

Descubrimiento y Visualización de Conocimiento en la Herramienta HACOPDS usando Tecnología de Web Semántica

Luis Alberto Nava-Ordoñez¹, César Enrique Rose-Gómez² y Ana María Chávez-Trejo³

^{1,3} División de Estudios de Posgrado e Investigación

Instituto Tecnológico de Orizaba, Orizaba, Ver., México

¹albert_nava007@hotmail.com, ³achavez@itorizaba.edu.mx

²Instituto Tecnológico de Hermosillo, Hermosillo, Son., México
crose@ith.mx

Paper received on 01/07/12, Accepted on 08/08/12.

Resumen. Dentro del proceso de desarrollo de software es fundamental la gestión de conocimiento, ayuda a los miembros del equipo de desarrollo a entender el conocimiento que existe en el dominio de la aplicación y proporciona información para futuros desarrollos, asimismo proporciona soluciones de problemas ya resueltos anteriormente con lo cual se reduce de manera significativa el desarrollo del mismo, esto se refleja en el oportuno cumplimiento del calendario y uso eficiente de los recursos. En este trabajo se describe un módulo para el descubrimiento y visualización de conocimiento que se integrará a la herramienta HACOPDS, la cual administra el conocimiento orientado al proceso de desarrollo de software.

Palabras Clave: Gestión de Conocimiento, Web Semántica, Ontología, Razonamiento.

1 Introducción.

La construcción de un sistema o producto de software, requiere de una serie de pasos, llamados procesos o subprocesos. Las actividades del proceso de desarrollo se realizan por ingenieros de software, básicamente son cuatro actividades fundamentales en el proceso de desarrollo de software: la especificación, el diseño e implementación, la validación y la evolución del software. Estas actividades generan una gran cantidad de información y conocimiento, lo cual ha motivado en la ingeniería de software a incluir un subproceso para la Gestión del Conocimiento. Este subproceso de gestión del conocimiento de la organización tiene como objetivo entre otros, mantener disponible y administrar la base de conocimientos, que contiene la información y los productos generados por la organización. Hoy en día las empresas que desarrollan software gestionan el conocimiento generado en todas las actividades del proceso de desarrollo de software, algunas muy formalmente, aunque no

todas gestionan el conocimiento de acuerdo a las 4 fases de creación del conocimiento (Obtención, uso/explotación, creación/descubrimiento y difusión del conocimiento). La fase que menos importancia tiene en las empresas, es la fase de creación y descubrimiento de conocimiento, ésta es de gran importancia dentro del desarrollo de un producto de Software, ayuda a la entrega oportuna del producto, presenta información de resolución de problemas, lecciones aprendidas necesarias para nuevos integrantes de los equipos de desarrollo, entre algunas. Para descubrir conocimiento se tiene que analizar una gran cantidad de información, lo que consume tiempo a una persona o grupo de personas, dicha información va aumentando conforme pasa el tiempo, lo que complica el uso de dicha información, esto reditúa en pérdida de tiempo, esfuerzo, recursos humanos, dentro de la organización.

Lo expuesto anteriormente indica que es necesario tener un sistema que permita la gestión de conocimiento para que apoye en este tipo de procesos. Por tal motivo, en el Instituto Tecnológico de Orizaba se desarrolla la herramienta HACOPDS (Herramienta de Administración de Conocimiento para el Proceso de Desarrollo de Software) que permite llevar a cabo la administración del conocimiento orientado a procesos de desarrollo de software.

Este trabajo presenta los avances en el diseño e implementación del módulo de descubrimiento de conocimiento y su visualización, su diseño plantea el uso de tecnologías de la Web Semántica. Este módulo se integrará a la herramienta HACOPDS lo que ha permitido alcanzar el objetivo de la fase de creación y descubrimiento de conocimiento en la teoría de Gestión de conocimiento.

2 Gestión del Conocimiento en Ingeniería de Software.

El Conocimiento según la Real Academia Española [1] se define como *acción y efecto de conocer*, y conocer como *averiguar por el ejercicio de las facultades intelectuales la naturaleza, cualidades y relaciones de las cosas*. El conocimiento se subdivide como conocimiento explícito (es decir artefactos, documentos, reportes y el producto de SW a desarrollar) y conocimiento implícito o tácito (es decir las experiencias obtenidas a lo largo de uno o varios desarrollos de productos de Software).

