedes.

A mis tutores, Delvis y Odelkys, Gracias por guiarnos y apoyarnos en la realización de nuestros sueños, son personas incomparables, nunca los olvidaremos, Gracias por todo.

A mis amigos, a Wisel, por ser el otro hermano que no tengo, Gracias de verdad, a los amigos del aula, Roberto, Javy, Rey, Gerardo, Ernesto, Yoa, Pedro y el PG, Yisell, Lizi, Indira, Annia y Yami, Gracias a todos.

A mi flaca Yito, Gracias por ser más que mi compañera de tesis, gracias por soportarme todos estos años de la carrera, no habrá jamás mejor amiga y compañera de tesis que tú. Gracias por todo.

A la Revolución por permitirme hacerme un hombre de bien, íntegro y útil, a este enorme proyecto que es la UCI y a mis profesores, por permitirme ver el verdadero color de la vida.

A todos Gracias.

Leonel Montero Alvarez.

Resumen

En este trabajo se hace una propuesta para gestionar el conocimiento utilizando Ontologías. Primeramente se analizan algunas de las metodologías existentes para construir Ontologías y se propone una serie de pasos con este fin, luego se especifican los elementos necesarios para la fase de inferencia y por último se plantea cómo lograr que interactúe el usuario con el sistema. Esta propuesta se pone en práctica en el Grupo de Calidad de FORTES en donde se define como dominio de la Ontología las pruebas de software y se emplean como herramientas: el lenguaje OWL, Protégé como editor, para visualizar el contenido de la Ontología se aprovecha Jena, el razonador HermiT1.2.3.

Palabras Claves: Calidad, Pruebas de Software, Gestión del Conocimiento, Ontología, Metodología.

Índice de Contenidos

Índice de Figuras

Índice de Tablas

Introducción

En la actualidad, con el crecimiento de la tecnología, el uso y manejo de la información ha cambiado vertiginosamente con respecto a años anteriores, lo que ha permitido un crecimiento en el uso de la computación y de la informática como ciencia que maneja la información. El tratamiento de la información mediante el uso de la computación, permitió la creación de sistemas informáticos que ofrecieran la posibilidad al ser humano, de resolver disímiles problemas que de manera manual, sería imposible o demasiado complicados, al punto que las soluciones a los mismos durarían tiempo indefinido.

El surgimiento de Internet ha hecho posible que exista mucha información acerca de distintos temas y que la misma sea accesible desde cualquier parte del mundo, provocando que las personas acudan a este medio para obtener y compartir recursos. La Web actual a pesar de haber alcanzado un gran éxito, presenta algunas limitaciones, las cuales, si se pudiesen mejorar, permitirían un mayor grado de aceptación por parte de los usuarios. La existencia de mucha información en la web y la falta de organización de la misma dificulta el trabajo a la hora de realizar alguna búsqueda sobre un tema determinado, pues en ocasiones se busca algo sobre un tema y se obtiene como resultado documentos cuyo contenido no guarda relación alguna con lo que se quiere encontrar o se encuentra lo que se buscaba sólo después de visitar varias páginas.

Una posible solución a estos problemas es el uso de la Web Semántica (WS), la cual no es más que “Una visión de una futura Web en la cual la información que en la versión actual de la Web es comprensible solamente por los seres humanos también esté disponible de una manera formal para sistemas inteligentes”. [Sánchez 2005]

El uso de la Web Semántica no sería la completa solución a este problema, es necesario que la información se encuentre estructurada de manera que facilite el razonamiento automatizado de la misma. Una forma de estructurar la información es mediante Ontologías. Esta palabra viene del griego ontos “el ser” y logos “estudio de”. [Diccionario Océano]

La ontología como "el estudio metafísico de la naturaleza de ser y la existencia" es tan antigua como la disciplina de la filosofía. Recientemente, la ontología se ha definido como "la ciencia de lo que es, de los tipos y estructuras de objetos, propiedades, eventos, procesos, y relaciones en cada área de la realidad". [Smith, 2003]

Las ontologías generalmente se usan para especificar y comunicar el conocimiento del dominio de una manera genérica y son muy útiles para estructurar y definir el significado de los términos.

En el desarrollo de software a través de la Ingeniería de Software, se tienen en cuenta una serie de etapas o pasos a seguir para que la construcción de dicho sistema sea lo mayor eficiente posible, económico y fiable a la vez. Dentro de las etapas en toda la elaboración del software, el Aseguramiento de la Calidad es fundamental para velar por la correcta realización del sistema, para que haga lo que debe hacer, según la etapa en la que se esté trabajando, hasta que sea entregado al cliente.

La etapa de pruebas es una etapa crucial para el desarrollo de software, donde los sistemas o componentes de estos sistemas se ejecutan bajo una condición o requerimiento especificado. Estas contribuyen, junto a otros procesos, al aseguramiento de la calidad del software que se está haciendo, permite verificar la interacción e integración de los componentes de la aplicación, verificar que se cumplan los requisitos definidos para el sistema y que los errores detectados, sean resueltos antes que sea desplegado, esto en la menor cantidad de tiempo y con el menor esfuerzo posible.

La Industria Cubana de Software, con la Universidad de las Ciencias Informáticas (UCI) a la vanguardia del desarrollo informático, trata de mantenerse al corriente con las políticas y estándares con los que debe cumplir una aplicación informática para satisfacer las necesidades del usuario.

Dentro de la universidad existen varios centros productivos que se dedican a satisfacer las necesidades de los clientes que mantienen contratos con la empresa ALBET.SA a la cual pertenece la UCI, centrándose cada uno a diferentes ramas de la informática, como son: la bioinformática, la teleinformática, realidad virtual, software educativo, entre otros. El centro rector para las aplicaciones de corte educativo es el Centro de Tecnologías para la Formación (FORTES).

Dentro del centro se encuentra el Grupo de Calidad de FORTES (GCF), el cual es un proyecto que brinda servicios de Aseguramiento de Calidad a otros proyectos productivos del centro y fuera de este. El principal servicio que se brinda son las pruebas de software, pruebas de funcionalidad, además de auditorías y revisiones a la documentación de los proyectos.

El GCF se enfrenta a gran cantidad de información y a los artefactos necesarios para realizar los diferentes tipos de pruebas, dígase documentación, que en muchos casos, son demasiados, haciendo que el acceso a la información y a la documentación realmente relevante se convierta en un problema crítico. Se puede observar que la gestión del conocimiento y la información relacionada con los fundamentos de las pruebas de software, se limitan solamente al almacenamiento de grandes volúmenes de información, usualmente sin clasificar, desactualizada y poco estructurada, lo que hace difícil el trabajo de acceso o recuperación de la información. La falta del hábito de documentar todas las actividades que se desarrollan en el proceso de prueba, conlleva a que se pierda el conocimiento adquirido por el personal y deba invertirse nuevamente en su creación.

Las búsquedas bibliográficas son realizadas solamente a través de internet, ocurriendo errores como la Polisemia, donde los resultados que arroja la búsqueda incluye documentos con el término utilizado para realizar la búsqueda pero con disímiles significados, arrojando resultados ajenos al deseado, Sinonimia, donde la búsqueda muestra documentos en que el contenido no es el esperado, esto debido a que aparece en los documentos sinónimos del término utilizado en la búsqueda y no el término en sí, para mismos tipos y niveles de pruebas existen diferentes conceptos distanciándose unos de otros. Las lecciones aprendidas solamente queda en los especialistas involucrados en las pruebas, además se realizan análisis y no se documentan ni se almacenan.

Por lo que el **Problema de la Investigación** está enmarcado en ¿Cómo mejorar la comunicación y organización del conocimiento en el Grupo de Calidad de FORTES?

El **objeto de estudio** sería la gestión del conocimiento basado en ontologías y enmarcando como **campo de acción** la ontología en el proceso de pruebas del Grupo de Calidad de FORTES.

Idea a defender

Con la creación de una ontología basada en el procedimiento de pruebas de software para el GCF se contribuirá a mejorar la calidad del proceso de prueba a través del uso de una fuente de conocimiento común para el uso de los integrantes del GCF.

Por tanto el **objetivo general** del presente trabajo de diploma sería: Desarrollo de una ontología en apoyo al procedimiento de pruebas en el Grupo de Calidad de FORTES.

Los **objetivos específicos** son:

- Fundamentar los conceptos, técnicas, metodologías, y herramientas relacionadas con la ingeniería ontológica.
- Definir y formalizar un modelo de conocimiento para representar conocimiento ontológico.
- Desarrollo de la ontología de apoyo a los procedimientos de pruebas de software dentro del GCF.
- Validar la propuesta ontológica.

Con el objetivo de dar respuesta a los objetivos anteriores se definieron las siguientes tareas:

- Analizar los diferentes conceptos relacionados con el tema de la elaboración de ontologías.
- Realizar un estudio de las herramientas utilizadas para la confección de ontologías para un dominio específico.
- Realizar una propuesta teórica para la vinculación de mecanismos de obtención de ontologías con la gestión del conocimiento.
- Caracterizar herramientas que intervienen en la propuesta teórica para la generación de ontologías.
- Realizar un estudio sobre los aspectos teóricos generales relacionados con las pruebas de software, específicamente dentro del Grupo de Calidad de FORTES, dominio en el que será aplicada la propuesta.

Para dar cumplimiento a las tareas de investigación se utilizaron métodos teóricos:

Dentro de los métodos teóricos:

Análisis Histórico y Lógico: Se usa este método para el análisis de conceptos relacionados con las ontologías, la gestión del conocimiento y las pruebas de software.

Análisis y Síntesis: Se usa este método para definir y formalizar un modelo de conocimiento para representar conocimiento ontológico.

Modelación: Se utiliza este método para el estudio de las herramientas utilizadas para la confección del modelo de conocimiento y las ontologías.

El presente trabajo consta de introducción, tres capítulos, conclusiones generales, recomendaciones, referencias bibliográficas y bibliografía consultada durante el desarrollo del trabajo, un glosario de términos, y los anexos que complementan el cuerpo del trabajo.

Capítulo 1: Fundamentación Teórica. Se analizan los principales conceptos relacionados con el tema de investigación, así como las características fundamentales de las herramientas y lenguajes existentes para la creación de ontologías, la gestión del conocimiento y las pruebas de software. Además, se muestran varias formas de representar las ontologías y se realiza un estudio sobre los aspectos teóricos relacionados con las pruebas de software, específicamente en el Grupo de Calidad de FORTES, dominio en el que será aplicada la propuesta.

Capítulo 2: Diseño de la Propuesta. Se llega a la formulación y conceptualización de una teoría para el desarrollo de una ontología para el proceso de pruebas de software. Además, se presentan herramientas que intervienen en la propuesta teórica y se emiten valoraciones sobre ellas.

Capítulo 3: Validación de la ontología. Estará destinado a la validación del diseño de la propuesta, implementación de los métodos de validación de la ontología.

Capítulo 1: Fundamentación Teórica.

Introducción al Capítulo

El objetivo fundamental de este capítulo es presentar de manera detallada los conceptos más importantes relacionados con la Calidad de Software, Pruebas de Software, Gestión del Conocimiento y Ontologías a nivel internacional, nacional y en la universidad. En el capítulo se tratarán los principales conceptos en los que se enmarca el dominio de nuestro problema, las ontologías, su importancia, utilidad, relación con otras áreas del conocimiento como: Web Semántica, Entornos de Ingeniería de Software (*Software Engineering Enviroments* (SEE)), Ingeniería de Dominios (*Domain Engineering*), Inteligencia Artificial, tipos, características fundamentales, metodologías para su definición; adicionalmente se describen los lenguajes que se utilizan para definirlas, y algunas herramientas existentes para la definición de las mismas en el Grupo de Calidad de FORTES (GCF).

1.1 Calidad

Es necesario definir el concepto con claridad, ya que si la calidad no puede ser definida, no puede ser medida; y donde la calidad no puede ser medida entonces no puede ser controlada. Para trabajar sobre un esquema consistente y evitar ambigüedades, a continuación se ofrece la definición de calidad ofrecida por la organización ISO, definida como “Grado en el que un conjunto de rasgos distintivos inherentes cumple con las necesidades o expectativas establecidas, generalmente implícitas u obligatorias”. [ISO 9000:2005]

Para complementar la definición, dado que el concepto calidad puede ser subjetiva y debido a que las necesidades explícitas o implícitas varían de organización en organización o de usuario en usuario, es esencial identificar dichas necesidades para el usuario o para la organización.

Calidad de Software

El concepto calidad de software puede tener diferentes acepciones y miradas que dependen del usuario o receptor de un servicio o producto. Depende de numerosos factores que van desde la cultura hasta el modo de producción donde se realiza. Por eso

se hace imprescindible definirlo correctamente para lo cual se tienen que elaborar indicadores de medición.

Dentro del contexto de Ingeniería de Software, se tomará la definición de calidad de software propuesta por la organización internacional de estándares (ISO/IEC DEC 9126): La totalidad de características de un producto de software que tienen como habilidad, satisfacer necesidades explícitas o implícitas. Otra definición bastante completa de calidad de software es la que se presenta más adelante: Se puede decir que el software tiene calidad si cumple o excede las expectativas del usuario en cuanto a:

1. Funcionalidad (que sirva un propósito)

2. Ejecución (que sea práctico)

3. Confiabilidad (que haga lo que debe)

4. Disponibilidad (que funcione bajo cualquier circunstancia)

5. Apoyo, a un costo menor o igual al que el usuario está dispuesto a pagar.

Resumiendo podemos decir, que la calidad de software se refiere a: Los factores de un producto de software que contribuyen a la satisfacción completa y total de las necesidades de un usuario u organización. [1]

1.2 Pruebas de Software

La prueba del software es un elemento crítico para la garantía de la calidad del software. El objetivo de la etapa de pruebas es garantizar la calidad del producto desarrollado. Además, esta etapa implica:

- Verificar la interacción de componentes.
- Verificar la integración adecuada de los componentes.
- Verificar que todos los requisitos se han implementado correctamente.
- Identificar y asegurar que los defectos encontrados se han corregido antes de entregar el software al cliente.

Diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo. [2]

La prueba es un proceso que se enfoca sobre la lógica interna del software y las funciones externas. La prueba es un proceso de ejecución de un programa con la intención de descubrir un error [4]. Una prueba tiene éxito si descubre un error no detectado hasta entonces. Las pruebas brindan confianza en la calidad del software al encontrar defectos y a su vez la calidad del software se incrementa a medida que los defectos encontrados por las pruebas son reparados.

Según la IEEE las pruebas constituyen "Una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan o almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente." [IEEE, 1991]

Particularmente Pressman plantea que "Las pruebas del software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación." [Pressman, 2002]

Dentro de las pruebas de software, la verificación se refiere a las actividades que demuestran que el software funciona adecuadamente, ¿Se está construyendo el software correctamente? y la validación responde a las actividades que aseguran que el software se ajusta a los requisitos del cliente, ¿Se está haciendo el software correcto?

Los productos software al ser intangibles salen a la venta con alto nivel de defectos. Se estima que el 80% del costo de desarrollo se invierte en identificar y corregir defectos pues se dedican esfuerzos de hoy a arreglar lo que se hizo mal ayer.

Las pruebas nos permiten:

- Eliminar la incertidumbre sobre el avance del proyecto.
- Mejorar la calidad del producto final.
- Mejorar la rentabilidad en la producción del software.
- Reducir costos en mantenimiento.

Todos coinciden en que es mejor prevenir los errores que detectarlos y corregirlos, pues se ganaría en tiempo y recursos, sin embargo, lo cierto es que aún estamos muy lejos de alcanzar esa perfección en el trabajo.

1.3 Gestión del Conocimiento

El conocimiento está en las mentes de las personas, no siempre está disponible cuando es necesario para la organización. Para tratar este problema surge la Gestión del Conocimiento (GC) cuya principal misión es crear un ambiente en el que el conocimiento y la información disponible en una organización sean accesibles a todos, permitiendo ser usados para estimular la innovación y mejorar las decisiones. [Simón 2006]

La GC *"es el proceso sistemático de buscar, organizar, filtrar y presentar la información con el objetivo de mejorar la comprensión de las personas en un área específica de interés."* [Figueroa 2006]

Resulta importante aclarar que existen diferencias entre conocimiento, datos e información. Los datos son una representación simbólica que por sí solo no tiene un valor semántico y constituyen la base de la pirámide del conocimiento. Al conjunto de datos organizados y analizados en un contexto determinado se le denomina información, que no es lo mismo que conocimiento. Recopilar y organizar datos, es algo que puede hacer un software informático. El conocimiento es un paso adelante, es identificar, estructurar y sobre todo utilizar la información para obtener un resultado. Requiere aplicar la intuición y la sabiduría, propios de la persona a la información. La capacidad de interpretar esos datos es lo que provoca que la información se convierta en conocimiento [Figueroa 2006].

En la GC hay dos procesos fundamentales, la creación y la transmisión de conocimiento. La transmisión se puede ver cuando se intenta poner de forma explícita el conocimiento en una base de datos, lo que se hace es ponerlo allí para que al cabo del tiempo alguien pueda recogerlo, de esta forma se está transmitiendo en el tiempo. Cuando se utilizan herramientas de comunicación se intenta transmitir el conocimiento en el espacio. Algunos de estos instrumentos son las intranets, los portales, las bases de datos relacionales y las documentales. [Canals 2003]

La creación y transmisión del conocimiento, que pueden pensarse por separado también están totalmente interrelacionados, pues la creación de conocimiento no es algo que se hace partiendo de la nada, sino que para crear conocimiento se utiliza conocimiento que proviene de otras personas y de otros lugares, por lo tanto, ha habido un proceso de transmisión previo. Son procesos que están muy interrelacionados y que juntos hacen que el conocimiento dentro de las organizaciones mejore y se utilice. Otro elemento importante

que ayuda a hacer que el conocimiento funcione es el contexto, este va a permitir interpretar el conocimiento y transmitirlo. [Canals 2003]

Existen dos tipos de conocimiento: explícito y tácito, el primero se ha definido como la noción objetiva y racional que puede ser expresada con palabras, números, fórmulas, etc. El segundo es aquel que una persona, comunidad, organización o país, tiene incorporado o almacenado en su mente, en su cultura y es difícil de explicar. Es necesario explicar que este conocimiento puede estar compuesto por: ideas, experiencias, destrezas, habilidades, costumbres, valores, historias y creencias [Figueroa 2006].

Cuando estos conocimientos permiten actuar se llaman competencias o conocimiento en acción. El problema que presenta este tipo de conocimiento es su dificultad a la hora de transmitirlo, por ello es necesario gestionarlo creando códigos que faciliten su transmisión. [Figueroa 2006]

Los mapas conceptuales (MC) facilitan la organización y representación del conocimiento de forma gráfica, formando una red de conceptos, en la que los nodos representan los conceptos y los enlaces las relaciones entre ellos.

