Memoria Organizacional Basada en Ontologías y Casos para un Sistema de Recomendación en Aseguramiento de Calidad

María de los Ángeles Martín

Grupo de Investigación y Desarrollo en Ingeniería de Software (GIDIS)

Calle 9 y 110, (6360) General Pico, La Pampa, Argentina

martinma@ing.unlpam.edu.ar

Presentada a la Facultad de Informática de la UNLP como parte de los requisitos para la obtención del título de
Doctora en Ciencias Informáticas

Director : *Dr.* Luis **OLSINA**

Co-Director : *Dr.* Gustavo **ROSSI**

La Plata, Octubre de 2010

Facultad de Informática
Universidad Nacional de La Plata - Argentina

Resumen

Administrar el conocimiento organizacional representa una ventaja clave en las empresas de software. Para que dicha administración sea eficiente es importante contar con una Memoria Organizacional que de soporte a la captura, estructuración, reuso y diseminación del conocimiento creado por sus empleados.

La utilidad de una Memoria Organizacional radica en que el conocimiento que almacena pueda ser fácilmente compartido entre sistemas dentro de la organización o entre organizaciones colaborativas. En esta tesis mostramos cómo se puede implementar una Memoria Organizacional Basada en Ontologías y Casos que administre en forma consistente, automática y distribuída el conocimiento presente en una empresa.

A medida que el área de Gestión del Conocimiento va progresando hacia una disciplina más madura, se van obteniendo importantes resultados de investigación relacionados a Memoria Organizacional, Gestión del Conocimiento y Razonamiento Basado en Casos. Sin embargo, se observa que el desarrollo rápido y caótico de tanta información relacionada a estas áreas, proveniente de fuentes heterogéneas, y la falta de consenso en la terminología utilizada, dificulta su aplicación y reuso de forma eficiente en los procesos de Gestión del Conocimiento. Con el propósito de dar solución a esta problemática, uno de los aportes de esta tesis es proveer una conceptualización del dominio de Memoria Organizacional Basada en Casos (MOBC) que se formaliza en una Ontología para dicho ámbito.

La Ontología de MOBC, combina aspectos de Memoria Organizacional con Razonamiento Basado en Casos para representar cada ítem de conocimiento informal. La estructuración del conocimiento informal en casos, facilita la captura, transferencia, y reuso a través de un procesamiento automático. La ontología propuesta, es usada como base para el diseño y construcción de un sistema de Memoria Organizacional Basada en Casos y un sistema de recomendación para proyectos de medición y evaluación en aseguramiento de calidad, que usa la MOBC como su componente central.

En Ingeniería de Software/Web, el aseguramiento de la calidad y como consecuencia la medición y evaluación, son procesos de soporte claves para el desarrollo y mantenimiento de aplicaciones de software. Por lo tanto la administración en forma consistente y eficiente del conocimiento relacionado a dichos proyectos es muy importante para la toma de decisiones, el reuso y la recomendación en nuevos proyectos, tomando ventaja de las experiencias anteriores, lecciones aprendidas y las buenas prácticas.

Agradecimientos

A Dios, fuente de todo conocimiento y sabiduría, por acompañarme e iluminar mi mente para la realización de este trabajo.

Al Dr. Luis Olsina, mi director, por acompañar mi trabajo con aportes enriquecedores y por el apoyo constante a mi tarea. Y a mi co-director Dr. Gustavo Rossi, por su soporte.

A la Facultad de Ingeniería, de la Universidad Nacional de La Pampa, por el soporte brindado que facilitó en buena parte este estudio.

A mis hijos Alejandro y Natalí, por soportar varias veces mi ausencia y acompañar mi esfuerzo con cariño y comprensión.

A mis colegas del grupo GIDIS_Web, por haber compartido momentos de trabajo y compañerismo, y por alentarme constantemente. Especialmente, a Belén Rivera por la importante contribución con la implementación del Sistema de Recomendación.

Índice

PARTE I: INTRODUCCIÓN

Lista de Figuras

Lista de Tablas

PARTE I

INTRODUCCIÓN

Capítulo 1- Introducción

1.1 Introducción

El conocimiento es un recurso que está convirtiéndose en una ventaja estratégica en materia de competitividad empresarial. Las organizaciones se van dando cuenta de la importancia de identificar qué es lo que saben y de hacer el mejor uso de este conocimiento ([Dogson, 1993], [Ermine, 2000]). El conocimiento organizacional está reconocido como el activo más importante de una empresa, y se han desarrollado numerosos trabajos de investigación para definir cómo adquirirlo, representarlo, retenerlo, administrarlo y transferirlo [Bolloju et al., 2002], [Conklin, 1996], [Mica et al., 2004].

Una organización que aprende es una compañía que constantemente construye estructuras y estrategias tales que incrementen y maximicen el conocimiento organizacional. Posibles clasificaciones de conocimiento organizacional son público / privado, explícito / implícito (o tácito), y formal (sintácticamente y semánticamente estructurado) / informal (no estructurado) [Nonaka et al., 1995]. Una de las metas principales de las estrategias de gestión de conocimiento es hacer explícito el conocimiento implícito de los individuos, tratar de formalizar el conocimiento informal, para permitir su procesamiento automático, y hacer al conocimiento público o privado dependiendo de la estrategia política en diferentes niveles de la organización. El beneficio del conocimiento o experiencia explícito es que se puede almacenar, organizar, y diseminar a terceros sin la participación del creador. El inconveniente que encuentran las organizaciones en este sentido es el considerable esfuerzo que involucra producir el conocimiento explícito y formal.

Para mantener la efectividad y competitividad de la empresa, una organización necesita aprender de las experiencias pasadas y presentes, sacar ventaja de las lecciones aprendidas y formalizar una Memoria Organizacional con el fin de permitir hacer explícito el conocimiento tácito del individuo (y por qué no también el conocimiento tácito de los grupos y la comunidad organizacional). Por lo tanto, en ambientes

colaborativos de trabajo es muy importante contar con una infraestructura robusta de Tecnologías de Información (TI) que permita la adquisición, transmisión y explotación de este conocimiento de una manera explícita y formal.

Las compañías dedicadas al software no son una excepción, son empresas donde las personas continuamente expanden su capacidad para crear los resultados que ellos realmente desean, donde nuevos patrones de pensamiento se nutren, y donde las personas continuamente aprenden de sus experiencias. La administración de este valioso activo intangible debe permitir aprovechar al máximo el conocimiento existente, que normalmente se encuentra distribuido entre los colaboradores de la organización o centros de investigación.

A pesar de la importancia de la Gestión del Conocimiento en una organización, a menudo los gerentes no pueden identificar dónde reside el valor del conocimiento o cómo usarlo como ventaja competitiva (el conocimiento es abstracto, no se ve, no se sabe ni cuánto mide, ni cuánto pesa, es decir, es intangible). Como resultado, el conocimiento organizacional contenido en las personas y las comunidades formadas dentro de la organización raramente está suficientemente detallado para ser valioso y frecuentemente se pierde cuando las personas abandonan la organización, ocasionando la *amnesia organizacional*.

Referido al tratamiento y manejo del conocimiento organizacional se han desarrollado importantes avances dentro del campo multidisciplinar llamado: la Administración del Conocimiento (KM por Knowledge Management) [Ackerman et al., 2000]. Si observamos las tecnologías existentes que dan soporte a la KM, se pueden identificar dos vistas principales [Abecker et al., 2000] que se corresponden a dos dimensiones distintas de la administración del conocimiento que detallamos a continuación:

La vista centrada en el proceso, que entiende la KM principalmente como un proceso de comunicación social que puede mejorarse atendiendo a los aspectos de apoyo al trabajo colaborativo en grupo. Está basado en el principio de que la fuente de conocimiento más importante en una empresa son los empleados y apunta a resolver los problemas de cooperación entre ellos, aplicando un proceso para lograr el compromiso

social de transmitir y compartir el conocimiento de cada uno. Las técnicas básicas para este enfoque intentan estimular la comunicación humana y la colaboración, como son el Trabajo Cooperativo Asistido por Computadora (CSCW), Administración de flujos de trabajo (Workflow Management), entre otras.

La vista centrada en el producto se enfoca en los documentos de conocimiento, su creación, almacenamiento, y reuso en memorias organizacionales apoyadas en tecnologías de la información. Está basada en la idea de explicitar, documentar, y formalizar el conocimiento para tenerlo como un recurso tangible, e intentar presentarle las fuentes de conocimiento correctas en el momento apropiado al usuario. En este caso las técnicas básicas derivan de la Gestión de Documentos, Sistemas basados en Conocimiento y Sistemas de Información, entre otros.

En esta tesis se aborda principalmente la última vista, a saber: la Gestión del Conocimiento centrado en una Memoria Organizacional. La Memoria Organizacional debe proveer mecanismos para el almacenamiento y explotación de todo el conocimiento formal e informal presente en la organización [Conklin, 1996]. El primero surge de los documentos, guías de buenas prácticas, manuales y libros que apoyan a los miembros de la organización a tener un buen desempeño. El conocimiento informal, en cambio, surge de la experiencia y el proceso de creación de los miembros. Aquí se incluyen las ideas, hechos, significados, presunciones, preguntas, puntos de vista y decisiones. Teniendo en cuenta este enfoque, observamos la importancia que ha adquirido la KM incluso desde el punto de vista estrictamente económico, que ha desencadenado la aparición de numerosas herramientas específicas basadas en tecnologías de la información. Estas herramientas proporcionan los mecanismos para la estructuración del conocimiento individual de los empleados hacia el conocimiento colectivo de la comunidad.

Pueden considerarse varias técnicas de TI para implementar una Memoria Organizacional, por ejemplo: los enfoques basados en documentos, basados en reglas, basados en workflow, centrados en Razonamiento Basado en Casos (RBC), basados en agentes inteligentes, basados en ontologías, basados en redes semánticas, memorias organizacionales distribuidas, etc. Cada enfoque tendrá sus ventajas o desventajas de acuerdo al ámbito de aplicación, uso y cultura de la organización.

El aporte de esta tesis, es la formalización de una ontología de Memoria Organizacional basada en casos (MOBC), la definición de una estrategia de administración del conocimiento basada en dicha ontología, y para validar la propuesta, la construcción de un prototipo de KM centrado en la ontología de Memoria Organizacional definida, y su aplicación en la construcción de una herramienta de recomendación para el área Aseguramiento de Calidad en Ingeniería de Software. La motivación es tratar de sacar ventaja de los métodos y tecnologías más potentes y que mejor se adapten (como son ontologías, CBR, Web Semántica, contexto) a un sistema de recomendación en aseguramiento de calidad, cubriendo una necesidad no satisfecha totalmente por las soluciones actuales (como se discutirá en el capítulo 2 respecto al estado del arte).

1.2 Motivación

Son varios los argumentos que motivaron la definición de una estrategia de administración del conocimiento basada casos y en ontologías. A continuación presentamos algunos problemas que se resuelven con la solución planteada en esta tesis:

En primer lugar, la mayoría de los Sistemas de Administración del Conocimiento (KMS) que existen en la actualidad capturan y almacenan el conocimiento en repositorios de documentos como manuales, memorandos, y en sistemas informáticos de archivos de texto, y su transferencia se hace por medio de reuniones de trabajo, cursos de capacitación y lecturas individuales de manuales y guías. Esta forma tradicional de almacenar y transferir este conocimiento, ocasiona grandes pérdidas de tiempo y alta inversión en recursos humanos, ya que no contemplan mecanismos potentes de procesamiento semántico y automático de dicho conocimiento. Una forma de resolver este problema es almacenar el conocimiento en forma estructurada en lo que se denomina Memoria Organizacional basada en casos. La estructuración del conocimiento en casos, le imprime mayor capacidad de procesamiento y facilita su captura, recuperación, transferencia y reuso.

Por otro lado, y considerando que el conocimiento especializado normalmente es generado por personal experto de la organización, e introducido en herramientas que dan apoyo a los procesos de los proyectos de la empresa, es conveniente que los

sistemas de KM se integren a dichas herramientas de manera que permita la explotación de ese conocimiento organizacional en forma pro-activa, favoreciendo potencialmente la productividad y competitividad de sus integrantes. Esto es particularmente aplicable al ámbito de desarrollo software, donde las actividades son intensivas en conocimiento, y donde nuevos patrones de conocimiento se generan a diario. Específicamente en el área de aseguramiento de calidad, la aplicación de un enfoque cuantificable al desarrollo, operación y mantenimiento del software es una tarea compleja que requiere disciplina, estudio y conocimiento de métodos, modelos, métricas e indicadores adecuados para los distintos objetivos de medición y evaluación, con el fin de garantizar la calidad [Martín 2005] [Olsina et al., 2007].

Como se mostrará en el próximo capítulo, muchos KMS actuales, carecen de estándares y descripciones formales, lo cual dificulta la compartición y reuso del conocimiento y la cooperación entre individuos, organizaciones y aplicaciones. Las ontologías pueden resolver potencialmente este problema a través de la definición formal de la semántica de los conceptos del mundo real (ver como ejemplo, la ontología de métricas e indicadores [Martin et al., 2003] [Olsina et al., 2004], desarrollada como base para un marco de medición y evaluación). La solución propuesta en esta tesis está basada en ontologías que operan en dos niveles distintos de abstracción. Por un lado, en el nivel de Memoria Organizacional genérico, se define la ontología de Memoria Organizacional en sí; y por otro lado, para caracterizar las experiencias y lecciones aprendidas de acuerdo al dominio de conocimiento y teniendo en cuenta su contexto [Molina et al., 2007], se necesita asociar a la Memoria Organizacional las ontologías de dominio y contexto respectivamente (ontologías de nivel de dominio). La estructuración de ontologías en niveles, permite instanciar fácilmente el sistema de gestión de conocimiento para distintos ámbitos de aplicación, reusando el mismo núcleo de los servicios de Memoria Organizacional, y adecuándolo a través de ontologías de dominio y de contextos específicos al área de conocimiento.

Debido a que el conocimiento organizacional puede ser un aspecto del sistema de información integral de una compañía, que puede tener subsistemas heterogéneos y distribuidos, es importante que el conocimiento pueda ser compartido y reusado en las distintas dependencias en forma consistente. Últimamente, se ha puesto mucha atención al desarrollo de sistemas que permiten diseminar el conocimiento publicándolo en la

Web. Sin embargo, su tamaño, complejidad y heterogeneidad hace que la distribución del conocimiento, no sea una tarea fácil ni escalable. Afortunadamente, el W3C (World Wide Web Consortium), ha impulsado el desarrollo de la "Web Semántica" [W3Csw 2001], con un conjunto de tecnologías que enriquecen la información disponible en la Web, proveyéndola de su semántica, y recomienda un nuevo paradigma para compartir e intercambiar conocimiento e información. Por lo tanto, para facilitar la disponibilidad y compartición del conocimiento en cualquier momento y lugar en forma distribuida e interoperable, la estrategia de KM propuesta en esta tesis hace uso de tecnologías de Web Semántica.

Por último (pero no menos importante), para que el conocimiento almacenado en una Memoria Organizacional pueda ser utilizado en forma consistente, es necesario que tenga en cuenta el contexto en el que se originó cada ítem de conocimiento. La información de contexto está inserta en las estructuras, rutinas, tecnologías y procedimientos de la organización, y acompaña las experiencias y aprendizaje de los individuos, influenciando la interpretación de experiencias pasadas para su reuso. Este es otro aspecto que también será tenido en cuenta en nuestra propuesta.

1.3 Principales Contribuciones

Las principales contribuciones de esta tesis son:

1.3.1 Especificación de una ontología de Memoria Organizacional Basada en Casos

Aunque los beneficios de usar sistemas de gestión de conocimiento en la empresa son bien conocidos, y la idea de aplicar métodos de Razonamiento Basado en Casos a las lecciones aprendidas y buenas prácticas no son nuevas en el área de KM ([Weber et al., 2001], [Yang et al., 2006]), no hay todavía consenso general en la definición de muchos de los conceptos y terminología usados en las áreas de Memoria Organizacional y Razonamiento Basado en Casos. A pesar de los esfuerzos hechos en nuevos desarrollos de investigación y estándares internacionales durante la última década, la terminología del área de Memoria Organizacional está todavía en fase de discusión, consenso y definición. En particular, hasta lo que nosotros hemos incursionado hasta el

presente, no hay ninguna ontología completamente especificada de Memoria Organizacional Basada en Casos, tal como fue propuesto en [Martín et al., 2008] [Martín et al., 2009].

Para alcanzar este objetivo se ha desarrollado una conceptualización común para una Memoria Organizacional Basada en Casos donde se especifican conceptos, atributos y sus relaciones explícitamente; tal especificación es uno de los pasos centrales para construir una ontología ([Fernández, 1999], [Gómez-Pérez 1996]). Esta ontología [Martín et al., 2009], surge como resultado de un trabajo de investigación sobre conceptos previamente definidos en el estándar australiano relacionado [AS5037], teniendo en cuenta artículos de investigación reconocidos en el área [Aamodt, 1994], [Kemp et al., 2001], [Kolodner, 1993]), y de reuniones entre investigadores de nuestro grupo para acordar la definición de la terminología.

El valor agregado de ontologías para modelar memorias organizacionales ha sido ampliamente discutida en la literatura ([Ale et al., 2007], [Martín et al., 2006]). Permite hacer el conocimiento explícito, compartido, formal, y facilita su procesamiento semántico y automático por computadoras.

1.3.2 Diseño de una Solución de Memoria Organizacional Basada en Ontologías y Casos para la Gestión del Conocimiento.

El Razonamiento Basado en Casos es ampliamente usado en la literatura como una tecnología para construir sistemas de información que den apoyo a la Gestión del Conocimiento [Althoff et al., 2005], donde se usa una representación estructurada para caracterizar los items de conocimiento. Estos items son representados por casos y se guardan en una base de casos que conforman la Memoria Organizacional. Sin embargo, un inconveniente de este enfoque, es que dichos sistemas de CBR se implementan a menudo en forma aislada y cerrada, en el sentido que no se desarrollan favoreciendo la cooperación con otros sistemas y por consiguiente no permiten un fácil intercambio de conocimiento.

Uno de los aportes de esta tesis es el diseño de una Memoria Organizacional distribuida y basada en ontologías, mostrando cómo los sistemas en forma cooperativa

se pueden beneficiar de una representación de conocimiento compartido y consensuado, que implica una interpretación inequívoca de los items de conocimiento (casos). En tal sentido se presenta el diseño modular y de la arquitectura en capas de la MOBC. Además se especifican las tecnologías recomendadas para lograr los objetivos propuestos y detalles de una implementación.

1.3.3 Sistema de Recomendación para Proyectos de Medición y Evaluación.

A través de un prototipo realizado para tal fin y de ejemplos concretos, se muestran los beneficios de la MOBC en la Gestión del Conocimiento (buenas prácticas y lecciones aprendidas). Particularmente se presenta un caso de estudio: esto es, su aplicación a un sistema de recomendación en etapas de decisión de un proyecto de medición y evaluación de calidad de productos software y Web. Además se muestra su flexible integración con herramientas existentes (en este caso la integración al marco de medición y evaluación C-INCAMI y su herramienta C-INCAMI-Tool [Olsina et al., 2007]). Se muestra además, la explotación de las bases de conocimiento por parte de herramientas automáticas y cómo este enfoque puede dar soporte a las características de interoperabilidad, generalidad y procesamiento automático tan importantes a la hora de reusar el conocimiento organizacional.

1.4 Estructura de la Tesis

La presente tesis se organiza de la siguiente manera: En el capítulo 2 se presenta el estado del arte, la problemática actual y las soluciones propuestas. Para una mejor comprensión de esta tesis, en los capítulos 3, 4, y 5 se introducen los principales conceptos de Gestión del Conocimiento y Memoria Organizacional, Razonamiento Basado en Casos y ontologías respectivamente. En el capítulo 6 se define la ontología de Memoria Organizacional Basada en Casos. En el capítulo 7 se presenta el diseño modular y la arquitectura en capas de un Sistema de Memoria Organizacional Basada en Casos (SMOBC) y se analizan sus beneficios. En el capítulo 8 se muestra la aplicación del SMOBC a un problema concreto, la construcción de un sistema de recomendación en proyectos de medición y evaluación en aseguramiento de calidad. Finalmente, en el capítulo 9 se presentan las conclusiones y se analizan las líneas de trabajos futuras.

Capítulo 2- Estado del Arte y Propuesta

2.1Introducción

En los últimos años numerosos trabajos de difusión científica y técnica han realizado importantes aportes de investigación y desarrollo en las áreas de *Memoria Organizacional* ,*Gestión del Conocimiento* y *Razonamiento Basado en Casos (CBR)*.

A pesar del indiscutible valor de los avances logrados por dichas investigaciones, a la hora de aplicar los resultados en el desarrollo de una Memoria Organizacional Basada en Casos y basada en ontologías para el área de Ingeniería de Software, nos encontramos con algunos de los siguientes problemas:

- ✓ No hay acuerdos de amplia aceptación en la información disponible sobre administración del conocimiento y Memoria Organizacional, además no existe una terminología común consensuada para el ámbito de Memoria Organizacional Basada en Casos que permita una fluida comunicación de información entre distintas bases de conocimiento y facilite su explotación entre distintas organizaciones y áreas de una misma organización. Es apropiado destacar que hasta el momento no existe ninguna norma internacional (del tipo de normas ISO) en el área de administración de conocimiento [Ferguson 2006].

- ✓ La mayoría de las estrategias de Gestión del Conocimiento que existen no cumplen con la condición de ser suficientemente genéricas como para que puedan ser aplicadas a distintos dominios facilitando la escalabilidad y adaptabilidad y al mismo tiempo suficientemente estructuradas y formales de manera que faciliten el procesamiento automático del conocimiento y se integren a las herramientas de la organización actuando en forma pro-activa.

- ✓ Específicamente, para el área de Ingeniería de Software, no existe un sistema de recomendación basado en Memoria Organizacional Basada en Casos para el

área de aseguramiento de calidad, que haciendo uso del conocimiento adquirido en experiencias anteriores, dé apoyo a la toma de decisiones en el diseño de proyectos de medición y evaluación.

El estudio, conceptualización y especificación de una estrategia de administración del conocimiento centrada en una Memoria Organizacional Basada en Casos y el sistema de recomendación en aseguramiento de calidad presentados en esta tesis tienen como objetivo dar solución a los problemas planteados anteriormente y cubrir una importante carencia que es la definición de una ontología para el dominio de Memoria Organizacional Basada en Casos.

2.2 Estado del Arte

La propuesta presentada en este trabajo consiste en usar el enfoque de *Razonamiento Basado en Casos* para desarrollar una ontología de *Memoria Organizacional*. El principal objetivo de la ontología propuesta es que sirva como base conceptual consensuada para ser aplicada a una eficiente *Gestión del Conocimiento* en una organización.

Existen muchos trabajos de investigación relacionados a estos tópicos, pero como no todos las integran totalmente, para una clara estructuración del estado del arte se decidió separar el estudio en dos áreas, a saber: Memoria Organizacional y Gestión del Conocimiento por un lado, y Razonamiento Basado en Casos por el otro.

2.2.1 Memoria Organizacional y Gestión del Conocimiento

Los sistemas de administración del conocimiento tienen como objetivo administrar y almacenar el conocimiento organizacional, de manera que después pueda ser utilizado para aprender, resolver problemas y como apoyo en la toma de decisiones [Dogson, 1993]. Estos sistemas soportan la creación, organización, almacenamiento y diseminación del conocimiento dentro de la empresa y para ello utilizan gran variedad de Tecnologías de Información.

Algunos enfoques con soporte tecnológico tales como: Sitios Web en Internet e Intranets, data mining, repositorios genéricos de documentos, foros de discusión y videoconferencia entre otros, dan apoyo a los procesos de captar, almacenar, compartir y distribuir el conocimiento organizacional. El inconveniente de aplicar estos medios en forma aislada, es que al no estar desarrollados específicamente para la Gestión del Conocimiento organizacional en forma integral, dificultan la explotación del mismo en forma automatizada en la difícil tarea de dar soporte a la recomendación en la toma de decisiones. Por tal motivo, y por no ser parte de la solución propuesta, no se analizan estas soluciones en el estado del arte.

Considerando las estrategias y herramientas desarrolladas específicamente para la Gestión del Conocimiento y Memoria Organizacional, se han presentado distintas propuestas y productos tanto en el ámbito académico como en la industria. Entre los ejemplos del área académica podemos citar: *Experience Factory* de Basili et al. [Basili et al., 1994]; *Know More* de Abecker et al. [Abecker et al., 1999]; *Knowledge Management in Software Engineering Environments* de Natali et al [Natali et al., 2002]; *Organizational Knowledge Sharing Networks* de Papoutsakis [Papoutsakis 2009]; *OntoDOM* de Ale [Ale 2009] y *Lessons Learned as Organizational Project Memories* de Abril et al. [Abril et al., 2009]. Como ejemplos de productos comerciales podemos citar el *QuestMap* de Touchstone Consulting, Inc. [Conklin, 2003]; *AskMe Enterprise* de AskMe - Realcom [AskMe, 2009] y *XpertRule Knowledge Builder* de XpertRule Software Ltd. [Xpert, 2009]. A continuación se presenta una breve descripción de dichas propuestas.

2.2.1.1 Experience Factory de Basili et al.

El modelo de la "fábrica de experiencias" [Basili et al., 1994], consiste en la definición de un marco para la administración de experiencias adquiridas en proyectos de desarrollo de software y establece la necesidad de contar con un mecanismo separado, acoplado a la organización de los proyectos, para la gestión de dichas experiencias (ver figura 2.1). De esta manera, los datos producidos como resultado de la ejecución del plan de proyecto son captados por la fábrica de experiencias para su análisis, síntesis y registro en una base de conocimientos, la cual va a dar soporte al negocio cuando se planeen proyectos similares en el futuro.

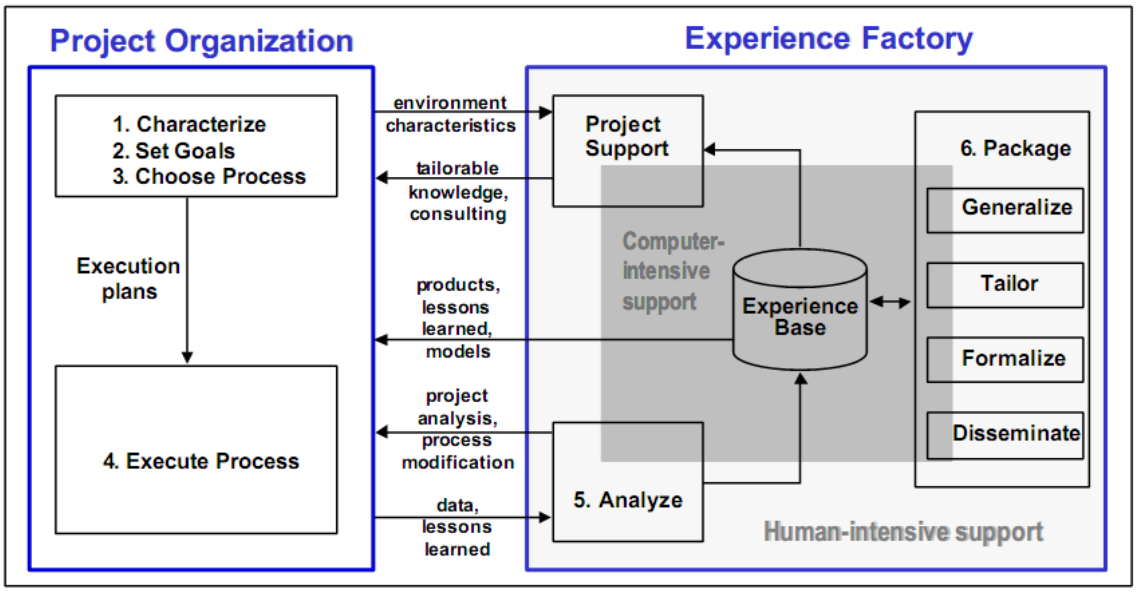


Figura 2.1. Estructura del marco Experience Factory (fuente: Basili et al., 2001)

La implementación física de un "Experience Factory" es un sistema de administración de experiencias, compuesto de contenido, estructura, procesos y herramientas [Basili et al., 2001].

Este enfoque permite dar solución a los siguientes problemas:

- Encontrar a la persona adecuada (especialista) para solucionar un problema.
- Crear modelos basados en la experiencia para mejorar la estimación de costos.
- Identificar patrones.
- Capturar, organizar y diseminar lecciones aprendidas.
- Reusar todas las categorías o tipos de experiencias documentadas, por ejemplo propuestas, planes, etc.

Esta propuesta facilita a los desarrolladores de software el acceso a documentos y modelos adecuados de experiencias pasadas para su reuso. Sin embargo, en la literatura se pueden encontrar algunas críticas. Chau y Maurer señalan que si bien el marco "Experience Factory" establece las actividades de gestión de conocimiento que se necesitan (elicitación, análisis, generalización, empaquetamiento y diseminación) falla

en no prescribir cómo tales actividades deben llevarse a cabo [Chau et al., 2005]. Una observación similar ha sido expresada por Zhu et al., quienes opinan que uno de los problemas en la investigación actual en los repositorios de experiencia es que la mayoría se enfoca en el aspecto tecnológico para construirlos en lugar de cómo realmente capturar y compartir las experiencias [Zhu et al., 2007].

2.2.1.2 Know More de Abecker et al.

Este trabajo incluye el desarrollo de un modelo para el procesamiento del conocimiento en tres capas [Abecker et al., 1998], centrado en una Memoria Organizacional (ver Figura 2.2), y una herramienta basada en dicho modelo [Abecker et al., 1999].