Thomas Davenport en [2] define la Gestión de Conocimiento como *un método para simplificar el proceso de Compartir, Distribuir, Capturar y entender el conocimiento en una Compañía*. El conocimiento se encuentra implícito en los procesos, y por ende en el desarrollo de un producto de software, es por eso que el conocimiento que se genera en el proceso, se tiene que gestionar, para obtener el mayor aprovechamiento. La Gestión del Conocimiento dentro del proceso de desarrollo del software es importante, dado que ayuda a entender el dominio de la aplicación, proporciona a los desarrolladores información útil para futuros proyectos, permite a desarrolladores novatos entender hasta qué punto de la aplicación se está trabajando y proporciona apoyo a diferentes miembros del equipo de desarrollo a visualizar el conocimiento generado de otras actividades. La figura 1, describe un modelo para sistemas de Gestión del Conocimiento comúnmente empleado en las empresas y la figura 2 muestra las fases del ciclo de creación de conocimiento, de acuerdo con Davenport en [2].

1. **La fase de obtención del conocimiento.** Conocimiento que se genera a través de los motores de búsqueda o de cualquier otra herramienta de filtrado y selección.
2. **Uso/Explotación del Conocimiento.** El conocimiento se utiliza como una parte del proceso de trabajo.

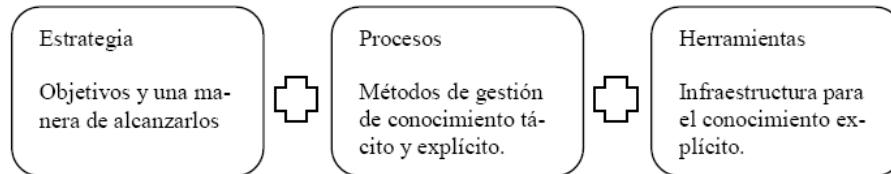


Figura. 1. Un modelo de componentes de un sistema de Gestión del Conocimiento.

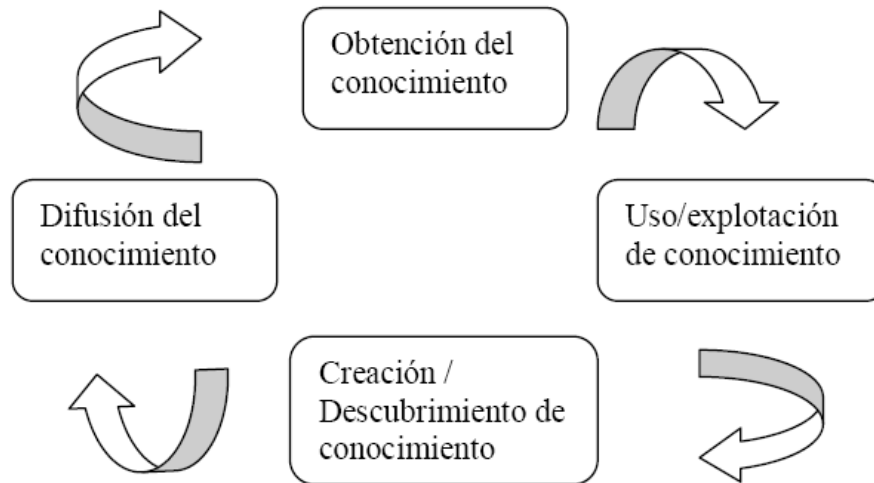


Figura. 2. Ciclo de creación de conocimiento.

3. **Creación/Descubrimiento de Conocimiento.** Permite la obtención de nuevo conocimiento, bien a partir de la creación directa del mismo, o bien a través de su descubrimiento.
4. **Difusión del Conocimiento.** El conocimiento se difunde entre los trabajadores para el aprovechamiento del mismo. Este conocimiento funge como punto de partida para la creación de nuevo, reiniciándose así el ciclo.