1.4 Ontología

Este concepto, cuyo significado es el de una explicación sistemática de la existencia, proviene del griego, en el que significa "*estudio del ser*". La ontología como "el estudio metafísico de la naturaleza de ser y la existencia" es tan antigua como la disciplina de la filosofía. [7] Recientemente, la ontología se ha definido como "*la ciencia de lo que es, de los tipos y estructuras de objetos, propiedades, eventos, procesos, y relaciones en cada área de la realidad.*" [Smith, 2003] No hay una definición aceptada unánimemente para el concepto de ontología, sino que cada una de las definiciones que aparecen en la literatura aporta diferentes y a la vez complementarias visiones del significado de esta palabra. Filósofos e Ingenieros de Software tienen puntos de vistas diferentes en cuanto a la definición de las ontologías.

Es decir, independientemente del ámbito en que se desarrollen, la base para una ontología es la conceptualización junto con un vocabulario para referirse a las entidades de un dominio particular. Es decir, las ontologías para representar el conocimiento

precisan los siguientes componentes [Gruber, 1993]: conceptos, relaciones, funciones, instancias y axiomas.

Mientras sigue siendo un área fecunda de investigación en el campo de la filosofía, la ontología es actualmente materia de investigación, desarrollo, y aplicación en disciplinas relacionadas con la computación, la información y el conocimiento. [7]

Las ontologías generalmente se usan para especificar y comunicar el conocimiento del dominio de una manera genérica y son muy útiles para estructurar y definir el significado de los términos. Para [Guarino, N 1998], ontología describe una cierta realidad con un vocabulario específico, por otro lado [Gruber, 1993] define una ontología como una especificación que proporciona una estructura y contenidos de forma explícita.

Apartándonos del aspecto filosófico, podemos definir a una ontología como un consenso necesario sobre cómo expresar y entender información de un determinado dominio. [Abad, M, 2004] Más profundamente una ontología o también llamada base de conocimiento (siglas en inglés KB Knowledge Base) es una especificación explícita de una o parte de una conceptualización que incluye vocabulario de términos y la especificación del significado de cada uno de ellos, define un vocabulario común para los que necesitan compartir información acerca de un dominio; haciendo una analogía al desarrollo de sistemas, es lo que una especificación es a un programa. Entendiéndose entonces conceptualización como una interpretación estructurada de una parte del mundo que usan los seres humanos para pensar y comunicarse sobre ella.

En el área de la informática, una conceptualización podría ser, por ejemplo, la clasificación de sistemas informáticos atendiendo su naturaleza física en sistemas hardware y software. Desde el punto de vista informático, las ontologías especifican un vocabulario relativo a un cierto dominio. Este vocabulario define: entidades, clases, propiedades, predicados y funciones y, la relación entre estos componentes.

Un indicador de la complejidad de una ontología es el conjunto de relaciones conceptuales. [Fox, M. Gruninger, M. 1998] sostienen que una ontología se define como un vocabulario más una especificación del significado de dicho vocabulario. Esta visión permite distinguir ontologías basadas en el grado de formalidad en la especificación del significado. Las ontologías informales usan un lenguaje natural, las ontologías semiformales proporcionan axiomatizaciones débiles tales como taxonomías y las

ontologías formales definen la semántica del vocabulario por una axiomatización completa y efectiva.

1.4.1 Tipos de ontologías

Según el grado de generalidad o nivel de dependencia a una tarea o visión en particular, las ontologías, según [Mizoguchi et al., 1999] se pueden clasificar en cuánto:

a) Al alcance de su aplicabilidad

- **Ontologías de Alto Nivel (genéricas):** describen conceptos muy generales como espacio, tiempo, materia, objeto, evento, acción, etc., los cuales son independientes de un problema o dominio en particular. Por lo tanto, parece razonable, al menos en teoría, tener ontologías unificadas de alto nivel para grandes comunidades de usuarios.

- **Ontologías de Dominio y Ontologías de Tarea:** describen, respectivamente, el vocabulario relacionado a un dominio genérico (como medicina o automóviles) o una tarea o actividad genérica (diagnóstico o venta), mediante la especialización de los términos introducidos en la ontología de alto nivel.

- **Ontologías de Aplicación:** describen conceptos que dependen tanto de un dominio como de una tarea en particular, los cuales frecuentemente son especializaciones de ambas ontologías. A menudo, estos conceptos corresponden a los roles desempeñados por entidades del dominio mientras realizan cierta actividad. Contienen conocimiento esencial para modelar una aplicación particular bajo consideración.

Otra clasificación de ontología propuesta por Van Heist [Van Heist, 1997] es la siguiente:

b) Cantidad y tipo de conceptualización

- **Terminológicas:** Especifican los términos que son usados para representar conocimiento en el universo de discurso. Son utilizadas para unificar vocabulario en un dominio determinado.

- **De información:** Especifican la estructura de almacenamiento de bases de datos. Ofrece un marco para el almacenamiento estandarizado de información (estructura de registros de la BD).
- **De modelado del conocimiento:** Especifican conceptualizaciones del conocimiento. Poseen una rica estructura interna y suelen estar ajustadas al uso particular del conocimiento que describen. [3]

1.4.2 Elementos que componen las ontologías

Las ontologías para representar el conocimiento precisan los siguientes componentes [Gruber 1993]:

- **Conceptos:** cualquier entidad sobre la que se puede decir algo, a la que se le puede poner un nombre. Además de esto también la descripción de una tarea, función, acción, estrategia, etc. Por ejemplo: coche, hombre, árbol.

- **Relaciones:** representan una interacción entre conceptos del dominio. Por ejemplo: subclase-de, parte-de, etc.

- **Funciones:** son un caso especial de las relaciones, en el que el elemento n-ésimo de la relación es único para los n-1 restantes. Por ejemplo: madre-de, precio-de, etc.

- **Axiomas:** son sentencias siempre verdaderas para dicho dominio. Por ejemplo: “el lunes va después del domingo”.

- **Instancias:** son conceptos concretos. Por ejemplo: David, Olmo.

No siempre todos estos elementos deben aparecer en una ontología. En el caso de que se tenga una ontología que posee solo conceptos y relaciones se llama taxonomía u ontología ligera. [Sheng-Tun, L. Huang Chih, S. 2003]

1.4.3 Lenguajes utilizados para la definición de Ontologías

Los primeros lenguajes para la representación de las ontologías estaban basados en representación del conocimiento, entre ellos se tienen:

1.- **CycL:** lenguaje utilizado en la ontología Cyca basado en marcos y en lógica de primer orden.

2.- **Ontolingua:** lenguaje construido sobre KIF (*Knowledge Interchange Format*- sintaxis formal utilizada para expresar conocimiento) y basado en marcos y en lógica de primer orden.

3.- **LOOM:** lenguaje basado en lógica de descripciones y reglas de producción.

4.- **OCML:** sistema de representación de conocimiento para el desarrollo de ontologías Ontolingua y métodos de resolución de problemas (PSMs – *Problem Solving Methods*) ejecutables.

5.- **OKBC (*Open Knowledge Base Connectivity*):** protocolo de acceso a bases de conocimiento construidas con diferentes sistemas de RC.
Con el auge de Internet surgen lenguajes de ontologías para la web con sintaxis basada en HTML o XML:

1.- **SHOE (*Simple HTML Ontology Extension*):** lenguaje basado en marcos y reglas diseñado como extensión del HTML.

2.- **RDFa:** modelo para la descripción de recursos Web basado en redes semánticas y con sintaxis XML.

3.- **RDF Schema:** lenguaje construido sobre RDF que incorpora primitivas de marcos (clases y propiedades).

Los lenguajes presentados anteriormente sirven tanto para la descripción de ontologías como para la publicación de contenidos web procesables por máquinas, de ahí que se les llame también lenguajes para el marcado ontológico.

Los lenguajes de ontologías más recientes son extensiones de RDF(S):

1.- **DAML+OIL:** fusión de DAML (DARPA *Agent Markup Language*) y OIL (*Ontology Inference Layer*) que amplía RDF(S) con primitivas de lógica de descripciones.

2.- **OWL (*Web Ontology Language*):** propuesta del W3C a partir de DAML+OIL para responder a las necesidades de la web semántica. Es un mecanismo para desarrollar temas o vocabularios específicos en los que asociar esos recursos. Lo que hace OWL es proporcionar un lenguaje para definir ontologías estructuradas que pueden ser utilizadas a través de diferentes sistemas. Incluye tres niveles: OWL Lite, OWL DL y OWL Full. Para los efectos de esta investigación y para soportar la continuidad de la misma, fue seleccionado este lenguaje para el desarrollo de la Ontología, ya que está diseñado para ser usado en aplicaciones que necesitan procesar el contenido de la información en lugar de únicamente representar la información para los humanos. OWL facilita un mejor mecanismo de interpretar el contenido Web que los mecanismos admitidos por XML, RDF y esquema RDF (RDF-S) proporcionando vocabulario adicional junto con una semántica formal.

A la hora de elegir un lenguaje para la definición de una Ontología deben tenerse en cuenta los siguientes aspectos:

- El lenguaje debe poseer una sintaxis bien definida para poder leer con facilidad la ontología definida.
- Debe tener una semántica bien definida para comprender perfectamente el funcionamiento de la ontología.
- Debe tener suficiente expresividad para poder capturar varias ontologías.
- Debe ser fácilmente mapearle desde/hacia otros lenguajes ontológicos.
- Debe ser eficiente a la hora de realizar razonamiento.

Existen 3 sublenguajes de OWL, los cuales van creciendo respecto al nivel de expresión:

- OWL Lite: Útil para la creación de jerarquías y restricciones simples, sólo permite valores de cardinalidad 0 y 1, pierde en expresividad. [6]

- OWL DL: (*Description Logic*), es el lenguaje más sencillo e indicado para los usuarios que requieren el máximo de expresividad y decibilidad (todos los cálculos acaban en un tiempo finito). Una clase puede ser a la vez subclase de muchas clases, no puede ser una instancia de otra clase. [6]

- OWL Full: Máximo nivel de expresión y la libertad sintáctica de RDF. Permite expresiones de segundo orden. Por ejemplo, una clase puede ser tratada simultáneamente como una colección de individuos y como un individuo por sí mismo. Es el más completo por lo que se necesita mucho poder computacional para poder hacer inferencias es por eso que se dice que no tiene garantía computacional. [6]

3.-**KIF:** *Knowledge Interchange Format* es un lenguaje para representar Ontologías basadas en la lógica de primer orden. KIF está basado en la lógica de predicados con extensiones para definir términos, metaconocimiento, conjuntos, razonamientos no monotónicos, etc.; y pretende ser capaz de representar la mayoría de los conceptos y distinciones actuales de los lenguajes más recientes de representación del conocimiento. Está diseñado para intercambiar conocimiento entre distintos sistemas de computación.

1.4.4 Metodologías para la creación de ontologías

Actualmente existe una gran cantidad de metodologías para la construcción de ontologías, pero de todas ellas las más usadas son:

Metodología Cyc: Nace como un proyecto de inteligencia artificial que busca la construcción de una ontología comprensible para habilitar el razonamiento humano.

Los pasos para la construcción de la ontología usando esta metodología son:

- Extracción manual del conocimiento común (de diversas fuentes)
- Utilización de herramientas de procesamiento de lenguaje natural o aprendizaje natural para la adquisición de nuevo conocimiento en la ontología.

El proyecto Cyc surgió en el año de 1984, por parte de la Corporación de Tecnología en Computación y Microelectrónica. La base de conocimiento de Cyc es propietaria, aunque

una pequeña versión fue liberada y está disponible como OpenCyc, la misma que busca definir un vocabulario común para el conocimiento automatizado.

Actualmente Cyc cuenta con más de un millón de aserciones en su base de conocimientos, y que han sido definidas por el humano mediante el lenguaje CycL (un lenguaje de programación parecido a Lisp).

Metodología de Uschold y King: Permite la creación de ontologías en base a otras ya existentes. Recomienda los siguientes pasos:

- Identificación del propósito para el cual se construye la ontología.
- Capturar los conceptos y las relaciones entre ellos.
- Codificación de la ontología.
- Evaluación de la ontología.
- Documentación de la ontología.

Metodología de Gruninger y Fox: En esta metodología se proponen los siguientes pasos:

1. Definición de los escenarios motivadores, es decir identificación de las posibles aplicaciones en las que la ontología será usada.
2. Formulación de preguntas en lenguaje natural, a las que se les denomina *cuestiones de competencia*, esto con la finalidad de determinar el ámbito de la ontología.
3. Especificación de la terminología, es decir en base a las preguntas realizadas en el paso anterior, se define conceptos principales, relaciones, propiedades, etc.
4. Formalización de las interrogantes.
5. Especificación de axiomas formales.
6. Verificación de la ontología.

Dentro de esta metodología las cuestiones de competencia hacen referencias a consultas a las cuales la ontología debería responder.

Metodología Kactus: Esta metodología centra la construcción de la ontología sobre una base de conocimiento, mediante un proceso de abstracción, para ello hace uso de:

- Especificación del contexto de la aplicación y el punto de vista de modelado, lo primero hace referencia a la descripción del dominio que tendrá la aplicación, así como también los objetos de interés y tareas que realizará la ontología, mientras que el punto de vista de modelado se refiere a definir qué tipo de modelado vamos a realizar: dinámico-estático, funcional-causal.
- Realizar un diseño preparatorio en base a una ontología existente, lo que implica realizar un estudio de ontologías que se hayan construido (mapeo).
- Refinamiento y estructuración de la ontología.
- Finalmente la documentación y reutilización de la ontología.

Metodología Methontology: Es una metodología creada en el Laboratorio de Inteligencia Artificial de la Universidad Técnica de Madrid. La creación de la ontología puede empezar desde cero o en base a la reutilización de otras existentes. Methontology incluye la identificación del proceso de desarrollo de la ontología (calendario, control, aseguramiento de calidad, adquisición de conocimiento), un ciclo de vida basado en la evolución de prototipos, para la cual sigue los pasos definidos en el estándar IEEE 1074 de desarrollo de software. Sus pasos principales son:

- Especificación.- Definir el alcance y granularidad de la ontología.
- Conceptualización.- Permite organizar y estructurar el conocimiento adquirido mediante tablas, lenguaje UML, jerarquías etc.
- Implementación.- Representa la formalización de la ontología, es decir pasar la conceptualización de la ontología a un lenguaje como RDF, OWL, etc.
- Evaluación.- Comprobar el funcionamiento de la ontología.

Metodología On-to-Knowledge: Es un proyecto de la IST (Tecnologías de la Sociedad de la Información), mediante este proyecto se ha desarrollado herramientas y métodos que soporten la administración de conocimiento, apoyado en una ontología compartible y usable.

Esta metodología aplica ontologías a la información electrónica con la finalidad de mejorar la administración de conocimiento. Incluye los siguientes aspectos

- Identificación de metas, las cuales deberán ser cumplidas por herramientas de gestión de conocimiento.

- Evaluación de la ontología a partir de casos de estudio [Cartuche, M.A, 2009].

1.4.5 Herramientas para definir ontologías

Entre las herramientas más destacadas utilizadas en la actualidad para la construcción de las ontologías se encuentran:

- **Protégé 2000:** herramienta a través de la cual el usuario puede construir ontologías de dominio, generar usuarios de entrada de datos y efectuar la propia entrada de datos. Es una herramienta que permite acceso a aplicaciones externas basadas en conocimiento. Además es una biblioteca a la que otras aplicaciones pueden acceder, permitiéndoles acceder a las bases de conocimiento de las cuales se dispone. Está disponible en: http://protege.stanford.edu/.

- **Ontology Server:** herramienta que permite al usuario la construcción de ontologías que comparten grupos geográficamente distribuidos. Fue desarrollado en el laboratorio de Sistemas de Conocimiento en la Universidad de Stanford. Este servidor es una extensión de Ontolingua. Al comienzo el término Ontolingua se usaba para referirse tanto al lenguaje para representar ontologías como a la herramienta utilizada para construirlas. Hoy, el término se utiliza para referirse al lenguaje proporcionado por el *Ontology Server* .Su arquitectura permite el acceso a una librería de ontologías, traductores de lenguajes y a un editor para crear y navegar por una ontología. Su servidor se encuentra disponible en la URL: http://www-ksl-svc.stanford.edu:5915

- **OntoEdit:** es un ambiente para soportar las actividades involucradas en los procesos relacionados con la Ingeniería Ontológica. Está basado en un poderoso modelo de ontologías, el cual puede ser serializado utilizando XML (*Extensible Markup Language*). Se encuentra disponible es la siguiente: URL: http://www.ontoknowledge.org/tools/ontoedit.shtml.

- **OilEd:** es un editor de ontologías gratuito que permite al usuario construir ontologías utilizando el lenguaje DAM+OIL. Aunque no posee todas las funcionalidades de otros editores de ontologías, proporciona suficiente

funcionalidad para permitir a los usuarios construir ontologías. Se encuentra disponible en la siguiente dirección URL: http://oiled.man.ac.uk/.

1.4.6 Principios para el diseño de ontologías

Para diseñar una ontología, es necesario considerar algunas de las características deseables que éstas deberían exhibir. Los principios de diseño a considerar son los siguientes:

- **Claridad y Objetividad:** Definir los conceptos en forma clara y objetiva utilizando lenguaje natural para evitar ambigüedades.
- **Coherencia:** Garantizar que todas las inferencias derivadas sean consistentes con los axiomas.
- **Completitud:** Los conceptos deben ser expresados en términos necesarios y suficientes.
- **Estandarización:** Siempre que sea posible, los nombres asignados a los términos deberán seguir un estándar, definiendo y respetando reglas para la formación de los mismos.
- **Máxima extensibilidad monótona:** Deberá ser posible incluir en la ontología especializaciones o generalizaciones, sin requerir una revisión de las definiciones existentes.
- **Principio de distinción ontológica:** Las clases de la ontología con diferente criterio de identidad, deberán ser disjuntas.
- **Diversificación de las jerarquías:** Para que la ontología se vea favorecida con los mecanismos de herencia múltiple, es conveniente usar tantos criterios de clasificación como sea posible, de manera de representar la mayor cantidad de conocimiento.
- **Minimización de la distancia semántica:** Conceptos similares deberán ser agrupados y representados utilizando las mismas primitivas.
- **Mínimo compromiso ontológico:** Una ontología debería imponer las menores exigencias posibles sobre el dominio que modela, es decir, se deben construir sólo los axiomas necesarios para representar el mundo a ser modelado.

- **Modularidad:** Al especificar una ontología se hacen definiciones de diferentes elementos como clases, relaciones y axiomas; tales definiciones se pueden agrupar en *teorías* que reúnen los objetos de una ontología más relacionados entre sí. Se puede lograr una organización altamente modular con máxima cohesión en cada módulo y mínima interacción, considerando que cada teoría es un módulo en la organización de la ontología. La modularidad permite flexibilidad y posibilidad de rehusar algunos módulos de la ontología.
- **Mínima dependencia con respecto a la codificación:** Una ontología debería permitir que los agentes que compartan los conocimientos, puedan ser implementados en diferentes sistemas y estilos de representación.