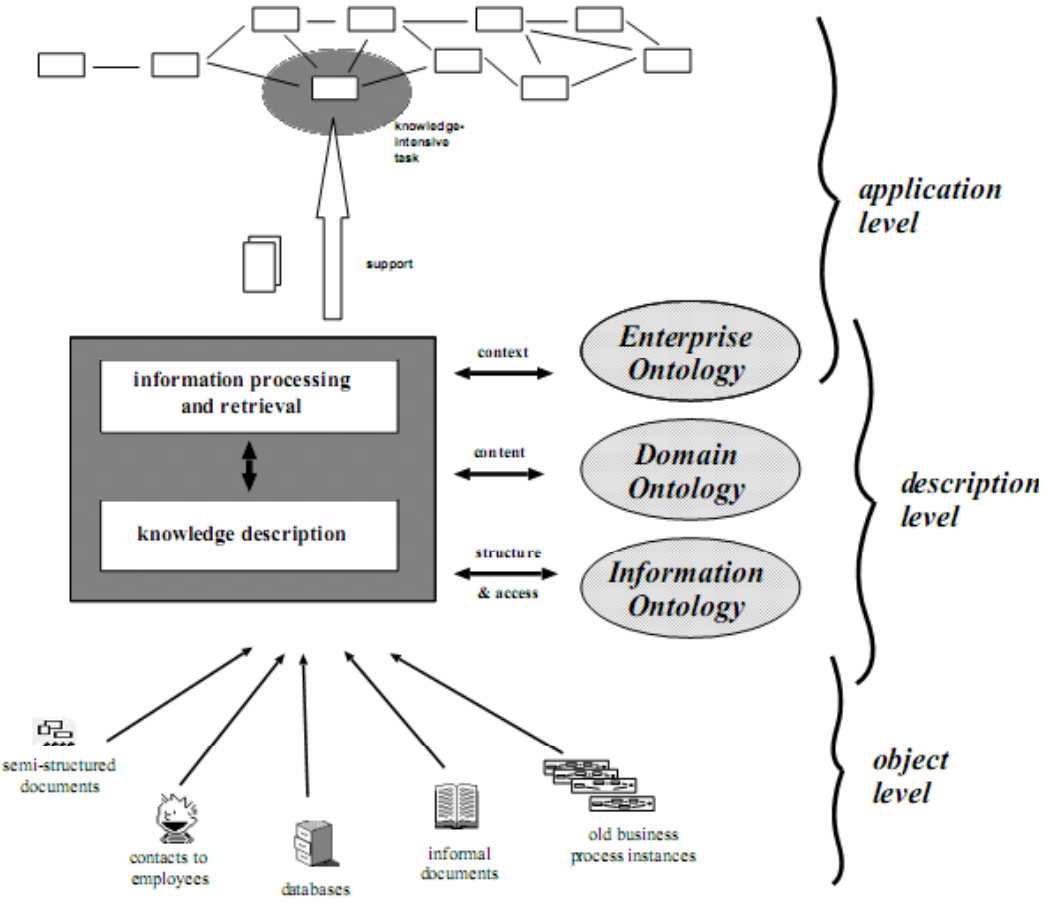


Figura 2.2 Niveles de Memoria Organizacional en el proyecto KnowMore (fuente: Abecker et al., 1999)

El sistema está orientado al usuario que ejecuta tareas intensivas en conocimiento en un proceso organizacional. En general, estas tareas, son complejas e importantes

por naturaleza y para su ejecución se necesita la participación de especialistas que posean tanto habilidades como capacidades, y entre las capacidades se destaca el conocimiento. Con el fin de dar una solución a esta problemática, la propuesta de Abecker et al. modela y ejecuta los procesos y tareas, a un nivel superior al nivel de aplicación. Luego, cuando el usuario reconoce una necesidad de información, se realiza una consulta a la Memoria Organizacional.

La propuesta tiene otros dos niveles, un nivel de objetos, que está caracterizado por una variedad de información captada desde fuentes heterogéneas y el nivel de descripción del conocimiento que transforma los requerimientos del nivel de aplicación en solicitudes al nivel de objetos, utilizando tres tipos de ontologías, a saber: ontología de la empresa (estructura empresarial), ontología del dominio (específico al ámbito de aplicación) y ontología de la información (referente a las fuentes de información y documentos).

Vale destacar que esta propuesta está orientada a la recuperación de documentos y la ubicación de personas expertas en determinado tema específico, no guarda el conocimiento informal derivado de experiencias o lecciones aprendidas en forma estructurada (como podría ser la estructuración en casos), de manera de poder implementar mecanismos de recomendación específica en forma automática.

2.2.1.3 Knowledge Management in Software Engineering Environments de Natali et al.

En este trabajo se propone una infraestructura de administración del conocimiento centrada en ontologías y basada en una Memoria Organizacional para el contexto de desarrollo de software [Natali et al., 2002]. Un sistema de administración del conocimiento se usa para capturar y diseminar el conocimiento y experiencias generadas durante el desarrollo de software.

Según los autores, aunque cada proyecto de desarrollo de software es único en algún sentido, las experiencias similares pueden ayudar para que diseñadores realicen sus actividades eficazmente. Reusando el conocimiento se puede prevenir la repetición de fracasos del pasado y se puede guiar la solución de problemas recurrentes. De esta

manera, un KMS debe integrarse al proceso de desarrollo del software para que sea eficaz.

Actualmente, la infraestructura de gestión de conocimiento propuesta se ha integrado en el ambiente de desarrollo ODE [Falbo et al. 2003]. ODE es un entorno de desarrollo de software que integra un conjunto de herramientas de apoyo a las actividades de Ingeniería de Software a lo largo del ciclo de vida del proyecto.

Esta valiosa investigación se orienta particularmente al dominio específico de la infraestructura ODE (ambiente de desarrollo de software basado en ontologías). El trabajo no especifica cómo estructurar una Memoria Organizacional para distintos dominios de aplicación y cómo se lleva a cabo el proceso de recuperación y diseminación.

2.2.1.4 Organizational Knowledge Sharing Networks de Papoutsakis

Esta propuesta [Papoutsakis, 2009] explora las maneras en que las redes de comunicación del conocimiento dan soporte al flujo de conocimiento organizacional dentro de una empresa, basado en la asunción de que las herramientas que la gente necesita para trabajar con otros, son diferentes a las que necesita para trabajar solo. El autor demuestra cómo el uso de grupos colaborativos de trabajo (groupware) permite el intercambio de conocimiento "en cualquier momento" y "en cualquier lugar" dentro de la organización. Además, hace un análisis de las herramientas de tecnologías de la información que posibilitan a los dirigentes no sólo a incentivar a sus empleados a compartir el conocimiento personalmente, sino también a resguardarlos de manera que puedan ser fácilmente accedidos por otros en el futuro. Presenta una clasificación de los tipos de herramientas groupware que se deberían usar para cada situación (es decir, para el intercambio de conocimiento en el mismo momento y lugar; mismo momento y distinto lugar, distinto momento pero mismo lugar y distinto momento y lugar).

El trabajo concluye que si se usan las herramientas groupware adecuadas, las redes de comunicación del conocimiento juegan un rol importante en la preservación de la Memoria Organizacional de la empresa. Si bien el autor afirma que para sacar

provecho del compartir el conocimiento se requiere usar más de una tecnología, no presenta una forma o modelo de cómo integrar las herramientas que propone.

2.2.1.5 OntoDOM de Ale

Una tesis doctoral reciente de Ale [Ale, 2009] propone un modelo conceptual de Gestión del Conocimiento organizacional, como una estrategia de administración del conocimiento basada en ontologías. Este enfoque utiliza un mecanismo de pregunta-respuesta que usa la interpretación del lenguaje natural para la recuperación del conocimiento (ver Fig. 2.3).

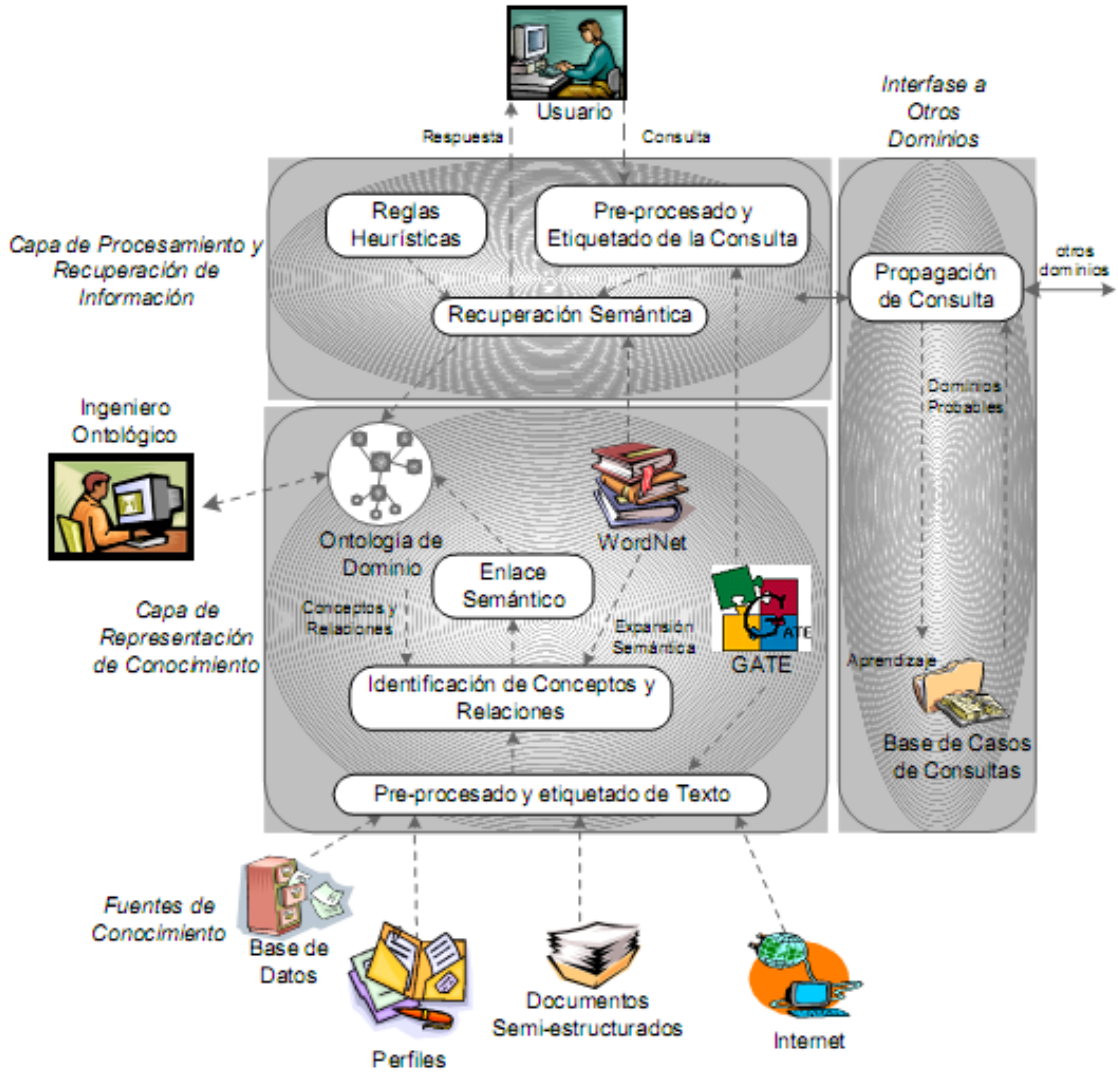


Figura 2.3 Modelo de representación del conocimiento y recuperación de información (fuente: Ale 2009)

El objetivo global de la propuesta es desarrollar un sistema de información de Memoria Organizacional para la representación del conocimiento de dominio y la

semántica ontológica contenida en consultas en lenguaje natural, y para la búsqueda basada en textos en una Memoria Organizacional.

Onto-DOM apunta a:

- El desarrollo de un marco para establecer una red de conocimiento organizacional formal.
- El desarrollo de métodos para el análisis semántico basado en ontologías de las fuentes de conocimiento. El principal objetivo es la representación homogénea de fuentes de conocimiento diversas en función de conceptos ontológicos.
- El desarrollo de metodologías para el procesamiento de la consulta basada en ontologías que compara de una forma eficiente una descripción formal interna de la consulta con la representación ontológica de las fuentes de conocimiento.

El objetivo de esta investigación es resolver el problema de manejar fuentes de conocimiento heterogéneas distribuidas basándose en ontologías para homogeneizar el etiquetado de las mismas y recuperar los documentos analizando semánticamente consultas en lenguaje natural.

Un prototipo construido sobre este modelo, *OntoDOM* (Ontology-based Distributed Organizational Memory) apunta a contribuir al desarrollo de soluciones generales para el tratamiento de consultas en lenguaje natural sobre objetos de conocimiento con descripciones textuales. El prototipo provee una funcionalidad de representación, consulta y recuperación basada en contenido, yendo mas allá del reconocimiento superficial de palabras claves típica en mucho de los motores de búsqueda actuales, aunque no realiza un análisis semántico completo de los textos fuente. Cabe destacar que esta propuesta está orientada a la recuperación de documentos de texto conteniendo conocimiento organizacional.

2.2.1.6 Lessons Learned de Abril et al.

Este trabajo sugiere establecer en la organización un enfoque de investigación para capturar lecciones aprendidas de sus proyectos. Según los autores [Abril et al., 2009], dada la naturaleza temporal de los proyectos, un manejo no apropiado de las lecciones aprendidas constituye un riesgo para los proyectos presentes y futuros. El trabajo argumenta que la investigación de cada caso de estudio es lo apropiado para el desarrollo de lecciones aprendidas y que se puede usar sobre ellas una metodología inductiva para generar hipótesis. Estas hipótesis son validadas a través de un análisis de cómo la lección aprendida se ajusta (es apropiada) a los problemas de negocio relacionados.

Para la investigación y captura de las lecciones aprendidas por parte del gerente de proyecto, se sugiere una metodología de seis pasos en la que se recomiendan plantillas para documentarlas.

Si bien los autores proponen una valiosa y consistente metodología para desarrollar y validar lecciones aprendidas, éstas son administradas en forma manual a través de plantillas textuales y no presentan ningún modelo de desarrollo de Memoria Organizacional que contemple el almacenamiento, compartición, reuso u explotación de dichas lecciones aprendidas en forma automática.

2.2.1.7 QuestMap de Touchstone Consulting, Inc.

Se trata de una herramienta que intenta capturar el conocimiento informal e implícito y convertirlo en explícito [Conklin, 2003]. El componente fundamental de esta propuesta es el uso de un sistema de interfaces de usuario que capturan preguntas e ideas claves durante las reuniones (presenciales o virtuales) creando un entendimiento compartido en el equipo de trabajo. El sistema posee los siguientes componentes:

- De Captura: el usuario apuntador (o intermediario) escribe sobre un cuadro (browser) las ideas claves.
- De estructura: se utiliza un modelo de conversión IBIS (Issue Based Information System) de Conklin & Begeman [Conklin et al., 1987], el

cual clasifica cualquier conversación en tres elementos simples: preguntas, ideas y argumentos (pros y contras).

- De interface de usuario: un sistema gráfico de hipertexto, que da soporte visual al árbol "*Questmap"* (ver figura 2.4)
- De base de datos: almacena lo mapas de discusión previos, posibilitando la búsqueda y navegación dentro de esta base de conocimientos informal.

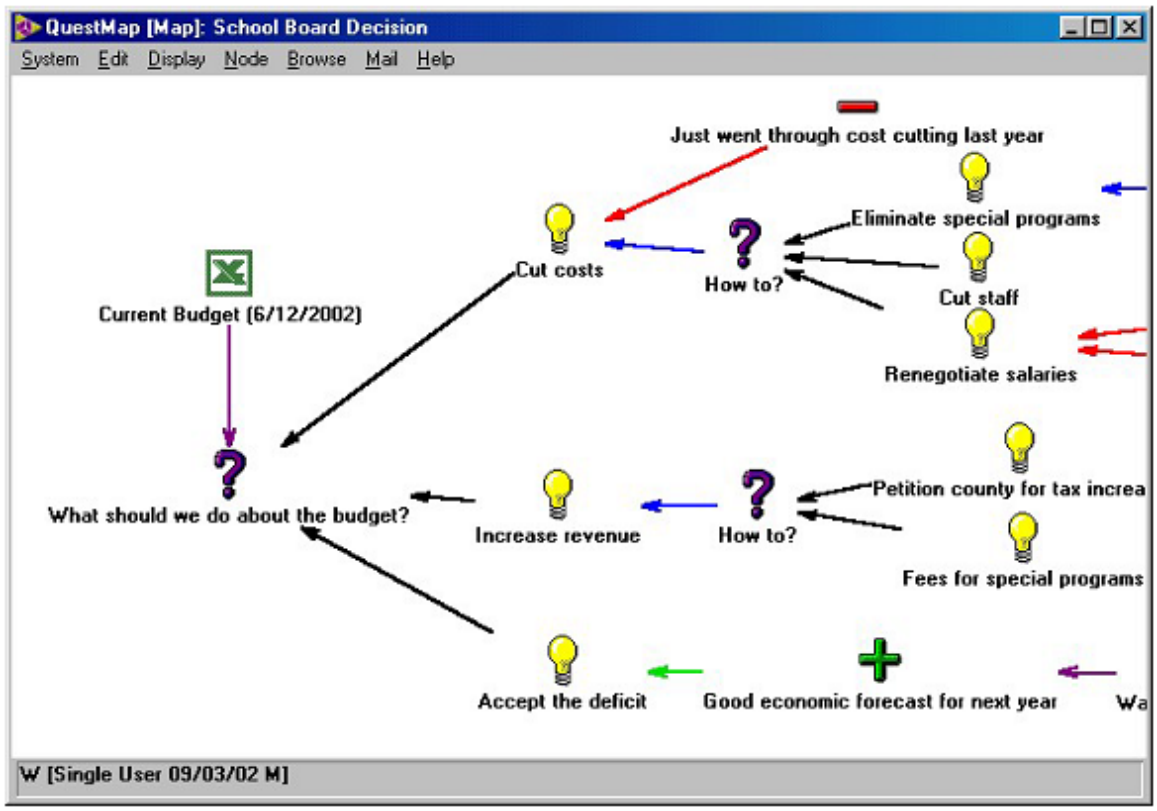


Figura 2.4 Segmento de mapa de diálogo mostrando un árbol de elementos IBIS (fuente: Conklin 2003)

.

Si bien esta herramienta administra el conocimiento informal permitiendo su búsqueda y navegación a través de una interface muy amigable, no permite la representación de la semántica de los elementos del árbol, limitando, por lo tanto el procesamiento semántico y automático del conocimiento.

2.2.1.8 AskMe Enterprise de AskMe - Realcom

AskMe Enterprise es un producto de software que según su proveedor, ofrece una solución comprobada para capturar, administrar y reutilizar el conocimiento de una empresa [AskMe, 2009], como así también el almacenamiento y recuperación de experiencias pasadas en forma de documentos. Utiliza una serie de algoritmos probabilísticos para el establecimiento de patrones, búsqueda y recuperación del conocimiento e inferencia, con la intención de:

- Compartir las mejores prácticas de la empresa
- Promover la innovación antes que la reinvención
- Incrementar la productividad de los funcionarios que realicen tareas intensivas en conocimiento
- Dar respuestas rápidas a problemas de negocio, mejorando la toma de decisiones
- Crear, en forma automática una base de conocimientos a partir del capital intelectual existente en los empleados a fin de ganar una ventaja competitiva sustancial.

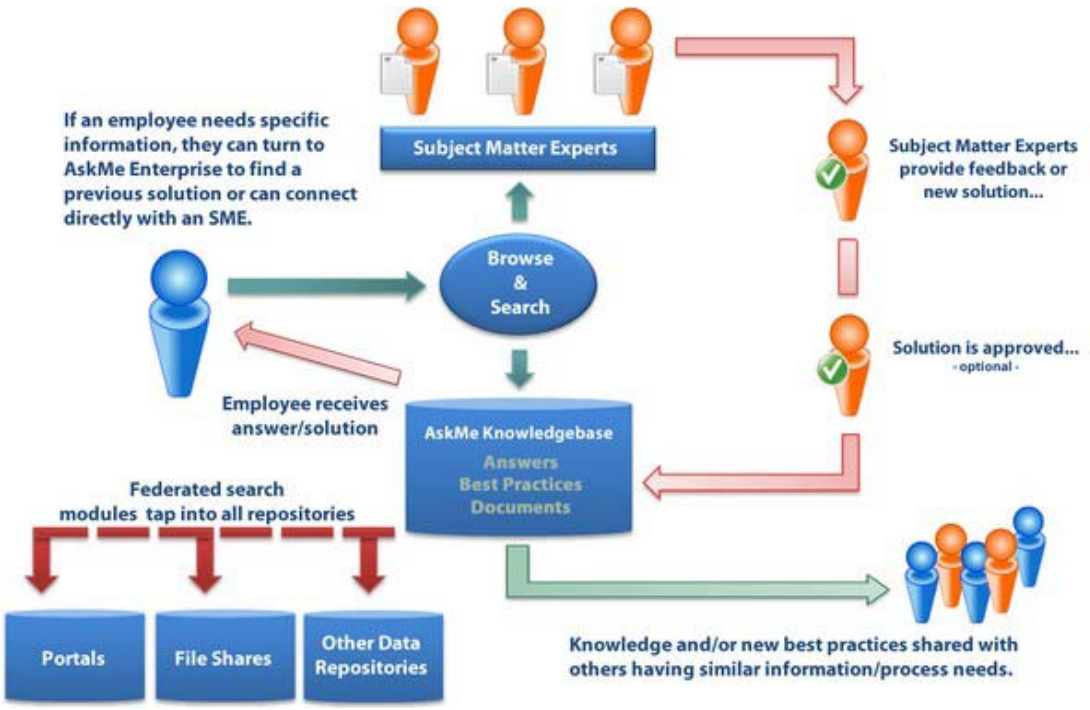


Figura 2.5 Infraestructura de la herramienta AskMe (fuente: AskMe-Realcom)

El objetivo general de este producto es ayudar a las corporaciones a que optimicen la administración del conocimiento organizacional en cualquier sector dentro de la empresa, conectando una red de computadoras. De esta manera AskMe proporciona una solución innovadora a las empresas intensivamente influenciadas por el conocimiento organizacional, para reducir la re-invención, contener los costos, acortar el ciclo de desarrollo de los proyectos y aumentar la innovación dentro de su organización. Además de proporcionar el acceso a la información de negocio, también distribuye la resultante de unir equipos, ideas y conocimiento a los procesos y workflows, habilitando a los integrantes de las empresas a acceder al conocimiento de más alto nivel. La figura 2.6 muestra algunas funcionalidades de la herramienta.

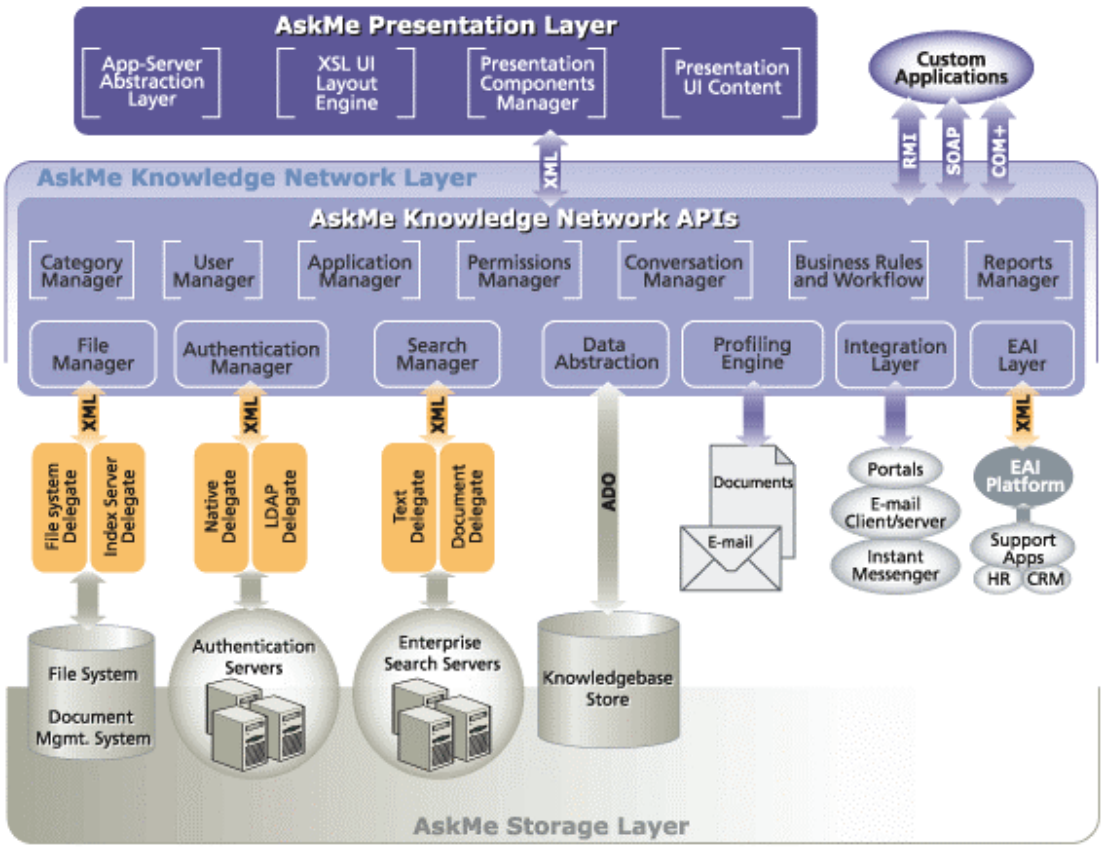


Figura 2.6 Arquitectura de la herramienta AskMe (fuente: AsMe- Realcom)

Desde el punto de vista tecnológico, la arquitectura de la herramienta AskMe está formada por tres capas lógicas que son: capa de presentación, capa de red de conocimiento y capa de almacenamiento. La comunicación entre las capas lógicas está implementada a través de las interfaces que usan tecnologías estándares basadas en XML.

La capa de presentación, permite conectar la infraestructura de AskMe a las tecnologías de información y aplicaciones de la empresa, como aplicaciones Web, motores de búsqueda, y directorios corporativos. Además incluye aplicaciones propias como son búsquedas, administración de documentos, creación de perfiles, etc.

La capa de red de conocimiento, recibe las peticiones de la capa de presentación y las resuelve a través de un conjunto de servicios que implementan funcionalidades como:

- La minería del conocimiento, que permite a los gerentes y directivos hacer inferencias y tomar decisiones basadas en el conocimiento compartido en la organización.
- El reuso y compartición del conocimiento, que facilita la búsqueda de soluciones a problemas de negocio a través de la administración y catalogación de documentos y mejores prácticas de los usuarios.
- La administración de aplicaciones, que permite al usuario jerarquizado (gerentes, administradores) gestionar taxonomías de conocimiento, usuarios, permisos y contenido.

La capa de almacenamiento, es donde se almacena la taxonomía de conocimiento, los perfiles de los especialistas y los documentos de tipos diversos (textos, artículos, mensajes que han sido intercambiados entre los empleados).

Esta aplicación es otra herramienta muy útil para la Gestión del Conocimiento empresarial, pero el almacenamiento del mismo se hace en forma semi-estructurada, en elementos de textos en lenguaje natural, dificultando el razonamiento semántico y automático.

2.2.1.9 XpertRule Knowledge Builder de XpertRule Software Ltd

XpertRule proporciona un entorno de desarrollo de aplicaciones y componentes basados en el conocimiento y permite la incorporación de reglas, procedimientos, políticas y reglamentos a los que le da el nombre de “Reglas de negocio” [Xpert, 2009].

La utilización de este ambiente, facilita la automatización de cualquier función (actividad) basada en conocimiento dentro de la organización, como por ejemplo:

- Recomendar servicios y cursos de acción, facilitando la compartición del conocimiento.
- Analizar defectos en las aplicaciones de soporte y ayuda a los clientes, facilitando la captura del conocimiento de los defectos y sus diagnósticos con el objetivo de dar soporte a los usuarios internos y clientes.
- Monitorear y validar condiciones de riesgo, por ejemplo en el área de fabricación y en los procesos donde se podría detectar anticipadamente las señales de advertencia de fallas.
- Servicio de "Workflow", las reglas pueden ayudar a decidir sobre las próximas acciones y tareas, basadas en los eventos actuales y datos disponibles.

El "Knowledge Builder" utiliza para la representación de las reglas, árboles de decisión. Los árboles de decisión están relacionados a un resultado o decisión dado un conjunto de atributos.

Esta herramienta brinda apoyo al desarrollo de aplicaciones intensivas en conocimiento basado en reglas de negocio, usa tecnologías propietarias y no provee mecanismos para la fácil integración con otros sistemas de información y procesos de la empresa.

2.2.2 Razonamiento Basado en Casos

Los sistemas de administración del conocimiento o de recomendación que usan CBR se basan en la asunción de que "los problemas similares tienen soluciones similares". Es decir, explotan el conocimiento adquirido a través de experiencias pasadas y lecciones aprendidas (casos), como un capital activo para la solución de un nuevo problema similar. Aamodt y Plaza [Aamodt et al., 1994], considerados referentes en el tema, presentan una definición para CBR, argumentando que implica "resolver un problema nuevo recordando una situación similar previa y reutilizando su información y conocimiento".

De acuerdo con los conceptos básicos de CBR, la recomendación de una solución a un problema determinado, está basada en la similitud del problema con experiencias pasadas guardadas en una base de casos. Típicamente un caso comprende: el problema y la solución a ese problema como dos objetos relacionados.

La resolución de problemas en CBR se compone de cuatro etapas, referidas en la literatura como las 4 R (Recuperar, Reusar, Revisar y Registrar):

- Recuperar los casos más similares.
- Reusar la información y conocimiento en los casos recuperados para resolver un problema similar.
- Revisar la solución propuesta, adaptándola si fuera necesario.
- Registrar o almacenar la nueva solución una vez que ha sido confirmada o validada de manera que pueda ser útil para resolver problemas futuros.