Como se mencionó previamente, en el ciclo de desarrollo de software se realizan diversas actividades en donde se tienen algunos problemas, uno de ellos es la falta de documentación que influye en la transferencia de conocimiento [3]. Otro problema es la rotación de personal: empleados nuevos no cuentan con un repositorio de conocimientos que les ayuden a resolver los problemas actuales en base a problemas afrontados en el pasado.

Lo anterior es una pequeña parte del porqué la administración del conocimiento ha sido utilizada cada vez más en las organizaciones de desarrollo de software [4], y su incorporación ha mostrado que tiene beneficios como la reducción de costos, mejoras en la calidad de los procesos y productos [5]. Como se menciona en [6], “existen diversos trabajos centrados en aplicar la administración del conocimiento en organizaciones de desarrollo de software, la mayoría de los cuales se enfocan en la reutilización de experiencias, tales como mejores prácticas o lecciones aprendidas, con el fin de mejorar la calidad de los nuevos productos, del proceso de desarrollo, o para facilitar la reutilización de artefactos de software. Comúnmente, este tipo de trabajos se enfoca en la captura de conocimiento relacionado con el desarrollo de nuevos productos, dejando de lado el mantenimiento del software actual”.

Un claro ejemplo que contrasta con lo anterior, es la metodología KoFI [6], esta metodología se basa en la “identificación de flujos de conocimiento por medio del estudio y modelado de los procesos de una organización, con enfoque en la identificación del conocimiento requerido y generado en las principales actividades del proceso, así como las fuentes donde éste es almacenado y obtenido” [7]. De esta manera, es posible para las personas relacionadas con los procesos, contestar a preguntas como ¿Qué conocimiento es importante para las personas que realizan los procesos? ¿Cuáles son las fuentes que proveen de conocimiento a un proceso? ¿Dónde se encuentra almacenado el conocimiento de un proceso?, entre otras. La metodología KoFI se compone de cuatro etapas: 1) Identificación de fuentes de conocimiento. 2) Identificación de tipos/temas de conocimiento. 3) Identificación de flujos de conocimiento. 4) Identificación de problemas que afecten el flujo de conocimiento. Esta metodología ha sido aplicada en diversos dominios [8], [9], [10] y [11].

Asimismo se han desarrollado herramientas para apoyar la gestión del conocimiento de software, una de ellas es la herramienta SKM [12] la cual está basada en la Web, usa ontologías y gestiona la actividad de diseño dentro del proceso de desarrollo de software. Esta herramienta resuelve una problemática en la actividad de diseño del proceso de desarrollo de software, registra las decisiones de diseño y permite visualizarlas. En HACOPDS se registran las decisiones en las cuatro actividades básicas del desarrollo de Software, la figura 3 visualiza la arquitectura lógica de la herramienta HACOPDS, esta implementa la gestión de usuarios, estos están relacionados con uno o más proyectos en los que gestionaran el conocimiento. Por otro lado la gestión de proyectos y la gestión de artefactos están estrechamente relacionados, ya que cada proyecto que se genere tendrá artefactos que se van a ir generando en cada fase del proceso de desarrollo de Software, las fases que se tomaron en cuenta en HACOPDS son *Requisitos*, *Análisis*, *Diseño*, *Implementación* y *Pruebas*, la herramienta crea un directorio de cada proyecto adjuntándosele los artefactos representativos y clasificados por las fases en las que pertenecen, en el mejor escenario la gestión de experiencias, se conforma de todos los problemas que se le presentan a los desarrolladores en cada fase del proceso de desarrollo.

3 Modelo de Conocimiento.

El modelo de gestión del conocimiento de Nonaka y Takeuchi [13], se fundamenta en el proceso de transformación del conocimiento, es decir, en las diferentes

fases por las que éste pasa en su transformación para ser utilizable por la organización. Desde este enfoque, si tenemos en cuenta la clase de conocimiento, el punto de vista de Nonaka trata la accesibilidad del conocimiento.