Un diseño ontológico ideal debería cumplir con todos estos criterios, pero no siempre es posible. [Ramos, E y Nuñez, H. 2007]

1.4.7 Utilidad de las Ontologías

Se mencionan a continuación el uso que, en la actualidad, tienen las ontologías:

- Sirven para entender como diferentes sistemas comparten información.
- Se utilizan para descubrir distorsiones que puedan presentarse en los procesos cognitivos de aprendizaje en un mismo contexto.
- Sirven para formar patrones para el desarrollo de Sistemas de Información. En el ámbito del software se viene utilizando hace algunos años para describir las propiedades del software (componentes, arquitecturas, lenguajes de definición).
- Permiten compartir y reutilizar conocimiento común.
- Ayudan a establecer comunicación entre personas y organizaciones con el fin de unificar diferentes áreas de investigación.
- Permiten la interoperabilidad entre sistemas de software usando ontologías como un lenguaje intermedio para unificar diferentes lenguajes y herramientas.
- Aumentan los beneficios de la Ingeniería de Sistemas ya que el uso de ontologías facilita la construcción de software clásico o basado en el conocimiento porque permite que los sistemas se puedan reutilizar. [Abad, M, 2004]

1.4.8 Importancia de las Ontologías

La importancia de las ontologías se puede resumir en dos aspectos fundamentales:

- El análisis ontológico clarifica la estructura del conocimiento. Dado un dominio, su ontología constituye el corazón de cualquier representación de un sistema de conocimiento para ese dominio. Sin ontologías o sin las respectivas conceptualizaciones que soportan el conocimiento no puede existir un vocabulario para representar el conocimiento. Así el primer paso para obtener un efectivo sistema de representación del conocimiento y su vocabulario es llevar a cabo un efectivo análisis ontológico del campo o dominio de estudio, un análisis pobre conlleva a bases de conocimiento incoherentes.

- Las ontologías permiten compartir conocimiento. Supongamos que se realiza un análisis y se obtiene un conjunto satisfactorio de conceptualizaciones y la representación de sus términos para un área en específica, por ejemplo el dominio de los componentes electrónicos, la ontología resultante debe incluir términos específicos del dominio como: transistores y diodos; términos generales como funciones y modos; y términos que describen el comportamiento como, voltajes. Para construir un lenguaje de representación del conocimiento basado en el análisis previo, se necesita asociar los términos con los conceptos y las relaciones, y definir una sintaxis de codificación del conocimiento en término de los mismos. Es posible compartir un lenguaje de representación de conocimiento con otros que tengan similares necesidades para el mismo dominio, de tal modo que se elimine la necesidad de replicar el proceso de análisis del conocimiento. Compartir ontologías puede de este modo significar la base de la definición de un lenguaje de representación del conocimiento. [Abad, M, 2004]

1.4.9 Relación con otras áreas del conocimiento

Las ontologías se relacionan y son aplicables en la actualidad a diversas áreas del conocimiento, en las cuales contribuyen a la realización de una actividad más efectiva, se seleccionaron algunas áreas consideradas importantes para describir la forma en que las ontologías trabajan en conjunto con las mismas:

- **Web Semántica:** la Web Semántica es una Web extendida, dotada de mayor significado en la cual los usuarios de Internet pueden encontrar respuestas a sus preguntas de manera más rápida y sencilla gracias a una información bien definida.[5] Esto permite compartir, procesar y trasmitir información de forma sencilla. Cuando se habla de que es una Web dotada de significado, quiere decir, de más semántica, lo cual le permite procesar sus contenidos, razonar con este, combinarlo y realizar deducciones lógicas para resolver problemas cotidianos automáticamente. Es una infraestructura basada en metadatos los cuales permiten razonar en la Web extendiendo de esta forma su capacidad, proporcionando a las máquinas la capacidad de resolver problemas bien definidos, a través de operaciones bien definidas que se llevan a cabo sobre datos bien definidos. La manera como se relaciona este nuevo enfoque de la Web con las ontologías es que estas últimas son utilizadas como estándar de la Web Semántica para obtener una adecuada definición de los datos a través del lenguaje OWL (*Ontology Web Language*), el cual permite definir ontologías estructuradas que pueden ser utilizadas a través de diferentes sistemas, usuarios, bases de datos y aplicaciones que necesitan compartir información específica.

- **Ingeniería de Dominio (*Domain Engineering*)**: un dominio es un área funcional diferenciable con requisitos y rasgos similares (*feature*s) que puede ser soportada por algún tipo de sistema de software. La Ingeniería de Dominios se encarga de capturar, organizar y representar la información útil para el desarrollo de sistemas de software, de manera que esta pueda ser reutilizada para crear nuevos sistemas. La idea principal es lograr desarrollar un conjunto de productos de software que comparten información y tienen aspectos comunes entre sí, pudiendo de esta forma ser reutilizados por los ingenieros de software. Su propósito es identificar, modelar, construir y catalogar un conjunto de productos de software que pueden ser aplicados a existentes o futuros sistemas en un dominio de aplicaciones particular. Un proceso de Ingeniería de Dominios debe contemplar por lo menos tres actividades principales: análisis de dominio, especificación de la infraestructura e implementación de la infraestructura. En la Ingeniería de Dominio, las ontologías pueden ejecutar distintos roles, en esta área el interés principal se basa en el uso de las ontologías como una especificación, en la cual la ontología de un dominio proporcione un vocabulario para requerimientos específicos de una o más aplicaciones. [Laguna, 2002]

- **Software Engineering Enviroments (SEEs):** representan colecciones integradas de herramientas que facilitan las actividades de la ingeniería de software a través de todo su ciclo de vida. En este contexto las ontologías son utilizadas para mejorar el proceso de integración de los SEEs, ya que permiten conceptualizar un dominio a través de la definición de clases de objetos y sus relaciones. El enfoque que se sigue para el caso de los SEEs se centra principalmente en que las herramientas que se desean integrar estén basadas en ontologías, esto con el objetivo de facilitar este proceso. [Falbo, R. Guizzardi, G. Duarte K]

- **Gestión del Conocimiento (*Knowledge Management*):** es un concepto utilizado para la transmisión de conocimiento y experiencia entre los empleados dentro de una organización. Usualmente este proceso requiere capturar, organizar y almacenar el conocimiento para transformarlo en un activo intelectual que se comparte. La actividad de la gestión de conocimiento es amplia y compleja, esta puede ser manejada como un conocimiento individual o como un conocimiento empresarial. Para lograr estos objetivos, las ontologías son consideradas una metodología adecuada para soportar la variedad de actividades de la gestión del conocimiento, como la recuperación del conocimiento, su almacenamiento y distribución. Las ontologías pueden ser vistas como la clasificación de conocimiento, definiendo el vocabulario que facilita la comunicación del conocimiento, almacenamiento, búsqueda y distribución de los sistemas basados en gestión del conocimiento.

- **Inteligencia Artificial (IA):** es una de las áreas de la ciencia de la computación encargadas de que la creación de hardware y software tenga comportamientos inteligentes. Una de las principales áreas de estudio de la Inteligencia Artificial está representada por los Agentes Inteligentes, los cuales son entidades de software con una arquitectura robusta que puede funcionar en distintos entornos o plataformas de computación y que son capaces de realizar de forma “inteligente” y autónoma distintos objetivos, intercambiando información con el entorno, o con otros agentes humanos o computacionales. En esta área, las definiciones de las ontologías juegan un papel importante ya que permiten establecer un consenso de comunicación sobre cómo se expresa y entiende la información de un determinado dominio en el cual se desenvuelven los agentes de software. [Sheng-Tun, L. Huang Chih, S. 2003]

1.4.10 Razonadores de ontologías

Una de las herramientas más utilizadas para trabajar con las Ontologías son los razonadores, que sirven para realizar inferencia, a través de los conceptos y en algunos casos las instancias, obteniendo nuevo conocimiento. Generalmente difieren en el lenguaje formal en el que se especifica el conocimiento, así como los lenguajes de consulta que puedan utilizar. En la actualidad, son varios los razonadores o sistemas deductivos basados en lógica descriptiva que permiten el razonamiento y la inferencia en las Ontologías. Los principales son:

- **FaCT** (*Fast Classification of Terminologies*): desarrollado por Ian Horrocks, que puede ser usado para chequear el grado de satisfacción de las descripciones. Permite reglas transitivas, inversas, restricciones cualificadas, jerarquías etc. Es lo suficientemente expresivo para soportar y razonar sobre cualquier base de conocimiento. Escrito en Common Lisp, fácilmente ejecutable por cualquier programa Lisp de forma local, tiene escrita una versión servidor FaCT para ser usada vía interfaz CORBA sobre cualquier sistema con acceso a la red. Actualmente es el razonador empleado por defecto en el editor de Ontologías OilEd para clasificar los conceptos en una jerarquía según las descripciones que tengan. [Samper 2005]

- **JTP** (*Java Theorem Prover*): es una arquitectura para razonar sobre conocimiento descrito en DAML + OIL. Escrito en Java, soporta el modelo de axiomas de DAML + OIL, realiza una pre computación cuando se carga la Ontología, y ha añadido un clasificador para realizar la jerarquía de conceptos de la Ontología. Por el momento, es el único razonador que permite el lenguaje de consulta DQL específico para DAML + OIL. [Samper 2005]

- **HermiT:** es un razonador de ontologías, escrito utilizando el Lenguaje de Ontologías Web (OWL). Dado un archivo OWL, HermiT puede determinar si la ontología es consistente, identificar las relaciones de subsunción entre las clases, y mucho más. HermiT es el primer razonador de OWL a disposición del público basado en "hypertableau", cálculo que proporciona razonamiento mucho más eficiente que cualquier algoritmo previamente conocido.

- **Pellet**: es un razonador de OWL-DL basado en Java. Puede ser utilizado conjuntamente con bibliotecas del API de Jena o del OWL. Mediante su uso es posible validar, comprobar la consistencia de Ontologías, clasificar la taxonomía y contestar a un subconjunto de consultas RDQL (conocido como consultas a ABox en terminología del DL). Se trata de una razonador DL basado en los algoritmos tableaux desarrollados para DL expresiva. Soporta todas las construcciones del OWL DL incluyendo las relacionadas con los nominales, es decir, owl: oneOf y owl: hasValue. [Samper 2005]

1.4.11 Sistemas de almacenamiento

Los sistemas de almacenamiento posibilitan mantener las Ontologías en bases de datos e ir añadiendo nueva información, y con la ayuda de razonadores probar la consistencia de la Ontología. Aunque existen varios sistemas entre ellos el más utilizado es Jena.

- **Jena:** es un framework, de código libre programado en Java. Su fin es manipular metadatos desde aplicaciones escritas en Java. Proporciona una API para extraer y escribir datos de un grafo RDF. Los modelos pueden ser consultados mediante SPARQL. Se puede utilizar OWL con Jena. Proporciona varios razonadores internos y se pueden añadir otros mediante una interfaz para Descripción Lógica (DIG).

- **Sesame:** es un repositorio para RDF-Schema que permite añadir y eliminar información, para ser almacenada en cualquier tipo de base de datos (MySQL, Oracle etc.). Soporta los lenguajes de consulta RQL, RDQL y SeRQL, para acceder al conocimiento.

- **KAON Tool:** Implementa una interfaz independiente del sistema en el que se almacenarán las Ontologías, ya sean cualquier base de datos o un fichero texto. Implementa un API para leer las descripciones de los recursos, emplea RQL para realizar consultas y soporta tanto Ontologías DAML + OIL como RDF.

1.4.12 Ontologías existentes

En la actualidad hay muchas ontologías, enfocadas a la geografía y a la medicina la mayoría, aun así no existe una ontología definida para los procesos de pruebas de software, y menos alguna que se ajuste a los procesos que están definidos para el Grupo de Calidad de FORTES.

1.5 Conclusiones del capítulo

En este capítulo se expusieron los principales conceptos relacionados con esta investigación. Fueron estudiados los aspectos relacionados con las Ontologías, gestión del conocimiento, Pruebas de Software y Calidad de Software. Se mostraron las principales herramientas y lenguajes utilizados para construir y representar Ontologías.

- ✓ Para la realización de este trabajo se ha decidido utilizar las siguientes tecnologías:

- **OWL** debido a que está diseñado para usarse cuando la información contenida en los documentos necesita ser procesada por programas o aplicaciones, permite definir clases mediante condiciones sobre sus miembros, mediante combinación booleana de clases o por enumeración de las instancias que pertenecen a la clase. Además OWL permite atribuir ciertas propiedades a las relaciones, como cardinalidad, simetría, transitividad, o relaciones inversas.

- **Protégé** herramienta a través de la cual el usuario puede construir ontologías de dominio, generar usuarios de entrada de datos y efectuar la propia entrada de datos. Es una herramienta que permite acceso a aplicaciones externas basadas en conocimiento. Además es una biblioteca a la que otras aplicaciones pueden acceder, permitiéndoles acceder a las bases de conocimiento de las cuales se dispone. Está disponible en: http://protege.stanford.edu/

- **Methontology** es una metodología para construir ontologías tanto partiendo desde cero o a través de un proceso de reingeniería. Este entorno permite la construcción de ontologías a nivel de conocimiento, e incluye: (1) identificación del proceso de desarrollo de la ontología donde se incluyen las principales actividades (evaluación, gestión, de configuración, conceptualización, integración,

implementación). Esta metodología ha sido usada en la construcción de múltiples Ontologías como: negocio, comercio electrónico, e-learning, ingeniería y en la medicina.

- **Jena** para visualizar el contenido de las Ontologías, permite trabajar con OWL, está dentro de la línea de Java, es gratuito y de código abierto.

- **NetBeans 6.9** por ser este un IDE multiplataforma, multilenguaje y de código abierto. Utilizado para el trabajo con la ontología exportada desde el modelo ontológico al modelo relacional.

- **HermiT** como razonador para validar sintácticamente la Ontología, es un plugin de Java y se puede trabajar utilizando OWL y Jena.

- **Visual Paradigm 6.4** se utilizó para el modelado de diagramas de clases de la ontología.

La creación de una ontología es un paso fundamental para lograr una correcta gestión del conocimiento la cual permite además, mantener organizada la información que se va a tratar, de manera estructurada, actualizada y clasificada. El uso de una buena ontología, le permitiría a los buscadores semánticos, y aplicaciones de corte semántico, manejar y presentar la información de una manera más rápida, eficiente y asequible a los usuarios.

Capítulo 2: Diseño de la Propuesta

Introducción al Capítulo

La utilidad fundamental de una ontología como se mencionó en el capítulo anterior, radica en el hecho de clasificar conceptos y permitir compartir conocimiento en un área de un dominio específico. Se presenta a continuación el proceso de creación de una ontología basado en la metodología Methontology; cuyo objetivo primordial es compartir y organizar conocimiento recopilados en este trabajo, así como también servir de base para el inicio de investigaciones relacionadas con la interpretación automática de estos conceptos, usando sistemas de software o agentes de software inteligentes.

2.1 Descripción de la Propuesta

La propuesta ontológica a diseñar, pretende, facilitar los procesos de pruebas para el Grupo de Calidad de FORTES. Se describen en ella, los principales conceptos, acepciones y temas en general que se tratan para el procedimiento de pruebas de software diseñado en el GCF. Se diseñará en esta propuesta, relaciones entre los diferentes conceptos planteados, facilitando al lector y a los sistemas donde se aplicará la ontología a diseñar, una manera más entendible y computable de los datos del dominio en cuestión. Permitirá además, estructurar, organizar y actualizar la información y el conocimiento a manejar dentro del GCF.

2.2 Metodología Methontology

Es una metodología creada en el Laboratorio de Inteligencia Artificial de la Universidad Técnica de Madrid. La creación de la ontología puede empezar desde cero o en base a la reutilización de otras existentes. Methontology propone un ciclo de vida basado en la evolución de prototipos que permite añadir, cambiar y eliminar términos en cada nueva versión (prototipo) de la ontología, para la cual sigue los pasos definidos en el estándar IEEE 1074 de desarrollo de software. [Ramos, Esmeralda. 2007].

El ciclo de vida de Methontology se muestra en la **Figura. 1**. Las actividades de control, aseguramiento de calidad, adquisición de conocimiento, integración, evaluación documentación y manejo de configuración se realizan simultáneamente con las actividades de desarrollo. La conceptualización debe ser evaluada cuidadosamente para

evitar la propagación de errores a las siguientes etapas del ciclo de vida de la ontología [Fernández, 1999].

La planificación se realiza antes del desarrollo de la ontología, por lo tanto no forma parte de su ciclo de vida. Las actividades de adquisición de conocimiento, integración y evaluación requieren un mayor esfuerzo en la etapa de conceptualización.

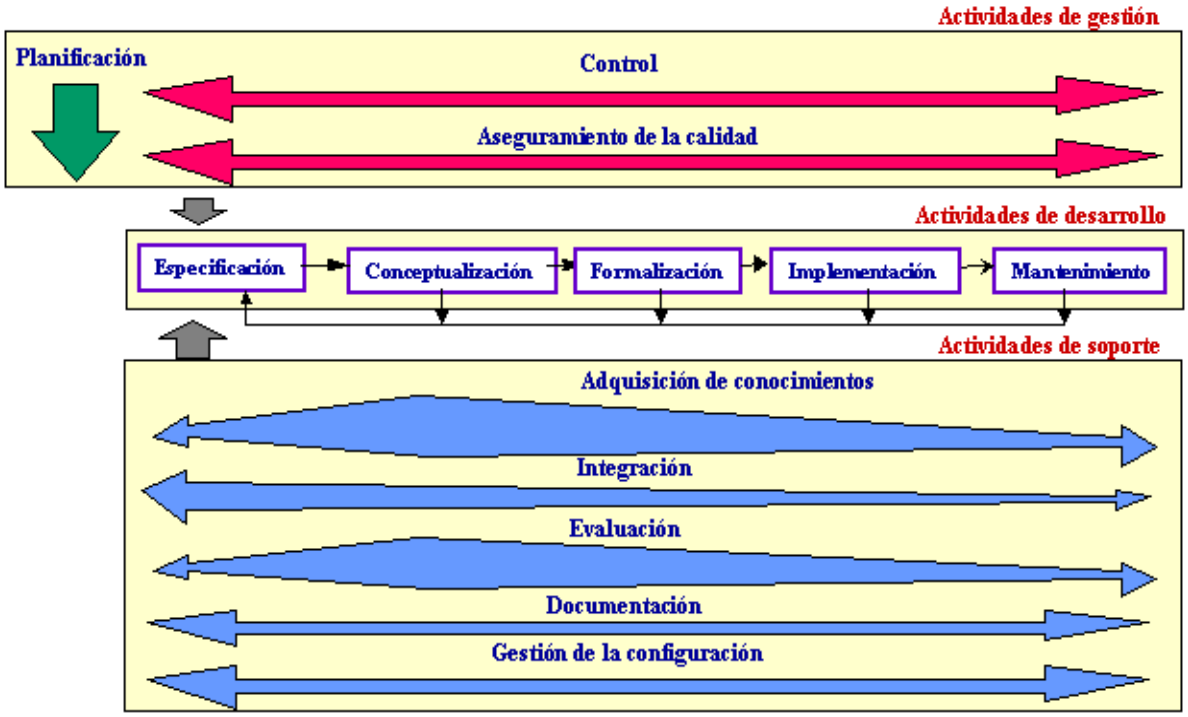


Figura. 1: Actividades de desarrollo de ontologías propuestas por METHONTOLOGY.