Contrariamente a los métodos de razonamiento basado en reglas, el enfoque de Razonamiento Basado en Casos es más apropiado para el ámbito donde el conocimiento es impreciso y difícil de describir, como ocurre con el conocimiento informal adquirido a través de la experiencia [Karacapilidis, 2006].

El CBR se ha utilizado de manera satisfactoria en muchas aplicaciones tanto académicas como industriales. En esta sección, revisaremos algunos sistemas tradicionales desarrollados en el ámbito académico que han sido ampliamente referenciados en la literatura [Kolodner, 1993], [Pal et al., 2004], por ejemplo: CHEF, CASEY, JULIA y PROTOS; soluciones comerciales contemporáneas ya revisadas en algunos trabajos recientes dedicados a la comparación de herramientas CBR [Schwabe, 2004], [Abdrabou et al., 2008] como son: Remind, CBR-Works, CBR3 y KATE. Finalmente se analizan los marcos de trabajo de amplia difusión: CBR Tools, CAT-CBR, jColibri e IUCBRF.

2.2.2.1 CHEF , CASEY, JULIA y PROTOS

CHEF [Hammond, 1990], es un planificador basado en casos para recetas de cocina. La aplicación crea las nuevas recetas a partir de otras ya conocidas, como respuesta a la especificación de determinados requisitos (submetas) de platos con ingredientes específicos, sabores particulares, etc. CHEF tiene que construir planes que satisfagan un número dado de metas simultáneamente. Cuando se solicita una nueva receta, con características determinadas, lo primero que hace es buscar un plan (una de las recetas que ya conoce y dispone en su base de casos) que satisfaga el mayor número posible de características o metas. Una vez que se ha encontrado ese plan, lo modifica para que se cumplan todas aquellas características que aun no satisface. CHEF altera estas recetas con reglas de modificación y un conjunto de "objetivos críticos". Estas reglas son especificas de la clase de plato de que se trata (frito, soufflé, etc.) y de la característica a modificar. Los objetos críticos, son los que cambian los tiempos de cocción de los ingredientes, teniendo en cuenta los cambios que ha sufrido la receta inicial.

CASEY es un sistema de diagnósticos médico basado en casos [Koton, 1989]. Como entrada toma una descripción de su nuevo paciente, incluyendo muestras normales y presentándolas como síntomas. Su salida es una explicación causal de los desordenes que tiene el paciente. La diagnosis la realiza en dos pasos: primero busca en la memoria casos y utiliza reglas de evidencia basadas en un modelo del dominio para determinar cuales de los casos que ajustan parcialmente son suficientemente similares al nuevo problema para proporcionar una diagnosis precisa. Después aplica reglas de reparación basadas en el modelo (estrategias de adaptación) para adaptar el diagnóstico antiguo a la nueva situación.

JULIA es un sistema de diseño basado en casos que opera en el dominio de planificación de comidas [Hinrichs, 1992]. Como en los casos anteriores, los problemas se describen en términos de las restricciones que tienen que cumplir, y las soluciones describen la estructura y un artefacto que cumple la mayor cantidad de restricciones posibles. Como los problemas suelen ser muy extensos y en general no pueden resolverse encontrando un caso anterior que se pueda adaptar para resolver el nuevo problema, habitualmente éstos se dividen en partes que se resuelven

independientemente. Obviamente se debe tener en cuenta las dependencias entre las partes en que se descompone. Para ello JULIA refuerza el razonamiento basado en casos con un proceso de propagación de restricciones, usando las restricciones como índices del diseño. Las restricciones provienen de diversas fuentes, unas del conocimiento general del artefacto a diseñar (componentes incompatibles, por ejemplo) y otras del nuevo problema (los componentes que quiere emplear el usuario). JULIA mantiene un conocimiento general del tipo de artefacto a diseñar almacenándolo en prototipos de objetos, es decir, los prototipos de objetos se corresponden con los diferentes tipos de comida que conoce, por ejemplo, cenas americanas, cenas europeas, comidas de catering, etc. Cada prototipo tiene una estructura específica y relaciones entre sus partes. La herramienta usa los prototipos de comida para proporcionar frameworks como soluciones cuando los casos no son la opción correcta.

PROTOS [Bareiss, 1989], implementa la adquisición y clasificación de conocimiento basado en casos. Dada una descripción de una situación u objeto la clasifica por su tipo. Cuando PROTOS clasifica un elemento de forma incorrecta, su consultor experto le informa del error y del conocimiento que debe usar para clasificarlo correctamente. El dominio de PROTOS es el de los desórdenes auditivos. Dada una descripción de síntomas y los resultados de algunas pruebas de un paciente, PROTOS determina qué problema auditivo tiene. El consultor experto es un otorrinolaringólogo. Puede emplearse para otras tareas relacionadas, como el reconocimiento del estado emocional de un agente. Para hacer la clasificación de una situación u objeto, PROTOS busca el objeto o situación ya conocido, que mejor ajusta a la nueva situación y le asigna la misma clasificación. Si lo encuentra, el proceso termina informando la solución encontrada. Si no, utiliza los resultados de este proceso de ajuste para seleccionar una hipótesis mejor. Este proceso se guía por su conocimiento sobre los tipos de errores de clasificación más comunes en este dominio.

Estas cuatro aplicaciones de Razonamiento Basado en Casos, han sido desarrolladas para dominios muy específicos: de cocina en los casos de CHEF y JULIA de diagnósticos médico en el caso de CASEY y de desórdenes auditivos en el caso de PROTOS. Al no estar diseñadas con un enfoque genérico más amplio, son de difícil aplicación a otros dominios, por ejemplo al de Ingeniería de Software.

2.2.2.2 ReMind de Cognitive System Inc.

ReMind fue inicialmente desarrollada [Bareiss, 1991] con el soporte del programa DARPA de EEUU.

Los casos en ReMind se representan como pares atributo:valor y ofrece varias posibilidades de recuperación:

- Por el vecino mas próximo, con ayuda de los pesos definidos por el usuario para las características de cada caso.
- Por patrones, soportando consultas SQL simples.
- Recuperación inductiva, bien automáticamente sin interacción del usuario, o bien inductiva guiada por el conocimiento, donde el usuario crea un modelo cualitativo para guiar al algoritmo de inducción con un cierto trasfondo de conocimiento.
- Por medio de modelos cualitativos que se crean gráficamente para indicar qué conceptos dependen de otros. Cuando el algoritmo de inducción se guía por estos modelos, los árboles de decisión serán un mejor reflejo de la relación causal de los conceptos en los casos.

Además, cabe destacar la posibilidad de importar bases de casos desde Bases de Datos convencionales.

ReMind no usa una representación totalmente estructurada de casos ni guarda información de su contexto. La representación de los atributos de tipo texto es en formato libre. Esto limita la potencia de procesamiento semántico de los casos.

2.2.2.3 CBR-Works de TECINNO GmbH

CBR-Works es una herramienta que usa CBR para brindar apoyo en actividades de recuperación de información [Schulz., 1999], diagnóstico, y selección de productos. Aprovecha el uso de bases de datos existentes como fuentes de conocimiento. Usa una representación de conocimiento orientada a objetos que se basa en normas

internacionales como CORBA para garantizar la integración de componentes de software futuros.

CBR-Works fue creado en el marco de proyecto INRECA [INRECA]. Está preparado para encontrar soluciones inteligentes en una variedad de dominios y ambientes. Incluye a editores gráficos que dan soporte al usuario para diseñar modelos de conocimiento sofisticadamente complejos. El sistema trata conceptos, tipos, medidas de similitud, pesos y filtros. Cuatro interfaces separadas proporcionan una manera de usar los elementos que modelan los conceptos, tipos, la administración de la base de casos y recuperación de casos:

- La interfaz de jerarquía de conceptos es un editor para construir el modelo de conceptos.
- La interfaz de jerarquía de tipos es un editor para definir los tipos y su medida de similitud a ser usado en el modelo.
- La interfaz de base de casos proporciona las herramientas para administrar la base de casos.
- La interfaz de consultas ofrece funciones para recuperar los casos ya sea proporcionando una consulta o dejándose llevar (guiándose) por un diseñador de consultas.

El usuario o bien puede definir sus propios elementos de modelo o bien puede usar el diseñador de modelo y utilizar los elementos predefinidos que el sistema ha establecido por defecto. El sistema está disponible para la plataforma de Windows.

Esta herramienta puede ser aplicada a distintos dominios de conocimiento y brinda la flexibilidad de permitir definir los atributos de los casos y su medida de similitud. El problema es que los atributos no pueden ser de tipos de datos estructurados, limitando su potencial de representación y no está basado en ontologías ni en tecnologías estándares, lo que dificulta su integración a otras herramientas del negocio y el compartir fácilmente el conocimiento.

2.2.2.4 CBR3 de Inference Corp.

Los productos de la familia CBR3 son, para algunos autores [Price, 1999], los más exitosos y mayormente usados.

La familia de herramientas CBR3 está formada por:

- *CBR Express*, un entorno de desarrollo y creación para bases de casos.
- *CasePoin*t, un motor de búsqueda en memoria, eficiente para bases de casos desarrolladas usando CBR Express.
- *Generator*, una herramienta que automatiza la creación de bases de casos a partir de una colección de archivos de texto, como ASCII, Ms Word, etc.
- *Tester*, una herramienta que provee una variedad de métricas para desarrolladores de base de casos que usan CBR3
- *CasePoint WebServer*, una herramienta que soporta la funcionalidad de CasePoint en Internet.

CBR3 usa una estructura de registros simples que son almacenados en una base de datos relacional. Los casos incluyen un título, una descripción, un conjunto de atributos con un cierto peso, y un conjunto de acciones. Los casos pueden compartirse a través de la red de una organización. Los desarrolladores no usan CBR3 programando, sino a través de la interfaz de CBR Express, que es una herramienta amigable para usuarios sin experiencia en programación.

Esta herramienta propietaria, no está basada en ontologías ni en tecnologías estándares, lo que dificulta su integración a sistemas de información la empresa y el reusar fácilmente el conocimiento en actividades de recomendación. Si bien permite la recuperación de casos a partir de archivos de texto, la captura de los mismos no está integrada a los procesos del negocio.

2.2.2.5 KATE Advisor de Kaidara Software

La herramienta KATE Advisor de la empresa Kaidara Software (anteriormente AcknoSoft) [Kaidara, 2009] provee puntos de acceso a servicio técnico para la

resolución de problemas. Los usuarios de Kaidara Advisor, que son típicamente agentes de centros de llamada, personal de asistencia técnica, clientes o distribuidores entran texto en el idioma cotidiano describiendo un problema que quieren solucionar. El sistema automáticamente responde, interactuando con ellos para consultar una serie de preguntas pertinentes relevantes para guiar al usuario recomendando posibles diagnósticos y soluciones.

La meta de Kaidara Advisor es ayudar al usuario a encontrar la solución más apropiada al problema presentado, y hacerlo en el tiempo más corto posible. El sistema soporta resolución de problemas en múltiples idiomas y puede incluir elementos de búsqueda multimediales como imágenes, iconos y sonidos para simplificar el proceso.

El motor CBR de la herramienta KATE usa una recuperación basada en un algoritmo de cálculo del vecino más cercano, donde las funciones de similitud y los pesos de los descriptores pueden personalizarse para ajustarse a las necesidades de cada aplicación específica. Además posee un módulo de *data mining* que extrae conocimiento oculto en los datos, construyendo automáticamente árboles de decisión a partir de los casos.

En esta herramienta, los descriptores de los casos se guardan en formato de texto libre, lo que la hace muy útil para su uso más frecuente: el acceso a servicio técnico a través de agentes de centros de llamada, para la resolución de problemas. Sin embargo cuando se quiere aplicar a un dominio de diseño y desarrollo en ingeniería, como puede ser el caso de aseguramiento de calidad en Ingeniería de Software, se necesita una descripción más formal de los casos para contar con una representación mas potente desde el punto de vista del procesamiento semántico y automático.

2.2.2.6 CBR*Tools

CBR*Tools es un framework orientado a objetos (OO) para CBR [Jaczynski, 1998] especificado con el Lenguaje de Modelado Unificado (UML) [Booch, 1994] y escrito en Java. Ofrece un juego de clases abstractas para modelar los principales conceptos necesarios para desarrollar aplicaciones que integran las técnicas del

Razonamiento Basado en Casos como son: caso, base de casos, índice, medida de similitud, control de razonamiento.

También ofrece un juego de clases concretas que llevan a cabo métodos tradicionales de CBR. CBR*Tools contiene en total más de 220 clases. La programación de una nueva aplicación se hace por la especialización de clases existentes, agregación de objetos o usando los parámetros de las clases existentes.

CBR*Tools delega cada etapa CBR (recuperar, reusar, revisar y retener), a un objeto diferente. Cada clase define una interfaz abstracta para cada una de las etapas del razonamiento mientras la clase *Reasoner* define cómo controlar el razonamiento. Las clases de cada etapa deben especializarse para implementar un motor de CBR para un dominio específico. La clase *Reasoner* permite la aplicación de métodos de control de razonamiento diferentes. La Figura 2.7 muestra parte del diagrama de clases de CBR*Tools.

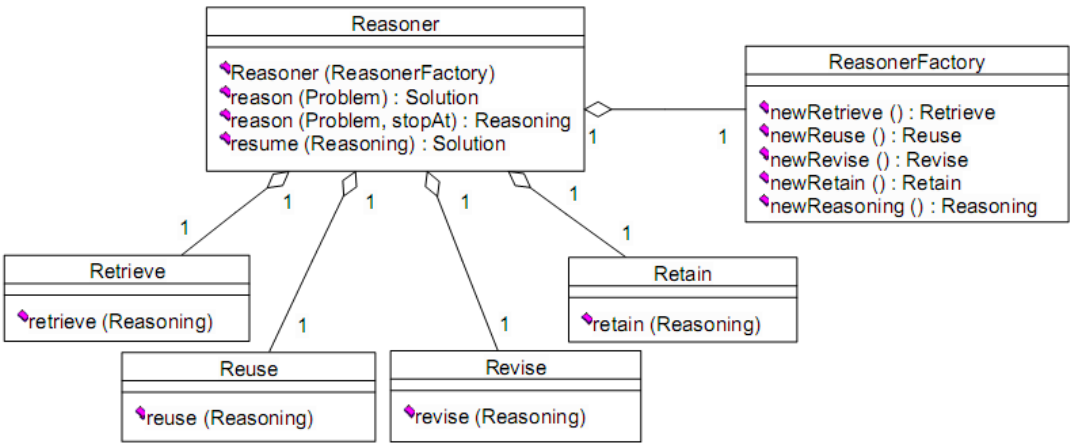


Figura 2.7 parte del diagrama de clases de CBR*Tools (fuente: Jaczynski, 1998)

Los marcos de trabajo OO de CBR presentados en esta subsección y en las tres subsecciones siguientes permiten el reuso del diseño y código para aplicaciones CBR, habilitando al programador no experto a escribir rápidamente aplicaciones complejas. Estos frameworks también permiten el desarrollo de prototipos que podrían extenderse en el futuro por especialización o composición.

Sin embargo, una vez que las clases del framework se especializan para un dominio específico, constituyendo una aplicación, ésta no puede aplicarse directamente a otro dominio, porque las caracterización del dominio está embebido en el código especializado y no en bases de conocimiento formal externas a la aplicación, como son por ejemplo las ontologías de dominio.

2.2.2.7 CAT-CBR

La plataforma CAT-CBR constituye una biblioteca de componentes de CBR para guiar al usuario en el desarrollo de una aplicación [Abasolo et al., 2002]. Estos componentes describen las distintas tareas que pueden aparecer en un sistema de CBR y también los métodos de resolución de problemas que pueden aplicarse a estas tareas. CAT-CBR se ha desarrollado para la plataforma de Noos [Arcos, 1997].

Para describir los componentes de CBR que integran el framwork se usó el Lenguaje Universal de Métodos de resolución de problemas (UPML). CAT-CBR usa dos procesos para permitir a los usuarios desarrollar una aplicación, a saber: el proceso de la configuración y el proceso de implementación operacional.

El proceso de configuración consiste en seleccionar los distintos componentes y conectarlos para especificar una aplicación, esto se puede hacer a través de una herramienta interactiva donde los usuarios eligen los componentes que necesitan ser incluidos en la aplicación. El proceso de implementación operacional toma una especificación de aplicación y genera una aplicación ejecutable. La plataforma genera un archivo que invoca métodos de Noos siguiendo la estructura de la configuración de componentes definida.

2.2.2.8 JColibri

El marco de trabajo JColibri [Bello-Tomas et al., 2004] se compone de una jerarquía de clases de Java y varios archivos de XML. Está organizado alrededor de cuatro elementos principales: tareas y métodos, base de casos, casos, y métodos de resolución de problemas.

Archivos XML describen las tareas soportadas por el framework y los métodos para resolver estas tareas. Las tareas son elementos claves que representan la meta del método y se pueden identificar por el nombre y descripción en un archivo de XML. Los usuarios pueden agregar tareas en cualquier momento al framework.

jColibri tiene una interfaz de organización de memoria que permite leer cualquier base de casos en la memoria para que el motor de CBR pueda trabajar sobre ella. Esto no es factible para tamaños de base de casos demasiado grande. También implementa una nueva interfaz que permite recuperar los casos que satisfagan satisfactoriamente una pregunta de SQL. Por encima de la base de casos posee una segunda capa que es una estructura de datos que organiza los casos que se carguen en memoria en índices. Este enfoque de dos capas es bastante eficaz para permitir distintas estrategias de recuperación de casos.

Este framework representa los casos de una manera muy simple. Un caso es un individuo que tiene un número de relaciones con otros individuos. El framework se apoya en diferentes tipos de datos que definen cualquier caso simple.

2.2.2.9 IUCBRF

El marco de trabajo de Razonamiento Basado en Casos de la Universidad de Indiana (IUCBRF) es un paquete de software libre, escrito en Java, para facilitar el desarrollo de sistemas CBR [Bogaerts et al., 2005a]. IUCBRF proporciona código para los distintos aspectos de CBR independiente del dominio de aplicación, dejando en mano de los desarrolladores sólo la implementación de aspectos del sistema dependientes del dominio. Este framework fue diseñado para facilitar un desarrollo modular y rápido de sistemas de CBR, proveyendo los pilares fundacionales para la compartición de código entre los desarrolladores de sistemas CBR.

El marco de trabajo también se pensó como un punto de partida para usos pedagógicos [Bogaerts et al., 2005b], e incluye código completo de aplicaciones de demostración simples para ilustrar su uso y el proceso de CBR. Además, provee

capacidades iniciales de generación automática de conjuntos de prueba simples de casos y generación de estadísticas de performance para ayudar a evaluar distintos enfoques.

2.3 Conclusión del Estado del Arte

Algunos de los sistemas revisados ([Abecker et al., 1999], [Ale 2009], [AskMe, 2009]) almacenan el conocimiento en repositorios de fuentes de conocimiento no estructuradas o semiestructuradas, como archivos de texto, perfiles o documentos de marcas entre otros, a los que adicionan importantes mecanismos de indexación y búsqueda (en algunos casos semántica) para la gestión y reuso del conocimiento. El problema de esta forma semiestructurada de almacenar el conocimiento, es que dificultan mecanismos potentes de procesamiento semántico y automático del contenido. Un trabajo de Weber [Weber 2007] analiza los factores que pueden hacer fracasar a los sistemas de administración del conocimiento existentes, y entre ellos el autor destaca que los KMS pueden fallar cuando guardan el conocimiento en representaciones textuales sin restricciones. Según el autor, "*Artefactos de conocimiento guardados en el formato textual pueden ser largos y difíciles interpretar*".

Otra de las características observadas en muchos de los KMS analizados, es que se basan en memorias organizacionales monolíticas. Las empresas que intentan desarrollar una Memoria Organizacional única y centralizada para la compañía en su conjunto han fallado [Ackerman et al., 2000], entre otras razones, porque tales organizaciones son distribuídas y pueden tener objetivos contradictorios y requerimientos de conocimiento distintos. Algunas propuestas, como por ejemplo OntoDom [Ale, 2009], muestran un modelo de Gestión del Conocimiento centrada en una Memoria Organizacional distribuída basándose en ontologías. Esta propuesta está orientada a la recuperación de documentos de texto conteniendo conocimiento organizacional, pero no administra el conocimiento informal en sí mismo. La autora también destaca la importancia y ventaja de la gestión distribuída del conocimiento.

Como se mencionó en el párrafo anterior, una de las características deseables de la Memoria Organizacional, es que permita gestionar y compartir el conocimiento de toda la organización que puede estar distribuido geográficamente y administrado localmente con sistemas o plataformas heterogéneas. Este intercambio de conocimiento tanto

dentro de la organización o entre distintas empresas, es difícil de lograr si no se formaliza una conceptualización común, consensuada que defina los principales conceptos relacionados a la Memoria Organizacional y sus componentes, (es decir, si no está basada en una ontología). Esta es otra importante carencia observada dentro de la bibliografía revisada. Algunos autores han observado que luego de varios años de investigación, la "*Memoria Organizacional*" se ha tornado un tópico demasiado complejo y confuso [Ackerman et al., 2000]; los autores también afirman que la literatura existente sobre Memoria Organizacional sostiene muchas definiciones distintas, y a veces contradictorias. Si bien se ha avanzado mucho en propuestas de Memoria Organizacional, su estructura y conformación es incierta en la literatura de investigación. A pesar de los esfuerzos hechos en nuevos desarrollos y estándares internacionales durante la última década, la terminología del área de gestión de conocimiento está todavía en fase de discusión, consenso y definición [Ferguson 2006]. En particular, hasta lo que nosotros sabemos, no hay ninguna ontología completamente especificada de Memoria Organizacional Basada en Casos, tal como propondremos en el capítulo 6.

Teniendo en cuenta los productos comerciales analizados, muchas de las herramientas revisadas (por ejemplo AskMe, XpertRule, KATE, Remind y CBR3) están disponibles como un conjunto de componentes (librerias C o API´s) que proporcionan un juego del funciones para la administración del conocimiento y pueden ser embebidas en KMS de las empresas. En general, apuntan a ayudar a los programadores a diseñar e implementar fácilmente sistemas de Gestión del Conocimiento. En el mejor de los casos, estos componentes pueden ser extendidos por el programador para adecuar los algoritmos al dominio de aplicación. Sin embargo, ninguna de estas herramientas está diseñada para ser fácilmente instanciada y que a su vez proporcione el conocimiento del dominio a través de ontologías.

Si bien la idea de aplicar métodos de Razonamiento Basado en Casos a las lecciones aprendidas y buenas prácticas no es nueva en el área de administración de conocimiento ([Weber et al., 2001], [Yang et al., 2006]), algunos de los sistemas de CBR analizados (por ejemplo CHEF, CASEY, JULIA, PROTOS), han constituido un valioso aporte a la ciencia, pero están definidos para dominios muy específicos, como son recetas de cocina, diagnósticos médicos, etc.

Afortunadamente también se han desarrollado frameworks genéricos de CBR, como por ejemplo CBR Tools, CAT-CBR, jColibri e IUCBRF. Estos facilitan el diseño y desarrollo de sistemas de Gestión del Conocimiento basados en CBR para distintos dominios, con pocas adecuaciones al código. El problema que encontramos en estos marcos es que las adaptaciones deben ser embebidas en el código, ya que no cuentan con una forma simple de tomar las definiciones de los atributos que caracterizan a los casos desde una ontología de dominio.

Una de las definiciones de caso más aceptadas en la literatura establece "*Un caso es una pieza* ***contextualizada*** *de conocimiento que representa una experiencia*" [Kolodner, 1993]. Contiene la lección pasada que es el contenido del caso y el contexto en el cual la lección puede ser utilizada. En las memorias organizacionales basada en casos, la relevancia de la información de experiencias similares, está muy ligada al contexto en que es válido aplicar dicha experiencia. Para que un KMS sea exitoso en el reuso y explotación del conocimiento, debe administrar los contextos en que ocurre cada ítem de conocimiento. La mayoría de las propuestas analizadas de Memoria Organizacional y sistemas de Razonamiento Basado en Casos, no contemplan un mecanismo para administrar el contexto en pro de un mejor uso de dicho conocimiento. Según un estudio de Szulanski, un impedimento que hace fracasar los KMS es que el conocimiento a ser transferido típicamente no puede comunicar los factores que hacen una estrategia (experiencia, buena práctica) aplicable a las distintas situaciones (contextos) eficazmente [Szulanski, 1996].

Respecto a la aplicación de técnicas de Gestión del Conocimiento al área de Ingeniería de Software existen varios trabajos que proponen soluciones interesantes ([Basili et al., 1997], [Falbo et al. 2003]) pero no presentan un modelo genérico de Memoria Organizacional que pueda ser aplicado también a otros dominios.

Como conclusión final, si bien ninguna de las herramientas y trabajos analizados cumple todas las características deseables y sirve para todos los propósitos; cada una proporciona un aporte interesante a la ciencia para construir una Memoria Organizacional Basada en Casos que esperamos conduzca a un mejor resultado que el que se hubiera logrado comenzando desde cero.

2.4 Necesidad de una Memoria Organizacional basada en Ontologías y Casos.

Teniendo en cuenta el análisis del estado del arte en Memoria Organizacional, Gestión del Conocimiento y Razonamiento Basado en Casos de la sección anterior, y evaluando la importancia de contar con sistema de recomendación centrado en el conocimiento adquirido en experiencias previas para dar soporte a los procesos de medición y evaluación en el área de aseguramiento de calidad en Ingeniería de Software, observamos la necesidad de contar con un sistema robustamente estructurado para explotar el valioso conocimiento presente en la organización y obtener así ventajas competitivas.

Para que este conocimiento pueda ser procesado automáticamente e integrado a los sistemas de información de la empresa es necesario guardar el conocimiento de manera estructurada en lo que se denomina Memoria Organizacional Basada en Casos (MOBC). La estructuración del conocimiento informal en casos, facilita la captura en forma automática, la recuperación, transferencia y reuso.

Los sistemas basados en CBR se pueden beneficiar de una representación formal y compartida de una conceptualización (ontología) del dominio de Memoria Organizacional, y Razonamiento Basado en Casos. La ontología implica una interpretación inequívoca del significado de conceptos como son *Memoria Organizacional, base de conocimiento, ítem de conocimiento, contexto, casos, problema, solución, resultado, modelo de similitud*, entre otros y las relaciones entre todos estos conceptos. Entre los beneficios de basar el sistema en ontologías, se destaca la posibilidad de compartir el conocimiento entre sub-sistemas heterogéneos de la misma empresa, tener una interpretación semántica y unificada de dicho conocimiento y además (como se explicará en el capítulo 7), pueda ser fácilmente extensible y aplicable a distintos dominios.

Por último, pero no menos importante, es conveniente que dicho sistema sea distribuído para que el conocimiento esté siempre disponible y sea de fácil acceso, aún cuando se haya generado en distintos lugares, distribuidos geográficamente. Para facilitar el desarrollo de herramientas que automaticen los procesos de captura, consulta

y reuso de casos, de una manera interoperable aún cuando estén implementadas en plataformas distintas, se propone su implementación bajo estándares de marcado (XML, SXML), estándares de Web Semántica (como RDF, RDFS, OWL) y de arquitecturas de ambientes distribuidos de amplia difusión. Esto permitirá que el conocimiento pueda ser procesado no sólo por seres humanos sino también por agentes, como herramientas de software para potenciar su explotación automática.

Con el fin de solucionar los problemas que se plantean en el estado del arte, esta tesis contribuye en la especificación, diseño e implementación de un Sistema de de Memoria Organizacional con estas características. Y para ilustrar la propuesta se desarrolla la aplicación a un sistema de recomendación para el área de medición y evaluación en Ingeniería de Software.

PARTE II

FUNDAMENTOS

Capítulo 3- Gestión del Conocimiento y Memoria Organizacional

3.1 Introducción

Las organizaciones modernas han comenzado a valorar y gestionar el conocimiento de sus empleados, ya que hoy en día, muy pocos cuestionan la afirmación de que vivimos en una sociedad basada en el conocimiento. Los expertos en gestión empresarial coinciden en que el conocimiento es uno de los recursos más importantes que contribuyen a las ventajas competitivas de una organización ([Handzic et al., 2004], [Malhotra 2005]). Prusak afirmaba lo siguiente "*Los investigadores en áreas de ventajas competitivas sostenibles han llegado a la conclusión de que la única cosa que da a una organización ventajas competitivas duraderas, es lo que sabe, como utiliza lo que sabe y su capacidad de aprender cosas nuevas rápidamente.*" [Prusak, 1996]. De este modo, y debido a la especial relevancia del conocimiento, la sociedad actual también recibe el nombre de "sociedad del conocimiento".

La Gestión del Conocimiento organizacional representa un activo fundamental para el soporte al proceso de toma de decisión en cualquier organización. Sin embargo, una organización no puede crear conocimiento por ella misma [Nonaka et al., 1995], por el contrario, el inicio de la creación del conocimiento en las organizaciones es el conocimiento tácito individual adquirido por los empleados a través de experiencias y lecciones aprendidas, y este conocimiento es compartido entre interacciones interpersonales. Por lo tanto, en pro de alcanzar y mantener la competitividad organizacional y la efectividad, una organización necesita aprender de estas experiencias pasadas y presentes y formalizar memorias organizacionales para permitir hacer explícito el conocimiento tácito individual y, por que no, el conocimiento tácito de la comunidad también. La Memoria Organizacional, por lo tanto, constituye el almacenamiento de todo el conocimiento formal e informal [Conklin, 1996] presente en la organización.

En este capítulo introduciremos conceptos de conocimiento y su gestión, Memoria Organizacional y algunos temas relacionados de interés para una mejor comprensión de la tesis.

3.2 El Conocimiento

Si bien la Gestión del Conocimiento es un tema que en las últimas décadas comenzó a cobrar importancia, sus orígenes se remontan desde mucho tiempo antes, por lo tanto, la literatura al respecto es amplia y diversa, a tal punto que suelen encontrarse muchas definiciones de la terminología con distintos enfoques. En esta sección presentamos un resumen de los principales conceptos relacionados al conocimiento extraídos de bibliografía de amplia difusión.