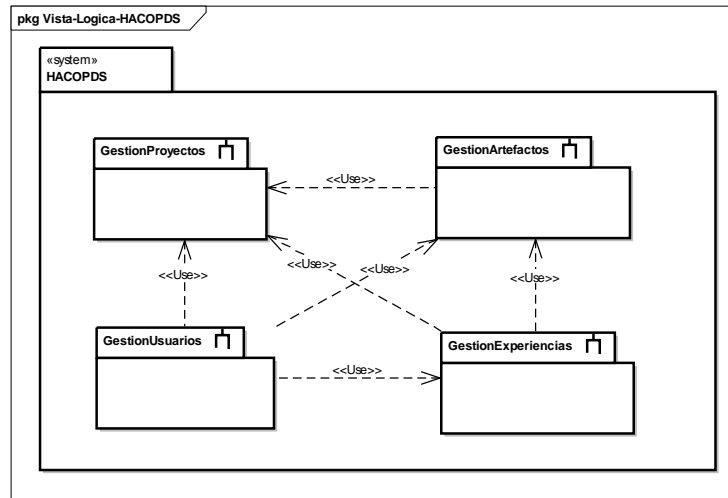


Figura. 3. Arquitectura lógica de la herramienta HACOPDS.

Por lo tanto, éste puede ser de dos tipos: tácito y explícito. Es un proceso de interacción entre el conocimiento tácito y el explícito de naturaleza dinámica y continua. Se conforma mediante una espiral, que define la permanente transformación ontológica del conocimiento, desarrollada en cuatro fases. En la figura 4 se muestra dicho proceso.

Socialización (Conocimiento Acordado): es el proceso de adquirir el conocimiento tácito a través de la compartición de las experiencias mediante: exposiciones orales, reuniones, documentos, manuales y tradiciones, que añade el conocimiento nuevo o individual a la base colectiva que posee la organización.

Exteriorización (Conocimiento Conceptual): es el proceso de convertir conocimiento tácito en conceptos explícitos, requiere hacer tangible dicho conocimiento, integrándolo en la cultura de la organización (por ejemplo, en bases de datos de conocimiento).

Combinación (Conocimiento Sistémico): es el proceso de generar conocimiento explícito al reunir conocimiento explícito proveniente de diversas fuentes, y que se puede categorizar, confrontar y clasificar para añadirse a las mencionadas bases de datos de conocimiento.

Interiorización (Conocimiento Operacional): es el proceso de incorporación del conocimiento explícito al conocimiento tácito individual (o grupal).

El módulo de descubrimiento de conocimiento y su visualización a la herramienta HACOPDS, considera la implementación de un modelo se diseñó una ontología, la cual es una representación del conocimiento de un dominio, para el diseño se usó la metodología Methontology [14] para formalizar dicho conocimiento, la ontología es una taxonomía que contiene los conceptos del dominio, además se definen las re-

laciones y atributos. En la figura 5 se muestra de manera parcial la ontología, ésta ha sido implementada en Protégé [15].

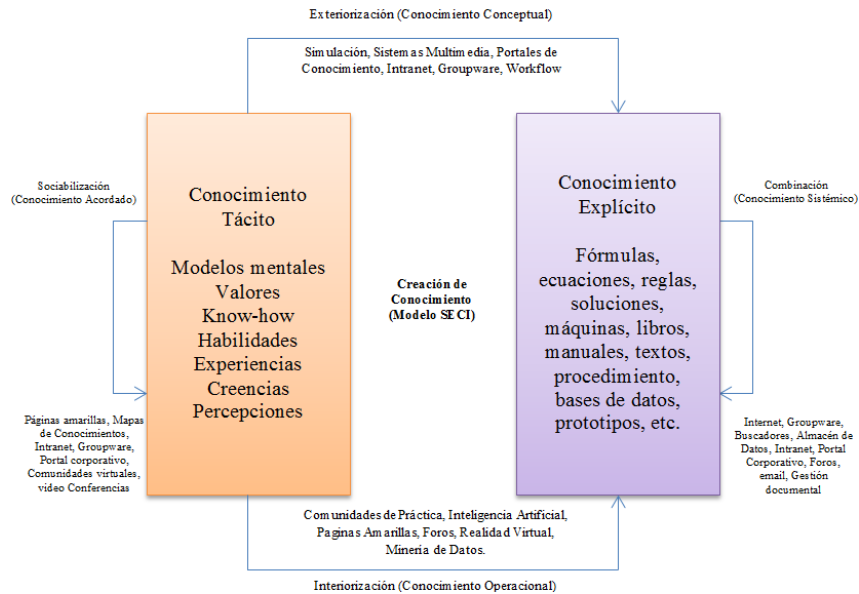


Figura. 4. Creación de conocimiento [13].