Las actividades de desarrollo identificadas para Methontology y por las que se realizará la ontología son las siguientes:

- ✓ La actividad de **especificación** permite determinar por qué se construye la ontología, cuál será su uso, y quiénes serán sus usuarios finales.

- ✓ La actividad de **conceptualización** se encarga de organizar y convertir una percepción informal del dominio en una especificación semi-formal, para lo cual utiliza un conjunto de representaciones intermedias (pueden ser: tablas, diagramas, gráficas) que pueden ser fácilmente comprendidas por los expertos de dominio y los desarrolladores de ontologías. También existen componentes de modelado de ontologías que deben describirse detalladamente:

- **Tarea 1: Construir el glosario de términos.** El glosario de términos debe incluir todos los términos relevantes del dominio (conceptos, instancias, atributos, relaciones entre conceptos, etc.)
- **Tarea 2: Construir la taxonomía de conceptos.** Cuando el glosario de términos tenga una cantidad importante de elementos, se debe construir una taxonomía que defina la jerarquía entre los conceptos. Se debe evaluar que la taxonomía creada no contenga errores.
- **Tarea 3: Construir un diagrama de relaciones binarias.** El objetivo de este diagrama es establecer las relaciones entre los conceptos de una o más taxonomías de conceptos. Se debe evaluar que el diagrama creado no contenga errores.
- **Tarea 4: Construir el diccionario de conceptos.** El diccionario de conceptos contiene los conceptos del dominio, sus relaciones, instancias, atributos de clases y atributos de instancias. Las relaciones, atributos de instancias, y atributos de clases son locales al concepto, lo que significa que sus nombres pueden repetirse en diferentes conceptos.
- **Tarea 5: Definir las relaciones binarias en detalle.** Se crea la tabla de relaciones binarias en la que se describe detalladamente todas las relaciones binarias incluidas en el diccionario de conceptos. Para cada relación binaria se debe especificar: nombre, conceptos fuente y destino, cardinalidad y relación inversa.
- **Tarea 6: Definir axiomas formales.** Son expresiones lógicas siempre verdaderas que suelen utilizarse para definir restricciones en la ontología.
- **Tarea 7: Definir reglas.** Se utilizan normalmente para inferir conocimientos en la ontología, tales como valores de atributos, instancias de relaciones, etc.

✓ La actividad de **formalización** se encarga de la transformación del modelo conceptual generado en la actividad anterior en un modelo formal o semi-computable.

✓ La actividad de **implementación** construye modelos computables en un lenguaje de ontologías (Ontolingua, RDF Schema, OWL, etc.).

- ✓ La actividad de **mantenimiento** permite la actualización y corrección de la ontología.

2.3 Especificación

Se pretende construir una ontología con el objetivo de proveer una mejor estructura para la base de conocimientos existente en el Grupo de Calidad de FORTES. Esta permitirá además, organizar, estructurar, jerarquizar y actualizar la información existente en el GCF de una manera comprensible, tanto para los usuarios, como para las aplicaciones semánticas donde en un futuro pueda ser usada. Esta ontología será aplicada en una futura aplicación semántica a realizar dentro del grupo, y los usuarios finales serán los propios integrantes del Grupo de Calidad de FORTES.

2.4 Conceptualización

En la **Figura. 2** se muestra el Diagrama de Clases el cual está confeccionado con la herramienta Visual Paradigm UML 6.4 Enterprise Edition, en el mismo se presentan todas las clases y relaciones que se utilizarán para diseñar la ontología.

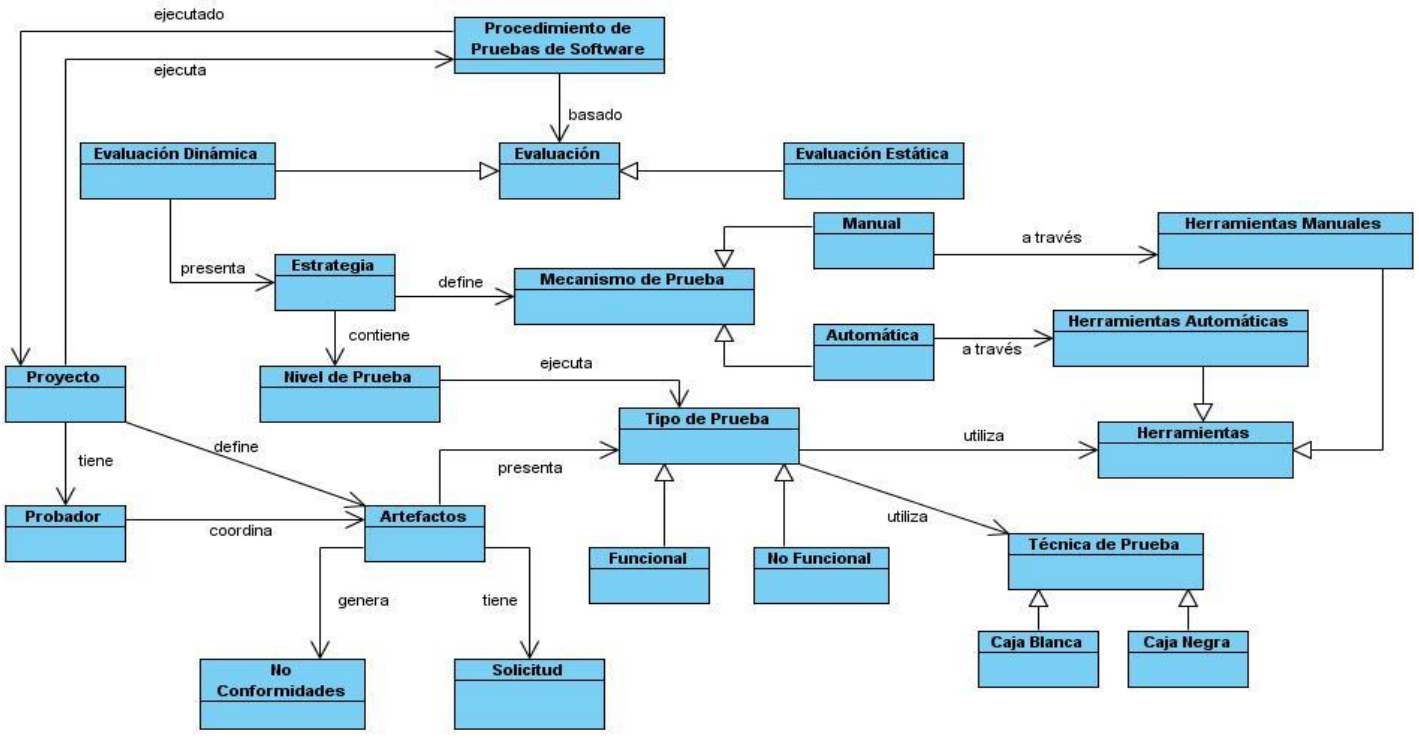


Figura. 2: Diagrama de Clases donde se muestran las clases y relaciones de la ontología Procedimiento de Pruebas de Software.

El **Visual Paradigm** para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

- ✓ **Principales componentes para el modelado de ontologías**

Las tareas mencionadas anteriormente se describen detalladamente para la ontología Procedimiento de Pruebas de Software de la siguiente manera.

2.4.1: Construir el glosario de términos

En el trabajo el glosario de términos debe incluir todos los términos relevantes del dominio (conceptos, instancias, relaciones entre conceptos).

Nro.	Término	Descripción	Tipo
1	Evaluación	El procedimiento de prueba de software está basado en una Evaluación.	Concepto
2	Evaluación Dinámica	Es una instancia del concepto Evaluación.	Instancia
3	Evaluación Estática	Es una instancia del concepto Evaluación.	Instancia
4	Estrategia	La instancia Evaluación Dinámica presenta una Estrategia	Concepto
5	presenta	Es la relación que existe entre la instancia Evaluación Dinámica y el concepto Estrategia, además la relación del concepto Artefactos y Tipo de Prueba.	Relación
6	Mecanismo de Prueba	Lo define una Estrategia y presenta dos instancias, Mecanismos de Prueba Manual y Automático.	Concepto
7	Manual	Es una instancia de Mecanismo de Prueba que se realizan a través de la instancia Herramientas Manuales.	Instancia
8	Automático	Es una instancia de Mecanismo de Prueba que se realizan a través de la instancia Herramientas Automáticas.	Instancia
9	define	Es la relación que se establece entre	Relación

		el concepto Estrategia y el concepto Mecanismo de Prueba.	
10	Nivel de Prueba	El concepto Estrategia contiene un Nivel de Prueba.	Concepto
11	contiene	Es la relación que existe entre el concepto Estrategia y el concepto Nivel de Prueba.	Relación
12	Tipo de Prueba	El concepto Nivel de Prueba ejecuta un Tipo de Prueba las cuales presentan dos instancias, pueden ser Funcionales y No Funcionales.	Concepto
13	ejecuta	Es la relación que se establece entre el concepto Nivel de Prueba y Tipo de Prueba, además establece relación entre Proyecto y un procedimiento de prueba de software.	Relación
14	Funcional	Es una instancia del concepto Tipo de Prueba.	Instancia
15	No Funcional	Es una instancia del concepto Tipo de Prueba.	Instancia
16	Herramientas	Los Tipos de Prueba utilizan Herramientas, las mismas pueden ser Herramientas Manuales y Automáticas.	Concepto
17	Herramientas Manuales	Es una instancia del concepto Herramientas.	Instancia
18	Herramientas Automáticas	Es una instancia del concepto Herramientas.	Instancia
19	utilizan	Es la relación que existe entre el concepto Tipo de Prueba con los conceptos Herramientas y Técnica de Prueba.	Relación
20	Técnica de Prueba	El concepto Tipo de Prueba utiliza Técnicas de Prueba y la misma contiene tres instancias, pueden ser, Caja Blanca y Caja Negra.	Concepto
21	Caja Blanca	Es una instancia del concepto Técnica de Prueba.	Instancia
22	Caja Negra	Es una instancia del concepto Técnica de Prueba.	Instancia
23	Proyecto	El concepto Proyecto ejecuta un procedimiento de prueba de software y tiene Probadores.	Concepto
24	Probador	El concepto Probador coordina los Artefactos.	Concepto
25	Artefactos	El concepto Artefactos genera No	Concepto

		Conformidades y tiene Solicitud.	
26	No Conformidades	Es generado por los Artefactos.	Concepto
27	Solicitud	Todo Artefacto tiene una Solicitud.	Concepto
28	tiene	Es la relación que existe entre el concepto Proyecto y Probador, además en Artefactos y Solicitud.	Relación
29	coordina	Es la relación entre el concepto Probador y Artefactos.	Relación
30	genera	Es la relación entre el concepto Artefactos y No Conformidades.	Relación

Tabla 1: Glosario de Términos.

2.4.2 Construir taxonomía de conceptos

Una vez construido el glosario de términos se procede a construir la taxonomía de conceptos que define su jerarquía. Primeramente, las taxonomías representan a través de las clases y las instancias propias de esas clases una realidad contextualizada que permitirá la organización y recuperación efectiva de los conceptos representados pertenecientes a un determinado dominio. Para construir la taxonomía de conceptos, se seleccionan del glosario de términos aquellos términos que son conceptos, tener en cuenta que un concepto puede ser subclase de más de un concepto en la taxonomía. Por ejemplo *Artefacto* es subclase de los conceptos *No Conformidades* y *Solicitud*. En la **Figura. 3** se muestra cómo quedaría esta taxonomía.

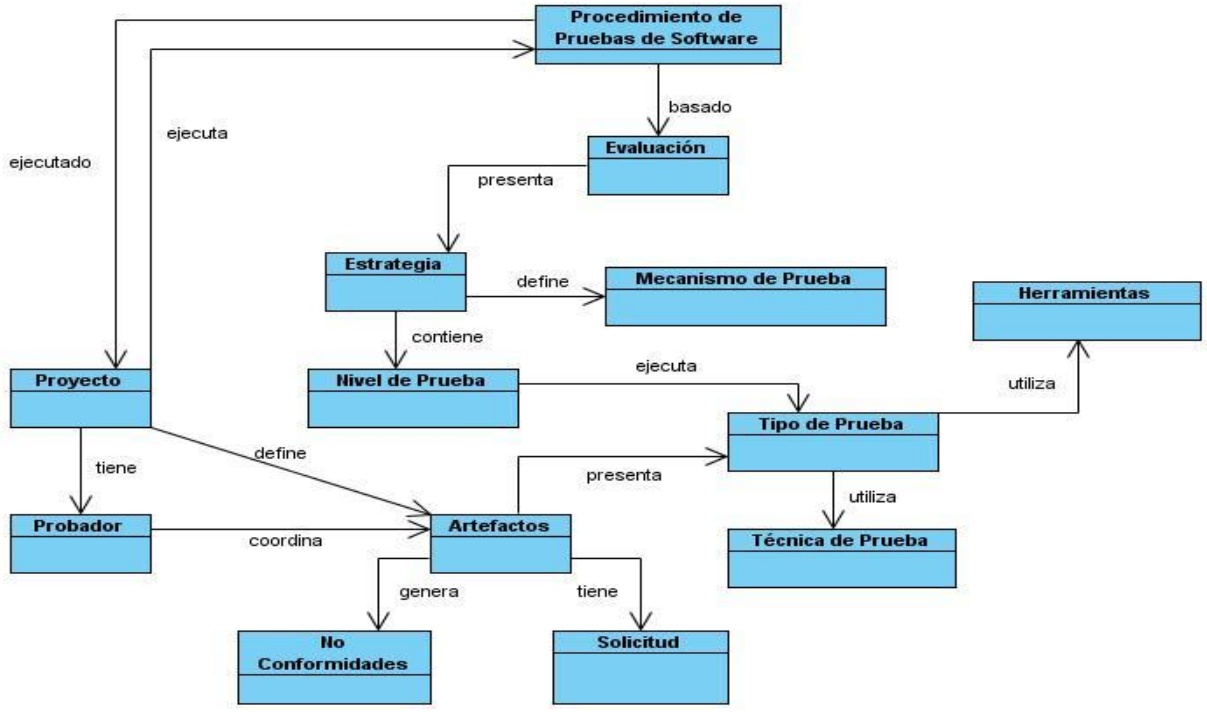


Figura. 3: Taxonomía de Conceptos de la ontología Procedimiento de Pruebas de Software.

2.4.3 Construir diagrama de relaciones binarias

Una vez construida y evaluada la taxonomía, la actividad de conceptualización propone construir diagramas de relaciones binarias. El objetivo de este diagrama es establecer las relaciones existentes entre conceptos de la misma o de distintas taxonomías de conceptos. En el diagrama de clases mostrado anteriormente todas las relaciones que existen son relaciones binarias pero en la **Figura. 4** se muestra la relación binaria inversa de uno de los conceptos del diagrama, *Procedimiento de Pruebas de Software* y *Proyecto*.

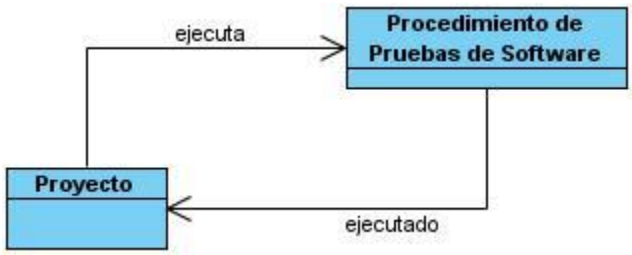


Figura. 4: Diagrama de Relaciones Binarias de la ontología Procedimiento de Pruebas de Software.

2.4.4 Construir el diccionario de conceptos

Una vez que las taxonomías de conceptos y los diagramas de relaciones binarias han sido creados, deben especificarse cuáles son las propiedades que describen cada concepto de la taxonomía, así como las relaciones identificadas en el diagrama anterior y las instancias de cada uno de los conceptos. El diccionario de conceptos contiene todos los conceptos del dominio, sus relaciones, sus instancias, y sus atributos de clase y de instancia. Las relaciones especificadas para cada concepto son aquellas en las que el concepto es el origen de la misma.

Nombre del concepto	Instancia	Atributos de clase	Atributos de instancia	Relaciones
Evaluación	Evaluación Dinámica y Evaluación Estática		Tipo	presenta
Estrategia			Tipo	define contiene

Nivel de Prueba			Nombre Tipo de Nivel	ejecuta
Mecanismo de Prueba	Manual y Automática			---
Tipo de Prueba	Funcional y No Funcional	Función Seguridad Volumen Usabilidad Fiabilidad Rendimiento Soportabilidad	Nombre	utiliza
Herramientas	Herramientas Manuales y Automáticas	JMeter Selenium IDE	Nombre Tipo	---
Técnica de Prueba	Caja Blanca y Caja Negra		Tipo	---
Proyecto		RHODA CRODA MOODLE	Nombre	ejecuta tiene
Probador		Probador 1 Probador 2 Probador 3	Nombre Año	coordina
Artefactos		Listas de Chequeo Casos de Prueba.	Nombre	presenta tiene genera
No Conformidades		Aplicación Documentación Nivel de Importancia.	Nombre	---
Solicitud		Plan de Pruebas Plantilla de No Conformidades Plantilla de Liberación	Fecha	---

Tabla 2: Diccionario de Conceptos de la ontología Procedimiento de Pruebas de Software.

2.4.5 Definir las relaciones binarias

El objetivo de esta tarea es describir en detalle todas las relaciones binarias identificadas en el diagrama de relaciones binarias e incluidas en el diccionario de conceptos. Para cada relación binaria, se debe especificar su nombre, los nombres de sus conceptos origen y destino, su cardinalidad y su relación inversa, si existe.

En la **Tabla. 3** se definen las relaciones binarias de la ontología Procedimiento de Pruebas de Software.

Nombre de la Relación	Concepto Origen	Cardinalidad	Concepto Destino	Relación Inversa
ejecuta	Proyecto	Min	Procedimiento de Prueba de Software	es ejecutado
ejecutado	Procedimiento de Prueba de Software	Min	Proyecto	se ejecuta

Tabla 3: Relaciones Binarias de la ontología Procedimiento de Pruebas de Software.