3.2.1 Dato, Información y Conocimiento

La definición de conocimiento también permite hacer referencia a otros conceptos con los que se halla relacionado (e incluso algunas veces confundido dentro de lo que puede considerarse un proceso continuo de aprendizaje).

Según Davenport y Prusak [Davenport et al., 1998], un dato es un conjunto discreto de factores objetivos sobre un hecho real. Un dato no dice nada sobre el porqué de las cosas y por sí solo tiene poca o ninguna relevancia o propósito. Las mediciones de, por ejemplo, un análisis clínico constituyen datos referidos a niveles de glucosa, colesterol, etc. Estos valores leídos de manera aislada de un contexto y sin ninguna interpretación, son totalmente carentes de sentido. La importancia de los datos radica justamente en su capacidad de asociarse dentro de un contexto para que se le otorgue una interpretación que los nutra de sentido. Dado que no tienen capacidad de comunicar un significado, no pueden afectar el comportamiento de quien lo recibe.

Los datos para ser útiles deben procesarse y obtener un valor agregado del mismo en un contexto determinado. Hace falta de la información para poder otorgar al dato un significado relevante en un contexto determinado. La información, al igual que un mensaje en un sistema de comunicación, tiene un emisor y un receptor. La

información es capaz de cambiar la forma en que el receptor percibe algo, es capaz de impactar sobre sus juicios de valor y comportamientos. Su descendencia del latín[1] acusa el significado de "*dar forma a"*; precisamente la información es capaz de formar a la persona que la consigue, proporcionando ciertas diferencias en su interior o exterior. Por lo tanto, es el receptor, y no el emisor, el que decide si el mensaje que ha recibido es realmente información. Sirve de ejemplo considerar un informe repleto de gráficos inconexos, que puede ser considerado información por el que lo escribe, pero que, sin embargo, pueda ser juzgado como ruido por quien lo recibe. Los datos se convierten en información añadiéndoles valor de la siguiente manera:

- *Contextualizando*: saber en qué contexto y para qué propósito los datos se generaron.
- *Categorizando*: se conocen las unidades de medida que ayudan a interpretar los datos.
- *Calculando*: los datos pueden haber sido analizados matemáticamente o estadísticamente.
- *Corrigiendo*: los errores han sido eliminados de los datos.
- *Condensando*: los datos se han podido resumir de forma más concisa.

Existe una confusión colectiva en creer que la información es conocimiento, pero en verdad, la información en sí no es conocimiento sino es transformada y utilizada. El conocimiento se deriva de la información, y ésta de los datos. Dato, información y conocimiento constituyen entonces tres grandes conceptos que no deben confundirse, y aun más, el buen manejo de ellos depende en gran medida el éxito de una organización en materia de toma de decisiones. De hecho, se procesan los datos para tomar algún tipo de decisión con respecto a la información obtenida y al conocimiento del entorno que se posea. Podría decirse que el dato carece de significado, sino conlleva a determinada acción.

[1] Según la Real Academia Española, proviene del latín *informare*

El enfoque de esta tesis gira principalmente en uno de estos tres conceptos: el conocimiento. Existe una sensación intuitiva de que el conocimiento es algo más profundo, más amplio y por qué no, más rico que los datos y la información. El conocimiento es una mezcla de experiencias, valores, información contextual y entendimiento de experto que provee un marco para evaluar e incorporar nuevas experiencias e información [Davenport et al., 1998]. Se origina y es aplicado en la mente de los conocedores. En las organizaciones, está embebido no sólo en documentos o repositorios sino también en rutinas, procesos, prácticas y normas organizacionales.

Las definiciones dadas permiten imaginar una jerarquía entre estos conceptos. En dicha jerarquía, los datos son vistos como hechos simples aislados que, cuando son puestos en un contexto y combinados en una estructura, emerge la información [Ale et al., 2006]. Cuando a la información se le otorga un significado mediante la interpretación, surgida de experiencias, lecciones aprendidas, reglas prácticas y heurísticas, se convierte en conocimiento.

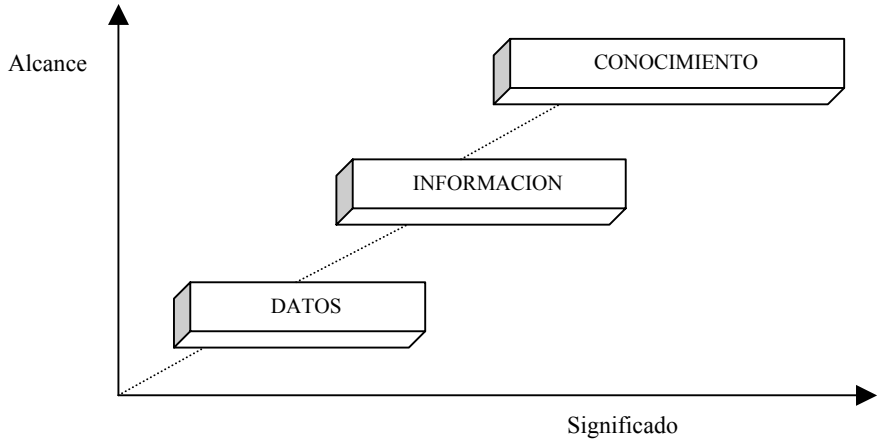


Figura 3.1. Datos, Información y Conocimiento

Algunos autores destacan un nivel superior de esta jerarquía constituyendo la *sabiduría*. El conocimiento se convierte en sabiduría mediante un proceso de asimilación cultural [Bellinger, 2004]. A los nuevos conceptos y modelos de la realidad se le asignan valores y éstos son relacionados a principios, buenas costumbres e ideales en contexto con los sistemas de creencias establecidos. La sabiduría puede ser incrementada o desafiada por nuevo conocimiento, pero tiene una fuerte base emocional

y no siempre está sujeta a cambio por argumentos estrictamente lógicos y racionales. Lo que la gente cree es una función que depende fuertemente de sus prejuicios, preferencias, tradiciones y normas sociales.

3.2.2 El Conocimiento y el Contexto

Un concepto relacionado con datos, información, conocimiento y sabiduría y que con frecuencia es utilizado para diferenciar uno del otro es el *contexto*. Por contexto se entiende la colección de condiciones relevantes e influencias circundantes que hacen que una situación sea única y comprensible. Bellinger argumenta que a medida que uno se mueve hacia arriba en la jerarquía de los datos hacia la sabiduría, el nivel de independencia del contexto crece [Bellinger, 2004], según se observa en la figura 3.2.

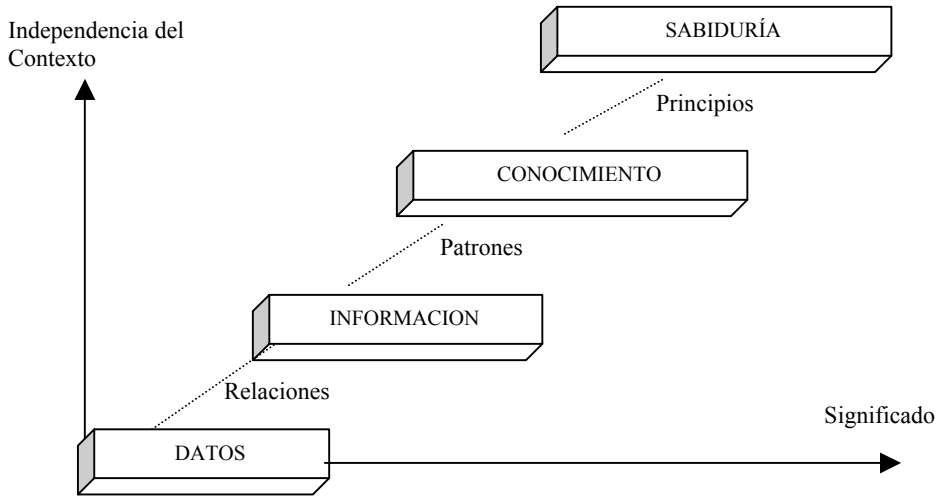


Figura 3.2. Relación entre contexto y significado (fuente: Bellinger, 2004)

Tomando una interpretación ligeramente diferente se puede considerar cuán embebido está el contexto en la construcción. En este caso a medida que nos movemos hacia arriba en la jerarquía hay un creciente nivel de significado implícito o contexto embebido dentro de las relaciones, patrones y principios asociados con los diferentes niveles.

Los datos por sí mismos no tienen contexto hasta que alguien los interpreta y contextualiza. La información tiene algo de contexto embebido por la naturaleza de las

relaciones que resume y organización del conjunto de datos que la constituye. El conocimiento tiene más contexto embebido a través de los patrones que son formados mediante el arreglo de la información que contiene. Finalmente, la sabiduría tiene la mayor cantidad de contexto embebido dentro de los principios que forman sus valores y arquetipos.

3.2.3 Conocimiento Explícito y Conocimiento Tácito

En el campo de la Gestión del Conocimiento, Nonaka y Takeuchi constituyen uno de los referentes pioneros. Entre sus trabajos [Nonaka et al., 1995], han realizado una clasificación del conocimiento distinguiéndolo en dos tipos: el explícito y el tácito. El conocimiento explícito es el saber que puede ser transmitido o compartido entre las personas o en el seno de la organización con relativa facilidad. Es un conocimiento formal y sistemático y puede ser estructurado, almacenado y distribuido. Es el conocimiento capturado y codificado en manuales, normas y procedimientos, por lo tanto resulta sencillo de transmitir. Como ejemplo puede mencionarse un procedimiento para resolver una ecuación matemática de segundo grado o la fórmula para hallar el área de un rectángulo.

Por su parte, el conocimiento tácito es el saber que presenta cierta dificultad y complejidad para ser transmitido o comunicado interpersonalmente. Es aquél que forma parte de las experiencias de aprendizaje personales que reside en cada individuo y que, por tanto, resulta difícil de capturar, estructurar, almacenar y distribuir. Son modelos mentales, intuiciones, creencias, perspectivas relacionadas con la concepción personal del mundo, habilidades técnicas o know how. Es difícil de transmitir verbalmente, puede difundirse a través de aprendizaje o experiencias personales. Es el conocimiento que se sabe que uno lo tiene, pero no se da cuenta de que lo está utilizando, simplemente se lo lleva a la práctica de una manera habitual. Como ejemplo puede mencionarse habilidades, como andar en bicicleta, o desarrollar un programa en Java.

3.2.4 Conocimiento Formal y Conocimiento Informal

El conocimiento formal es aquel que aparece en libros, manuales, documentos y cursos de perfeccionamiento. Es el producto primario del conocimiento del trabajador en forma de informes y plantillas. Las organizaciones confían en este conocimiento, sin mucho éxito, como su propia Memoria Organizacional [Conklin, 1996].

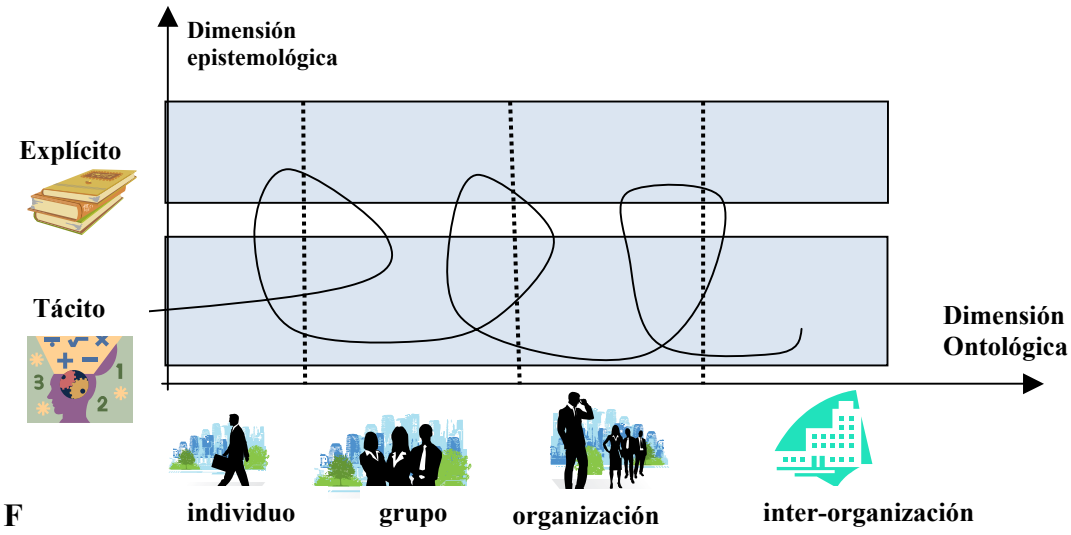
El conocimiento informal es el conocimiento creado y usado en el proceso de creación de resultados formales. Si el conocimiento formal es la parte visible, el conocimiento informal es el que queda atrás de este otro. Incluye ideas, testimonios, presunciones, significados, preguntas, decisiones, incógnitas, historias y puntos de vista. En una organización, es tan importante como el conocimiento formal pero es más efímero y transitorio y difícil de capturar [Conklin, 1996].

3.2.5 Proceso de Creación del Conocimiento

El modelo propuesto por Nonaka y Takeuchi respecto a la creación de conocimiento organizacional, es uno de los más aceptados y reconocidos [Nonaka et al., 1995]. Describe al proceso de creación de conocimiento que se inicia analizando dos dimensiones:

- *La dimensión epistemológica, que diferencia dos tipos de conocimiento: el conocimiento explícito y el conocimiento tácito.*

- *La dimensión ontológica, que reconoce cuatro niveles de agentes creadores de conocimiento: el individuo, el grupo, la organización y el nivel inter-organizacional.*



F

F

Figura 3.3 Las dos dimensiones en la creación del conocimiento

El nuevo conocimiento siempre se inicia en forma tácita en el individuo, pero ese conocimiento individual a través de un proceso de exteriorización se transforma en conocimiento explícito valioso para toda la empresa (ver figura 3.3).

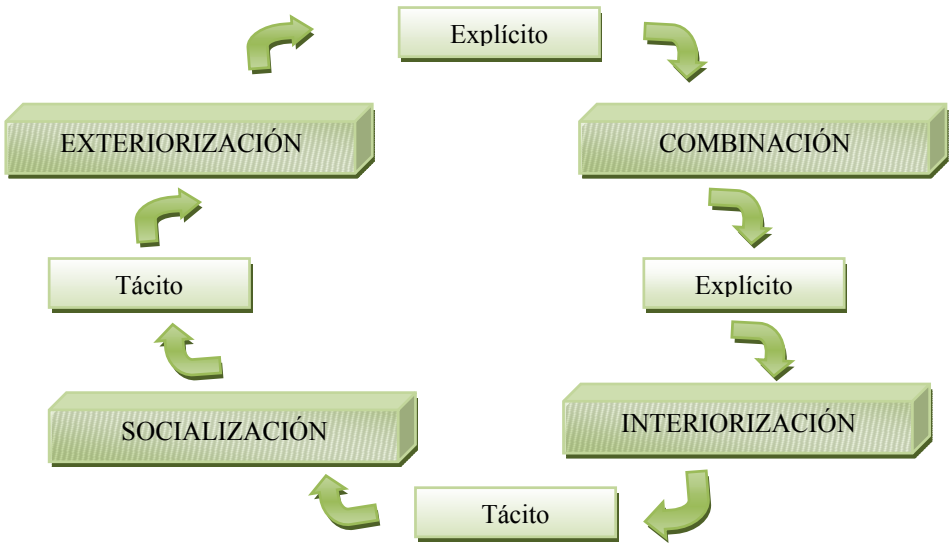


Figura 3.4 Proceso de conversión del conocimiento en la organización

Las dos dimensiones conforman un modelo en "espiral" de conocimiento, en el cuál el conocimiento es creado a través de la interacción dinámica entre los diferentes modos de conversión del conocimiento. De esta manera, el proceso es iterativo en su dimensión epistemológica, e incremental en su dimensión ontológica, formando una

espiral permanente de transformación del conocimiento, desarrollada siguiendo cuatro fases, que pueden ser observadas en la figura 3.4.

A continuación se describen las cuatro fases que se llevan a cabo en la creación del conocimiento:

- *Socialización* (de tácito a tácito): La socialización es el proceso de adquirir conocimiento tácito a través de compartir experiencias por medio de exposiciones orales, documentos, manuales, tradiciones y creencias (cultura organizacional) y que añade el conocimiento novedoso a la base colectiva que posee la organización. Un individuo puede adquirir conocimiento tácito directamente de otros sin usar el lenguaje (a través de la observación, la imitación y la práctica). La clave para obtener conocimiento tácito es la experiencia.

- *Exteriorización* (de tácito a explícito): La exteriorización es el proceso de convertir conocimiento tácito en conceptos explícitos. El conocimiento tácito se vuelve explícito adoptando la forma de metáforas, analogías, conceptos, hipótesis o modelos. La metáfora es una forma de percibir o entender intuitivamente una cosa imaginando otra cosa simbólicamente. La exteriorización es la actividad esencial en la creación del conocimiento y es generada por el diálogo o la reflexión colectiva.

- *Combinación* (de explícito a explícito): La combinación es el proceso de crear conocimiento explícito al reunir conocimiento explícito proveniente de cierto número de fuentes mediante el intercambio de conversaciones telefónicas, reuniones, correo, etc. Y se puede categorizar, confrontar y clasificar para formar bases de datos para producir conocimiento explícito. La combinación es un proceso de sistematización de conceptos con el que se genera un sistema de conocimiento. Como ejemplo puede mencionarse el conocimiento que se dan en escuelas gracias a la educación y al entrenamiento formal.

- *Interiorización* (de explícito a tácito): La interiorización es el proceso de incorporación de conocimiento explícito en conocimiento tácito (muy relacionada con "aprender haciendo"), que analiza experiencias adquiridas en la puesta en práctica de los nuevos conocimientos y que se incorpora en las bases de conocimiento tácito de los miembros de la organización.

3.3 Gestión del Conocimiento

Los Sistemas de Administración del tienen como objetivo administrar y almacenar el conocimiento organizacional, de manera que después pueda ser utilizado para aprender, resolver problemas y como apoyo en la toma de decisiones [Dogson, 1993] [Conklin, 1996]. Además, promueven el ahorro de tiempo y costo, permitiendo conocer los errores y aciertos del pasado para mejorar el presente sin repetir esos errores, tomando ventaja de las lecciones aprendidas.

Las empresas modernas deben ser capaces de gestionar eficientemente el conocimiento que poseen para hacer frente a los desafíos que presenta el mercado actual a las organizaciones [Drucker, 1998]. Sin embargo, con frecuencia, los gerentes organizacionales no pueden identificar dónde reside el valor del conocimiento que poseen, como gestionarlo, ni cómo usarlo como ventaja competitiva. Como se mostró en el capítulo anterior, en la literatura asociada se describen una multiplicidad de modelos e iniciativas para administrar el conocimiento organizacional, cada uno de ellos se enfoca en ciertos elementos de la Gestión del Conocimiento desde distintas perspectivas.

Una estrategia eficiente de Gestión del Conocimiento debe estar basada en un entendimiento exhaustivo de lo que ello implica. En la siguiente sección se muestra un resumen de algunas consideraciones entorno a la Gestión del Conocimiento.

3.3.1 Definición y Objetivos de la Gestión del Conocimiento Organizacional

Si bien los orígenes de la ciencia relacionada al conocimiento se remontan a muchos años atrás, la mayoría de los trabajos estaban situados en un contexto filosófico y se enfocaban fundamentalmente en la definición y análisis del conocimiento, pero no en el esfuerzo sistemático de gestionarlo.

La conceptualización de la Gestión del Conocimiento tuvo su principal impulso cuando conocimiento se convirtió en un factor central para la producción y la innovación en las empresa a partir de los 90s. En ese sentido, Peter Drucker fue uno de los primeros que abogó por el advenimiento de la sociedad del conocimiento, documentando la transformación desde el Capitalismo a una Sociedad del Conocimiento, puntualizando que el recurso más importante ya no era el capital, la tierra o el trabajo, sino más bien, el conocimiento [Drucker, 1998]. El dominio de la Gestión del Conocimiento también ha sido modelado por la experiencia y filosofía de la sociedad oriental. Nonaka y Takeuchi han sido pioneros en el modelado para mostrar la dinámica de la creación de conocimiento (como se mostró en la sección 3.2.5).

Aunque existe un creciente reconocimiento de la importancia de la Gestión del Conocimiento organizacional, muchas organizaciones no han desarrollado actividades de gestión de una forma estructurada porque todavía no comprenden el alcance de este tipo de iniciativa o qué actividades implica. No existe un consenso de lo que significa la "*Gestión del Conocimiento*", y esta es la razón por la que se encuentran tantas definiciones en la literatura.

De acuerdo a O'Leary, la Gestión del Conocimiento puede ser definida como los esfuerzos hechos por la organización para capturar el conocimiento, convertir el conocimiento personal (tácito) en conocimiento disponible grupalmente (explícito), conectar personas con personas, personas con conocimiento, conocimiento con conocimiento, y medir el conocimiento para facilitar la gestión de recursos y para ayudar a entender su evolución [O'Leary, 2002]. Smith, por su parte, describe al objetivo de la Gestión del Conocimiento como el mejoramiento del desempeño organizacional permitiendo que los individuos capturen, compartan y apliquen su conocimiento colectivo para tomar decisiones óptimas en tiempo real [Smith et al., 2000]. Skyrme

sugiere que la Gestión del Conocimiento es la gestión intencionada y sistemática del conocimiento vital junto con sus procesos asociados de creación, recolección, organización, difusión, uso y explotación de ese conocimiento [Skyrme, 2003].

En definitiva, la Gestión del Conocimiento tiene por objeto detectar, organizar, compartir, presentar y usar el conocimiento organizacional con miras a su reuso y explotación en forma cooperativa en pro de la mejora de los procesos y productos de la empresa.

Si bien Gestión del Conocimiento no es gestión de información, y sobre todo, no es implementar un sistema informático, el uso de las tecnologías de la información puede ser un gran facilitador de la Gestión del Conocimiento. En este sentido, definimos un Sistema de Gestión del Conocimiento como una infraestructura creada por la organización para implementar los procesos y los procedimientos que, actuando sobre un repositorio de información y de conocimientos (es decir una *Memoria Organizacional*), permitan generar, compartir y reutilizar el conocimiento formal e informal existente en la organización para dar respuesta a las necesidades de los individuos y de las comunidades y de esta manera contribuir a la mejora de la competitividad de la empresa.

3.3.2 Ventajas e Inconvenientes de la Gestión del Conocimiento Organizacional

El principal objetivo de la Gestión del Conocimiento es apoyar y mejorar los procesos y la competitividad de las empresas. Los gerentes de las organizaciones no ignoran que ellos mismos y sus empleados necesitan aprender y formarse continuamente para ser más eficaces, mejorar el servicio a sus clientes, reducir costos e innovar.

De acuerdo a Davenport [Davenport et al., 1998] las ventajas de la Gestión del Conocimiento pueden enumerarse de la siguiente manera:

- Genera ventaja y diferenciación competitiva, en el sentido que dicha diferenciación no proviene de la cantidad de conocimiento que una empresa pueda reunir, sino del uso que se le dé. La Gestión del

Conocimiento permite diferenciar la información valiosa de la que no lo es, agilizando los tiempos de respuesta y permitiendo tomar ventajas sobre la competencia por el valor de sus activos intangibles.

- Incrementa el conocimiento individual y grupal de la empresa.
- Reduce la "amnesia corporativa" que se produce cuando el único que conoce cómo desarrollar alguna tarea se retira de la empresa.
- Las tecnologías de la información y comunicaciones se aprovechan de manera más productiva.
- Fomenta una cultura de compartir experiencias y conocimientos entre los miembros de la empresa.
- Reduce los tiempos de las decisiones ya que permite una solución semejante a problemas parecidos.

También existen posibles inconvenientes que se han encontrado al gestionar el conocimiento en una organización; algunos de ellos son:

- Exige compromiso por parte de los empleados. Los empleados deben comprometerse y estar dispuestos a aprender a compartir el conocimiento que ellos poseen y que es beneficioso para la organización. Esto será posible sólo si la organización posee una cultura orientada hacia esta práctica y políticas que la apoyen. Se debe encontrar la forma de motivar a la gente para que esté dispuesta a compartir aquellos conocimientos útiles para la organización.
- Fomentar el aprendizaje continuo y el compartir en una cultura organizacional es un cambio difícil de sobrellevar y requiere mucho tiempo.
- Se debe saber capturar. No toda la información que circula dentro de la organización es valiosa, por lo cual se deberá encontrar a la gente apropiada para que maneje el proceso de implementación y administración de la Gestión del Conocimiento. Deberá ser personal con

la habilidad y capacidad para poder seleccionar sólo aquello relevante y aprovechar así el conocimiento para obtener una ventaja competitiva y lograr una eficiente administración del conocimiento acumulado.

- Obtener conocimiento de los clientes no siempre es factible. Si bien esto puede realizarse a través de encuestas, los clientes no siempre están dispuestos a revelar su información.

3.3.3 El Proceso de Gestión del Conocimiento

Existe una importante y creciente literatura que enfatiza la importancia del proceso de Gestión del Conocimiento y, en el campo académico, los investigadores han desarrollado propuestas para ayudar a entender la problemática asociada [Nonaka et al., 1995], [Wiig, 1997], [Alavi, 1997], [Van der Spek et al., 1997], [Davenport et al., 1998b], [Chen et al., 2006]. Pueden distinguirse una variedad de modelos que presentan metodologías a seguir en la conducción del proceso de Gestión del Conocimiento. Por ejemplo, el modelo pilares de Wiig identifica las principales funciones necesarias para la Gestión del Conocimiento y las organiza en tres pilares [Wiig, 1997], estos pilares se basan en un entendimiento amplio de lo que es la creación, manifestación, uso y transferencia de conocimiento.

Por otro lado, el modelo de Van der Spek y Spijkerver identifica un ciclo de cuatro etapas en la Gestión del Conocimiento: conceptualización, reflexión, actuación y retrospección [Van der Spek et al., 1997]. La etapa de conceptualización se enfoca en ganar comprensión de los recursos de conocimiento. Esto es alcanzado mediante la captura y clasificación del conocimiento existente. Durante la etapa de reflexión, se evalúa el conocimiento conceptualizado y se establecen requerimientos de mejora. La etapa de actuación involucra desarrollar nuevo conocimiento, además de la distribución y combinación del conocimiento desarrollado. La última etapa, la retrospección, evalúa los resultados alcanzados en la etapa de actuación y compara la situación anterior y la nueva. El establecimiento de estas etapas de Gestión del Conocimiento está orientada hacia un ciclo de solución de problemas.

El modelo de Alavi define a la Gestión del Conocimiento como la creación, apalancamiento y distribución del conocimiento laboral (know-how) y bienes

intelectuales por todos los individuos a lo largo de la organización para mejorar el servicio a clientes [Alavi, 1997]. El modelo de proceso de Gestión del Conocimiento consiste en una secuencia de seis etapas: adquisición, indexado, filtrado, enlazado, distribución y aplicación. Por otro lado, Chen y Chen, clasifica estas actividades en cuatro categorías: creación, conversión, circulación y desarrollo [Chen et al., 2006].

Como podemos concluir de la literatura citada (aunque no es exhaustiva), que si bien todos los modelos son diferentes, en la mayoría de ellos, la Gestión del Conocimiento presupone como mínimo cuatro etapas básicas [Chu et al., 2008] que en esta tesis serán llamadas: adquisición, resguardo, distribución y aplicación (o reuso), como puede observarse en la figura 3.5.

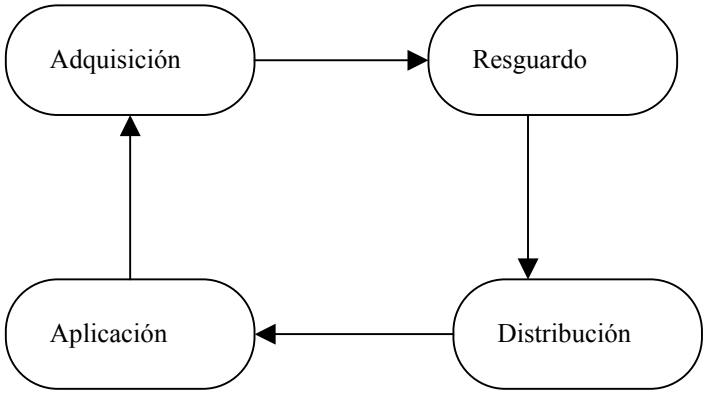


Figura 3.5 Proceso de Gestión del Conocimiento

La **adquisición** es la etapa de detectar modelos cognitivos de valor para la organización que residen en las personas, de la creación del conocimiento y desarrollo de contenido. Esto es conseguido mediante la captura de las experiencias y lecciones aprendidas y mediante la recolección, sintetizando e interpretando una gran variedad de conocimiento que puede tener su origen en fuentes internas (I+D, proyectos, descubrimientos, etc) o externas (publicaciones periódicas, Internet, cursos, libros, etc). Se asemeja a la etapa de creación en los modelos de Wiig, y Nonaka y Takeuchi. Con respecto al modelo de Alavi, esta etapa se corresponde con las etapas de adquisición e indexación, y comparándola con los modelos de Davenport et al., y Chen et al. denominan a esta instancia creación. En el gráfico de la figura 3.6, se resume una comparación de los modelos de procesos de distintos autores, con las etapas propuestas en esta tesis.