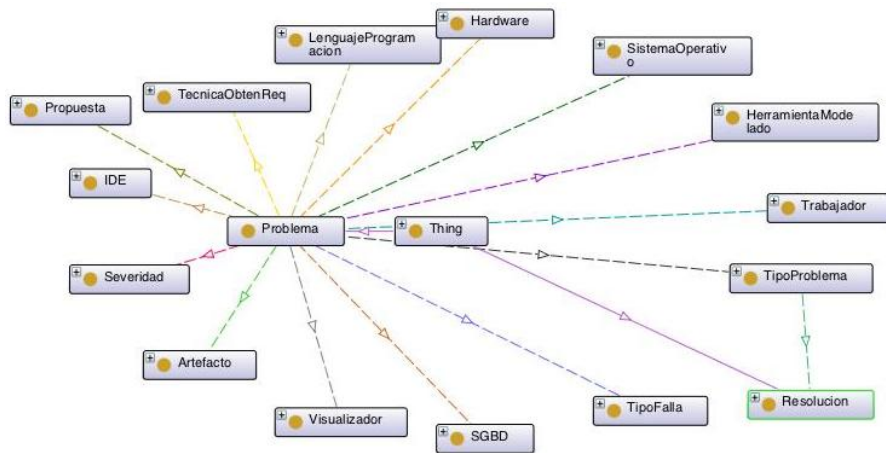


Figura. 5. Vista parcial de la ontología.

Como se puede apreciar en la figura 4, el dominio que se ha considerado es acerca de los problemas que se presentan en las diversas etapas del desarrollo de software. De tal manera que se puede apreciar la relación del concepto *Problema* con diversos conceptos que participan en el desarrollo de algún producto de software, por ejemplo el sistema operativo, el lenguaje de programación, tipo de falla, entre

algunas. Estos conceptos contienen instancias que se usarán para determinar *el tipo de problema* que se presenta.

4 Aplicación del Conocimiento.

La aplicación del conocimiento se refiere al proceso de utilizar el conocimiento en la práctica para alcanzar un desempeño competente. El objetivo final de la gestión de conocimiento es el de utilizar el conocimiento para beneficio de la organización. Solamente el uso productivo del conocimiento traducirá los activos intangibles en resultado tangibles. Sin una efectiva utilización, todos los esfuerzos en el desarrollo, almacenamiento y transferencia del conocimiento son en vano.

El uso de la tecnología en la aplicación del conocimiento tiene una influencia positiva al facilitar la captura y accesibilidad del conocimiento organizacional. La tecnología también puede incrementar la cantidad de memoria organizacional disponible, así como permitir la aplicación del conocimiento a través del tiempo y espacio y agilizar la aplicación del conocimiento a través de la automatización.

En nuestro trabajo la aplicación del conocimiento está asociada con el descubrimiento y la visualización del conocimiento acerca de los problemas que se presentan en las fases de desarrollo de software. De tal manera que nuestra primera pregunta es ¿cómo obtenemos conocimiento?, la respuesta a esta pregunta en particular para nuestro caso de estudio es a través del uso de la inferencia a partir de nuestro modelo de conocimiento y con el uso de mecanismos de razonamiento. En la figura 6, se visualiza el diagrama de bloques que muestra el procedimiento general para la obtención del conocimiento, la herramienta HACOPDS cuenta con una base de datos relacional y adicionalmente se tiene una ontología, es por ello que se realizó un mapeo de la base de datos (donde se obtienen el conocimiento tácito) a la ontología como instancias y poder realizar procesos de inferencia como también consultas.