2.4.6 Definir axiomas formales

Methontology propone especificar la siguiente información para cada axioma formal: nombre, descripción en lenguaje natural, expresión que define de manera formal el axioma, los conceptos y las relaciones utilizadas en el axioma.

Nombre del axioma	Descripción	Expresión	Conceptos	Relaciones
axioma Tipo de Prueba	un Tipo de Prueba utiliza solamente una Técnica de Prueba y una Herramienta	isUsedBy only (Tecnica_de_Prueba and Herramienta)	Tipo de Prueba Técnica de prueba Herramienta	utiliza

Tabla 4: Ejemplo de axiomas formales de la ontología Procedimiento de Pruebas de Software.

En la **Tabla. 4** se muestra un axioma formal de la ontología Procedimiento de Pruebas de Software, que establece que; un *Tipo de Prueba* utiliza solamente una *Técnica de Prueba*

y una *Herramienta*, en la columna de la expresión se muestra la sintaxis utilizada en el Protégé. Las columnas que corresponden a conceptos y relaciones referidas contienen los conceptos y relaciones utilizados en la expresión formal del axioma.

2.4.7 Definir reglas

De manera similar a la tarea previa, en esta tarea el desarrollador de la ontología debe identificar en primer lugar qué reglas se necesitan en la ontología, y entonces describirlas en la tabla de reglas. Para cada regla, METHONTOLOGY propone incluir la siguiente información: nombre, descripción en lenguaje natural, expresión que describe formalmente la regla, conceptos, y relaciones utilizados en la regla, así como las variables usadas.

Nombre de la regla	Descripción	Expresión	Conceptos	Relaciones
regla Tipo de Prueba	un Tipo de Prueba utiliza alguna Técnica de Prueba y alguna Herramienta	(isUsedBy some Tecnica_de_Prueba) and (isUsedBy some Herramienta)	Tipo de Prueba Técnica de prueba Herramienta	utiliza

Tabla 5: Ejemplo de una regla en la ontología Procedimiento de Pruebas de Software.

En la **Tabla. 5** se muestra la regla *Tipo de Prueba* con la expresión *utiliza algunas Técnicas de Pruebas y algunas Herramienta,* definiéndose en las columnas de conceptos las clases Tipo de Prueba, Técnica de prueba y Herramientas y en la columna de relaciones se muestra la relación *utiliza.*

2.5 Formalización

Para llevar la ontología a la transformación del modelo conceptual al modelo formal o semi-computable utilizamos el editor Protégé 4.1, donde se muestran los conceptos, relaciones entre conceptos, instancias y atributos de la ontología.

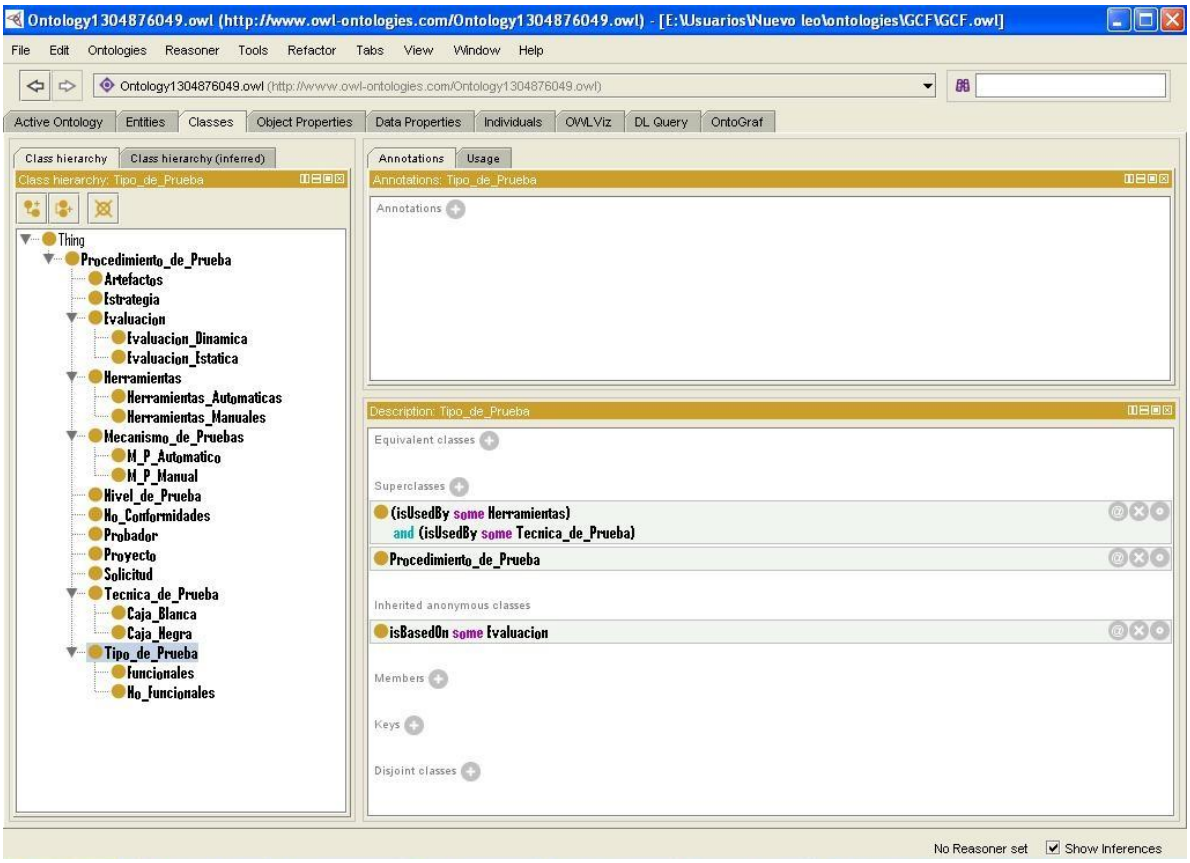


Figura. 5: Editor Protégé.

Cuando se inicializa el editor Protégé se trabaja primeramente en la pestaña *Classes,* como se muestra en la **Figura. 5**, jerárquicamente todas las clases dependen de owl: Thing, que es una palabra clave que hace referencia a la W3C, las pestañas más importantes son, *Classes*(Conceptos), *Object Properties*(Relaciones) y *OWL Viz*. [8]

Classes: en esta pantalla se crean las clases, subclases, restricciones, comentarios, disjoints y se pueden aplicar razonadores lógicos como HermiT1.2.3.

Las restricciones se usan para limitar las relaciones válidas entre individuos, pueden ser necesarias o suficiente y necesarias, adicionalmente se tienen cuantificadores universales y existenciales. En OWL las clases pueden solaparse (tener individuos comunes) a no ser que se diga explícitamente que son disjuntas. En la **Figura. 6** se muestra la taxonomía de las clases en Protégé.

Figura. 6: Pestaña de Classes del Protégé.

Properties: en esta pantalla se establecen las relaciones entre 2 individuos, las relaciones pueden ser de 2 tipos, "*Object properties*" las cuales se establecen entre individuos y "*Datatype properties*" la cual se establece entre individuos y esquemas xml, se recomienda nombrar las propiedades con la primera letra en minúscula, las propiedades tienen unas características que se pueden asignar por ejemplo: Funciona, Funcional inversa, simétrica y transitiva, también es posible asignar los dominios y los rangos a cada propiedad, estos son axiomas que se utilizan para que el razonador haga ciertas inferencias, violar una restricción de dominio o de rango no significa necesariamente que la ontología sea inconsistente o que contenga errores.

En la **Figura. 7** se muestra como quedan conformadas las *Properties*, es decir, las relaciones de la ontología.

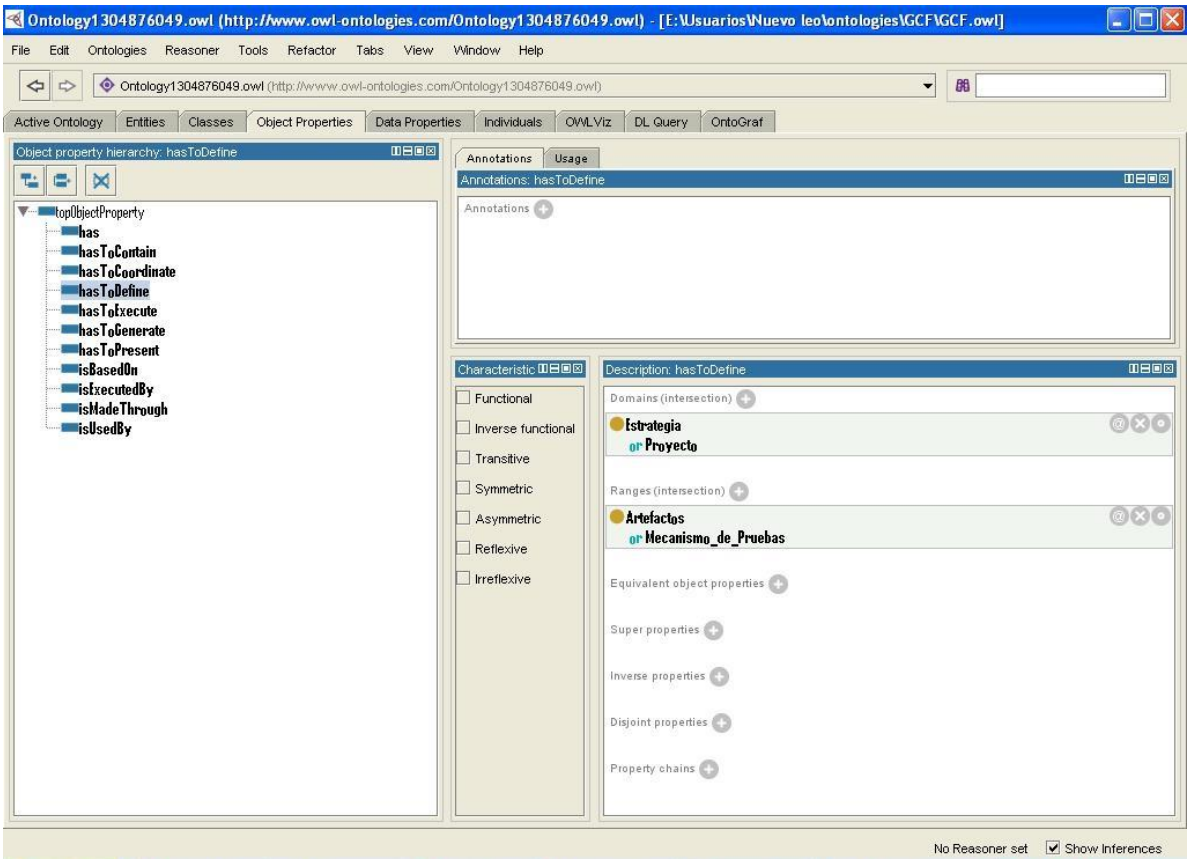


Figura. 7: Pestaña Object Properties del Protégé.

OWL Viz: es un plugin para Protégé que permite visualizar con grafos los conceptos y las relaciones que tienen creadas, adicional a este plugin se debe tener instalado el GraphViz[10], las principales funciones de esta pantalla son:

- Mostrar clases
- Mostrar subclases
- Mostrar superclases
- Ocultar clases
- Ocultar subclases
- Ocultar superclases
- Ocultar clases especificando el radio
- Ocultar todas las clases

En la **Figura. 8** se muestra la gráfica de cómo quedaría la ontología Procedimiento de Pruebas de Software en el editor Protégé una vez instalado el *GraphViz*.

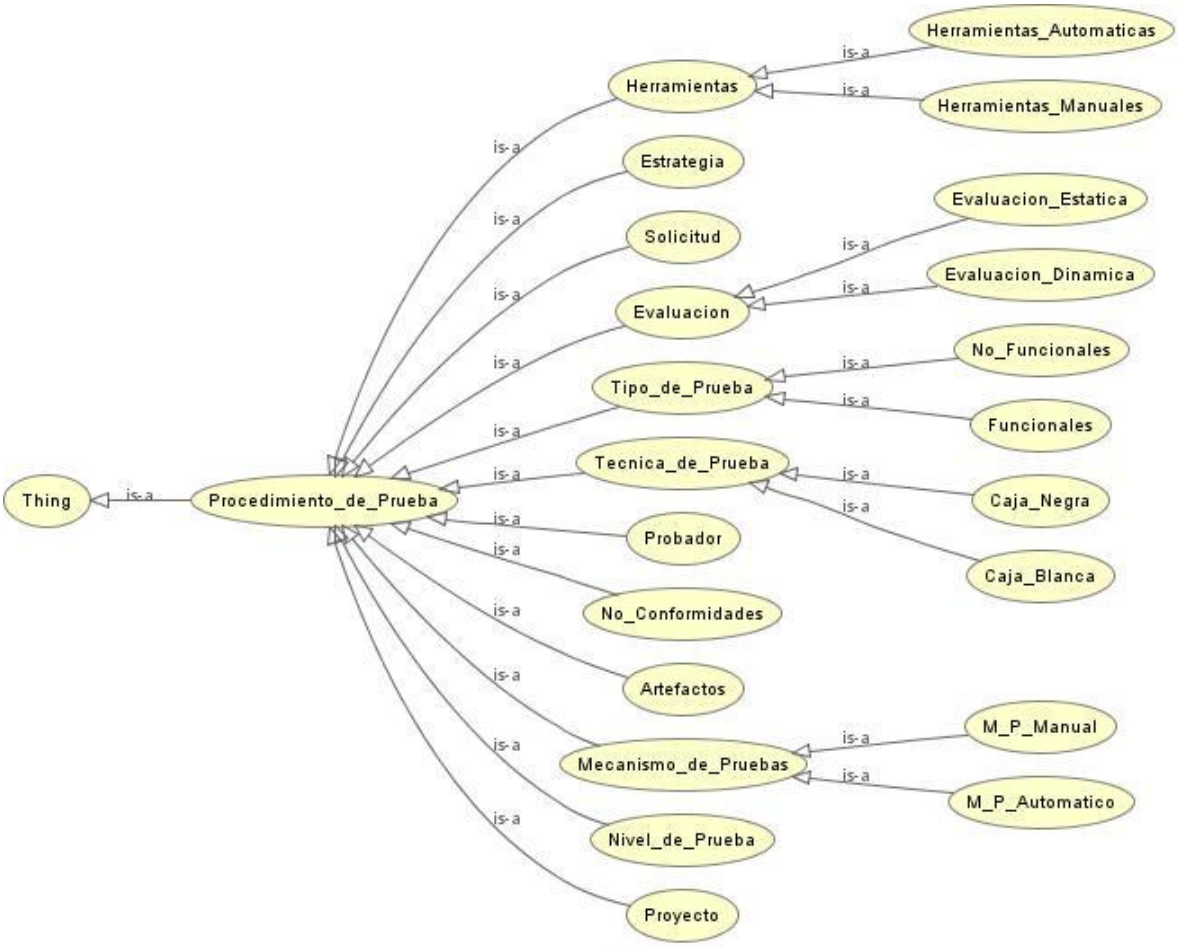


Figura. 8: Pestaña OWL Viz del Protégé.

En la siguiente **Figura. 9** se muestra cómo quedaría la ontología en la pestaña *OntoGraf* del editor Protégé pero aquí las clases están relacionadas unas con otras, cada color indica las diferentes relaciones que existen entre una clase y otra.

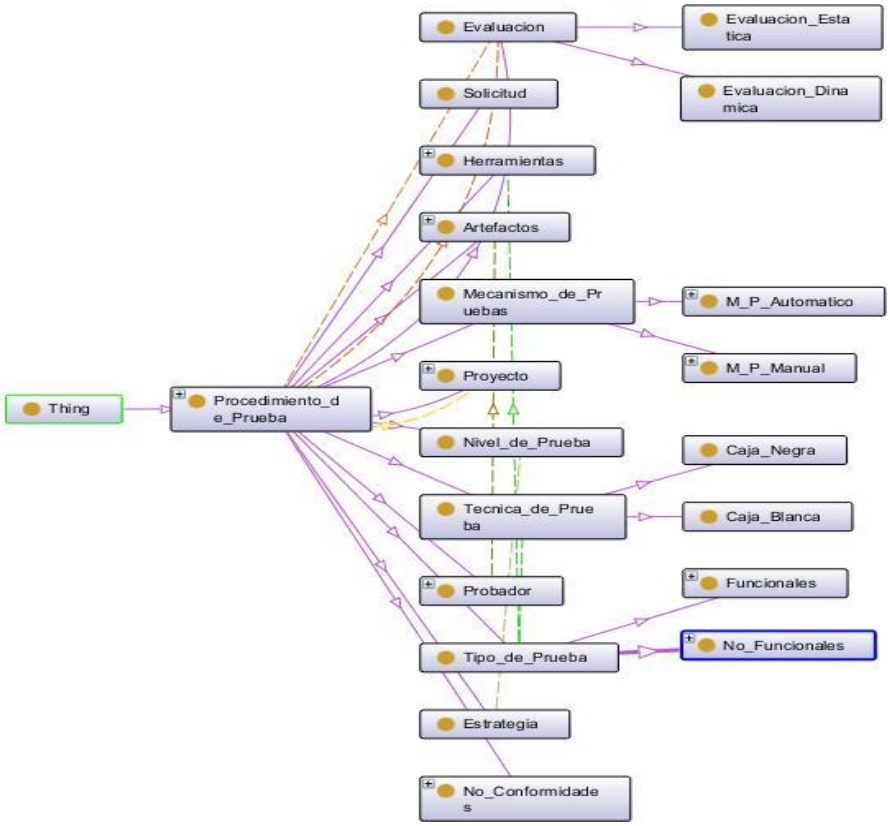


Figura. 9: Pestaña OntoGraf del Protégé.

En la **Figura. 10** se muestra el editor Protégé modelando como quedan las reglas y los axiomas entre diferentes clases, ejemplo de un axiomas sería, Tipo de Prueba utiliza solamente una Técnica de Prueba y una Herramienta, quedando en el Protégé *isUsedBy only (Tecnica_de_Prueba and Herramienta)* y la regla sería, un Tipo de Prueba utiliza alguna Técnica de Prueba y alguna Herramienta, y su expresión en el Protégé quedaría (isUsedBy some Tecnica_de_Prueba) and (isUsedBy some Herramienta).