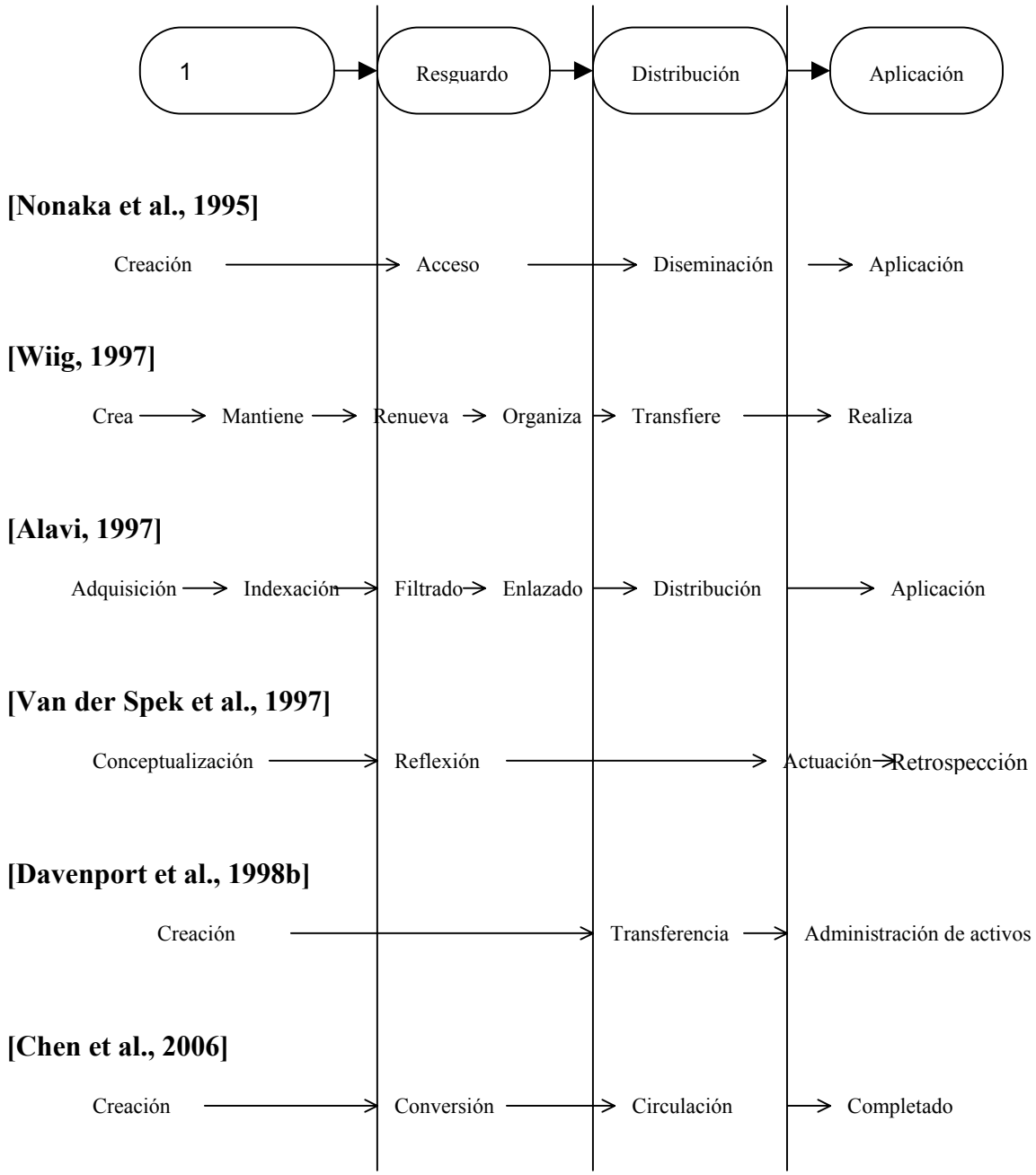


Figura 3.6 Comparación de procesos de gestión de conocimiento

La etapa de **resguardo** es el proceso de almacenar en forma estructurada la representación explícita del conocimiento. Tiene que ver con el filtrado, organización, codificación, interconexión y almacenamiento del conocimiento, incluye también las denominadas actividades de gestión de memoria como la clasificación, catalogación, integración, interrelación y visualización del contenido tanto formal como informal.

La etapa de **distribución** es el proceso de apertura de la base de conocimientos facilitando su transferencia y diseminación en los momentos y lugares en que se

requieran. Esto implica establecer interfaces de acceso masivo (por ejemplo Internet o intranets) y mecanismos de seguridad adecuados, implementar facilidades de búsqueda y filtrado (tanto para agentes humanos como para máquinas y considerar aspectos temporales (por ejemplo vencimiento) y de distancias.

La **aplicación** se refiere al uso del conocimiento que ha sido recolectado, capturado y distribuido para producir productos y servicios. Específicamente, es el acto de aplicarlo a la resolución de un problema o la toma de decisiones. Del uso del conocimiento en distintas experiencias surgirá nuevo conocimiento o el desarrollo del mismo, que generará una retroalimentación en la adquisición de nuevo conocimiento (ver figura 3.5).

Una actividad central en la Gestión del Conocimiento es el resguardo del mismo, de manera que luego pueda ser compartido y explotado en su aplicación. Una forma de almacenar en forma estructurada el conocimiento es lo que se denomina Memoria Organizacional. La Memoria Organizacional constituye el almacenamiento de todo el conocimiento formal e informal presente en la organización [Conklin, 1996]. En la siguiente sección, se introducen algunos conceptos de Memoria Organizacional.

3.4 Memoria Organizacional

De todos los aspectos de Gestión del Conocimiento vistos en la sección anterior, este trabajo de tesis se enfoca principalmente en el conocimiento en su dimensión explícita, en particular en los procesos de retención para su uso futuro (resguardo), búsqueda, recuperación y diseminación (distribución) y su eventual reutilización (aplicación). En la literatura , la disciplina que estudia estos procesos se conoce como "Memoria Organizacional". Teniendo en cuenta el enfoque de proceso de conversión del conocimiento de Nonaka y Takeuchi, centraremos nuestra atención especialmente en el modo de exteriorización del conocimiento que involucra principalmente la codificación del conocimiento y su posterior reutilización.

Como ya fue mencionado, el conocimiento organizacional se compone además del conocimiento formal que es fácilmente codificado y documentado, del conocimiento informal que consiste de experticias en la resolución de problemas, diseños técnicos y

lecciones aprendidas. La integración coherente de estas habilidades dispersas en una corporación, apuntando a facilitar su acceso y reuso, componen lo que se llama "memoria corporativa" o "Memoria Organizacional". La Memoria Organizacional se considera el requisito previo central como soporte tecnológico a la Gestión del Conocimiento y es el medio para la conservación de conocimiento, su distribución, y reuso; además, habilita el aprendizaje organizacional y la mejora continua de los procesos. Consecuentemente, el término Memoria Organizacional ha sido impregnado de distintos significados en teorías organizacionales de las más variadas.

3.4.1 Definición

De las diversas definiciones de Memoria Organizacional que existen en la literatura, aquella que va a guiar la conducción de este trabajo está relacionada con la capacidad de una organización de beneficiarse de las experiencias pasadas para guiarse en el futuro [Ackerman et al., 1990], caracterizada por la conversión del conocimiento informal tácito en conocimiento explícito y la incorporación del conocimiento individual a la organización [Yates, 1990]. Complementariamente, Huber [Huber, 1991] define Memoria Organizacional como "*el medio por el cuál el conocimiento es almacenado para su uso futuro*". Esta definición se asemeja a la de Ackerman y Malone [Ackerman et al., 1990]en la medida que la utilidad de la Memoria Organizacional "para qué forma parte de su propia definición.

Otros autores la definen como: "*una representación explícita, incorpórea y persistente de conocimiento e información en una organización para facilitar su acceso y reuso por los miembros de la organización, en sus tareas*" [Van Heijst et al., 1996], [Rabarijaona et al., 1999], [Despres, 2000], [Ale, 2009].

En esta tesis, y teniendo en cuenta la discusión precedente, se define Memoria Organizacional como : "*La manera en que una organización resguarda y lleva cuenta de lo que sabe para beneficiarse con su uso futuro"*.

También es importante resaltar la distinción entre Memoria Organizacional y conocimiento organizacional. Ackerman complementa la definición de Memoria Organizacional afirmando que es el conocimiento organizacional con persistencia

[Ackerman, 1994]. El mismo autor también hace una distinción bastante simple, basada en la temporalidad del conocimiento: "si el conocimiento organizacional es el ahora, la Memoria Organizacional es lo que antecede al ahora".

Si bien la diversidad de definiciones es amplia, en la mayoría de los casos observamos que se considera a una Memoria Organizacional desde dos aspectos diferentes [Ackerman et al., 2000]: como un objeto, "*representación explicita*", "*almacenamiento de lo que la organización sabe*" y como un proceso "*facilitar su acceso y reuso*", "*resguarda y lleva cuenta*". Es importante enfatizar la diferencia entre el aspecto estático, que consiste en el conocimiento capturado o "*contenido"* y el aspecto dinámico de la memoria, la capacidad de memorizar y recordar el *comportamiento*, ya que ambos aspectos deben estar presentes en una solución completa de Memoria Organizacional.

Respecto al *contenido* puede decirse que una Memoria Organizacional contiene todo el conocimiento formal (procesos de negocios, procedimientos, políticas, misión, reglas y normas) e informal, (conocimiento basado en la experiencia adquirida por los empleados relacionada con el trabajo que realizan). Es un repositorio de conocimiento y conocimiento laboral (know-how) de un conjunto de individuos que trabajan en una organización [Conklin, 1996], [Abecker et al., 1998]. Constituye una representación explícita, y persistente de la totalidad del conocimiento de la organización .

Respecto al *comportamiento* podemos decir que la principal tarea de una Memoria Organizacional es hacer disponibles los bienes intangibles de conocimiento de una empresa, en el tiempo y lugar que se requieran. De esta manera constituye un valioso soporte a la toma de decisiones, contribuye a mejorar la eficiencia y efectividad de los procesos intensivos en conocimiento de la organización y mejora la capacidad de aprendizaje de sus empleados. Para lograr estos objetivos, la Memoria Organizacional debe contar con mecanismos eficientes que faciliten el acceso, difusión y reutilización de la totalidad del conocimiento entre los miembros de una organización y los sistemas de información [Von Krogh, 1998].

Cabe destacar en este punto que el deseo de controlar y gestionar el conocimiento organizacional a través de las memorias organizacionales, contrasta con la naturaleza fluida, dispersa, intangible, subjetiva y con frecuencia tácita del conocimiento. Esto dificulta la implementación de soluciones de Memoria

Organizacional que sean completas en cuanto a su contenido y además eficientes y pro-activas en cuanto a su comportamiento. Frecuentemente, al no encontrar un modelo de gestión de Memoria Organizacional completo y eficiente que se adecue a una situación en particular, las organizaciones desarrollan soluciones propias que dan origen a una multiplicidad de tipos de memorias organizacionales y sistemas de gestión. En la siguiente sección, se presenta una clasificación de memorias organizacionales, teniendo en cuenta sus dos principales aspectos: contenido y comportamiento.

3.4.2 Clasificación

Algunos autores clasifican las memorias organizacionales dependiendo del conocimiento contenido en ellas [Dieng et al. 1999] en:

- *Memoria profesional*: compuesta por los métodos, técnicas y documentos de referencia usados en una profesión dada.

- *Memoria de gestión*: relacionada a la organización, sus actividades, sus productos. Captura las estructuras organizacionales pasadas y presentes (recursos humanos, gestión, etc.). Esta memoria está extremadamente cercana al modelado organizacional.

- *Memoria individual*: caracterizada por habilidades, competencias, conocimiento laboral (know-how) y actividades de un miembro dado de la organización.

- *Memoria de proyecto*: se adquiere en el contexto de un proyecto que debe ser almacenado junto con el conocimiento para preservar su significado. Comprende la definición de un proyecto, las actividades, historia y resultados. Se usa para capitalizar lecciones y experiencias de proyectos anteriores.

Desde un punto de vista del comportamiento, Van Heijst y colaboradores [Van Heijst et al., 1996] presentan una clasificación basada en cómo son los procesos de Gestión del Conocimiento que se pretenden implementar. Y agrupa las memorias según

sean estos procesos más o menos pro-activos (respecto a la adquisición y distribución del conocimiento), resultando en cuatro tipos de memorias organizacionales distintas [Van Heijst et al., 1996], tal como se observa en la Figura 3.7.

Desván de conocimiento: Es una Memoria Organizacional usada como un repositorio que puede ser consultado y actualizado cuando sea necesario. No es intrusivo, pero requiere de una gran disciplina por parte de los miembros de la organización para no quedar obsoleto.

	Adquisición Pasiva	**Adquisición Activa**
Distribución Pasiva	Desván de conocimiento	Esponja de conocimiento
Distribución Activa	Publicación de conocimiento	Bombardeo de conocimiento

Figura 3.7 Tipos de Memorias Organizacionales (adaptada de Van Heijst et al., 1996)

Esponja de conocimiento: Es una Memoria Organizacional que se alimenta activamente para mantenerse más o menos completa. Su uso queda como responsabilidad de los miembros de la organización.

Publicación de conocimiento: Es una Memoria Organizacional donde la contribución se deja a los trabajadores individuales mientras que los mantenedores de la memoria analizan el conocimiento entrante y lo combinan con el conocimiento ya almacenado y reenvían las noticias relevantes a miembros potencialmente interesados.

Bombardeo de conocimiento: es una Memoria Organizacional que asegura que el conocimiento desarrollado en la organización es efectivamente capturado y usado desde y por los miembros de la organización.

3.4.3 Las TIC como Factor Determinante para la Gestión de la Memoria Organizacional

La mayoría de la literatura relacionada a la Gestión del Conocimiento organizacional afirma que la Gestión del Conocimiento depende más de las personas que de las tecnologías, pero reconoce también que el avance y disponibilidad de nuevas tecnologías de información le han impreso un gran impulso a la Gestión del Conocimiento [Davenport et al., 1998b], [Grupta et al., 2000], [Alavi et al., 2001].

Como se mostró en el capítulo anterior, la asociación entre TI y Gestión del Conocimiento, normalmente está asociada al uso de sistemas de información para compartir información o conocimiento, los llamados sistemas de administración del conocimiento (KMS). Debido a que las TI no se aplican completamente a todas las áreas de Gestión del Conocimiento, puede dar soporte a esta práctica de muchas maneras distintas [Alavi et al., 2001]. Por ejemplo, en la búsqueda de un experto o una fuente de conocimiento utilizando un directorio o base de datos, dando apoyo a la compartición de conocimiento, facilitando el trabajo colaborativo, brindando acceso a la información de proyectos del pasado, entre otras. No existe un rol único de las TI en la Gestión del Conocimiento, como tampoco existe una única tecnología que soporte todo el proceso de Gestión del Conocimiento.

Respecto al uso de TI en la gestión de memorias organizacionales, la revisión de literatura hecha por Alavi et al., revela que existen tres áreas comunes en las que se aplican, a saber: (1) codificación y compartición de lecciones aprendidas y buenas prácticas, (2) creación de directorios de conocimiento organizacional y (3) creación de redes de conocimiento. El mismo autor afirma que el almacenamiento de lecciones aprendidas y buenas prácticas es la aplicación más común.

En este trabajo, el interés y desarrollo se centrará en el resguardo y compartición del conocimiento informal contenido en lecciones aprendidas y buenas prácticas. Este conocimiento, para que pueda ser recuperado para su reuso, antes debe ser registrado de alguna manera. Para ello, el conocimiento necesita trascender el nivel individual y transformarse en conocimiento colectivo de la organización [Davenport et al., 1998a].

El problema es que tanto el proceso de codificación y resguardo como de su conversión en su dimensión ontológica (pasaje de individual a colectivo), son costosos en cuanto al consumo de recursos humanos e influyen negativamente en la motivación por parte de las organizaciones para que sus integrantes inviertan un tiempo precioso en el registro y distribución de lecciones aprendidas [Fischer, 2002]. Aquí es donde las TIC tienen una indiscutible importancia como apoyo para la gestión de la Memoria Organizacional. En los próximos dos capítulos, analizaremos la importancia del Razonamiento Basado en Casos y las ontologías respectivamente, como tecnologías centrales para la gestión de la Memoria Organizacional.

Capítulo 4- Razonamiento Basado en Casos

4.1 Introducción

El Razonamiento Basado en Casos (CBR, del inglés *Case Based Reasoning*) surgió alrededor de los años ’70 en los Estados Unidos, como otro paradigma de resolución de problemas en ambientes de IA. Pero son precisamente sus diferencias con el resto de los enfoques de resolución de problemas en la IA lo que le otorga un especial interés en diversas comunidades. Así, en lugar de basarse en reglas que modelen el comportamiento del dominio del problema o en la generalización de relaciones entre descriptores de problemas y soluciones, el CBR utiliza el conocimiento específico obtenido a partir de situaciones concretas previamente experimentadas: los casos. Ante una nueva situación, considerada un nuevo caso, la misma se resuelve encontrando un caso pasado similar y adaptando su solución a la situación del problema nuevo mencionado .

Aamodt y Plaza [Aamodt et al., 1994], considerados referentes en el tema, realizan una definición para el CBR, argumentando que implica “*resolver un problema nuevo recordando una situación similar previa y reutilizando su información y conocimiento”*.

Esta definición no se aleja mucho al mecanismo de razonar que llevan las personas, de hecho, el Razonamiento Basado en Casos es una manera de razonar haciendo analogías [Sankar et al., 2004]. Sin percibirlo, el ser humano aplica Razonamiento Basado en Casos para solucionar problemas cotidianos, por ejemplo: un mecánico de automóviles que sabe cómo reparar un motor porque recordó que otro auto presentaba los mismos desperfectos, o un médico que, examinando un nuevo paciente recuerda un caso parecido un tiempo atrás, encuentra una similitud importante de los síntomas y decide asumir que posee la misma enfermedad y la trata de la misma manera, pues el tratamiento resultó efectivo en la ocasión anterior, están usando Razonamiento Basado en Casos. Y son numerosas las situaciones en las que, casi sin

intención alguna, el cerebro razona realizando comparaciones sobre experiencias pasadas exitosas conocidas y las aplica en la resolución de una nueva situación, un nuevo problema.

En la terminología de CBR, un caso denota la situación de un problema, en particular, una situación previamente experimentada, la cual ha sido capturada y aprendida de tal manera que pueda ser reutilizada para resolver problemas futuros, se denomina un caso previo, un caso almacenado o un caso guardado. Así, un caso nuevo o un caso sin resolver no es más que la descripción de un problema nuevo a resolver. Por lo tanto, el Razonamiento Basado en Casos se convierte en un proceso cíclico en el que se resuelve un problema, se aprende de esta experiencia, se resuelve un nuevo problema, etc. [Aamodt et al., 1994].

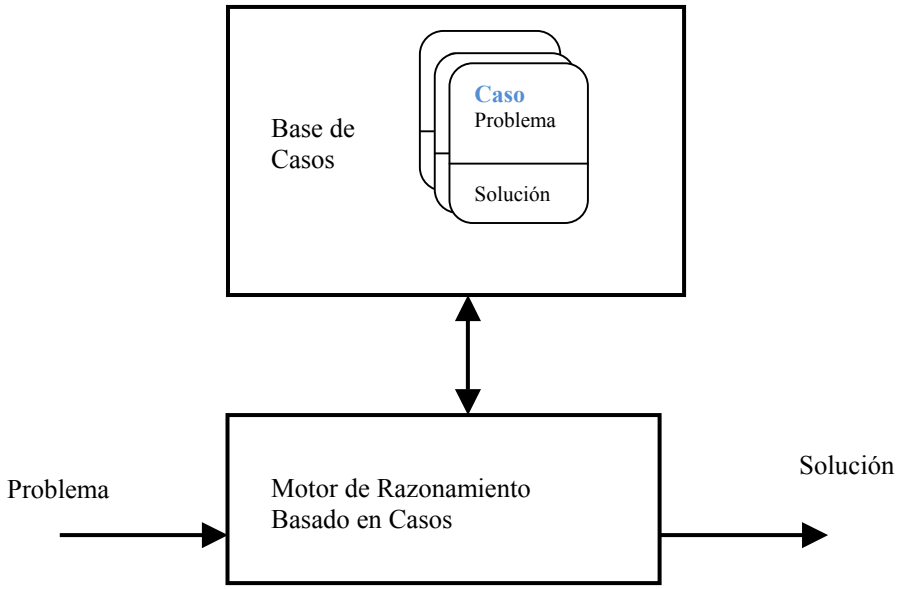


Figura 4.1. Sistema de Razonamiento Basado en Casos (adaptado de [Sankar et al., 2004])

A un nivel más alto de abstracción [Sankar et al., 2004], un sistema de Razonamiento Basado en Casos puede verse como una caja negra (ver Fig. 4.1) que incorpora el motor del razonamiento y las facetas externas siguientes:

- La entrada de una especificación de problema a resolver
- La salida que define una solución sugerida al problema de entrada

- La memoria de casos pasados, la base de casos, que es referenciada por el motor de razonamiento.

4.2 Características Generales de CBR

El Razonamiento Basado en Casos se sustenta en un modelo de razonamiento que incorpora los aspectos ya mencionados en la sección anterior de resolución de problemas, entendimiento y aprendizaje e integra todo ello en meros procesos de memoria (como será analizado en la sección 4.9).

En resumen, estas son las características subyacentes al modelo CBR [Sankar et al., 2004], [Watson, 2003] :

La referencia a casos pasados es muy importante y de gran utilidad para tratar situaciones que vuelven a darse. Por lo tanto, memorizar situaciones similares es a menudo necesaria para tratar la complejidad de una nueva situación. La actividad de recordar un caso para usarlo en un problema futuro (e integrar ambos) es, necesariamente, un proceso de aprendizaje.

Debido a que las descripciones de los problemas son, a menudo, incompletas, es necesario una etapa de entendimiento o interpretación, ya que no puede llevarse a cabo un razonamiento, una resolución adecuada de una nueva situación, si ésta no se entiende con cierta completitud. Se puede considerar que esta etapa es a la vez un prerequisito y una parte del ciclo de razonamiento, pues el entendimiento de las situaciones mejora conforme progresa el razonador.

No obstante, cualquier forma de razonamiento necesita que la situación sea elaborada con suficiente detalle y representada con suficiente claridad y con el vocabulario apropiado para que el razonador reconozca el conocimiento que necesita (sea conocimiento general del dominio o casos específicos) para razonar a partir de él. La práctica demuestra que no suele existir un caso pasado exactamente igual a un caso nuevo. Por ello, es usual tener que adaptar la solución pasada para que se ajuste a la nueva situación.

El aprendizaje es una consecuencia natural del Razonamiento Basado en Casos. Si se halla un nuevo procedimiento en el curso de la resolución de un problema complejo y su ejecución resulta positiva, entonces se aprende el nuevo procedimiento para resolver esta nueva clase de situaciones.

Estas premisas sugieren que la calidad de un sistema de Razonamiento Basado en Casos depende de:

- La experiencia que tiene (cantidad de casos previos).
- La habilidad para entender situaciones nuevas en términos de experiencias pasadas.
- Su capacidad de adaptación.
- Su habilidad para integrar nuevas experiencias en su memoria adecuadamente.

4.3 Orígenes y Antecedentes del CBR

El precursor de estas ideas fue Schank, [Aamodt et al., 1994]. Sus trabajos se basaban en la suposición de que las personas almacenan conocimiento en forma de guiones que permiten hacer predicciones y realizar inferencias [Schank, 1982]. Estos guiones describen situaciones típicas como por ejemplo, ir al médico o a comer en un restaurante. Aunque los guiones resultan ser un modelo incompleto de memoria, sirvieron como base para los trabajos posteriores de Schank y su grupo de investigadores en la Universidad de Yale a principios de los años ‘80.

El primer sistema al que se le puede apodar el nombre de “razonador basado en casos” fue CYRUS, desarrollado por Kolodner en la Universidad de Yale [Kolodner, 1983a], [Kolodner, 1983b]. CYRUS está basado en el modelo de memoria dinámica de Schank y en teorías sobre resolución de problemas (MOP) y aprendizaje. La aplicación almacena y recupera información sobre personas importantes, basándose en la manera en que las personas organizan y recuerdan información sobre ellos mismos. Almacena información relacionada a viajes y encuentros de los secretarios de estados de la época,

Cyrus Vance y Edmundo Muskie. El modelo de memoria basada en casos desarrollada para este sistema ha servido como base para otros sistemas de CBR desarrollados posteriormente.

Tras la aparición de CYRUS y su prominente acercamiento al modelo de Razonamiento Basado en Casos, el surgimiento de nuevos sistemas CBR no tardaron en llegar: Mediator, un sistema de arbitraje [Simpson, 1985]; CHEFF, un sistema que diseña recetas de comidas chinas [Hammond, 1990]; CASEY para el diagnóstico de enfermedades cardiovasculares [Koton, 1989]; y JULIA para el diseño de menús [Hinrichs, 1992]. Merece mención también el trabajo de Porter y su grupo en la Universidad de Texas que, a fines de los '80 desarrollaron el sistema PROTOS [Bareiss, 1989] aplicado al diagnóstico de enfermedades del aparato auditivo.

4.4 Representación del Conocimiento usando Casos

Como se expresó en el capítulo anterior, el contenido de una Memoria Organizacional va desde lo documentado (reportes de proyectos, modelos, planificación, manuales de procedimientos, etc.) hasta lo no documentado (experiencias, formas de pensar, anécdotas) pero que forman el acervo cultural, conocimientos y experiencias de los miembros de la empresa [Malhotra 1996]. Si bien el conocimiento documentado es un capital valioso dentro de una organización, es de vital importancia tener en la Memoria Organizacional aquel conocimiento que permita contestar preguntas como: ¿Porqué se hizo esto de cierta manera?, ¿Este problema ha sido resuelto antes?, ¿Qué aprendimos la última vez que sucedió ese problema?. Una de las maneras de lograrlo, es guardar estas experiencias por medio de casos.

Un caso constituye una pieza contextualizada de conocimiento que representa una experiencia. Contiene la lección pasada, que constituye el contenido del caso y el contexto en el cual la lección puede ser utilizada [Kolodner, 1993].

Los casos que representan un conocimiento específico relacionado a situaciones específicas, contienen un conocimiento en un nivel operacional; por ejemplo, hacer explícito cómo se efectuó una tarea o cómo se aplicó una parte del conocimiento o qué estrategias particulares se usaron para alcanzar una meta.

Típicamente un caso comprende:

- El problema que describe el estado del mundo cuando ocurrió el caso
- La solución que describe cómo se resuelve el problema.
- El resultado que describe la eficacia del resultado obtenido como consecuencia de la aplicación de determinada solución al problema.

CASO 1: Obtener el árbol de requerimientos para evaluar la eficiencia en desarrollo del grupo de trabajo Gidis_Web.

PROBLEMA:

Purpose	Evaluar
CalculableConcept	Eficiencia
ConceptModelType	ISO
UserView	Evaluador
EntityCategory	Grupo de Desarrollo

SOLUCION:

1. Eficiencia en Desarrollo
1.1. Productividad de Desarrollo
1.2. Completitud del Desarrollo
1.3. Duración del Desarrollo

Figura 4.2. Ejemplo de representación de un caso para el diseño de un árbol de requerimientos en aseguramiento de calidad.

De acuerdo a lo anterior, un caso se puede definir como la descripción detallada de una experiencia del pasado sobre una situación particular, formada por la descripción del problema, la solución tomada para resolver el problema y el resultado obtenido después de la aplicación de la solución. En la figura 4.2 se muestra un ejemplo de caso relacionado al área de medición y evaluación de calidad en Ingeniería de Software [Martín et al., 2008].

La información que se almacena para describir un problema depende tanto del dominio como del propósito para el que se usa el caso [Althoff et al., 1995]. En un sistema CBR es importante incluir también información acerca de la especificación

(estructura) del problema y metadatos para facilitar su procesamiento como así también atributos del ambiente que describan el entorno del problema (contexto) [Martin et al., 2009]. La descripción del problema debe incluir además las metas que se deben conseguir para resolver el problema, las restricciones de estas metas, así como las características de la situación del problema y las relaciones entre sus partes.

La descripción de la solución, será reusada cuando nos encontremos en una situación similar, por lo tanto hay que tener especial cuidado en la información que se almacena [Sankar et al., 2004]. Dependiendo de cómo el sistema razone con los casos, se debe incluir únicamente los hechos (atributos) que llevan (o caracterizan) a la solución, o información sobre pasos adicionales en el proceso de obtención de la solución. También es posible incluir soluciones alternativas junto con las razones y justificaciones que motivaron la elección de una a favor del resto, o también soluciones no admisibles rechazadas junto con sus motivos de su rechazo.

CASO 2: Casos de testing de un módulo con dos variables de entrada.

PROBLEMA: Diseñar Casos de testing para un módulo con dos variables de entradas X e Y .

SOLUCIÓN: Los casos de prueba diseñados fueron 9:

1) X=valor válido ; Y= valor válido
2) X=valor válido ; Y= valor en el límite de validez
3) X=valor válido ; Y= valor inválido
4) X=valor en el límite de validez ; Y= valor válido
5) X=valor en el límite de validez; Y= valor en el límite de validez
6) X=valor en el límite de validez; Y= valor inválido
7) X=valor inválido ; Y= valor válido
8) X=valor inválido ; Y= valor en el límite de validez
9) X=valor inválido ; Y= valor inválido

RESULTADO: El 95% los defectos existentes fueron detectados.

Figura 4.3 Ejemplo de Caso para el dominio de Testing en Ingeniería de Software.

Por último es importante incluir una medida del éxito de la solución si en la base de casos se ha logrado soluciones con diferentes niveles de éxito o fracaso y una medida del resultado obtenido, si se han cumplido las expectativas o no ante el problema.

Los casos que constituyen un problema y solución pueden usarse para derivar soluciones a nuevos problemas, basándose en experiencias similares del pasado. Si además tienen una descripción del resultado, pueden usarse para evaluar propuestas de solución y anticipar problemas potenciales antes de que ellos ocurran. En la figura 4.3 se muestra un caso donde se describen el problema, la solución y el resultado.

Tradicionalmente, hay varios tipos de métodos para representar casos, que van desde representaciones no estructuradas a totalmente formales y automáticamente procesables [Chen et al., 2003]. Dichos métodos pueden resumirse en:

- Método textual: un caso se representa en un formato de texto libre, o por una lista de preguntas y respuestas. Este método es el más potente en cuanto a la capacidad de representación y es el más simple, pero necesita la lectura humana para su interpretación.