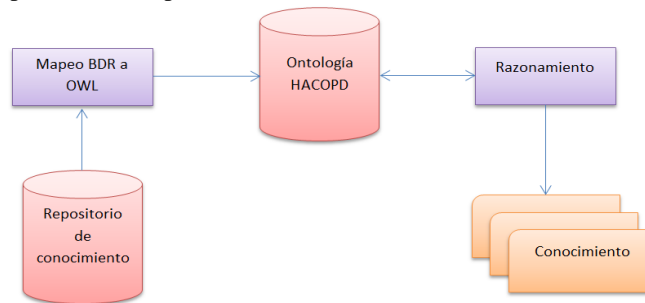


Figura. 6. Diagrama de bloques.

A través de la herramienta se introducen las características del problema (ver figura 7) que se ha presentado y se realiza un proceso de razonamiento que indicará si es un nuevo problema o uno ya existente. Este proceso inferencial se apoya en el uso de una base de conocimiento de reglas, estas reglas modelan los diversos problemas de los que se tiene un conocimiento previo. La figura 8 muestra parcialmente esta base de reglas.

Figura. 7. Interfaz de usuario para la consulta de experiencias.

```
@prefix prob: http://www.semanticweb.org/ontologies/2012/4/Ontology1338231475747.owl#

[problem1: (?A prob:tieneFalla prob:Error_de_Compilacion) , (?A prob:tieneLenguaje prob:Java) ,
(?A prob:tieneIDE prob:NetBeans) , (?A prob:tieneSistema prob:Windows) ,
(?A prob:tieneSGBD prob:PostgreSQL) , (?A prob:tieneHardware prob:PC) ,
(?A prob:tieneVisualizador prob:No_Aplica) , (?A prob:tieneSeveridad prob:Normal)

-> (prob:Problem1 rdf:type prob:TipoProblema) , (?A prob:tieneTipoProblema prob:Problem1),
print('El tipo del problema es Problem1')]

[solucion_problema1: (?A prob:tieneTipoProblema prob:Problem1), (prob:Problem1 prob:tieneResolucion ?B)
-> (?A prob:tieneResolucion ?B), print('La solucion del nuevo problema es: ',?B)]
```

Figura. 8. Reglas.

Dado que la ontología está representada en el lenguaje OWL [16], se puede aprovechar que el subsistema de inferencia de Jena 2 [17] está diseñado para permitir un rango de máquinas de inferencia o razonadores que pueden ser conectados a Jena. Tales máquinas se emplean para derivar afirmaciones adicionales RDF [18], las cuales se obtienen desde alguna base RDF junto con cualquier información adicional de la ontología, así como los axiomas y reglas asociadas con el razonador. EL uso primario de este mecanismo es soportar el uso de lenguajes como RDFS y OWL, los cuales permiten hechos adicionales a ser inferidos desde las instancias de datos y descripciones de clases. Aquí se usa el término inferencia para referirse al proceso abstracto de derivar información adicional y el término razonador para referirse a un objeto de código específico que ejecuta esta tarea. En nuestro caso de estudio se usa la plataforma Jena para llevar a cabo el proceso de inferencia para descubrir nuevo conocimiento. La figura 9 muestra parcialmente un programa para llevar a cabo el razonamiento con el conjunto de reglas que describen los problemas y el aprendizaje de estos nuevos problemas.

En la herramienta HACOPDS se realizarán diferentes tipos de consulta a la ontología utilizando el lenguaje de consulta SPARQL, este realiza consulta sobre documentos RDF, la estructura de estos documentos está formada por un conjunto de tripletas (las tripletas están compuestas por un sujeto, predicado y objeto), para la creación del documento que contiene la ontología de la herramienta HACOPDS se