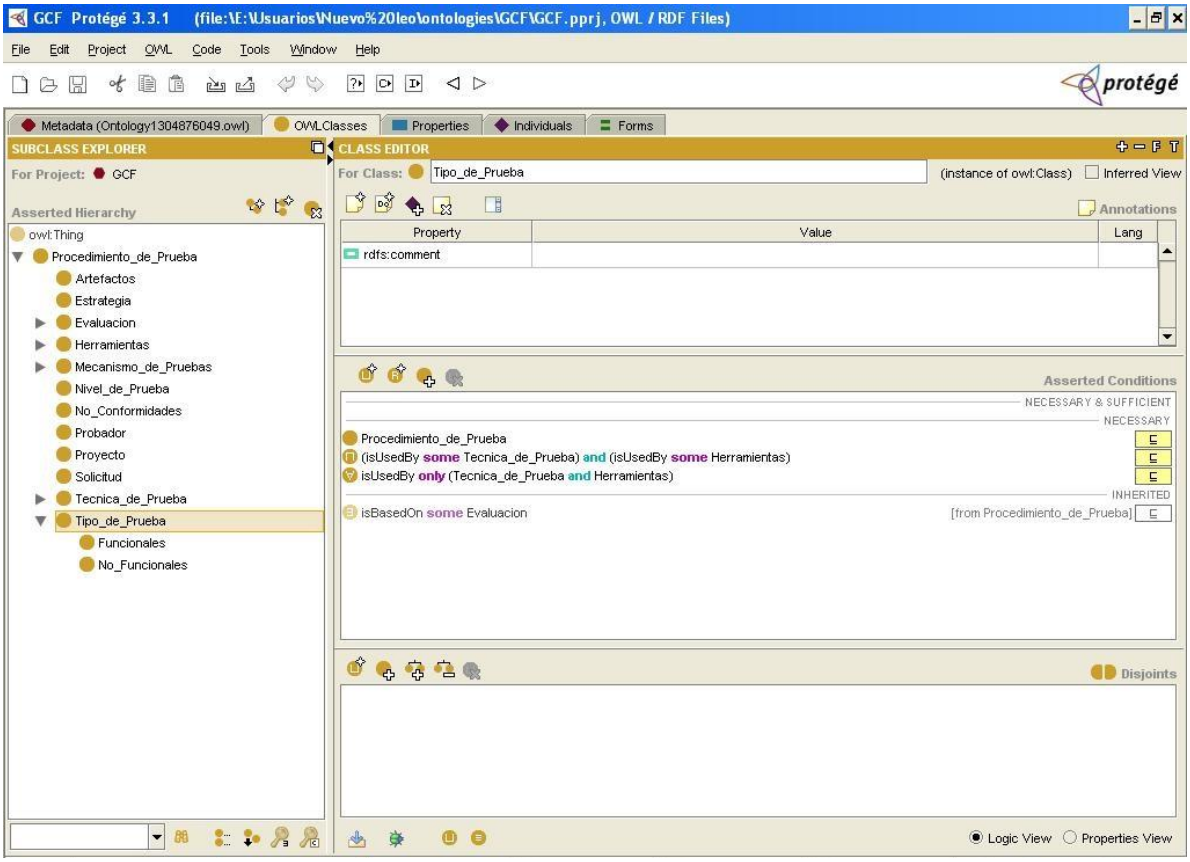


Figura. 10: Ejemplo de las reglas en el Protégé.

2.6 Implementación

El Lenguaje de Ontologías Web (OWL) proporciona un lenguaje para definir ontologías estructuradas basadas en Web. OWL pretende facilitar un lenguaje para ser usado con el fin de describir las clases, las propiedades de las clases y la relación entre las clases. OWL se construye sobre RDF, quien ofrece la base adecuada para desarrollar ontologías. Las ontologías basadas en RDF podrán ser distribuidas en muchos sistemas y serán compatibles con otros estándares web. [9]

Aquí se muestran algunos fragmentos que describen la manera en que se codifican los datos de la propuesta ontológica.

La declaración de las clases se hace de la siguiente manera, donde se declaran las clases *Procedimiento de Pruebas* y *Proyecto*:

```
<Declaration>
    <Class IRI="#Procedimiento_de_Pruebas"/>
</Declaration>
<Declaration>
    <Class IRI="#Proyecto"/>
</Declaration>
```

La declaración de una subclase de otra clase, se realiza de la siguiente manera, mostrando que las clases, *Caja Negra* y *Caja Blanca* son subclases de la clase *Técnica de Prueba*:

```
<SubClassOf>
    <Class IRI="#Caja_Blanca"/>
    <Class IRI="#Tecnica_de_Prueba"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="#Caja_Negra"/>
    <Class IRI="#Tecnica_de_Prueba"/>
</SubClassOf>
```

La declaración de las relaciones se realiza de la siguiente manera, donde se describe las relaciones que existen en la propuesta ontológica, *hasToPresent* e *isBasedOn*:

```
<Declaration>
    <ObjectProperty IRI="#hasToPresent"/>
</Declaration>
<Declaration>
     <ObjectProperty IRI="#isBasedOn"/>
</Declaration>
```

Para la ver el código OWL de la propuesta ontológica dirigirse a los **Anexos**.

2.7 Mantenimiento

El mantenimiento de software se define como "*cualquier modificación de un producto de software, después de su entrega, para corregir errores, mejorar el rendimiento u otros atributos, o a la acción de adaptar el producto a un entorno que cambia"* [S. Mamone]*; "los cambios en la gestión de productos de software para mantenerlos actualizados y en pleno funcionamiento".* [R. Singh]

Aun cuando los procesos de desarrollo de software duran pocos meses y años, los procedimientos de mantenimiento de software duran varios años, por tanto, es esta una fase fundamental, que debe ser planificada correctamente para evitar gastos excesivos para la empresa que crea o diseña el software. [Montoya, Edgar]

Para el mantenimiento de la ontología se diseñará un mecanismo para preparar en el trabajo con el editor Protégé a varios integrantes del GCF, para que en caso de alguna modificación del procedimiento de pruebas, algún error en el sistema que utilice la ontología o algún malfuncionamiento en la propia ontología, pueda ser corregido y actualizado propiamente.

2.8 Conclusiones del capítulo

- ✓ Se realizó en este capítulo el diseño e implementación de la propuesta ontológica para el Grupo de Calidad de FORTES, haciendo uso de la Metodología *Methontology*, usando cada uno de los pasos y etapas definidos en la misma.

Capítulo 3: Validación de la Propuesta

Introducción al Capítulo

En el presente capítulo se pondrá en práctica la propuesta planteada en el capítulo anterior. Se ejemplificará el diseño de la propuesta a través de una Ontología para un dominio específico, en este caso, pruebas de software. Se abordarán algunas de las características principales relacionadas con el contenido del dominio, se mostrará de manera física el funcionamiento de la ontología creada, y dos métodos para validar la propuesta ontológica, la validación mediante el Servicio de Validación Web de la W3C (*World Wide Web Consortium*) y la exportación de la propuesta ontológica a un modelo de almacenamiento persistente basado en Base de Datos.

3.1 Validación mediante el Servicio de Validación Web de la W3C.

El **World Wide Web Consortium** (**W3C**) es una comunidad internacional que desarrolla estándares que aseguran el crecimiento de la Web a largo plazo. Está dirigida por Tim Berners-Lee, el creador original de **URL** (*Uniform Resource Locator*, Localizador Uniforme de Recursos), **HTTP** (*HyperText Transfer Protocol*, Protocolo de Transferencia de HiperTexto) y **HTML** (Lenguaje de Marcado de HiperTexto) que son las principales tecnologías sobre las que se basa la Web. Fue creada el 1 de octubre de 1994 por Tim Berners-Lee en el MIT, actual sede central del consorcio. Está asociada con la compañía francesa ERCIM y la universidad japonesa Keiō (Shonan Fujisawa Campus).

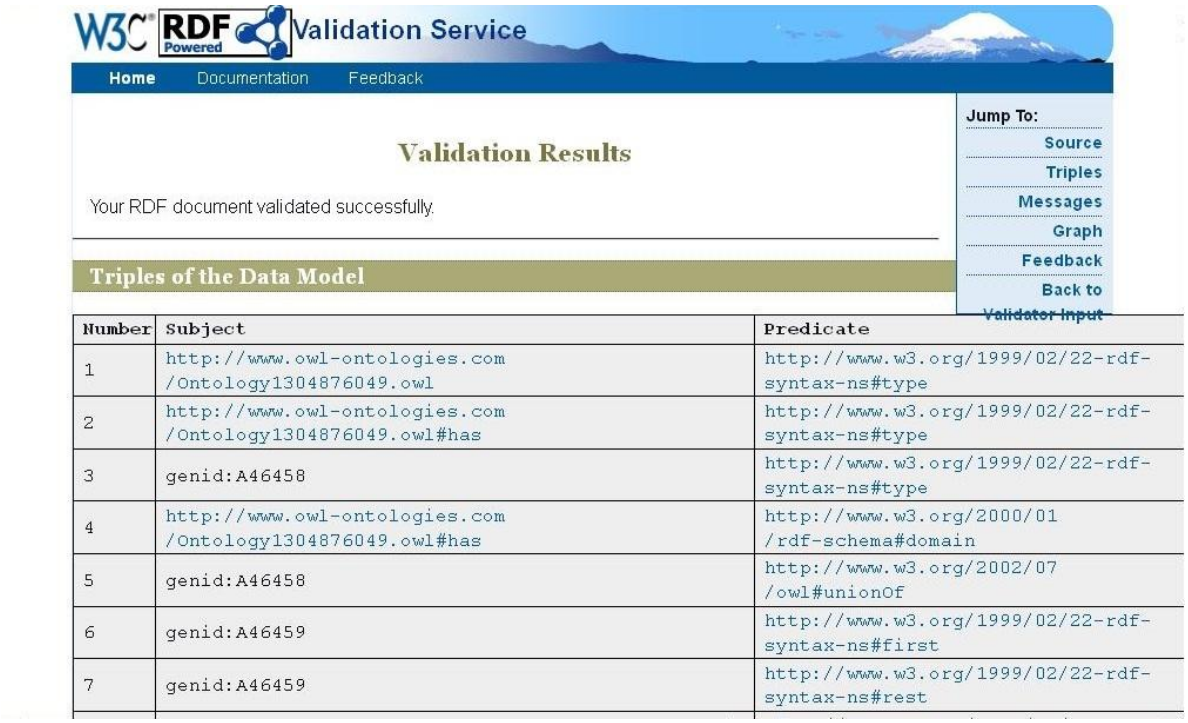
Además de la clásica “Web de los documentos” El W3C está ayudando a construir un conjunto de tecnologías para apoyar una “Red de datos,” el tipo de datos que se encuentran en bases de datos. El objetivo último de la Web de los datos es permitir a las computadoras hacer el trabajo más útil y desarrollar sistemas que puedan apoyar las interacciones de confianza en la red. El término “Web Semántica” se refiere a la visión del W3C de la Web de los datos vinculados. Las tecnologías de la Web Semántica permiten a las personas crear recopilaciones de datos en la Web, crear vocabularios, y escribir las reglas para el manejo de datos. Los datos vinculados se encuentran facultados por tecnologías como RDF, SPARQL, OWL, y SKOS.

La W3C cuenta con un servicio de validación de archivos RDF, **Figura. 11**, servicio que permite al usuario conocer cuando la ontología o el archivo poseen inconsistencias, verifica la taxonomía de dicha ontología, etc.

Figura. 11: Validador W3C.

En esta página el usuario puede verificar, insertando íntegramente el código RDF de su ontología, lo anteriormente mencionado. El sistema muestra, luego de parsear el código RDF, los datos de la ontología, las relaciones, etc. Mostrando además el resultado de la revisión, si se realizó y validó correctamente o si por algún error no se pudo validar el archivo.

Luego de generar el código RDF de la propuesta ontológica en cuestión, y siguiendo los pasos descritos a lo largo del epígrafe, el validador arrojó el resultado que se muestra en la **Figura. 12**, mostrando que la ontología y el archivo RDF como tal se validó correctamente y sin incongruencias.



W3C RDF Powered Validation Service

Home Documentation Feedback

Jump To: Source Triples Messages Graph Feedback Back to Validator Input

Validation Results

Your RDF document validated successfully.

Triples of the Data Model

Number	Subject	Predicate
1	http://www.owl-ontologies.com /Ontology1304876049.owl	http://www.w3.org/1999/02/22-rdf-syntax-ns#type
2	http://www.owl-ontologies.com /Ontology1304876049.owl#has	http://www.w3.org/1999/02/22-rdf-syntax-ns#type
3	genid:A46458	http://www.w3.org/1999/02/22-rdf-syntax-ns#type
4	http://www.owl-ontologies.com /Ontology1304876049.owl#has	http://www.w3.org/2000/01 /rdf-schema#domain
5	genid:A46458	http://www.w3.org/2002/07 /owl#unionOf
6	genid:A46459	http://www.w3.org/1999/02/22-rdf-syntax-ns#first
7	genid:A46459	http://www.w3.org/1999/02/22-rdf-syntax-ns#rest

Figura. 12: Resultado de la ontología por el Validador W3C.

3.2 Exportación a un modelo de almacenamiento persistente basado en base de datos.

Para lograr la exportación a un modelo de almacenamiento persistente, es necesario utilizar el plugin de la librería Jena para Protégé, *protege2jena* disponible en: *http://semweb.krasu.ru/protege2jena/* el cual permite copiar los datos del archivo OWL a un modelo relacional.

La facilidad que brinda la utilización de este plugin es, que el mismo permite realizar este proceso desde el mismo Protégé. Para realizar esta acción es necesario seguir los siguientes pasos:

1. Descargar el plugin desde la dirección mostrada y extraerlo dentro de la carpeta plugins de la instalación del Protégé.

2. Abrir el editor Protégé con el archivo OWL que se quiera exportar a almacenamiento persistente.

3. Seleccionar en el menú *File*, la opción *Export to format*, dentro de esta seleccionar la opción *Jena Persistent Model.*

4. Completar la información solicitada en el cuestionario de la ventana emergente.

Figura. 13: Ventana para exportar a almacenamiento persistente.

En este caso se utilizará una base de datos en MySQL, con el nombre *gcf*, en equivalencia el nombre del Grupo de Calidad de FORTES.

Una vez hecho esto, ya se habrá exportado el modelo ontológico a un modelo persistente en una base de datos. Para el trabajo con el modelo persistente, se utilizó el **IDE NetBeans 6.9**, por ser este un IDE multiplataforma, multilenguaje y de código abierto u *open source.* Seguidamente se crea la conexión a la base de datos, especificando el nombre de la base de datos, el driver de conexión que en este caso es el **JDBC**, y el user y el password para la conexión. Quedando como muestra la **Figura. 14**.

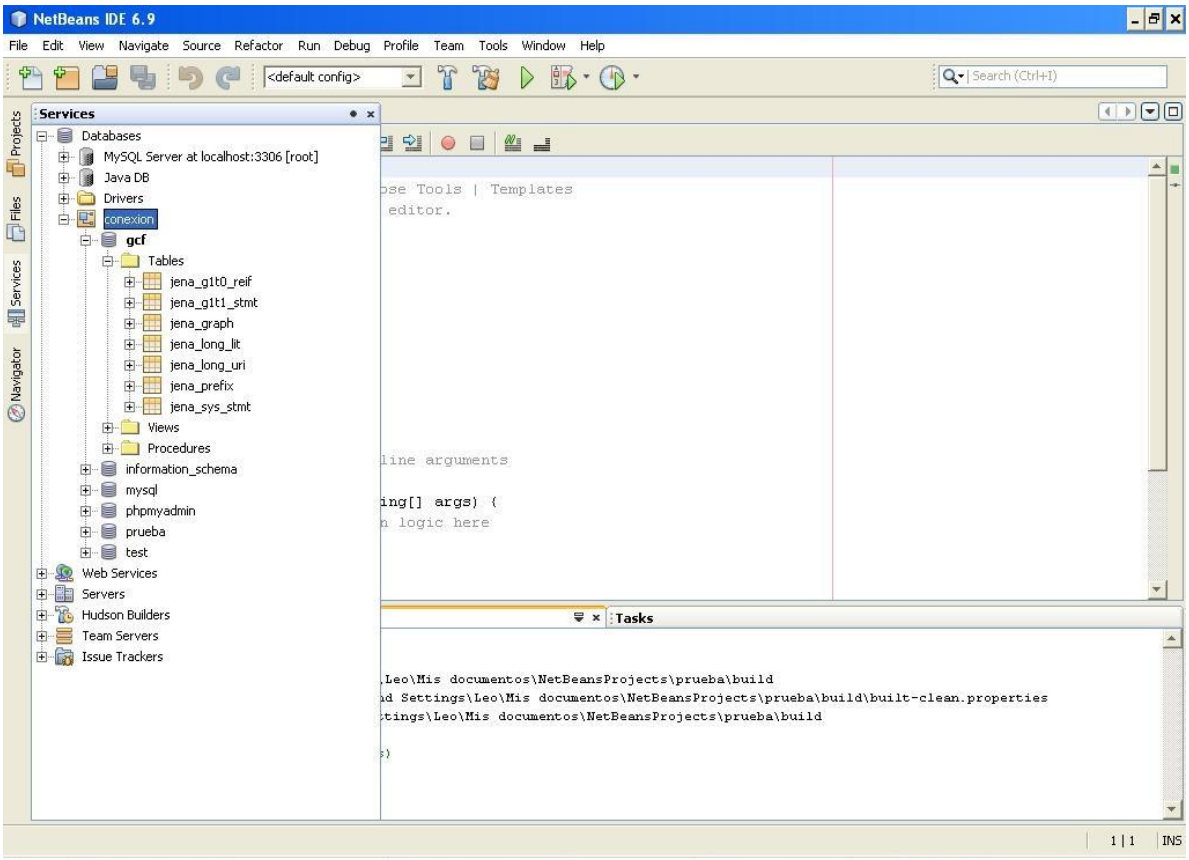


Figura. 14: Ventana del modelo persistente en una base de datos.

Una vez hecho esto se procede a crear el proyecto el cual permitirá hacer las consultas y validar que la propuesta ontológica en cuestión funciona correctamente, de manera fiable y segura.

El primer paso sería crear el proyecto, seleccionando en el menú *File* la opción *New Project*, se selecciona la opción para el tipo de proyecto Java, y de tipo *Java Application* una vez creado el proyecto se debe agregar la librería Jena, que es la que permitirá exportar desde el modelo ontológico al modelo relacional, tal y como se muestra en la **Figura. 15**.

Esto se realiza ejecutando clic derecho sobre el proyecto creado en la pestaña Proyectos o *Projects*, seleccionando la opción propiedades o *Properties* luego en la ventana que aparecerá a continuación, se selecciona en la parte izquierda el menú *Libraries* y se selecciona la opción *Add Library* para crear la librería.