- Representación a través de atributos: un caso se representa con pares (atributo, valor) que describen el problema y las características de la solución. Por ejemplo, la descripción de la solución en el caso representado en de la figura 4.3, es representado por un conjunto de pares: (Xi, ValorXi), (Yi, ValorYi), etc. Cada atributo tiene un tipo y rango de valores asociados. Esta representación, si bien puede ser procesada automáticamente no guarda información estructural y relacional, por lo tanto falla al describir objetos complejos, por ejemplo, un módulo de código, una métrica, etc. Estos constituyen objetos estructurados que no pueden ser representados con un par (atributo, valor).

- Representación estructurada: consiste en aplicar, por ejemplo, técnicas orientadas a objetos para representar casos. Este tipo de representación es la más adecuada para dominios complejos, en el cual los casos involucran variables estructuradas. Los casos son representados como

una colección de objetos, cada uno de los cuales es descripto por un conjunto de atributos, que a su vez pueden ser simples o estructurados. Por ejemplo, en el caso de la figura 4.2, la solución (en este caso el diseño de un árbol de requerimientos adecuado) es una instancia de una clase compleja que representa un modelo de requerimientos para un proyecto de medición.

Los casos pueden obtenerse directamente del experto humano, aunque también es posible obtenerlos de la documentación que conservan la mayoría de las instituciones o empresas, o mejor aún, de los propios sistemas de información que llevan la historia de las decisiones tomadas.

4.5 Ciclo del Razonamiento Basado en Casos

El Razonamiento Basado en Casos opera a partir de casos almacenados, previamente resueltos. Como ya se mencionó, es un método para resolución de problemas por medio del razonamiento por analogía. El proceso de razonamiento consiste en proponer una solución a un problema planteado, en función de su similitud con casos resueltos anteriormente.

La resolución de problemas en CBR puede ser descripta mediante los cuatro etapas siguientes (ver figura 4.3 [Aamodt et al., 1994]):

- **Recuperar** los casos más similares.
- **Reusar** la información y conocimiento en los casos recuperados para resolver un problema.
- **Revisar** la solución propuesta.
- **Registrar** o almacenar la nueva solución una vez que ha sido confirmada o validada de una manera que pueda ser útil para resolver problemas futuros.

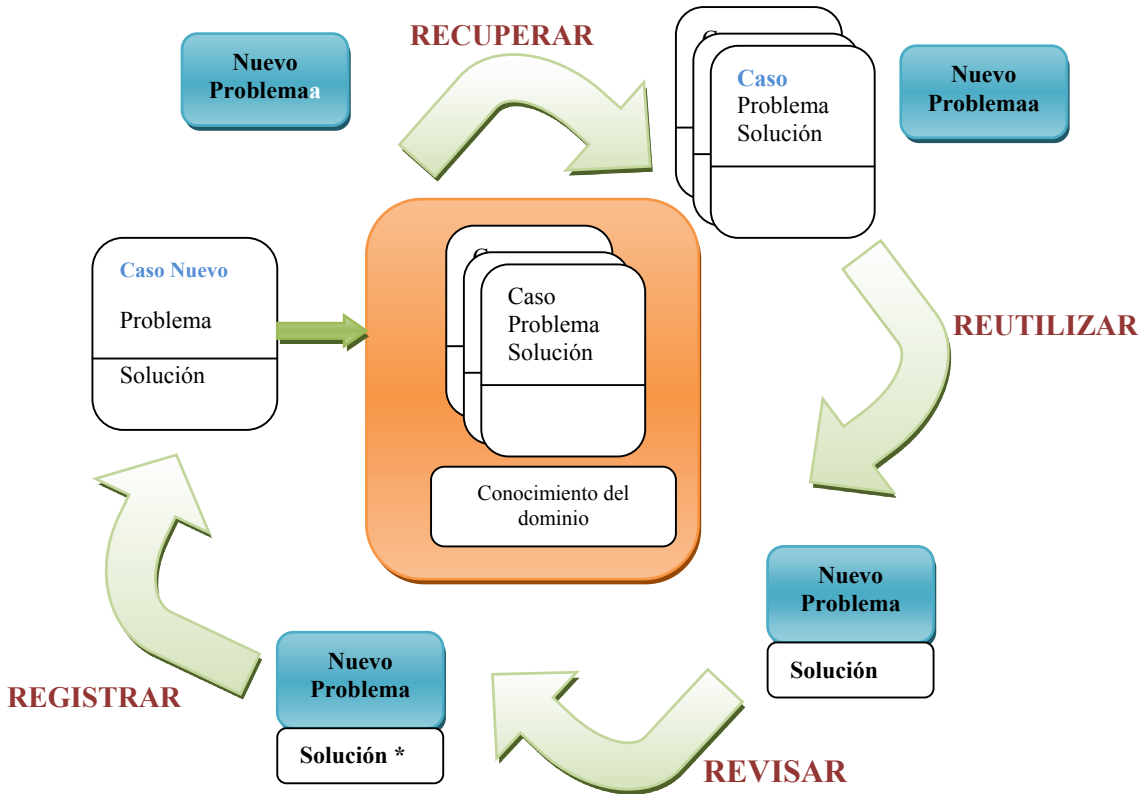


Figura 4.4: Ciclo de CBR (adaptada de Aamodt et al., 1994)

Un nuevo caso se resuelve *recuperando* uno o más casos pasados similares al caso actual, *reusando* la información recuperada para resolver el problema, *revisando* la solución propuesta y *almacenando* la nueva experiencia, incorporándola a la base de conocimiento existente (base de casos). La figura 4.4 ilustra dichos procesos según como fueron definidos por los autores citados.

4.5.1 Recuperación de Casos

Es el proceso de recuperar casos similares desde la memoria. En general, consiste en dos pasos:

- Recuperar un conjunto de casos previos.
- Seleccionar el mejor caso del conjunto.

Una descripción inicial de un problema define un nuevo caso o situación que se quiere resolver. Este nuevo caso es usado para buscar en la colección de casos pasados almacenados, aquellos que sean similares. Del conjunto de casos recuperados se selecciona el que más se adecue a la nueva situación teniendo en cuenta la similitud del problema y el contexto.

Se debe definir un proceso de matching (correspondencia) lo suficientemente general para permitir determinar los casos similares [Sankar et al., 2004]. Existen numerosos algoritmos utilizados en la literatura (con bastante éxito) para la comparación de casos, (en el capítulo 6 definiremos con más detalle el algoritmo propuesto). Además se debe definir una función que mida (cuantifique) el grado de similitud (analogía) entre los casos. Esta función se apoya en el proceso de correspondencia y debe tener en cuenta la similitud de cada uno de los atributos de los casos, que a su vez pueden ser de distintos tipos.

4.5.2 Reuso de Casos

El caso recuperado es combinado junto con el nuevo caso a través del reuso de información, y mediante mecanismos de similitud que definen la cercanía o no del caso recuperado con el nuevo, y se propone una solución al caso.

En este paso la solución recuperada de la base de casos es adaptada a una nueva situación. Hay dos pasos importantes involucrados en la adaptación [Aamodt et al., 1994] :

- Descifrar las necesidades que van a ser adaptadas del caso.
- Hacer la adaptación.

Para llevar a cabo la adaptación del caso se empieza por considerar la adaptación por sí misma para cualquier tarea o dominio en particular, podemos proponer un grupo de estrategias de adaptación. Se debe evaluar qué parte de la solución anterior puede ser adaptada al nuevo problema.

4.5.3 Revisión del Caso

En esta etapa, la solución del nuevo caso se revisa para confirmar que puede ser aplicada. Para ello, la solución (o su interpretación) debe ser justificada.

Cuando el conocimiento necesario para la evaluación es de dominio público, o la adaptación es realizada por un experto humano que está exteriorizando su experticia, uno puede pensar que este paso está validado, sin embargo en algunas situaciones hay muchas dudas para validar una solución.

Se pueden revisar las soluciones basándose en los resultados obtenidos en casos previos. Esto puede requerir de la recuperación de otros casos adicionales (ya no basándose en la similitud del problema sino en la similitud de la solución, para analizar sus resultados). Este método puede resultar en la necesidad de una adaptación adicional. Se pueden proponer también situaciones hipotéticas para probar la robustez de una solución.

La revisión de la solución propuesta es una etapa necesaria para completar el ciclo de razonamiento/aprendizaje. Este análisis (habitualmente llevada a cabo por un agente externo, léase humano) puede conllevar una reparación de posibles fallos.

4.5.4 Registro del Caso

Se almacena el nuevo caso con su solución sugerida y la base de casos se actualiza con una nueva lección aprendida, para su uso futuro. El proceso más importante de la actualización de la memoria es escoger las formas para indexar el nuevo caso en la memoria. Los índices pueden ser elegidos así como el nuevo caso puede ser recuperado.

4.6 Ventajas y Desventajas del CBR

El Razonamiento Basado en Casos es aplicable a una amplia gama de situaciones reales, abarcando desde el conocimiento de situaciones sencillas en las

cuales los casos proporcionan directamente el conocimiento requerido hasta las situaciones importantes en las cuales la construcción de soluciones es compleja.

Esto tiene varias ventajas, ya que permite proponer soluciones rápidas a los problemas, razonar en situaciones que no son bien entendidas, evaluar soluciones cuando los métodos algorítmicos no están disponibles, evitar repetición de problemas, y enfocarse en las partes importantes de una situación.

Es posible destacar numerosas ventajas de los sistemas de Razonamiento Basado en Casos:

- Reducen la tarea de adquisición del conocimiento: dicha tarea sólo consiste en una representación de experiencias/casos almacenados junto a su representación y solución.

- Aprenden con el tiempo, debido a que cuanto más casos son usados, encuentran más situaciones de problemas y crean más soluciones.

- Evitan cometer errores del pasado. Dado que se guarda los resultados de aplicar una solución, pueden descartarse aquellos casos que no han sido eficaces, de manera tal de aprender de ellos.

- Se pueden utilizar para muchos propósitos, como crear planes, diagnósticos, argumentación de puntos de vista, etc., de formas muy distintas, dependiendo básicamente de los métodos de recuperación y de adaptación que implementen.

Con respecto a las desventajas que pueden encontrarse, se destacan las siguientes:

- Puede que no se disponga del conjunto de casos más apropiado para el tratamiento de un problema concreto.

- La recuperación de casos inapropiados puede costar un tiempo considerable o llevar a errores, que podrían ser evitados por métodos más incrementales.

- Puede haber una tendencia a usar los casos previos ciegamente, confiando en la experiencia previa sin validarla con respecto a la nueva situación.

4.7 Razonamiento Basado en Casos vs. Razonamiento Basado en Reglas

En el área de IA, los sistemas basados en reglas son un modo de resolver problemas mediante la aplicación de un conjunto de reglas, comparación de resultados y aplicación de las nuevas reglas basadas en la situación modificada [Sankar et al., 2004].

También puede trabajar por inferencia lógica dirigida, ya sea empezando con una evidencia inicial en una determinada situación y dirigiéndose hacia la obtención de una solución, o bien con hipótesis sobre las posibles soluciones y volviendo hacia atrás para encontrar una evidencia existente (o una deducción de una evidencia existente) que apoye una hipótesis en particular. Una regla se compone de un par condición - acción, como por ejemplo "*if condición then acción*". Cientos de reglas se requieren para una tarea típica de diagnóstico o reparación.

En la construcción de sistemas expertos basados en reglas, la adquisición del conocimiento no es fácil. A pesar de la potencia de estos sistemas, presentan algunas limitaciones. El experto encargado del sistema no es capaz de enumerar las cientos de reglas necesarias para resolver un problema. Es decir, quizás pueda hacer una larga lista, pero es probable que esa lista este incompleta y eso se perciba al momento de resolver un problema. Por lo tanto, es difícil obtener un conjunto de reglas completo y correcto; a este problema se lo conoce como cuello de botella de elicitación del conocimiento que hace, justamente, muy difícil la recolección de conocimiento en ciertos dominios.

Otra limitación recae en el hecho de que los sistemas basados en reglas carecen de memoria. No pueden recordar problemas que han resuelto con anterioridad. Por ejemplo, si un programa de diagnóstico médico se utiliza en un paciente que presenta ciertos síntomas, el proceso puede disparar docenas, cientos o quizás, miles de reglas para dar con el diagnóstico o tratamiento. Del mismo modo, si se utiliza con otro paciente que presenta los mismos síntomas, dispararía el mismo conjunto de reglas, sin

saber si han resultado efectivas o no. Un programa sin memoria no puede recordar sus errores y está destinado a volver a cometerlos [Slade, 1991].

Como tercera y última limitación, los sistemas basados en reglas no son robustos en la medida que si un problema no puede ser coincidente con alguna de las reglas, directamente no se le brinda solución alguna. La base de conocimiento está limitada a las reglas, de manera que si ninguna regla puede ser aplicada, el sistema no tiene alternativas.

Los sistemas basados en casos solucionan los problemas encontrados en los sistemas basados en reglas. El primero, está relacionado con la adquisición del conocimiento. La unidad de conocimiento es el caso, no la regla. Es más fácil articular, examinar y evaluar casos que reglas.

Como segunda cuestión resuelta es la performance. Los sistemas basados en casos pueden recordar su propio desempeño y modificar su comportamiento para evitar errores cometidos en el pasado.

Tercero, es posible construir soluciones a nuevos problemas mediante el razonamiento de analogía de casos pasados. Tal como mostramos en el capítulo anterior, todo el acervo cultural, conocimientos y experiencia de los miembros de una organización conforman el contenido de la Memoria Organizacional. Toda experiencia pasada que permita ayudar a solucionar problemas actuales, constituyen casos. Por lo tanto, el conocimiento de una Memoria Organizacional puede ser estructurado en casos para poder reusarse cuando se requiera.

Por último, cabe mencionar que los sistemas CBR pueden ofrecer explicaciones y justificaciones de manera natural y completa. Al usuario se le presentan los casos relacionados al planteado por él, y por lo tanto obtiene directamente las razones que originaron la solución propuesta. Mientras que en los sistemas con Razonamiento Basado en Reglas (RBR), la explicación consiste generalmente en desplegar la secuencia de reglas utilizadas, lo cual no siempre representa una explicación adecuada de la solución; principalmente si se han aplicado numerosas reglas y la representación de las mismas no es clara.

A pesar de las ventajas enunciadas del CBR sobre los sistemas basados en reglas, no se puede afirmar que vayan a desplazarlos. Lo que sí puede afirmarse es que

permiten ser aplicados a dominios donde el conocimiento es incompleto, está mal definido o no existe una teoría causal [Sankar et al., 2004].

Algunas de las situaciones en las que es particularmente ventajoso utilizar el Razonamiento Basado en Casos [Sankar et al., 2004] son cuando:

- El dominio no puede formalizarse con reglas porque:
  - el dominio tiene un modelo causal débil o desconocido.
  - el dominio tiene términos subdefinidos
  - se aplican reglas contradictorias en situaciones diferentes
- La aplicación requiere una salida compleja, por ejemplo, planes de batalla.
- El dominio es, de por sí basado en precedentes, por ejemplo leyes, diagnósticos médico, demandas de asentamiento, etc.
- La formalización del dominio requiere demasiadas reglas.
- El dominio es dinámico, requiere de la rápida adquisición de soluciones a nuevos tipos de problemas.

El área de Gestión del Conocimiento para el aseguramiento de calidad en Ingeniería de Software es un dominio donde la aplicación de Razonamiento Basado en Casos es muy beneficiosa, teniendo en cuenta que cumple ampliamente con las características detalladas, como se mostrará en el capítulo 8.

4.8 Tipos de CBR

Las aplicaciones CBR se clasifican principalmente en dos tipos: tareas de clasificación y tareas de síntesis o planificación [Althoff et al., 1995].

4.8.1 Tareas de clasificación

En las tareas de clasificación, un caso nuevo se empareja (matching en su terminología original) con los de la base de casos para determinar qué tipo, clase o caso es. La solución del caso que mejor ajusta es el que se reutilizará.

La mayoría de las herramientas CBR disponibles dan un soporte aceptable para las tareas de clasificación, que suelen estar relacionadas con la recuperación de casos. Existe una gran variedad de tareas de clasificación, como por ejemplo:

- Diagnosis: médica o de fallos de equipos.
- Predicción: pronóstico de fallos de equipos o actuación sobre el stock de un mercado.
- Valoración: análisis de riesgos para bancos o seguros o estimación de costos de proyectos.
- Control de procesos: Control de fabricación de equipos.
- Planificación: reutilización de planes de viaje o planificadores de trabajo.

Podemos aplicar CBR fácilmente a problemas de clasificación puesto que pueden consistir en:

- La recuperación de un amplio conjunto de casos similares, por ejemplo aquellos en los que el antibiótico fue el tratamiento.
- Recuperar el mejor ajuste de este conjunto, quizás para sugerir penicilina como antibiótico específico.
- Adaptar la solución, por ejemplo alterando la dosis para diferentes edades o pesos de los pacientes.
- Almacenar el resultado del nuevo caso para un futuro uso.

Como se puede observar, las tareas de clasificación son fáciles de implementar porque se ajustan al ciclo CBR, los casos tienden a ser más fáciles de representar y

recuperar, y los algoritmos de recuperación utilizados en la mayoría de las herramientas CBR son clasificadores.

4.8.2 Tareas de síntesis

Las tareas de síntesis intentan crear una nueva solución combinando partes de soluciones previas. Éstas son inherentemente complejas a causa de las restricciones de los elementos usados durante la síntesis.

Los sistemas CBR que realizan tareas de síntesis deben realizar adaptación y son normalmente sistemas híbridos que combinan CBR con otras técnicas. Algunas de las tareas que realizan estos sistemas son:

- Diseño: La creación de un nuevo artefacto adaptando elementos de otros existentes.
- Planificación: La creación de nuevos planes a partir de otros previos.

Por norma general, los sistemas que implementan tareas de síntesis son difíciles de implementar. Esto se debe a que es más fácil ajustar un artefacto a un conjunto de artefactos prototípicos que construir un artefacto a partir de una especificación.

Las tareas de clasificación simplemente requieren reconocimiento de las características mientras que las tareas de síntesis requieren colocar las características correctas en el orden y lugar correcto.

Los sistemas de síntesis operan en dominios de diseño o planificación para intentar simplificar el proceso creativo produciendo un diseño o plan que se sabe que es bueno para producir el plan final a partir de él.

Para los diseñadores, esto es más rápido que empezar un diseño desde una hoja en blanco. Se asume que modificar un buen diseño o plan inicial es más fácil que crear uno desde el principio.

Las razones por las que los sistemas de síntesis son difíciles de construir son:

- La representación de un caso de un plan o diseño es compleja y altamente estructurada con muchas dependencias entre características. Los casos no se almacenan en un medio único y homogéneo, por lo tanto la recuperación de casos es más difícil.

- Las herramientas CBR tienden a no soportar indexación o recuperación de representaciones de casos altamente estructurados [Althoff et al., 1995]. Además la adaptación es a menudo un requisito clave en las tareas de síntesis.

4.9 CBR como Soporte para la Implementación de Memorias Organizacionales

Como fue analizado en el capítulo anterior, actualmente somos testigos de un creciente interés en el desarrollo de la Gestión del Conocimiento y Memorias Organizacionales. El primero se refiere a administrar y almacenar el conocimiento organizacional, de manera que después pueda ser utilizado para aprender, resolver problemas y como apoyo en la toma de decisiones [Dogson, 1993] [Conklin, 1996]. Las Memorias Organizacionales tienen como objetivo la retención del conocimiento para su uso futuro (resguardo) [Rabarijaona et al., 1999], [Despres, 2000].

En un alto nivel de abstracción, el objetivo de las Memorias Organizacionales es conservar el conocimiento mientras que el de la Gestión del Conocimiento apunta a desarrollar y desplegar el mismo. Claramente estas dos áreas están estrechamente interrelacionadas y se centran en el perfeccionamiento de la competitividad de una organización mejorando la manera en que ella maneja su conocimiento.

Hay varias maneras de llevar a cabo la Gestión del Conocimiento en una organización así como tecnologías para el desarrollo de Memorias Organizacionales de apoyo.

Algunos autores [Ale, 2009], [Martin et al., 2009], [Mica et al., 2004] [O'Leary, 1998b] proponen onntologías y CBR como los medios para dar soporte a KM y Memorias Organizacionales (MO´s), respectivamente. Uno de los objetivos de una ontología es servir como base conceptual consensuada para el intercambio de

conocimiento organizacional, facilitando su reuso e interoperabilidad semántica [Martin et al., 2009].

Por otro lado, el Razonamiento Basado en Casos se ha aplicado exitosamente como una forma de promover y gestionar el conocimiento adquirido en experiencias y lecciones aprendidas coleccionadas a lo largo del tiempo en una organización, es decir, una manera de proveer persistencia (memoria) al conocimiento.

Mientras que en el área de Gestión del Conocimiento, las ontologías han sido reconocidas por muchos autores como la tecnología central para apoyar los KMS [Abecker et al., 1998], [Ale, 2009], [Martin et al., 2009], [Mica et al., 2004], no hay tal tecnología discernible en el área de Memoria Organizacional. Una razón para esto puede ser la variedad de actividades que están involucradas en una Memoria Organizacional.

Teniendo en cuenta la definición presentada en el capítulo anterior de Memoria Organizacional como: "*La manera en que una organización resguarda y lleva cuenta de lo que sabe para beneficiarse con su uso futuro",* ésta debe dar apoyo a las actividades de retención del conocimiento para su uso futuro (resguardo), búsqueda, recuperación y diseminación (distribución) y su eventual reutilización (aplicación) [Rabarijaona et al., 1999], [Despres, 2000]. No hay una tecnología única que pueda dar soporte a todas estas actividades. Sin embargo, hay una área de investigación que está promoviéndose como una solución potencial por llevar a cabo MO´s. Esta es el área de Razonamiento Basado en Casos.

Además de las ventajas del Razonamiento Basado en Casos sobre el Razonamiento Basado en Reglas ya enunciadas en la sección 4.7, otro aspecto que lo hace beneficioso para la implementación de MO´s, es que las cuatro etapas del ciclo de CBR se pueden acomodar fácilmente a las principales actividades de apoyo que las Memorias Organizacionales deben brindar en cualquier KMS.

Teniendo en cuenta las actividades principales del proceso de Gestión del Conocimiento analizadas en la sección 3.3.3, describimos a continuación cómo una Memoria Organizacional que usa Razonamiento Basado en Casos como tecnología central, puede dar apoyo eficientemente a las actividades de Gestión del Conocimiento

(ver Fig. 4.5), basándose en las cuatro etapas del CBR (Recuperación, Reuso, Revisión, Registro).

Recuperación de casos: El proceso de recuperar casos similares desde la Memoria Organizacional Basada en Casos, para su posterior reuso, con sus mecanismos de indexado y filtrado brinda un importante apoyo a la actividad de *distribución del conocimiento.* Si además el proceso de recuperación de casos es pro-activo (actúa cuando detecta una necesidad) asegura que el conocimiento desarrollado en la organización sea efectivamente usado por todos sus miembros.

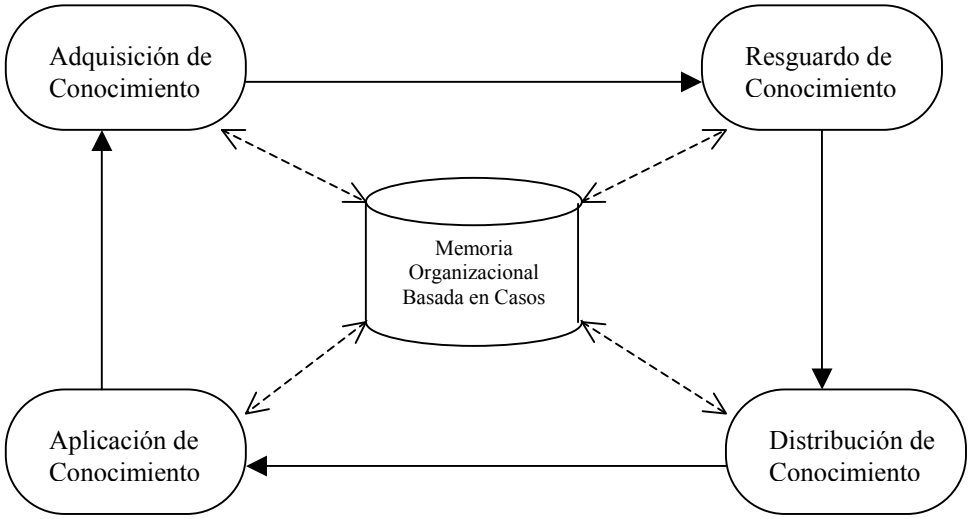


Figura 4.5 La Memoria Organizacional Basada en Casos asiste a las principales actividades de Gestión del Conocimiento

Reuso de casos: El caso recuperado es combinado junto con el nuevo problema y se propone una solución al mismo. Si bien la solución deberá ser adaptada a la nueva situación, esta etapa es fundamental como apoyo a la actividad de *aplicación del conocimiento.*

Revisión de casos: La solución del nuevo caso se revisa para confirmar que puede ser aplicada, o si necesita una adaptación. A veces, el conocimiento necesario para la revisión y adaptación es de dominio público; en otras, es realizada por un experto humano que está exteriorizando su experticia. También se pueden revisar las soluciones basándose en los resultados obtenidos en casos previos. De todas maneras,

esta etapa da un importante apoyo a la *adquisición de nuevo conocimiento*. La revisión de la solución propuesta es una etapa necesaria para completar el ciclo de razonamiento/aprendizaje.

Registro de nuevos casos: El almacenamiento del nuevo caso con su solución constituye la actualización de la base de conocimientos con una nueva lección aprendida o experiencia de utilidad para su uso futuro. Esta etapa tiene el mismo objetivo que la actividad de *resguardo* en un sistema de Gestión del Conocimiento. Esto es, la persistencia del conocimiento para su ulterior búsqueda, recuperación y diseminación.

Capítulo 5- Ontologías

"*Lo que nosotros observamos, no es la naturaleza en sí misma, sino la naturaleza expuesta a nuestro método de cuestionamiento*", Werner Heisenberg *, Físico y Filósofo*

5.1 Introducción

La vertiginosa evolución de Internet y la Web como un medio de publicación de documentos, diseminación de conocimiento y soporte para sistemas de información distribuidos conjuntamente con las tecnologías de la información y comunicación relacionadas, han creado las condiciones propicias para difundir y compartir conocimiento organizacional, documentación científica y datos e información catalogados.

Sin embargo, para que toda esta información y conocimiento pueda ser compartida, reusada, interpretada y aplicada uniformemente a distintos ámbitos, es necesario contar con una terminología común, y una estructura de metadatos (datos acerca de cómo están estructurados los datos), que permita tanto el procesamiento automatizado de la información, como su correcta interpretación.

Las ontologías son una solución a esta necesidad, ya que proveen los metadatos necesarios para una representación declarativa del conocimiento de un domino, que puede ser comunicado entre personas y computadoras; además, proporcionan una definición formal de conceptos consensuados que asegura la interpretación correcta del conocimiento compartido y, por último, brindan un vocabulario común, bien definido para el intercambio de información y conocimiento en el dominio en cuestión.

Desde el comienzo de los noventa, las ontologías se volvieron un tema común de investigación para algunas comunidades del área de IA, incluyendo Ingeniería del Conocimiento, procesamiento de lenguaje natural y representación del conocimiento.

Recientemente la noción de ontología se ha extendido a otras áreas, tales como integración de información desde orígenes heterogéneos, búsqueda semántica de información en la Web y Gestión del Conocimiento organizacional. La razón de su difusión y potencial adopción está sustentada básicamente en todo lo que se promete: una comprensión común y compartida de algún dominio que puede ser comunicado entre personas y computadoras.

5.2 Definición

La definición más citada del término Ontología en el área de IA es la de Gruber "*Una ontología es la especificación explícita de una conceptualización*" [Gruber, 1995].

Para la IA lo que existe es exactamente aquello que puede ser representado computacionalmente. Cuando el conocimiento de un dominio es implementado en un formalismo declarativo, el conjunto de objetos o entidades que pueden ser representados es llamado el universo de discurso [Bartsch, 2004]. En este contexto, una ontología es un tipo de base de conocimiento que describe conceptos a través de definiciones que son lo suficientemente detalladas para capturar la semántica de un dominio [Rodriguez et al., 2003]. Este conjunto de definiciones, y las relaciones descriptibles entre ellos, se refleja en el vocabulario con el cual se representa el conocimiento.

En la especificación de una ontología se asocian mediante definiciones, los nombres de las entidades del ámbito en cuestión, (por ejemplo clases, relaciones, funciones u otros objetos), con cierto texto legible por el ser humano que las describe y con axiomas que restringen su interpretación.

Borst [Borst, 1997], modificó ligeramente la definición de Gruber: "Las ontologías se definen como la especificación formal de una conceptualización compartida". Studer, Benjamins, y Fensel [Studer et al., 1998], agregaron expresividad a las definiciones de Gruber y Borst explicitando:

Conceptualización se refiere a un modelo abstracto de algún fenómeno en el mundo, proveniente de haber identificado los conceptos relevantes de dicho fenómeno.

Explícita se refiere a que los conceptos usados y las restricciones para su uso se definen explícitamente.

Formal se refiere al hecho de que la ontología debería ser legible o interpretable por computadoras.

Compartida refleja la noción de que una ontología captura conocimiento consensuado, es decir, no es conocimiento privado de un individuo, sino aceptado por un grupo o comunidad.