utilizo el lenguaje OWL, este por dentro esta estructurado como un documento RDF, es decir contiene un conjunto de tripletas utilizadas para realizar consultas sobre esta. En la figura 10, se muestra un ejemplo de consulta de SPARQL utilizando el marco de trabajo jena. Existen lenguajes de consulta para documentos OWL como OWL-QL, para este caso, no se utilizó de acuerdo a [19], es débil al manejo de operaciones aritméticas y de comparación.

```
public class UsoReglasHACOPDS {
    public static void main(String args[] ) {
        Model datos = ModelLoader.loadModel("C:/HACOPDS.owl");
        List reglas = Rule.rulesFromURL("C:/reglasProblemas.txt");
        GenericRuleReasoner razonador = new GenericRuleReasoner(reglas);
        razonador.setMode(GenericRuleReasoner.HYBRID);
        razonador.setOWLTranslation(true);
        razonador.setTransitiveClosureCaching(true);
        InfModel inf = ModelFactory.createInfModel(razonador, datos);
        inf.prepare();

        try{
            String outputFileName = "C:/HACOPDS.owl";
            FileOutputStream salida= new FileOutputStream(outputFileName);
            inf.write(salida, "RDF/XML");
        }catch(Exception e){
            System.out.println("Error "+e);
        }
    }
}
```

Figura. 9. Usando Jena para el razonamiento.

```
public static boolean verifica(OntModel modelo_mapeo,String titulo, String descripcion,
    String tipoFalla, String sistema,String hardware, String ide, String gestor,
    String visualizador, String severidad){

    String queryString =
    "PREFIX prob: <http://www.semanticweb.org/ontologies/2012/4/Ontology1338231475747.owl#>" +
    "prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>" +
    "SELECT ?z " +
    "WHERE { " +
    "?z prob:tieneFalla prob:"+tipoFalla+"." +
    "?z prob:tieneSistema prob:"+sistema+"." +
    "?z prob:tieneHardware prob:"+hardware+"." +
    "?z prob:tieneIDE prob:"+ide+"." +
    "?z prob:tieneSGBD prob:"+gestor+"." +
    "?z prob:tieneVisualizador prob:"+visualizador+"." +
    "?z prob:tieneSeveridad prob:"+severidad+"." +
    "}" ;

    com.hp.hpl.jena.query.Query query = QueryFactory.create(queryString);
    QueryExecution qe = QueryExecutionFactory.create(query, modelo_mapeo);
    ResultSet results = qe.execSelect();
    ResultSetFormatter.out(System.out, results, query);
    qe.close();

    if(results.getRowNumber()==0){
        return false;
    }else{
        return true;
    }
}
```

Figura. 10. Usando el lenguaje de consulta SPARQL dentro de la herramienta HACOPDS.

5 Conclusiones.

Una herramienta de gestión del conocimiento debe de emplear las 4 fases que menciona el ciclo de creación del conocimiento, la herramienta HACOPDS describe 3fases del proceso de creación del conocimiento, como consecuencia se creó el módulo encargado de descubrir el conocimiento para cubrir las 4 fases.

En el proceso de desarrollo de Software, los trabajadores o integrantes del equipo conforme van trabajando con el desarrollo de la herramienta y documentación se les presentan problemas que anteriormente ya se les ha dado solución, este artículo muestra una forma de darle al trabajador una posible solución, indicando los criterios y argumentos que fundamentaron la resolución del problema en el pasado, como consecuencia se reducirá el tiempo de desarrollo.

Para la obtención del conocimiento se utilizarán mecanismos de inferencia, estos mecanismos son una forma de encontrar y darle solución a estos problemas comunes. Cada problema tiene asociado una regla que permitirá darle solución a un problema con las mismas características del problema, el uso de reglas es una de las mejores alternativas para darle solución a problemas comunes ya que se aproximan más que a diferentes técnicas de similitud. La desventaja del uso de reglas es que para cada nuevo problema con características diferentes a los problemas ya almacenados que tienen asociada una regla, se debe realizar una nueva regla. Existen otras alternativas de solución que ayudaran a encontrar problemas que tengan o no asociada una regla y es el uso de métricas de similitud, como su nombre lo indica muestran similitud pero podrían ayudar a dar solución a un nuevo problema que no tenga asociada ninguna regla, esto se muestra como trabajo a futuro dentro del módulo de descubrimiento de conocimiento.