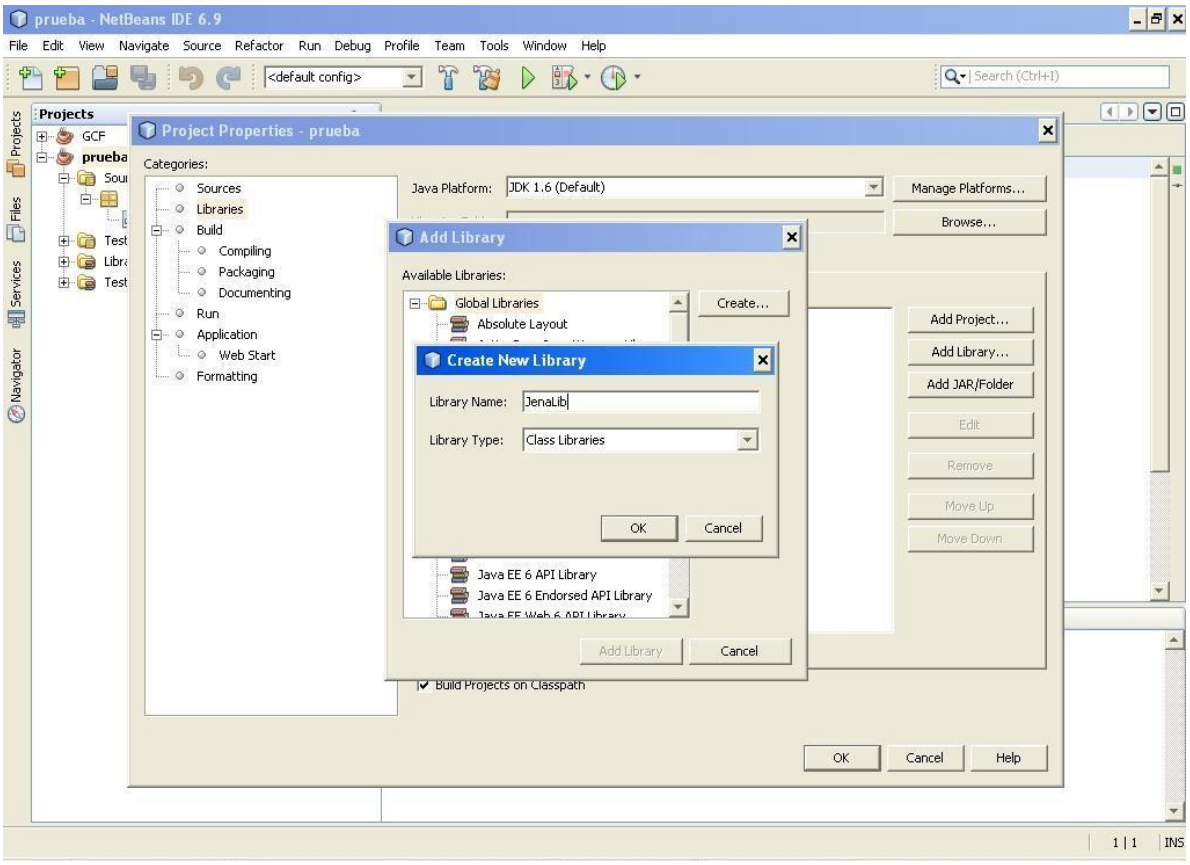


Figura. 15: Ventana donde se adicionan las librerías.

Una vez creada la librería la cual se nombrará *JenaLib* se procede a agregar los archivos que permitirán el funcionamiento correcto de la librería creada, esto se realiza seleccionando la dirección donde se encuentran las extensiones JAR eligiendo la opción *Add JAR/Folder*, luego se debe definir el *Source Location* y la documentación *JavaDoc,* tal y como muestra la **Figura. 16**.

Se debe agregar además el driver JDBC de conexión de la misma manera escogiendo la opción *Add JAR/Folder* una vez creada la librería y hecho todo lo anteriormente mencionado, esto para evitar incongruencias en la conexión con la Base de Datos creada.

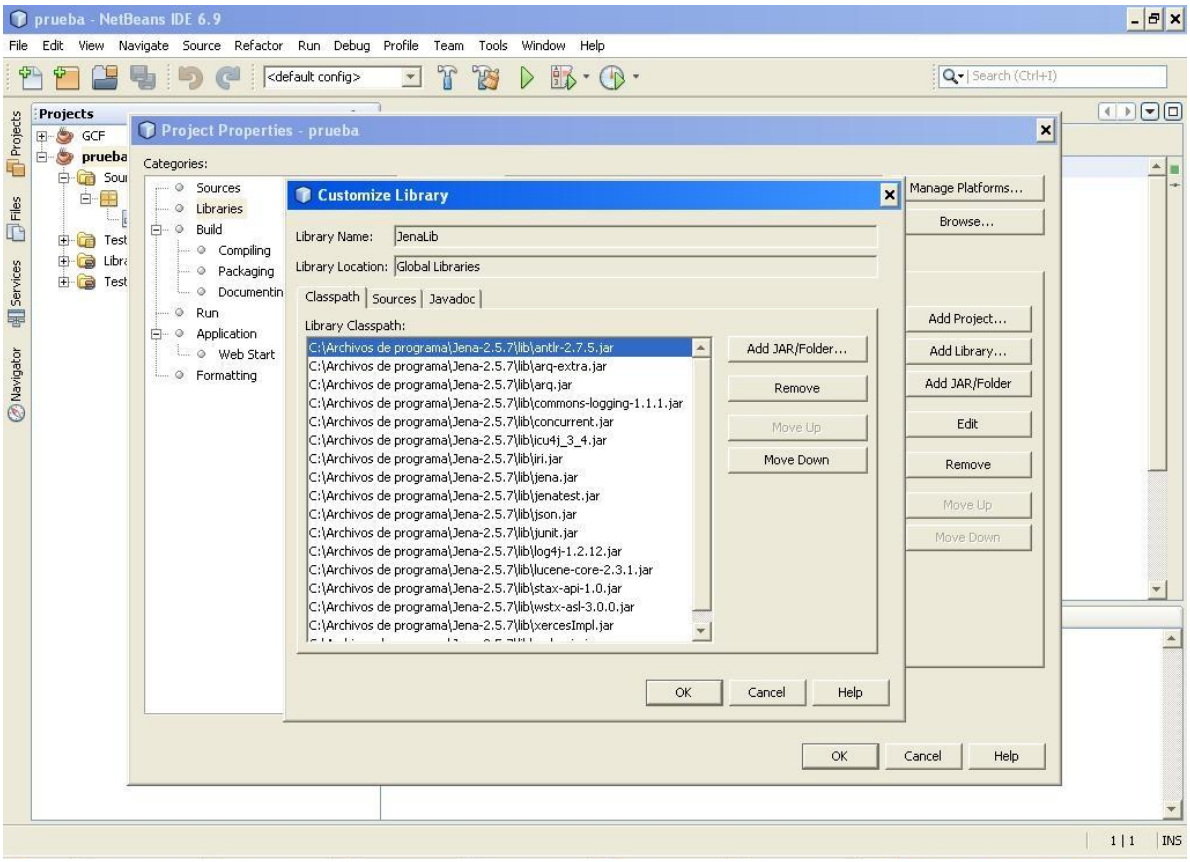


Figura. 16: Ventana donde se adicionan los JAR.

Una vez agregada la librería como se ha descrito a lo largo del epígrafe, el proyecto debe quedar como muestra la **Figura. 17,** con todo configurado para proceder a ejecutar las consultas a la ontología exportada del modelo ontológico al modelo relacional, mediante almacenamiento persistente basado en base de datos.

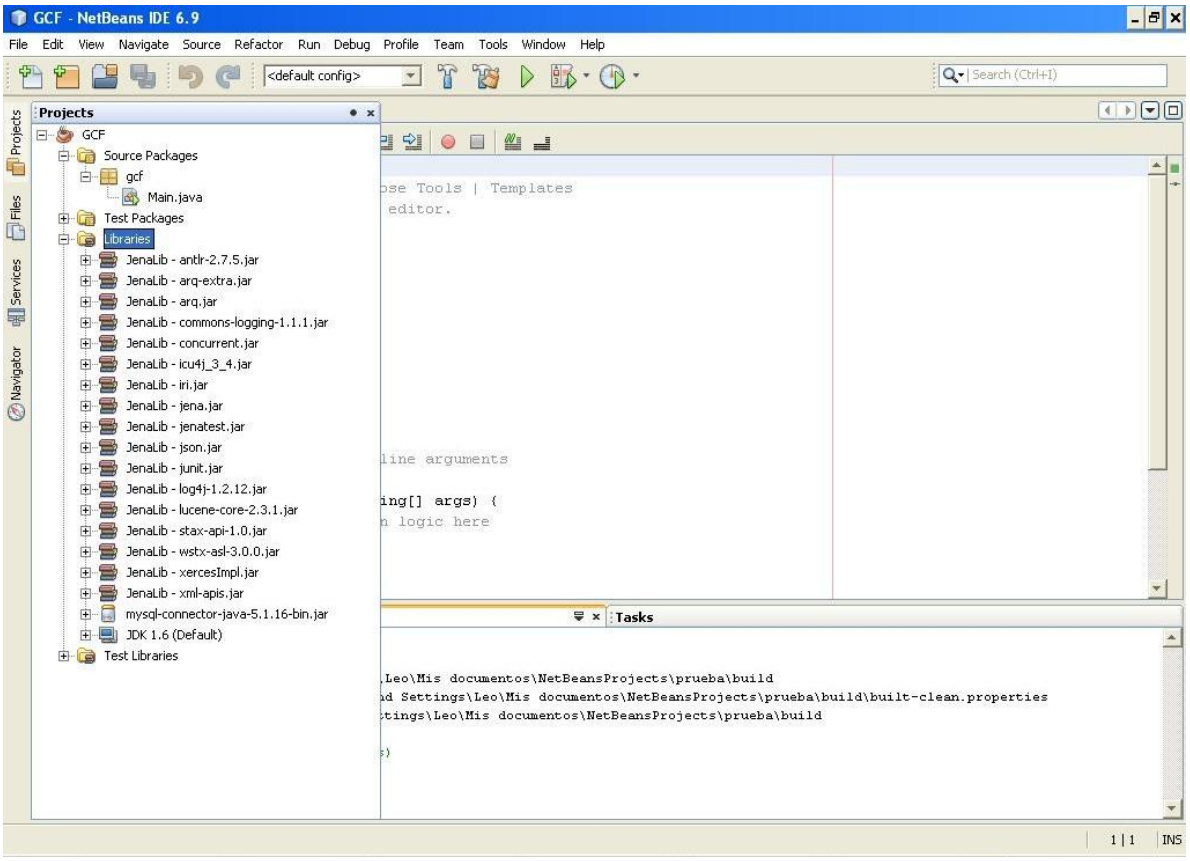


Figura. 17: Ontología exportada correctamente a almacenamiento persistente.

3.3 Validación mediante consultas a la ontología exportada al modelo persistente

Se puede decir que una ontología está bien elaborada, cuando esta es capaz de responder una serie de preguntas propuestas por el usuario o sistema que la utilice. Para validar la correcta realización de la ontología para el Grupo de Calidad de FORTES, se proponen las siguientes preguntas, a convertir en consultas para el modelo ontológico exportado al modelo relacional, mediante almacenamiento persistente con base de datos. Las preguntas son:

1- ¿Cuál son los probadores vinculados al proyecto Grupo de Calidad de FORTES?

2- ¿Qué artefactos coordina el probador Probador_1?

3- ¿Qué Tipos de Prueba deben presentarse ante el artefacto Casos de Prueba?

4- ¿Qué proyectos ejecutan el Procedimiento de Prueba definido por el Grupo de Calidad de FORTES?

5- ¿Qué Procedimiento de Prueba ejecuta el proyecto Alfaomega?

6- ¿Qué tipo de no conformidades se generan a partir del artefacto Casos de Prueba?

Para realizar las consultas a la ontología exportada a la base de datos mediante el uso del plugin Jena, como se describe en el epígrafe 3.2, es necesario adicionar todas las clases que van a utilizarse por el IDE para poder representar de una manera comprensible los datos almacenados en las tablas de la base de datos. Esto lo permite hacer el IDE una vez que se escriba el código y se utilicen las diferentes clases del API de Jena, cargado como una librería para nuestro IDE en este caso el NetBeans.

La declaración de las variables, la conexión con el modelo persistente, el uso del razonador sobre el modelo abierto y la consulta a ejecutar quedarían como se muestra en la **Figura. 18**.

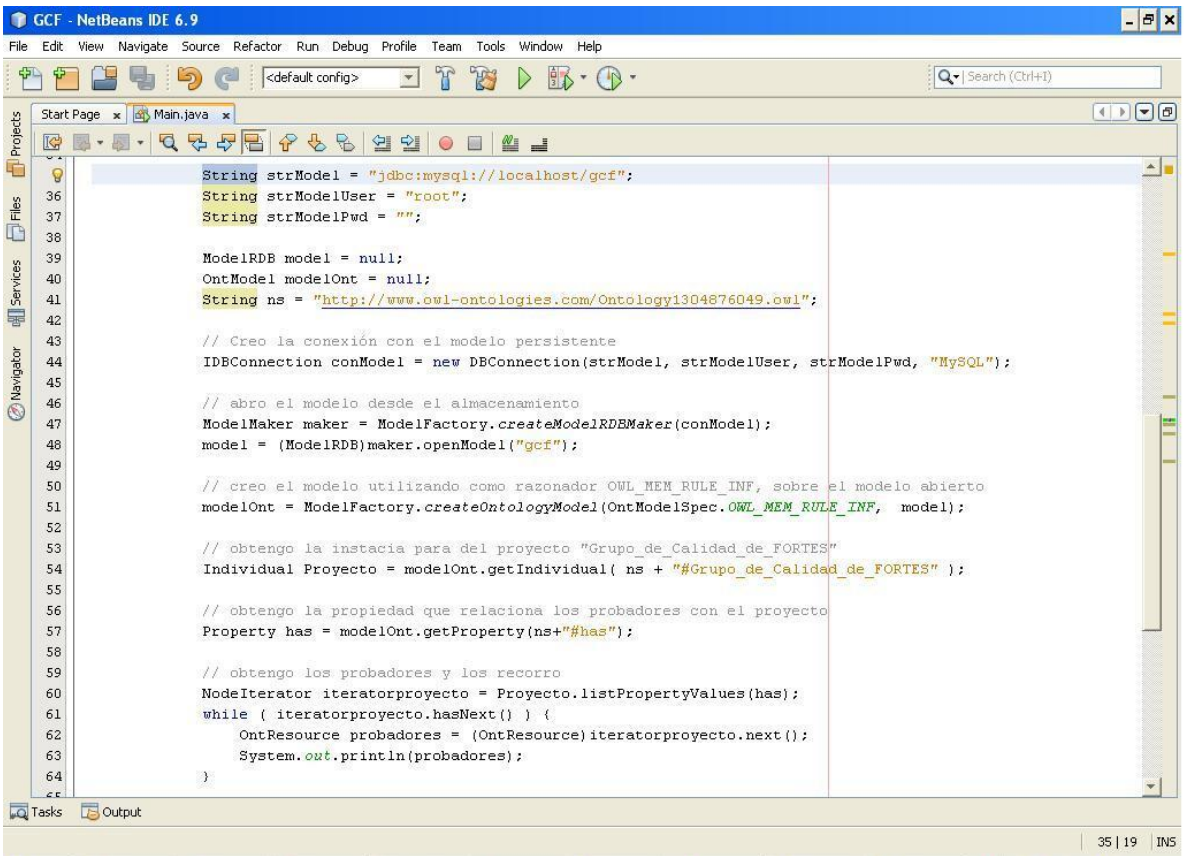


Figura. 18: Consulta al modelo persistente en el NetBeans.

La consulta que muestra la figura es la que responde a la Pregunta 1 la cual arrojó el resultado que muestra la **Figura. 19**.

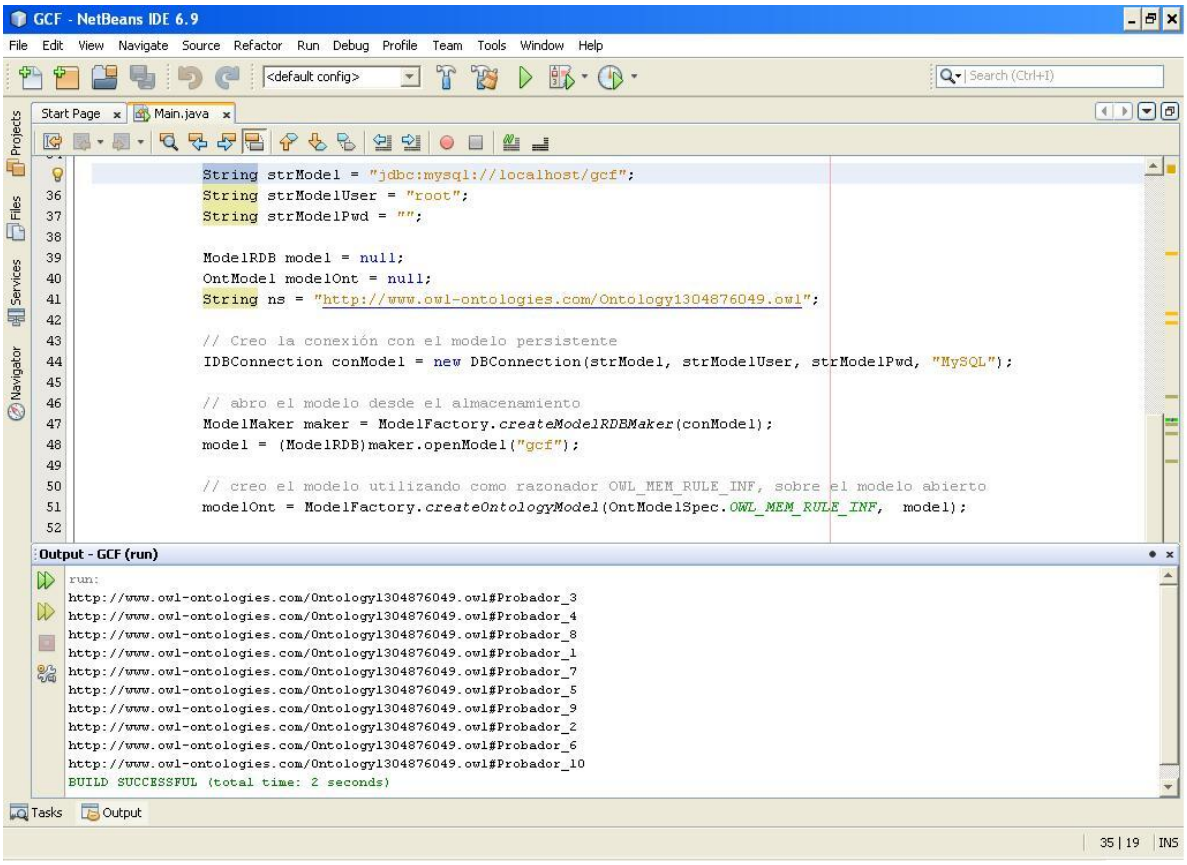


Figura. 19: Consulta y respuesta de la Pregunta 1.