Debido a que las ontologías son ampliamente usadas para diferentes propósitos (procesamiento de lenguaje natural, Gestión del Conocimiento, comercio electrónico, integración de información, la Web Semántica, etc.) en diferentes comunidades (Ingeniería del Conocimiento, Ingeniería de Software y base de datos, etc.), Uschold y Jasper proveen una nueva definición de la palabra ontología para popularizarla en otras disciplinas [Uschold et al., 1999]. Dichos autores definen una ontología como sigue: "una ontología puede tomar una variedad de formas, pero incluirá necesariamente un vocabulario de términos y algunas especificaciones de su significado. Esto incluye definiciones y una indicación de cómo los conceptos están interrelacionados lo cual colectivamente impone una estructura en el dominio y restringe las posibles interpretaciones de los términos".

Como conclusión general puede decirse que el objetivo de las ontologías es capturar conocimiento consensual de una forma genérica y formal, y pueden ser reutilizadas y compartidas a través de diferentes aplicaciones de software y por grupos de personas.

5.3 Componentes de una Ontología

Una ontología consta de un conjunto no vacío de conceptos identificados como entidades relevantes en el dominio a modelar, un conjunto de atributos que describen los conceptos y que pueden ser propios o heredados en una especialización, un conjunto de relaciones entre dichos conceptos, un conjunto de funciones (que son un caso especial de relaciones), un conjunto de axiomas que vinculan elementos de la ontología en condiciones que deben cumplirse siempre y un conjunto de instancias [Gruber 1993]. En las siguientes subsecciones pasaremos a describir cada uno de estos elementos.

5.3.1 Conceptos

Un concepto puede ser cualquier cosa acerca de la cual algo se pueda aseverar, y por tanto puede ser un objeto físico (tangible), u objetos intangibles como por ejemplo la descripción de una tarea, función, acción, estrategia, entre otros. Cada concepto tiene un término asociado como nombre y un conjunto de atributos que lo identifican.

En la figura 5.1 se muestra mediante un diagrama UML un ejemplo simple pero ilustrativo de ontología, donde los conceptos están representados con rectángulos que tienen en su parte superior el término asociado y en la parte inferior, la lista de sus atributos. Por ejemplo, en dicha figura se puede distinguir el concepto Caso, cuyos atributos en la ontología presentada son autor y fecha (en el capítulo 6 se describe con precisión los términos, atributos y relaciones para el dominio de Memoria Organizacional Basada en Casos).

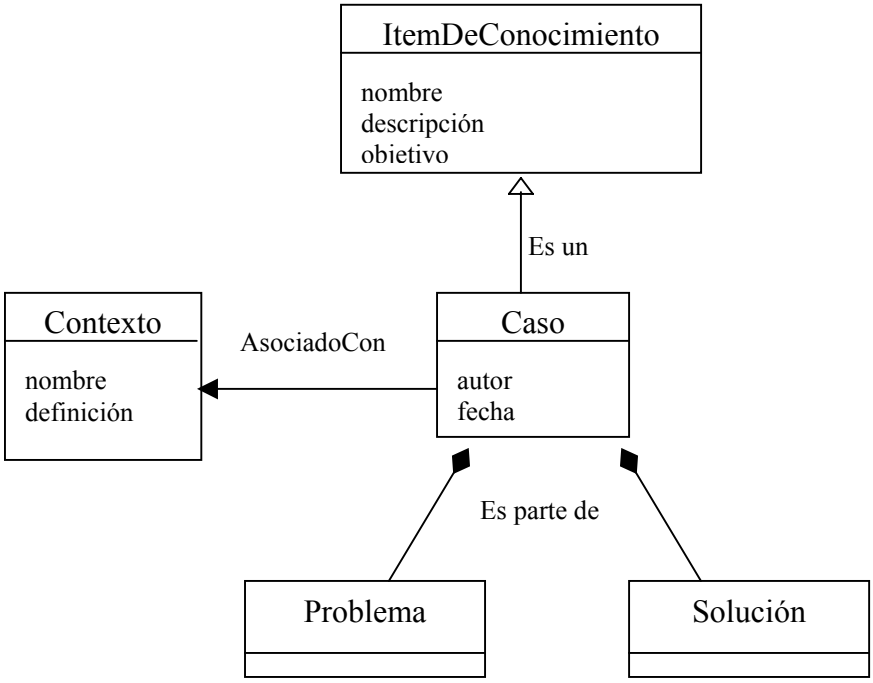


Figura 5.1 Conceptos, atributos y relaciones en una ontología.

5.3.2 Relaciones

Las relaciones representan el tipo de interacción entre los conceptos de un dominio y son formalmente definidas como subconjuntos del producto cartesiano de n

conjuntos, esto es R: C1 x C2 x … x Cn. En la figura 5.1 las relaciones están representadas con flechas, como por ejemplo la relación asociadoCon que asocia cada caso con el contexto al cual se aplica.

Algunas relaciones tienen un significado especial, a saber: las relaciones binarias de especialización (Es un) y de composición (Es parte de), que en el diagrama de la figura 5.1 están representadas con las convenciones establecidas por el modelo UML. En nuestro ejemplo, un caso es un tipo de Item de conocimiento, por lo tanto heredan todos los atributos del concepto ItemDeConocimiento a través de la relación taxonómica (Es un). Por otro lado, un caso tiene como partes componentes un problema y una solución, en este caso la relación es del tipo parte-todo (Es parte de).

En general los modelos ontológicos definen la relación taxonómica *Es un* como irreflexiva, transitiva y asimétrica. En tanto que la propiedad transitiva permite inferir una jerarquía (taxonomía) en la estructura, las propiedades restantes son útiles para chequear consistencia.

Para modelar la relación de que un conjunto de conceptos son partes que constituyen otro concepto, se usa la relación Es parte de (part-whole), que se definen con las propiedades irreflexiva y asimétrica.

5.3.3 Funciones

Son un caso especial de relaciones donde el enésimo elemento de la relación es único para los n-1 anteriores. Formalmente las funciones se definen como F: C1 x C2 x….x Cn-1 x Cn. Ejemplos de funciones son las relaciones Madre-de, o en el ejemplo de la figura 5.1 la relación asociadoCon es una función ya que para cada caso el contexto en el que ocurrió es único.

5.3.4 Axiomas

Los axiomas se usan para modelar verdades que se cumplen siempre en la realidad modelada [Bartsch, 2004]. Los axiomas definidos en una ontología pueden ser estructurales o no estructurales.

Un axioma estructural establece condiciones relacionadas a las jerarquías de la ontología, conceptos y atributos definidos. Un ejemplo de axioma estructural puede ser "Un Problema no puede a la vez ser una solución". Es decir

$P \cap S = \varnothing$

Siendo P = conjunto de Problemas y S conjunto de Soluciones

Los axiomas no estructurales establecen relaciones entre atributos de un concepto, y son específicos de cada dominio. Un ejemplo de axioma no estructural puede ser "El atributo nombre de caso no puede ser nulo".

5.3.5 Instancias

Se usan para representar elementos específicos del dominio de la ontología. En nuestro ejemplo de la figura 5.1, podrían definirse casos específicos, detallando los valores de sus atributos.

5.4 Clasificación de las Ontologías.

Algunos autores [Gómez-Pérez, 1999] [Van Heijst et al., 1997] [Mizoguchi et al., 1995] [Sowa, 2000], han clasificado las ontologías existentes según diferentes criterios. A continuación se exponen algunos de ellos y un resumen final que los relaciona por el grado de usabilidad y reusabilidad que supone cada clase.

5.4.1 Clasificación por Grado de Axiomatización

Según Sowa [Sowa, 2000], las ontologías se pueden clasificar en Terminológicas o Formales de acuerdo al grado de axiomatización que tenga la definición de sus categorías.

Ontologías Terminológicas: Una ontología terminológica define términos y sus relaciones en taxonomías que involucran tanto relaciones de subtipo y supertipo, como

las que relacionan partes con un todo (part-whole), pero no incluyen axiomas y definiciones expresadas en lógica o algún tipo de lenguaje formal procesable por computadoras. Esto hace que se disponga de menos información del universo modelado, pero permite a su vez, por la relativa simplicidad de su especificación, construir ontologías de gran tamaño.

La mayoría de los campos de la ciencia, ingeniería, negocios y jurídico, han desarrollado sistemas de terminología o nomenclatura para designar, clasificar y estandarizar sus conceptos en enormes ontologías terminológicas. Ejemplos de estas ontologías son:

EDR. (The Electronic Dictionary Research project) [Miyoshi et al., 1996], es un proyecto japonés que ha desarrollado un diccionario con más de 400.000 conceptos, con su traducción a palabras en inglés y japonés con muy poco detalle acerca de cada uno de sus términos.

WordNet [Fellbaum 1998], es una jerarquía de 166.000 palabras, muy usada para procesamiento de lenguaje natural por su temprana disponibilidad en Internet en http://wordnet.princeton.edu/index.shtml/.

Ontologías Formales: Una ontología formal tiene sus categorías restringidas por axiomas y definiciones expresadas en lógica formal o en algún tipo de lenguaje procesable por computadoras [Bartsch, 2004]. Las ontologías formales suelen tener menos cantidad de conceptos que las terminológicas, pero sus axiomas y definiciones pueden soportar computaciones e inferencias más complejas.

La diferencia entre una ontología terminológica y una formal, según Sowa [Sowa, 2000], es en el grado de especificación. Teóricamente, en la medida que se adicionen axiomas a una ontología terminológica, podrá evolucionar a una formal, pero la definición de axiomas no es una tarea trivial, y es por eso que en la práctica es difícil tal evolución.

5.4.2 Clasificación por Dependencia del Contexto

Según Mizoguchi, Vanwelkenhuysen e Ikeda [Mizoguchi et al., 1995] las ontologías pueden clasificarse conforme al grado de dependencia del contexto que presenten, en el sentido de que aquellas menos dependientes del contexto serán las candidatas a ser más reusadas. La clasificación que surge es:

Ontologías de dominio: Expresan conceptualizaciones que son específicas a un dominio particular, colocando restricciones en la estructura y contenido de un dominio de conocimiento mediante axiomas que se cumplen siempre entre los elementos de dicho dominio. Su principal objetivo es permitir el reuso de la ontología para diferentes aplicaciones que involucren al mismo dominio. Ejemplos de estas ontologías son:

The EngMath ontology [Gruber 1994a], que es una ontología para el ámbito de la ingeniería en matemáticas, disponible en la biblioteca de ontologías de Ontolingua[2].

The Enterprise Ontology [Uschold et al., 1998] que se ubica en el dominio del modelado de procesos empresariales. Es por lo tanto, una colección de términos y definiciones relevantes al entorno empresarial.

Ontología de Métricas e Indicadores [Martin et al., 2003] [Martin, 2005] que modela el dominio de Métricas e Indicadores para el área de medición y evaluación, definiendo términos como: *Indicador, Métrica, Medición, Evaluación, Unidad, Escala*, etc.

Ontologías generales o de sentido común: Definen un vocabulario relacionado a cosas, eventos, tiempo, espacio, causalidad, comportamiento, función, etc.

La ontología CYC[3] [Cyc 2002] es una ontología de este tipo, ya que provee una gran cantidad de conocimiento humano general. Además está dividida en micro teorías que permiten su mejor administración.

Meta ontologías, Ontologías genéricas (Core Ontologies): Son similares a las de dominio, pero los conceptos que definen se consideran genéricos a través de diferentes

[2] Disponible en http://ksl.stanford.edu/software/ontolingua/

[3] Disponible en http://www.Cyc.com

áreas de conocimiento y por ello reusables en diferentes dominios. Típicamente, las ontologías genéricas definen conceptos como estado, evento, proceso, acción, etc. Los conceptos de las ontologías de dominio son a menudo definidos como especializaciones de conceptos existentes en ontologías genéricas. Cabe destacar que el límite para considerar una ontología genérica no está bien definido, pero la distinción es intuitivamente significativa y útil cuando es necesario organizar ontologías en bibliotecas.

Un ejemplo de este tipo de ontologías es The Mereology Ontology (ontología de mereología) propuesta por Borst [Borst, 1997] y que define los conceptos relacionados a la relación part-of y sus propiedades.

5.4.3 Clasificación por el Sujeto de Conceptualización

Algunos autores como Gomez-Perez, entre otros [Gómez-Pérez, 1999] [Van Heijst et al., 1997], también clasifican las ontologías por el tipo de dominio que modelan, o por el sujeto de conceptualización. En esta subsección mostramos algunos tipos de ontologías comúnmente definidos en la literatura, las principales categorías de ontologías identificadas son:

Ontologías de tareas: Proveen un sistemático vocabulario de los términos usados para resolver problemas asociados con tareas particulares, ya sean dependientes o no del dominio. Por ejemplo, relacionados a la tarea de evaluación, tendremos términos que involucren a los conceptos de "medición", "cálculo" y "objetivo", así como el término que refiera a la acción "generación de informe".

Ontologías de aplicación: Contienen todas las definiciones que son necesarias para modelar el conocimiento requerido para una aplicación particular en un dominio dado. Típicamente son una mezcla de conceptos provenientes de ontologías de dominio y de ontologías genéricas. Las ontologías de aplicación no se construyen con el propósito de lograr reusabilidad en diferentes dominios.

Ontologías de dominio: Coincide con la clasificación propuesta por Mizoguchi, Vanwelkenhuysen e Ikeda [Mizoguchi et al., 1995] que fue explicada anteriormente en la clasificación por dependencia del contexto (sección 5.4.2) .

Ontologías genéricas: Coincide con la clasificación propuesta por Mizoguchi, Vanwelkenhuysen e Ikeda [Mizoguchi et al., 1995] que fue explicada anteriormente en la clasificación por dependencia del contexto (sección 5.4.2) .

Ontologías de representación de conocimiento: Describen las primitivas de representación usadas para formalizar conocimiento en paradigmas de desarrollo de sistemas basados en el conocimiento, es decir, explican la conceptualización que subyace en un formalismo de representación de conocimiento. Se pretende que sean neutrales con respecto a las entidades del mundo, es decir, que provean un marco representacional sin hacer aseveraciones acerca del mundo. Las ontologías genéricas y de dominio son descriptas usando ontologías de representación.

El ejemplo más representativo de esta clase de ontologías es la Frame-Ontology [Gruber et al., 1994b] disponible en el servidor Ontolingua: http://www-ksl-svc. stanford.edu. Esta ontología expone las primitivas de representación usadas en lenguajes basados en marcos (frames), definiendo términos como clases, subclases, atributos, valores, relaciones , axiomas, entre otros, y permitiendo que otras ontologías puedan ser especificadas usando convenciones basadas en frames.

Ontologías de más Alto Nivel (Top-Level) : Pretenden establecer una estructura básica, bajo la cual todos los términos de cualquier ontología existente, deberían poder relacionarse. Hasta ahora el principal problema es que no existe una ontología única de este tipo. Ejemplos de estas ontologías son:

Ontología Top-Level de Sowa [Sowa, 2000]: Esta ontología incluye las categorías básicas derivadas de una variedad de fuentes en lógica, lingüística, filosofía e inteligencia artificial. La ontología es presentada como una estructura reticulada donde el concepto de más alto nivel es el tipo universal. Como subtipos del tipo universal se definen conceptos primitivos: independiente, relativo, etc; combinando estos conceptos primitivos se obtienen nuevos conceptos, hasta obtener el reticulado completo. Para cerrar el reticulado, el tipo absurdo (absurd) se considera un subtipo de todos los conceptos que no tienen descendientes.

Ontología CYC [Cyc 2002]: (También clasificada como ontología general o de sentido común en la sección 5.2.2), contiene alrededor de 3000 términos captados de los conceptos más generales de la realidad humana tradicionalmente consensuada.

Wordnet [Fellbaum 1998]: Esta ontología es una gran base de datos léxica para el idioma inglés (por eso se categoriza también como ontología terminológica). Está organizada en 70,000 conjuntos de sinónimos ("synsets"), cada uno de éstos conjuntos representa un concepto léxico subyacente. Los synsets están enlazados entre sí vía relaciones. Wordnet divide el léxico en cinco categorías: nombres, verbos, adjetivos, adverbios y funciones.

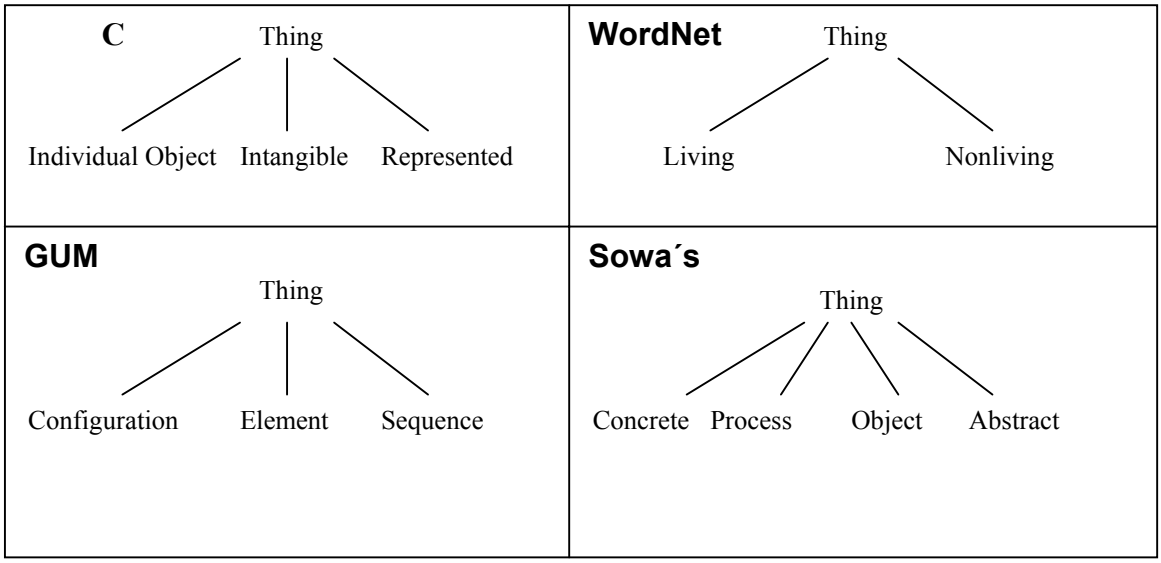


Figura 5.2 Diferencias entre ontologías top-level (adaptada de [Chandrasekaran, 1999]).

Modelo Superior Generalizado [Bateman et al., 1995]: La ontología Generalized Upper Model (GUM) es una ontología general independiente del dominio. GUM está dividida en dos jerarquías, la primera contiene todos los conceptos y la segunda contiene todos los roles.

En la figura 5.2 se puede ver cómo difieren algunas de las propuestas de ontologías de más alto nivel entre sí, en cuanto a la clasificación de los términos más generales.

5.4.4 Ontologías livianas (Lightweight) vs. pesadas (Heavyweight)

En la comunidad de la Ingeniería Ontológica es común también clasificar las ontologías como livianas (Lightweight) o pesadas (Heavyweight). Esta distinción está basada en el nivel de riqueza de su estructura interior. De esta manera, las ontologías

lightweight serán principalmente las taxonomías, mientras las ontologías heavyweight son aquéllas que modelan un cierto conocimiento "*de una manera más profunda y proporciona más restricciones en la semántica del dominio*" [Gómez-Pérez et al., 2004]. Las primeras incluyen conceptos, taxonomías de conceptos, relaciones entre conceptos, y propiedades que describen estos conceptos. Las últimas agregan axiomas y restricciones para clarificar el significado.

Las ontologías lightweight tiene la ventaja de ser simples y las heavyweight, de ser poderosas. No es posible poseer ambas ventajas al mismo tiempo, y no hay manera de determinar cuál de los tipos de ontologías es mejor que la otra, si las lightweight o las heavyweight. Todo depende de los objetivos de cada una y las necesidades basadas en cada caso particular. Por ejemplo, las ontologías lightweight son más útiles cuando el objetivo es, simplemente, para compartir conocimiento de un dominio entre las personas. Por otro lado, si es necesario ejecutar alguna clase de inferencia lógica o cálculo automático, será necesario utilizar las ontologías heavyweight. En todo caso, el siguiente consejo podría servirle a los distinto participantes: "use las ontologías más ligeras posible que puedan servir para las necesidades del proyecto específico".

5.4.5 Conclusiones sobre la Clasificación de Ontologías

A modo de resumen, considerando los distintos tipos de ontologías, se muestra un esquema extraído del trabajo de Benjamins y Gómez Pérez [Benjamins et al., 1996], en la figura 5.3, que relaciona los tipos de ontologías más comunmente usadas por su grado de usabilidad y reusabilidad. Se puede concluir que en la medida en que una ontología es más usable en un contexto dado, es menos reusable en otro contexto [Benjamins et al., 1996].

En la figura 5.3 se ubican de abajo hacia arriba las ontologías por su grado de generalidad y reusabilidad, las más generales abajo. El mismo esquema las clasifica también según sean ontologías orientadas a un domino específico (a la izquierda) o a la tarea (a la derecha).

Las Meta Ontologías, Ontologías de dominio y Ontologías de aplicación capturan el conocimiento estático del dominio en una forma independiente del método de resolución de problemas que se pueda utilizar. Por otro lado las Ontologías de tareas

y Ontologías de dominio-tarea están destinadas a conceptualizar y exponer conocimiento de resolución de problemas.

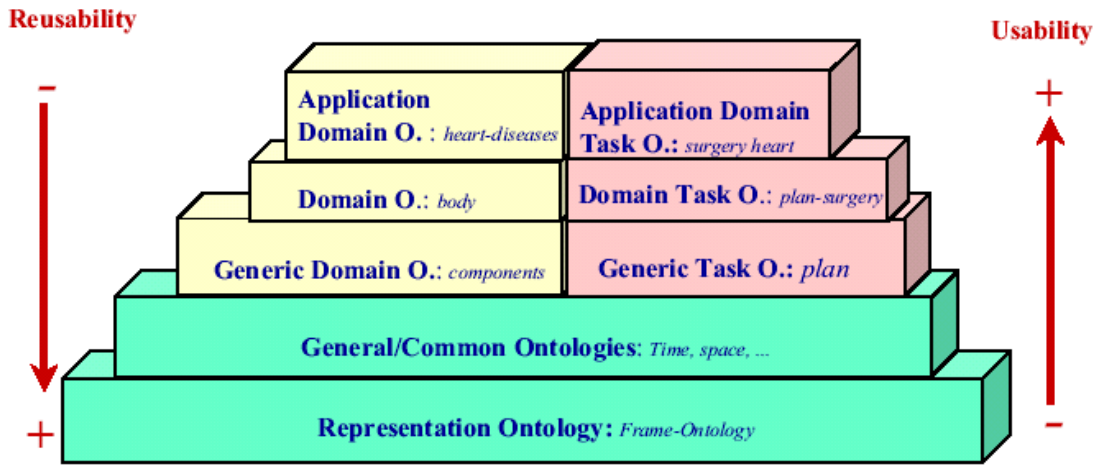


Figura 5.3 El equilibrio entre reusabilidad y usabilidad en ontologías (extraído de [Benjamins et al., 1996]).

Teniendo en cuenta las distintas clasificaciones de ontologías existentes en la literatura (para las que no existe un único criterio consensuado), la ontología de Memoria Organizacional Basada en Casos que se propone en esta tesis (desarrollada en el capítulo 6), se clasifica como **ontología de representación de conocimiento** según el sujeto de conceptualización, y como **Ontología general o de sentido común** según su dependencia del contexto. Es una ontología de representación de conocimiento porque define elementos de formalización usados para representar conocimiento y explica la conceptualización que subyace en una forma de codificación de conocimiento. Además es una ontología general ya que puede ser usada en distintas áreas de conocimiento y distintos contextos, ubicándose en un nivel intermedio de generalidad, (y por lo tanto, de grado de usabilidad y reusabilidad según [Benjamins et al., 1996]).

5.5 Recomendaciones de Diseño.

Algunos autores [Benjamins et al., 1996] [Gruber, 1995], establecen características deseables que deberían cumplir las ontologías, y resaltan la importancia de tenerlas en cuenta al momento de diseñar una ontología. A continuación se presentan algunas de estas recomendaciones.

Claridad. Según Gruber [Gruber, 1995], una ontología debería efectivamente comunicar el significado propuesto para los términos definidos en ella. Las definiciones deberían entonces ser tan objetivas como sea posible.

Completitud. Siempre que sea posible, debería proporcionarse una definición completa, es decir, definida por condiciones necesarias y suficientes, y no solamente por condiciones necesarias. Además, todas las definiciones deberían estar documentadas en lenguaje natural [Gruber, 1995].

Coherencia. Una ontología debería ser coherente, es decir, debería sólo concluir inferencias que sean consistentes con las definiciones que contiene la propia ontología. En definitiva, esto significa que sus axiomas deberán ser lógicamente consistentes [Gruber, 1995].

La coherencia debería alcanzar también a los conceptos que son definidos informalmente, como por ejemplo lo que es descripto en lenguaje natural y ejemplos que se proporcionan para clarificar ideas. Si una sentencia que puede ser inferida de los axiomas de la ontología contradice la definición de un ejemplo presentado informalmente, entonces se deberá considerar que la ontología es inconsistente.

Extensibilidad. Una ontología debería soportar la definición de nuevos términos basándose en el vocabulario existente, de manera tal que no requiera la revisión de las definiciones existentes [Gruber, 1995].

Esto se lograría manteniendo cierto equilibro entre ser lo suficientemente específicos en la definición como para permitir el uso en el ámbito para el cuál la ontología se define, pero no demasiado específicos, de manera que la ontología pueda ser de utilidad para otros usos futuros.

Mínimo compromiso ontológico. Una ontología debería hacer la menor cantidad de aseveraciones posibles acerca del mundo que está siendo modelado, permitiendo de esta forma que las partes que están comprometidas con la ontología (los diferentes agentes que la usarán) tengan libertad de especializar e instanciar la ontología cuando sea necesario [Gruber, 1995].

Diversificación de jerarquías. Para que la ontología se vea favorecida con el poder que otorgan los mecanismos de herencia múltiple [Benjamins et al., 1996], es

conveniente usar tantos criterios de clasificación como sea posible, de manera de representar más conocimiento en la ontología. De esta manera es muy sencillo agregar un término porque se puede definir fácilmente especificado desde los conceptos preexistentes y criterios de clasificación.

Minimización de la distancia semántica entre conceptos hermanos. Conceptos similares se deben agrupar y representar como subclases de una clase y deberían ser definidos usando las mismas primitivas. Por otro lado, conceptos que son menos similares se deberían representar aparte en la jerarquía [Benjamins et al., 1996].

Estandarización de nombres. La estandarización de nombres definiendo y respetando reglas para su formación siempre que sea posible, es una característica deseable para ayudar en el mantenimiento de una ontología [Benjamins et al., 1996].

Una posible estandarización es especificar el nombre de una relación como la concatenación del nombre de la ontología (o el del concepto que es primer elemento de la relación) con el nombre simple de la relación y con el nombre del concepto destino. Por ejemplo, recordando la figura 5.1, el nombre de la relación AsociadoCon, podría ser: Caso_AsociadoCon_Contexto, lo que ayudaría al entendimiento del significado de la relación.

5.6 Metodologías de Diseño y Construcción.

Ante la falta de un consenso entre las comunidades involucradas, que permita la formulación de una metodología estándar, cada grupo de desarrollo ha usado su propio conjunto de principios, criterios de diseño y etapas en el proceso de diseño y construcción de una ontología.

Afortunadamente varios autores [Fernández, 1999a] [Corcho et al., 2001] [Corcho et al., 2003] [Öhgren et al., 2005] han realizado trabajos de recopilación mostrando la perspectiva general y el estado del arte en desarrollo de metodologías de diseño y construcción para ontologías.

A continuación se presentan las metodologías más relevantes, conjuntamente con ejemplos de la aplicación de las mismas en diferentes experiencias.

5.6.1 Metodología de Grüninger y Fox

La metodología propuesta por Grüninger y Fox [Grüninger, 1995] recomienda un método formalizado para construir ontologías. La figura 5.4 ilustra los pasos de la metodología, que detallamos a continuación:

5.6.1.1 Pasos Recomendados para la Construcción de la Ontología

Paso 1: Definición de los escenarios motivadores. El desarrollo de ontologías es motivado por escenarios, que surgen en las aplicaciones. En particular, tales escenarios pueden ser presentados como la descripción de problemas que no han podido ser resueltos hasta el momento, posibles aplicaciones en las cuales la ontología podrá ser usada, entre otros.

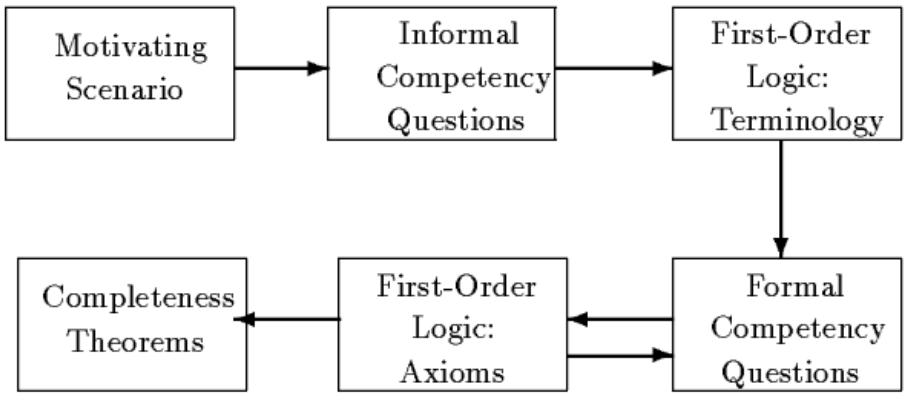


Figura 5.4 Etapas en la metodología de Grüninger y Fox (extraído de [Grüninger, 1995]).

Un escenario motivador suele proveer un conjunto de soluciones intuitivamente posibles a los problemas enumerados. Esas soluciones dan una semántica informal para los objetos y relaciones que serán luego incluidos en la ontología.

Cualquier propuesta de creación de una ontología o extensión de una existente, debería describir uno o más escenarios motivadores, y el conjunto de soluciones proyectadas a los problemas de dichos escenarios.