Referencias

1. Real Academia Española, <http://www.rae.es/rae.html>, (visitado el 01 de Marzo de 2011).
2. Davenport, T.; Prusak, L.: *Working Knowledge: How Organizations Manage What They Know*. Harvard Business School Press (1998)
3. Lethbridge, T. C., Singer, J., y Forward, A.: *How software engineers use documentation: The state of the practice*. IEEE Software, 20(6): 35- 39 p. (2003)
4. Aurum, A., Jeffery, R., Wohlin, C., y Handzic, M. (Eds.): *Managing software engineering knowledge*. Springer. Berlin, Germany. 406 p.(2003)
5. Rus, I., y Lindvall, M: *Knowledge management in software engineering*. IEEE Software, 19(3): 26-38 p. (2002)
6. Rodríguez, O.: *Metodología para el diseño de sistemas de administración del conocimiento: su aplicación en mantenimiento de software*, Centro de Investigación Científica y de Educación Superior de Ensenada, Tesis de Doctorado, (2007)
7. Rodríguez Elías, O., Martínez García, A., Favela Jara, J., Viscaíno Barceló, A., Piattini Velthuis, M.: *Facilitando el acceso a las fuentes de información de un grupo de mantenimiento de software por medio de un mapa de conocimiento*. (2005)
8. Rodríguez-Elias, Oscar M.; Morán, Alberto L.; Lavandera, Jaqueline I.; Vizcaíno, Aurora; Soto, Juan Pablo.: *Applying the KoFI methodology to Improve Knowledge Flow in a Manufacturing Process*. In: Proc. of the 3rd International conference on Software

- and Data Technologies (ICSOFTE'2008), Porto (Portugal), 5-8 de Julio, pp. 309-314, ISBN: 978-989-8111-51-7. (2008)
9. Rodríguez-Elias, Oscar M.; Morán, Alberto L.; Lavandera, Jaqueline I.; Vizcaíno, Aurora.: Analyzing Manufacturing Process Knowledge Flow with KoFI", In: Proc. of the 20th International Conference on Software Engineering and Knowledge Engineering (SEKE'2008), San Francisco Bay, USA, 1-3 de Julio. Pp. 428-433, ISBN: 1-891706-22-5. (2008)
 10. Rodríguez Elias, Oscar M.; Vizcaíno Barceló, Aurora; Martínez García, Ana Isabel, Favella Vara, Jesús, PiattiniVelhuis, Mario G.: Aplicando KoFI: Un Caso de Estudio", In: Calidad de Producto y Proceso Software, Calero, Coral; Moraga, Ma. Ángeles; Piattini, Mario (Eds.), Ra- Ma, p. 483-507, (ISBN: 978-84-7897-961-5).(2009)
 11. Zavala Díaz A., Rose GómezC.E.,Rodríguez Elías O. M., Serna Encinas M.T.: Modelo de Conocimiento para un Sistema de Gestión de Conocimiento de una División de Estudios de Posgrado e Investigación Tecnológica, Avances de Investigación en Ingeniería en el Estado de Sonora, Volumen 1, ISBN 978-0-578-09609-4, (2011)
 12. Quiroga J.A.:Gestión de conocimiento en grupos de desarrollo de software, Universidad de los Andes, Bogotá, Colombia. Obtenido Marzo 05, 2011 de <http://paradigma.uniandes.edu.co/media/articulos/jor-quir-1.pdf>. (2011)
 13. Nonaka I., Takeuchi H.: La Organización creadora del Conocimiento. Oxford (1999)
 14. Bernal J.A.: Modelo ontológico para la administración del currículo del personal académico de la Universidad de Sonora, Maestría en Ciencias de la Computación, Instituto Tecnológico de Hermosillo. (2009)
 15. Protégé, <http://protege.stanford.edu/>, (visitado el 8 de Noviembre de 2011).
 16. OWL, <http://www.w3.org/TR/owl-features/>, (visitado el 8 de Noviembre de 2011).
 17. Jena, <http://jena.sourceforge.net/>, (visitado el 8 de Noviembre de 2011).
 18. RDF, http://www.w3.org/standards/techs/rdf#w3c_all, (visitado el 8 de Noviembre de 2011).
 19. Estudio comparativo de lenguajes para la búsqueda y recuperación de información semántica, <http://www.sociedadelainformacion.com/15/lenguajes.pdf>, (visitado el 01 de septiembre 2012).