Las respuestas a las 5 preguntas siguientes arrojaron los siguientes resultados:

Pregunta 2

run:

http://www.owl-ontologies.com/Ontology1304876049.owl#Listas_de_Chequeo

http://www.owl-ontologies.com/Ontology1304876049.owl#Casos_de_Prueba

BUILD SUCCESSFUL (total time: 2 seconds)

Pregunta 3

run:

http://www.owl-ontologies.com/Ontology1304876049.owl#Funcion

http://www.owl-ontologies.com/Ontology1304876049.owl#Volumen

http://www.owl-ontologies.com/Ontology1304876049.owl#Seguridad

BUILD SUCCESSFUL (total time: 2 seconds)

Pregunta 4

run:

http://www.owl-ontologies.com/Ontology1304876049.owl#Alfaomega

http://www.owl-ontologies.com/Ontology1304876049.owl#CRODA

http://www.owl-ontologies.com/Ontology1304876049.owl#MOODLE

http://www.owl-ontologies.com/Ontology1304876049.owl#Multisaber_y_El_Navegante

http://www.owl-ontologies.com/Ontology1304876049.owl#Recursos_Didacticos

http://www.owl-ontologies.com/Ontology1304876049.owl#RHODA

BUILD SUCCESSFUL (total time: 7 seconds)

Pregunta 5

run:

http://www.owl-ontologies.com/Ontology1304876049.owl#Procedimiento_de_Prueba_definido_por_GCF

BUILD SUCCESSFUL (total time: 7 seconds)

Pregunta 6

run:

http://www.owl-ontologies.com/Ontology1304876049.owl#Aplicacion

BUILD SUCCESSFUL (total time: 2 seconds)

3.4 Conclusiones del Capítulo

Mediante la realización de este capítulo, se logró validar correctamente la propuesta ontológica diseñada en el capítulo anterior, mediante dos vías, el Servicio de Validación Web de la W3C, y con la realización de consultas a la ontología realizada, previamente transformada y exportada de modelo ontológico a modelo relacional de base de datos.

Conclusiones Generales

La realización de este trabajo, ha demostrado que la era de la gestión del conocimiento es una realidad cada vez más tangible, y esto vinculado con las TIC hace aún más vertiginoso el crecimiento de la misma, dándole a la información un nuevo significado, aproximándose lo más posible al pensamiento humano. Este trabajo derivó además las siguientes conclusiones:

- ✓ Para la ejecución de esta investigación se determinaron los conceptos de Calidad, Pruebas de Software, Gestión del Conocimiento y Ontología, así mismo la metodología METHONTOLOGY para la creación de la ontología.

- ✓ Se formalizó correctamente el modelo de conocimiento en el Grupo de Calidad de FORTES, que se utilizó para el desarrollo de la ontología.

- ✓ La realización de esta investigación permitió la creación de una ontología capaz de resolver las funciones definidas para nuestro campo de acción.

- ✓ Mediante la realización de este trabajo, se logró demostrar de manera práctica alguno de los usos de las ontologías, con la exportación de la ontología, desde el modelo ontológico a un modelo relacional, utilizando almacenamiento persistente basado en base de datos.

- ✓ Se validó la ontología creada mediante dos métodos, el validador web de la W3C y mediante consultas a la ontología previamente exportada a almacenamiento persistente, arrojando resultados satisfactorios para cada uno de los casos.

Recomendaciones

- ✓ Se recomienda la implementación de un Sistema de Información Basado en Ontologías (SIBO) o la creación de un sistema inteligente para la aplicación correcta de la ontología creada.

- ✓ Se recomienda extender el uso de la ontología para el Procedimiento de pruebas de software a otros grupos de Calidad de otros centros productivos en la universidad, para así facilitar la gestión del conocimiento en dichos proyectos.

- ✓ Promover el uso correcto de la ontología en el GCF, para todos los integrantes del mismo además de mantener actualizada la propuesta sobre el modelo de conocimiento creado.

Referencias Bibliográficas

[Abad, M, 2004]. Ontologías. Inteligencia Artificial. FBI-UPC. Disponible en: <http://www.slideshare.net/cuacua/3-rc2-ontologias-2> Fecha de acceso: 20 de enero de 2011.

[Canals 2003]. Canals, Agustí. La gestión del conocimiento. (Julio del 2003). Disponible en: <http://www.uoc.edu/dt/20251/>Fecha de acceso: 18 de enero de 2011.

[Cartuche Flores, M. A, 2009].Ontología para la recomendación de recursos educativos almacenados en el Repositorio de Objetos de Aprendizaje (ROA) DSpace. Disponible en: <http://repositorio.utpl.edu.ec/bitstream/123456789/3732/1/004X675.pdf> Fecha de acceso: 20 de enero de 2011.

[Diccionario Océano]. Diccionario Ilustrado Océano de la Lengua Española. Edición del Milenio. Disponible en: http://www.oceano.com. Fecha de acceso: 13 de enero del 2011.

[Falbo, R. Guizzardi, G. Duarte K]. An ontological Approach to Domain Engineering. Disponible en:<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.19.2577&rep=rep1&type=pdf> Fecha de acceso: 20 de enero de 2011.

[Figueroa 2006]. Figueroa, Liliana; Palavecino, Rosa. Aproximación a la diferencia entre Gestión de la Información y la Gestión del Conocimiento. (2006). Disponible en: <http://www.cibersociedad.net/congres2006/gts/comunicacio.php?id=618&llengua=es>. Fecha de acceso 18 de enero de 2011.

[Fox, M. Gruninger, M. 1998]. Enterprise modeling. AI Magazine, 19(3):109-121. Disponible en: <http://aaaipress.org/ojs/index.php/aimagazine/article/viewFile/1399/1299>. Fecha de acceso: 20 de enero de 2011.

[Gruber, T. R. 1993]. Toward Principles for the Design of Ontologies Used for Knowledge Sharing". Stanford Knowledge Systems Laboratory. Disponible en: <http://archive.itee.uq.edu.au/~infs3101/ Readings/OntoEng.pdf>. Fecha de acceso: 18 de enero de 2011.

[Guarino, N 1998]. Formal Ontology and Information Systems. Disponible en: <http://www.loa-cnr.it/Papers/FOIS98.pdf> Fecha de acceso: 20 de enero de 2011.

[IEEE, 1991]. Standard 610, Computer Dictionary. Nueva York, 1990. p.

[Laguna, M-A, García, F. López, O y Márquez, J, 2002]. Reutilización de requisitos en el modelo mecano. Disponible en: <http://giro.infor.uva.es/Publications/2001/LGLM01/JIRA_Laguna.pdf> Fecha de acceso: 20 de enero de 2011.

[Mizoguchi et al., 1999].Task ontology for reusable problem solving knowledge, Towards Very Large Knowledge Bases: Knowledge Building & Knowledge Sharing. IOS Press, pp. 46-59. Disponible en: <http://www.ei.sanken.osaka-u.ac.jp/pub/miz/KBKS95.pdf>. Fecha de acceso: 20 de enero de 2011.

[Polo Usaola, Macario]. Mantenimiento Avanzado de Sistemas de Información Pruebas del Software. Disponible en: <http://alarcos.inf-cr.uclm.es/doc/masi/doc/lec/parte5/polo-apuntesp5.pdf> Fecha de acceso: 17 de enero de 2011.

[Pressman, 2002]. Ingeniería del Software. Un enfoque práctico. (Parte 1: páginas_131-148).

[Ramos, Esmeralda. Nuñez, Haydemar. 2007].ONTOLOGÍAS: componentes, metodologías, lenguajes, herramientas y aplicaciones. Disponible en: <http://www.ciens.ucv.ve/escueladecomputacion/documentos/archivo/51> fecha de acceso: 20 de enero de 2011.

[Sánchez 2005]. Semántica: Sánchez Fernández, Luis y Fernández García, Norberto. La Web fundamentos y breve "estado del arte". (Revista Novática No 178, noviembre-diciembre de 2005). Disponible en: http://www.ati.es/novatica. Fecha de acceso: 13 de enero del 2011.

[Sheng-Tun, L. Huang Chih, H. I-Wei, S. 2003]. An Ontology-based Knowledge Management System for the Metal Industry. Disponible en: <http://www2003.org/cdrom/papers/alternate/P620/p620-li.html> Fecha de acceso: 20 de enero de 2011.

[Simón, 2006]. Simón, Alfredo; Estrada, Vivian; Rosete, Alejandro; Lara, Vladimir. GECOSOFT: UN Entorno Colaborativo para la Gestión del Conocimiento con Mapas Conceptuales. (2006). Disponible en: <http://cmc.ihmc.us/cmc2006Papers/cmc2006-p156.pdf>. Fecha de acceso: 18 de enero 2011.

[Smith, B. 2003]. Ontology and Information Systems. Disponible en: <http://ontology.buffalo.edu/ontology%28PIC%29.pdf>. Fecha de acceso: 16 de enero de 2011.

[Van Heist, 1997]. Using explicit ontologies in KBS development. International Journale of Human-Computer studies Vol. 47 pp.183 – 292. Disponible en: <http://www.cs.vu.nl/~guus/papers/Heijst97a.pdf> Fecha de acceso: 20 de enero de 2011.

[ISO 9000:2005]. International Organization for Standardization. Sistemas de gestión de la calidad, fundamentos y vocabulario.

[Fernández López M. 1999]. Overview of methodology for building ontologies. Proceedings of the IJCAI-99 workshop on Ontologies and Problem-Solving Methods (KRR5) Stockholm, Sweden, August. Disponible en: <http://www.lsi.upc.es/~bejar/aia/aia-web/4-fernandez.pdf> Fecha de acceso: 10 de marzo 2011.

[Montoya, Edgar]. Las ontologías en el mantenimiento del software. Disponible en: http://www.eatis.org/eatis2010/portal/paper/memoria/html/files/6.pdf Fecha de acceso: 18 de marzo de 2011.

[S. Mamone]. “The IEEE standard for software maintenance”. ACM SIGSOFT Software Engineering Notes. Vol. 19. 1994. pp. 75-76.

[R. Singh]. “ISO/IEC draft international standard 12207, software life-cycle processes”. IFIP Transactions. Vol. A-55. 1994. pp. 111-119.

[Samper 2005]. Samper Zapater, José Javier. Ontologías para Servicios Web Semánticos de Información de Tráfico: Descripción y Herramientas de Explotación. (5 de Julio de 2005). Disponible en: http://www.tesisenxarxa.net/TESIS_UV/AVAILABLE/TDX-0628106-085805//SAMPER.pdf

Bibliografía Consultada

[1] Colectivo de Autores. 2008. Mejores prácticas para el establecimiento y aseguramiento de la Calidad de Software. Disponible en: <http://www.eumed.net/libros/2008a/351/351.zip> Fecha de acceso: 17 de enero de 2011.

[2] Tipos de pruebas de software. Disponible en: <http://www.cetic.guerrero.gob.mx/pics/art/articles/113/file.TiposPruebasSoftware.pdf> fecha de acceso: 18 de enero de 2011.

[3] Colectivo de autores, 2006. Sistemas de información: Nuevos escenarios basados en ontologías. Disponible en: <http://www.jistem.fea.usp.br/index.php/jistem/article/download/10.4301%252FS1807-17752006000100001/42> Fecha de acceso: 17 de enero de 2011.

[4] Las pruebas en las aplicaciones Web, Conceptos y Clasificación. Disponible en: <http://www.web-mix.ws/webmaster/las-pruebas-en-aplicaciones-web-conceptos-y-clasificacion-fuente-las-pruebas-en-aplicaciones-web-conceptos-y-clasificacion/> Fecha de acceso: 17 de enero de 2011.

[5] Guía breve sobre Web Semántica. Disponible en: <http://www.w3c.es/Divulgacion/Guiasbreves/WebSemantica> Fecha de acceso: 20 de enero de 2011.

[6] Lenguaje de Ontologías Web (OWL). Vista General Disponible en: <http://www.w3.org/2007/09/OWL-Overview-es.html> Fecha de acceso: 20 de enero de 2011.

[7] Colectivo de autores, 2008. Ontologías en los sistemas de información/ conocimiento. Disponible en: <http://www.ing.unp.edu.ar/wicc2007/trabajos/ISBD/095.pdf> Fecha de acceso: 17 de enero de 2011.

[8] http://diegocaviedes.host56.com/docs/PROTEGE-Diego_Caviedes.pdf. Fecha de acceso: 15 de marzo de 2011.

[9] http://usuarios.multimania.es/obib/doc/doc_Web_Semantica.doc. Fecha de acceso: 15 de marzo de 2011.

[10] http://www.graphviz.org/. Fecha de acceso: 15 de marzo de 2011.

Anexos

Anexo.1

```
<?xml version="1.0"?>


<!DOCTYPE Ontology [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY xml "http://www.w3.org/XML/1998/namespace" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>


<Ontology xmlns="http://www.w3.org/2002/07/owl#"
     xml:base="http://www.owl-ontologies.com/Ontology1304876049.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     ontologyIRI="http://www.owl-ontologies.com/Ontology1304876049.owl">
    <Prefix name="xsd" IRI="http://www.w3.org/2001/XMLSchema#"/>
    <Prefix name="owl" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="" IRI="http://www.owl-ontologies.com/Ontology1304876049.owl#"/>
    <Prefix name="rdf" IRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>
    <Prefix name="rdfs" IRI="http://www.w3.org/2000/01/rdf-schema#"/>
    <Prefix name="p1" IRI="http://www.owl-ontologies.com/assert.owl#"/>
    <Declaration>
        <Class IRI="#Artefactos"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Caja_Blanca"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Caja_Negra"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Estrategia"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Evaluacion"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Evaluacion_Dinamica"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Evaluacion_Estatica"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Funcionales"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Herramientas"/>
```

```
</Declaration>
<Declaration>
   <Class IRI="#Herramientas_Automaticas"/>
</Declaration>
<Declaration>
   <Class IRI="#Herramientas_Manuales"/>
</Declaration>
<Declaration>
   <Class IRI="#M_P_Automatico"/>
</Declaration>
<Declaration>
   <Class IRI="#M_P_Manual"/>
</Declaration>
<Declaration>
   <Class IRI="#Mecanismo_de_Pruebas"/>
</Declaration>
<Declaration>
   <Class IRI="#Nivel_de_Prueba"/>
</Declaration>
<Declaration>
   <Class IRI="#No_Conformidades"/>
</Declaration>
<Declaration>
   <Class IRI="#No_Funcionales"/>
</Declaration>
<Declaration>
   <Class IRI="#Probador"/>
</Declaration>
<Declaration>
   <Class IRI="#Procedimiento_de_Prueba"/>
</Declaration>
<Declaration>
   <Class IRI="#Proyecto"/>
</Declaration>
<Declaration>
   <Class IRI="#Solicitud"/>
</Declaration>
<Declaration>
   <Class IRI="#Tecnica_de_Prueba"/>
</Declaration>
<Declaration>
   <Class IRI="#Tipo_de_Prueba"/>
</Declaration>
<Declaration>
   <ObjectProperty IRI="#has"/>
</Declaration>
<Declaration>
   <ObjectProperty IRI="#hasToContain"/>
</Declaration>
<Declaration>
   <ObjectProperty IRI="#hasToCoordinate"/>
</Declaration>
<Declaration>
   <ObjectProperty IRI="#hasToDefine"/>
</Declaration>
<Declaration>
```

```
    <ObjectProperty IRI="#hasToExecute"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#hasToGenerate"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#hasToPresent"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#isBasedOn"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#isExecutedBy"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#isMadeThrough"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#isUsedBy"/>
</Declaration>
<Declaration>
    <DataProperty IRI="#Anno_lectivo"/>
</Declaration>
<Declaration>
    <DataProperty IRI="#Nombre"/>
</Declaration>
<Declaration>
    <DataProperty IRI="#usuario"/>
</Declaration>
<SubClassOf>
    <Class IRI="#Artefactos"/>
    <Class IRI="#Procedimiento_de_Prueba"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="#Caja_Blanca"/>
    <Class IRI="#Tecnica_de_Prueba"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="#Caja_Negra"/>
    <Class IRI="#Tecnica_de_Prueba"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="#Estrategia"/>
    <Class IRI="#Procedimiento_de_Prueba"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="#Evaluacion"/>
    <Class IRI="#Procedimiento_de_Prueba"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="#Evaluacion_Dinamica"/>
    <Class IRI="#Evaluacion"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="#Evaluacion_Estatica"/>
    <Class IRI="#Evaluacion"/>
```

Glosario de Términos

1- **Calidad:** grado en el que un conjunto de rasgos distintivos inherentes cumple con las necesidades o expectativas establecidas, generalmente implícitas u obligatorias.

2- **Pruebas de Software:** son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación.

3- **Gestión del Conocimiento:** es el proceso sistemático de buscar, organizar, filtrar y presentar la información con el objetivo de mejorar la comprensión de las personas en un área específica de interés.

4- **Ontología:** estructura jerárquica que define formalmente las relaciones semánticas de un conjunto de conceptos. Se usa para crear vocabularios controlados/estructurados para la recuperación o el intercambio de información.

5- **Metodología:** es el conjunto de métodos por los cuales se regirá una investigación.

6- **Internet:** red mundial de ordenadores interconectados basada en el protocolo TCP/IP. Ofrece distintos servicios: e-mail, foros, FTP, chat, compartir ficheros, videoconferencia, etc.

7- **Dominio:** un dominio es un área funcional diferenciable con requisitos y rasgos similares que puede ser soportada por algún tipo de sistema de software.

8- **Usuario:** es la persona que utiliza o trabaja con algún objeto o que es destinataria de algún servicio público, privado, empresarial o profesional.

9- **Herramienta:** las herramientas se diseñan para cumplir uno o más propósitos específicos, por lo que son artefactos o requisitos con una función técnica.

10- **Ingeniería:** Se denomina con el nombre de ingeniería a aquella disciplina que se ocupa del estudio y de la aplicación de los conocimientos que de este y de la experiencia resultan, para que a través de diseños, técnicas y problemas puedan ser resueltos los diferentes problemas que afectan a la humanidad**.**

11- **Razonador:** para realizar inferencia, a través de los conceptos y en algunos casos las instancias, obteniendo nuevo conocimiento.

12- **Taxonomía:** un vocabulario controlado donde los términos están conectados a una o a múltiples jerarquías. Posibilita el control de ambigüedades, de sinónimos además de contener relaciones jerárquicas.

13- **Modelo:** es la representación real de los datos obtenidos a partir de la información disponible. Ejemplos son el modelo de estructura y el modelo de estilo de que representan la estructura analítica y la información de estilo asociada a un documento, el modelo podría ser un árbol, o un grafo orientado, o cualquier otra.

14- **Aplicación:** es el programa que el usuario activa para trabajar en el ordenador. Existen muchos programas de ordenador que pueden clasificarse como aplicación, generalmente se les conoce como Software.

15- **Información:** Conjunto de datos interrelacionados, debidamente estructurados y asociados a una experiencia.

16- **Proceso:** es un conjunto de actividades o eventos que se realizan o suceden con un determinado fin.