Paso 2: Especificación informal de las preguntas pertinentes (competency questions). Dado el escenario motivador, un conjunto de preguntas surgirán como demandas que la ontología a construir debería resolver. Se pueden considerar esas preguntas como requerimientos en forma de interrogantes (competency questions). Una ontología debe ser capaz de representar esos interrogantes usando su terminología, y ser capaz de caracterizar las respuestas a ellas usando axiomas y definiciones. Las preguntas pertinentes son usadas para determinar el alcance de la ontología.

Se consideran informales, pues no están aún expresadas en el lenguaje formal de la ontología. Las preguntas pertinentes o interrogantes, deberían ser definidas de una manera estratificada, de forma que la respuesta a un interrogante pueda ser usada para responder interrogantes más generales de la misma u otra ontología por medio de operaciones de composición y descomposición. Esta estratificación, en realidad es una forma de identificar conocimiento para reuso.

Las preguntas no generan compromisos ontológicos de diseño, sino que son usadas luego, para evaluar los alcances de la ontología en cuanto a la capacidad para satisfacer los requerimientos que plantean.

Paso 3: Especificación de la terminología. A partir de las preguntas formuladas en el paso anterior, se puede extraer el conjunto de términos usados para expresarlas y considerarlo como base para la terminología a usar en un lenguaje formal.

Si se está diseñando una nueva ontología, para cada interrogante informal planteado, debe haber objetos, atributos, y relaciones en la ontología en construcción que son intuitivamente requeridos para resolver la pregunta.

El primer paso en la especificación de la terminología de la ontología es identificar los objetos en el dominio de aplicación, que serán representados por constantes y variables del lenguaje.

Luego se definen los atributos de objetos y relaciones, usando predicados en lógica de primer orden, o en un formalismo equivalente.

Paso 4: Formalización de las preguntas pertinentes (competency questions). Una vez definida la terminología de la ontología, se definen formalmente las preguntas pertinentes en función de dicha terminología.

Paso 5 Especificación de axiomas formales. Los axiomas en una ontología especifican las definiciones de términos y restricciones en su interpretación. Se definen como sentencias en lógica de primer orden.

Para los autores de la metodología [Grüninger, 1995] un conjunto simple de objetos, o un conjunto de cláusulas básicas en lógica de primer orden, no constituye una ontología. Deben proveerse axiomas para definir la semántica del conjunto.

Si el conjunto de axiomas es insuficiente para representar las preguntas pertinentes y caracterizar sus soluciones, es un indicativo de que se deben agregar nuevos objetos o axiomas a la ontología hasta lograrlo. Se puede ver entonces, que la construcción de axiomas y la evaluación de las preguntas pertinentes es un proceso iterativo.

Paso 6: Verificar la completitud de la ontología. Una vez que las preguntas pertinentes han sido formalmente enunciadas se deben definir las condiciones bajo las cuales las soluciones son consideradas completas.

5.6.1.2 Ontologías Desarrolladas usando la Metodología

Esta metodología fue usada para construir las ontologías del proyecto TOVE (TOronto Virtual Enterprise) en el Enterprise Integration Laboratory de la Universidad de Toronto [TOVE 2009]. Dichas ontologías constituyen un modelo integrado y formalizado usando lógica de primer orden que incluye Enterprise Design Ontology, Project Ontology, Scheduling Ontology y Service Ontology.

5.6.2 Metodología de Unschold y King

Esta metodología se basa en la experiencia de los autores en el desarrollo de la ontología: "Enterprise Ontology" [Uschold et al., 1998], y provee guías para el desarrollo de ontologías.

5.6.2.1 Pasos Recomendados para la Construcción de la Ontología

Paso 1: Identificar el propósito. El objetivo de esta etapa es dejar claro porqué se construye la ontología y qué uso se le pretende dar. También es útil en este momento identificar y caracterizar el rango de posibles usuarios y aplicaciones de la ontología.

Paso 2: Construir la ontología. En esta etapa se distinguen tres sub-tareas:

Paso 2.1: Capturar la ontología. Por la tarea de capturar la ontología los autores entienden lo siguiente:

- Identificar los conceptos claves y relaciones en el dominio de interés.
- Producir definiciones textuales precisas y carentes de ambigüedad que describan tales conceptos y relaciones.
- Identificar los términos para referirse a tales conceptos y relaciones.

Los autores recomiendan que antes de buscar los conceptos más generales o particulares como conceptos clave, se identifiquen los conceptos más importantes, que luego serán usados para obtener el resto en la jerarquía por medio de generalización y especialización.

Paso 2.2: Codificación. Involucra explícitamente representar el conocimiento adquirido en el paso anterior, en un lenguaje formal.

Paso 2.3: Integrar ontologías existentes. Durante los procesos de captura y codificación debe plantearse la pregunta de si corresponde usar ontologías que ya existen, y cómo hacerlo.

Paso 3: Evaluación. Los autores adoptan la definición de Asunción Gómez Pérez, N. Juristo y J.Pazos [Gomez-Perez et al., 1995]: "hacer una evaluación de las ontologías, sus ambientes de software asociados, y documentación con respecto al marco de referencia ... .El marco de referencia puede ser especificaciones de requerimientos, preguntas pertinentes, y/o el mundo real".

Paso 4: Documentación. Se recomienda el establecimiento de pautas para documentar la ontología construida, posiblemente diferentes de acuerdo al tipo y propósito de la ontología.

5.6.2.2 Ontologías Desarrolladas usando la Metodología

El proyecto más importante que se desarrolló usando esta metodología es The Enterprise Ontology [Uschold et al., 1998], que es una colección de términos y definiciones relevantes para el dominio de empresas de negocios. La ontología fue desarrollada en el marco del Enterprise Project del Artificial Intelligence Applications Institute de la Universidad de Edimburgo.

5.6.3 METHONTOLOGY

Esta metodología fue desarrollada dentro del laboratorio de IA de la Universidad Politécnica de Madrid [Fernández et al., 1997] [Corcho et al., 2003]. El marco de trabajo (framework) METHONTOLOGY permite la construcción de ontologías a un nivel de conocimiento. Dicha metodología provee un conjunto de guías de cómo se deberían llevar a cabo las actividades identificadas en el proceso de desarrollo de una ontología.

5.6.3.1 Tareas Recomendadas para la Construcción de la Ontología

Tarea 1: Establecimiento del proceso de desarrollo de la ontología. En esta etapa se identifican qué actividades se realizaran al construir la ontología. Se deben definir las **Actividades de gerenciamiento del proyecto** (incluyen planificación, control y aseguramiento de la calidad), las **Actividades orientadas al desarrollo del proyecto** (incluyen especificación, conceptualización, formalización, implementación y mantenimiento) y las **Actividades de apoyo** que incluyen una serie de actividades ejecutadas al mismo tiempo que las actividades orientadas al desarrollo, sin las cuales la ontología podría no llegar a ser construida, (incluyen adquisición de conocimiento, evaluación, integración, documentación y administración de versiones).

Tarea 2: Desarrollo de la ontología. El ciclo de vida para el desarrollo de una ontología propuesto por METHONTOLOGY incluye las siguientes actividades orientadas al desarrollo [Fernández, 1999a] (ver figura 5.5):

Especificación: El propósito de la especificación es definir en un documento cuál es el principal objetivo de la ontología, con qué propósito se desarrolla (posibles aplicaciones), cuál es su nivel de generalidad y granularidad, y cuál es su alcance (establecimiento de los límites del dominio a cubrir).

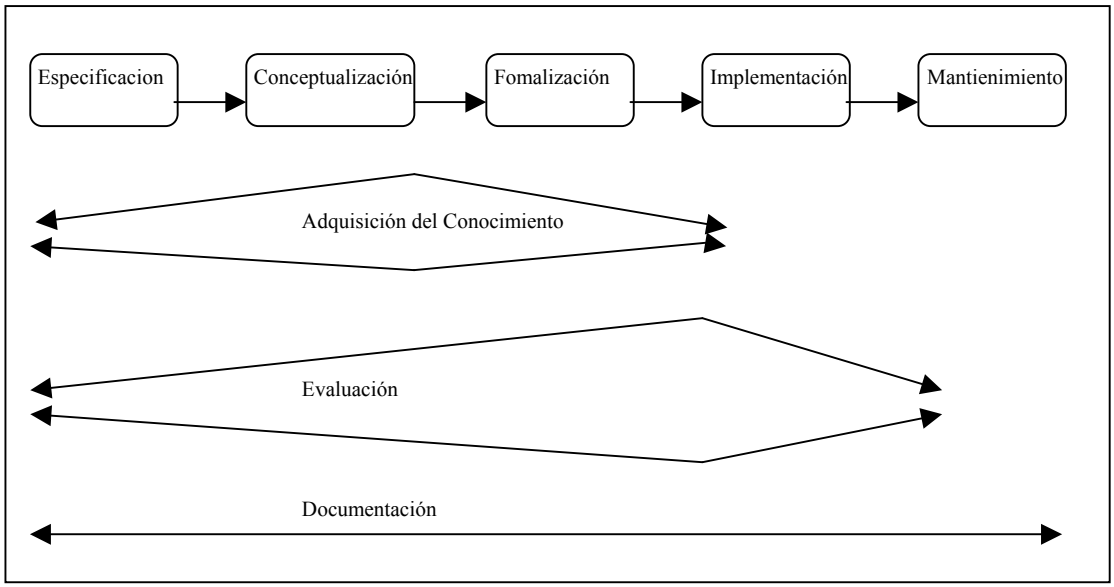


Figura 5.5 Ciclo de vida en METHONTOLOGY (Adaptada de [Corcho et al., 2003])

Conceptualización. Estructura el conocimiento del dominio usando modelos conceptuales en un nivel de conocimiento. Una vez que se ha adquirido la mayoría del conocimiento del dominio en cuestión, los desarrolladores tienen un conjunto de conocimiento no estructurado que debe ser organizado. La actividad de conceptualización tiene como objetivo organizar y estructurar el conocimiento adquirido, usando un conjunto de Representaciones Intermedias (IR) que son modelos para representar conceptos independientes de los lenguajes y ambientes de desarrollo que se usarán en la implementación de la ontología. Gómez-Pérez y otros [Gomez-Perez et al., 1996], en su trabajo "Towards a Method to Conceptualize Domain Ontologies", proponen un conjunto de representaciones intermedias (como Diccionario de Datos, Árboles de Clasificación, Tablas de Atributos, entre otros) para la conceptualización de ontologías de dominio.

Formalización. Transforma el modelo conceptual en un modelo formal o semi-formal computable.

Implementación. Construye modelos computables en un lenguaje computacional, como podría ser Ontolingua, RDF/S (Resource Description Framework/Schema) u OWL (Ontology Web Language).

Mantenimiento. Actualiza y corrige la ontología.

Paralelamente a las actividades de desarrollo de la ontología se deben llevar a cabo actividades de apoyo al desarrollo, importantes para la construcción de la ontología, dichas actividades son:

Adquisición del conocimiento. Consiste en la adquisición del conocimiento del dominio. Esta actividad es independiente de las actividades de desarrollo de la ontología.

Evaluación. Consiste en hacer un valoración técnica de la ontología, su ambiente de software y documentación, con respecto al marco de referencia durante cada fase y entre fases del ciclo de vida.

Documentación. La documentación detallada, clara y exhaustiva es necesaria para cada una de las fases completadas y productos generados.

5.6.3.2 Ontologías Desarrolladas usando la Metodología

La ontología más citada construida con esta metodología es **CHEMICALS** [Fernández et al., 1999b], la cual contiene conocimiento sobre el dominio de elementos químicos y estructuras cristalinas.

Otra ontología desarrollada usando esta metodología es **The Reference-Ontology,** una ontología en el dominio de las ontologías (una meta-ontología) que juega el rol de una especie de páginas amarillas de ontologías. Reúne, describe y relaciona ontologías existentes, usando una organización lógica común.

Finalmente, relacionado al dominio de medición y evaluación, se ha desarrollado la Ontología de Métricas e Indicadores [Martin et al., 2003], que provee una

terminología consensuada para ser usada en proyectos de medición y evaluación y en la catalogación de métricas e indicadores [Martin et al., 2003b].

5.6.4 Resumen

Sintetizando y comparando las metodologías analizadas en las secciones anteriores, presentamos un cuadro comparativo (ver tabla 5.1) basado en el trabajo realizado por Fernández [Fernández, 1999a], en donde se hace un estudio de las metodologías más difundidas de desarrollo de ontologías, estableciendo un conjunto de criterios para comparar dichas metodologías.

Algunos de los criterios seleccionados como base de comparación (se tuvieron en cuenta aquellos criterios del estudio de [Fernández, 1999a] considerados de interés para el desarrollo de nuestra ontología de Memoria Organizacional Basada en Casos) son los siguientes:

- **Detalle de Actividades y Técnicas:** Consideración del grado de detalle en el que están especificadas las actividades y técnicas propuestas por la metodología.

- **Recomendaciones para la formalización del conocimiento:** formalismos propuestos para representar los conocimientos del dominio.

- **Estrategia para construir ontologías:** se refiere a la relación existente entre la ontología y la aplicación que la va a explotar. Fernández las clasifica en: dependientes, semi-dependientes e independientes de la aplicación.

- **Estrategia para identificar conceptos:** Esta puede ser [Uschold, 1996b]: desde los conceptos más concretos a los más abstractos (bottom-up); desde los más abstractos a los más concretos (top-down) o un enfoque intermedio que combina a ambos, también llamado desde dentro hacia fuera (middle-out).

- **Ciclo de vida:** cuál es el ciclo de vida recomendado por la metodología.

- **Diferencias con el estándar IEEE 1074-1995:** el estándar IEEE 1074-1995 describe el proceso de desarrollo de software, las actividades que deben realizarse y las técnicas que se pueden usar en el desarrollo del software. En este ítem se analizan las diferencias con dicho estándar[4] (en los párrafos siguientes se muestran algunas diferencias entre las metodologías de desarrollo de Ingeniería de Software, y las metodologías para el desarrollo y construcción de ontologías).

- **Técnicas recomendadas:** si se proponen técnicas particulares para cada actividad recomendada del ciclo de vida.

- **Ontologías desarrolladas con la metodología:** ejemplos de ontologías desarrolladas usando la metodología.

Tabla 5.1 Comparación de metodologías de desarrollo de ontologías

Criterio	Grüninger	Unschold y King	METHONTOLOGY
Detalle de Actividades y técnicas	Ningún	Muy poco	Muy detalladas.
Recomendaciones para formalizar el conocimiento	Lógica de primer orden	Ninguna	Ninguna
Estrategia para construir ontologías	Semidependiente de la aplicación	Independiente de la aplicación	Independiente de la aplicación
Estrategia para identificar conceptos	Middle-out	Middle-out	Middle-out
Ciclo de vida	No definido totalmente	Ningun	Prototipos evolutivos
Diferencias con el estándar IEEE 1074-1995	Faltan muchos procesos y actividades	Faltan muchos procesos y actividades	Faltan algunos procesos y actividades
Técnicas recomendadas	Desconocido	Desconocido	Recomienda algunas técnicas
Ontologías desarrolladas con la metodología	TOVE	Enterprise ontology	CHEMICALS. Ontología para medio ambiente. Reference Ontology

[4] De acuerdo a la definición de la IEEE [IEEE 1990], software es "Programas de computadoras, procesos, además de documentación y datos posiblemente asociados, que pertenecen al funcionamiento de un sistema de computadora"; las ontologías son parte de un producto de software, por lo tanto deberían ser desarrollados de acuerdo a los estándares propuestos para desarrollo de software general, el cual deberá ser adaptado a las características especiales del desarrollo de ontologías [Fernández, 1999a].

Como conclusión, podemos observar que no existe ninguna metodología totalmente madura si se compara con el nivel de detalle especificado para los procesos en el estándar IEEE 1074 o en ISO 12207 [ISO 12207], por citar sólo algunos. Tampoco existe un consenso entre los desarrolladores de ontologías sobre qué metodología es conveniente usar para distintas necesidades. Se podría indicar que cada grupo aplica su propia metodología.

Según Fernández [Fernández, 1999a], METHONTOLOGY es la metodología más madura, además es la metodología recomendada por la "Fundación para los Agentes Físicos Inteligentes" (FIPA), que promueve la interoperabilidad entre aplicaciones basadas en agentes (http://www.fipa.org). Sin embargo, algunas actividades y técnicas deberían ser especificadas con más detalle.

Cabe destacar que aunque las actividades propuestas por cada metodología en el ciclo de vida de una ontología, tienen alguna influencia proveniente de las actividades especificadas en el proceso de Ingeniería de Software, existen diferencias, y las principales son:

- En Ingeniería de Software luego de las actividades de especificación de requerimientos y análisis, el diseño no es dividido en conceptualización y formalización como la mayoría de las metodologías proponen.

- Las actividades de adquisición de conocimiento no existen con tanto énfasis en el proceso de Ingeniería de Software, y son en cambio una parte esencial de cualquier proceso para la especificación y construcción de ontologías.

Para la construcción de la ontología de Memoria Organizacional Basada en Casos, que será detallada en el próximo capítulo, se usó la metodología METHONTOLOGY [Fernández et al., 1997] por ser una de las metodologías más difundidas y maduras y por contar con actividades y técnicas detalladas para el proceso de desarrollo de ontologías.

5.7 Areas de Aplicación de las Ontologías

A continuación se citan algunas de las áreas en las que actualmente se usan exitosamente las ontologías.

Gestión del Conocimiento. La Gestión del Conocimiento ha emergido como una disciplina diferente, pero estrechamente relacionada a la de Ingeniería de Software y Ontologías. Entre sus aspectos distintivos podemos enumerar a un conjunto de técnicas de elicitación (extracción) y modelado del conocimiento, un conjunto de formalismos para representar y resguardar el conocimiento, y un conjunto de mecanismos para automatizar razonamiento.

El aporte de las ontologías en el proceso de Ingeniería del Conocimiento está en las siguientes etapas:

Modelado conceptual, donde se crea un glosario de la terminología del dominio de la aplicación (los conceptos), se definen relaciones entre dichos términos y restricciones en su uso. Este modelo conceptual explícito es la ontología.

Construcción de la base de conocimiento. Usando la ontología definida en la etapa anterior como un conjunto de esquemas o contenedores de conocimiento, se genera la base de conocimiento con instancias del dominio bajo la forma de reglas, hechos, y restricciones.

Como esta área de aplicación (la de Gestión del Conocimiento y Memoria Organizacional) es justamente el objetivo de esta tesis, la próxima sección la dedicaremos a analizar la aplicación de ontologías en la Gestión del Conocimiento, con más detalle.

Procesamiento de lenguaje natural. En el área de procesamiento de lenguaje natural, una ontología puede mantener la definición de elementos gramaticales del lenguaje y la relación entre ellos, permitiendo por ejemplo el análisis sintáctico de un texto.

Interoperabilidad entre sistemas de información heterogéneos. En la interoperabilidad de sistemas de información, las ontologías se presentan como una importante solución para lograr una integración inteligente, en particular en el área de

bases de datos, una ontología puede ser un elemento clave asociado a un mediador (puente), que integra datos provenientes de fuentes heterogéneas.

Con una ontología terminológica, se pueden organizar los términos que son usados en interacciones entre sistemas heterogéneos, de manera de reconocer cuando una aplicación está usando un término que es más general o más específico que otro que está en uso por otra aplicación.

Si la ontología es formal, se puede contar con una definición más completa de cómo se relaciona un término de un origen con el de otro, y eventualmente usar axiomas definidos que los vinculen por igualdad o que expresen un término exactamente en función del otro, lo que permitiría establecer correspondencias seguras y automáticas entre ellos.

Búsqueda semántica en sitios Web. Usando ontologías asociadas a los motores de búsqueda se puede reemplazar la búsqueda tradicional basada en palabras claves, por una búsqueda semántica, encontrando páginas cuyo contenido difiere sintácticamente, pero semánticamente son similares. De esta forma las consultas no serán procesadas a un nivel terminológico, sino conceptual, con los beneficios que esto supone en el grado de acierto de las respuestas retornadas.

Modelado de empresas. En el área de modelado de empresas, las ontologías desempeñan, entre otros, el rol de mantener una Memoria Organizacional colectiva que permita a los distintos niveles de la empresa interoperar con un lenguaje común y con reglas únicas.

Las Ontologías TOVE (Toronto Virtual Enterprise) desarrolladas en la universidad de Toronto y la Enterprise Ontology desarrollada en la Universidad de Edimburgo, son ejemplos de este tipo de ontologías.

Aplicaciones en la Web Semántica. Las ontologías se volverán un elemento clave en la Web Semántica, ya que permitirán explicitar la semántica de los contenidos dispersos en la Web. Es decir, las ontologías proveen un vocabulario de marcas consensuado y con un significado bien definido, permitiendo su procesamiento automático y la inferencia de nuevos conocimientos.

5.8 Ontologías como soporte en la Gestión del Conocimiento

Muchos investigadores ya han analizado el importante papel de las ontologías en la Gestión del Conocimiento [Ale, 2009], [Davies et al., 2003], [Martin et al., 2009], [Mica et al., 2004] [O'Leary, 1998b]. También la comunidad de investigadores de la Web Semántica [W3Csw 2010], considera a las ontologías una pieza fundamental para publicar contenidos semánticos en la Web, procesables por computadoras [Torres et al., 2004] .

El trabajo de O'Leary en [O'Leary, 1998b] describe el uso de ontologías en sistemas de Gestión del Conocimiento mostrando casos ilustrativos del sector privado, en particular empresas consultoras. Además el mismo artículo defiende varios factores que justifican la necesidad de ontologías en la Gestión del Conocimiento, a saber: la formación de grupos de discusión en temas de intereses particulares, capacidades de búsquedas mejoradas por medio de preguntas/respuestas enriquecidas semánticamente, facilidades de filtros para capturar el conocimiento deseado, el reuso de items de conocimiento, y habilitar la comunicación entre diferentes sistemas y/o personas por medio del uso de un lenguaje común.

Por otra parte, Benjamins y colegas [Benjamins et al., 1998] defienden la Gestión del Conocimiento basada en ontologías dado que resulta en un acceso inteligente a los recursos de conocimiento, sus argumentos son mostrados con un caso de ejemplo para una organización virtual. Otros autores, Davies y colaboradores afirman en la introducción de su libro "*...como tal, el uso de ontologías y las herramientas de apoyo ofrecen una oportunidad a la mejora significativa de las capacidades de Gestión del Conocimiento en las organizaciones grandes...*" [Davies et al., 2003].

El trabajo de Mica y colaboradores, [Mica et al., 2004] afirma que una de áreas de aplicación más significativas de las ontologías es la disciplina comercial de Gestión del Conocimiento y proporciona un análisis del estado del arte en las tecnologías de KM para explicar el papel y la potencial contribución de las ontologías al desarrollo de la Gestión del Conocimiento.

Las ontologías han surgido como la herramienta ideal para desarrollar un modelo conceptual detallado de un dominio seleccionado. Con el creciente acceso a repositorios de conocimiento independientes y heterogéneos, el tratamiento de las diferencias en la estructura y semántica de los objetos de conocimiento juega un rol principal en los KMS [Ale, 2009].

Desde los primeros estudios en sistemas de información interoperables se han hecho progresos relacionados a la heterogeneidad sintáctica (tipos de datos y formatos) y estructural (integración de esquemas, lenguajes de consulta e interfaces) [Aparicio et al., 2005]. A medida que los sistemas de información interoperables fueron confrontados con tareas de Gestión de Conocimiento cada vez más complejas, la tecnología necesaria para tratar con éxito estos temas debió enfocarse en la semántica subyacente a los datos utilizados por estos sistemas. Algunos autores han enfatizado el uso de ontologías y funciones de similitud semántica como mecanismo de comparación de objetos que pueden ser recuperados o integrados a través de repositorios heterogéneos [Guarino et al., 1999] [Rodriguez et al., 2003]. En este contexto, una ontología es un tipo de base de conocimiento que describe conceptos a través de definiciones que son lo suficientemente detalladas y formales como para proveer interoperabilidad semántica en un dominio.

La utilidad de las ontologías en el área de Gestión del Conocimiento es evidente. Las personas, organizaciones y sistemas de software deben comunicarse entre ellos. Sin embargo, debido a las diferentes necesidades y contextos de fondo, puede haber puntos de vista y presunciones ampliamente variados relacionados esencialmente con un mismo tema. Cada uno utiliza diferentes terminologías, cada uno puede tener conceptos, estructuras y métodos diferentes, superpuestos y/o mal emparejados [Uschold et al., 1996a]. En el ámbito organizacional, existe una necesidad de reducir o eliminar la confusión conceptual o terminológica y llegar a un entendimiento compartido. El desarrollo e implementación de una ontología en un dominio dado, puede mejorar la comunicación, lo cual favorece una mayor reutilización, distribución, e interoperabilidad del conocimiento disponible. Desde esta óptica, una ontología es un marco unificador para diferentes puntos de vista y sirve como la base conceptual para la comunicación entre las personas, entre personas y sistemas, y entre sistemas. Las ontologías son consideradas por lo tanto, como poderosas herramientas para solucionar la ambigüedad inherente a la heterogeneidad del conocimiento organizacional ya que

proveen una base semántica y un vocabulario conceptual consensuado, sobre el cual uno puede construir descripciones y actos de comunicación, en fin, gestionar el conocimiento.

La representación final (implementación) de una ontología puede hacer uso de lenguajes más o menos formales, dependiendo del uso que se le intenta dar. Sin embargo, un mejor aprovechamiento automatizado del conocimiento encerrado en una ontología por parte de un sistema de Gestión del Conocimiento, implicará la formalización de la misma en algún lenguaje que pueda ser procesado por computadoras, para permitir una manipulación automática de este conocimiento.

PARTE III

MEMORIA ORGANIZACIONAL BASADA EN CASOS

Capítulo 6- Ontología de Memoria Organizacional Basada en Casos

6.1 Introducción

A medida que el área de Gestión del Conocimiento va progresando hacia una disciplina más madura, se van obteniendo importantes resultados de investigación relacionados a Memoria Organizacional, bases de conocimiento y Razonamiento Basado en Casos. De hecho ya se ha publicado una gran cantidad de información relacionada a organización de memorias corporativas, representación del conocimiento mediante casos, Razonamiento Basado en Casos y medición de similitud para distintos propósitos y dominios.

Sin embargo, se observa que el desarrollo rápido y caótico de tanta información relacionada a Memoria Organizacional y Razonamiento Basado en Casos, proveniente de fuentes heterogéneas, y con frecuencia, la falta de consenso en la terminología utilizada, dificulta su aplicación y reuso de forma eficiente en los procesos de Gestión del Conocimiento distribuido.

Por lo tanto se considera importante llegar a un acuerdo entre investigadores, profesionales y otros participantes relacionados al ámbito de Memoria Organizacional y Razonamiento Basado en Casos, sobre la terminología y significado de conceptos primitivos como son: *Memoria Organizacional, base de conocimientos, ítem de conocimiento, caso, problema, solución, modelo de similitud*, entre otros. Con este propósito, en este capítulo se realiza una conceptualización del dominio de Memoria Organizacional Basada en Casos que se formaliza en una ontología para dicho ámbito.

La ontología, combina aspectos de Memoria Organizacional con Razonamiento Basado en Casos para representar cada ítem de conocimiento informal. La estructuración del conocimiento informal en casos, facilita la captura, recuperación, transferencia, y reuso a través de un procesamiento automático. Además, evita la dispersión de la experticia concentrando el conocimiento de todos los expertos en

diversos casos, y permite la continua evolución de la memoria gracias a la adición progresiva de nuevo conocimiento [Dieng et al. 1999].

En el área de representación del conocimiento, la idea de aplicar métodos de Razonamiento Basado en Casos para recuperar experiencias exitosas no es novedosa [Weber et al., 2001]. Y, aunque los beneficios derivados de la aplicación de dichos métodos sean bien conocidos, es también notoria la falta de consenso en los conceptos y terminología empleados tanto sea en administración del conocimiento, como en áreas de Razonamiento Basado en Casos. A pesar de numerosos esfuerzos durante la última década por parte de grupos de investigación por tratar de hallar algún estándar, lo cierto es que la administración del conocimiento, hoy en día, se encuentra en un estado en el cual la terminología está aún por definirse, consolidarse y lograr un consenso.

La ontología propuesta, es usada como base para el diseño y construcción de una Memoria Organizacional Basada en Casos (MOBC) y un sistema de Gestión del Conocimiento distribuido, que permitirá explorar, reusar y compartir el conocimiento informal capturado de experiencias, buenas prácticas y lecciones aprendidas (es decir casos). En el próximo capítulo se mostrará la arquitectura y la construcción de un prototipo de dicho sistema.

Posteriormente, en el capítulo 8, a través de una aplicación realizada para tal fin y de ejemplos concretos, se muestran los beneficios de la MOBC en la Gestión del Conocimiento. Particularmente se presenta su aplicación a un sistema de recomendación en etapas de decisión de proyectos [Olsina et al., 2007] de medición y evaluación de calidad de productos software y Web. Un tipo de empresas donde la eficiencia en la Gestión del Conocimiento organizacional es un factor clave para su competitividad y productividad, son las empresas dedicadas al desarrollo de software, ya que son organizaciones donde las personas continuamente expanden sus capacidades para crear los resultados que ellos realmente desean, donde nuevos patrones de pensamiento se nutren, y donde los individuos continuamente aprenden de sus experiencias. Con este caso de estudio, mostraremos los beneficios del uso de la Memoria Organizacional basada en ontologías y casos.

6.2 Fuentes de Conocimiento

Las fuentes de conocimiento que se tuvieron en cuenta para el desarrollo de la ontología MOBC [Martín et al., 2006], [Martín et al., 2008], [Martin et al., 2009] provienen de varios recursos a saber:

- El estándar australiano relacionado a KM [AS5037, 2005], que proporciona una guía no prescriptiva y ahonda en la comprensión de conceptos en Gestión del Conocimiento.

- Artículos de investigación reconocidos en el área de Gestión del Conocimiento y Memoria Organizacional como son [Wiig, 1993], [Nonaka et al., 1995]; [Prusak, 1996], [Conklin, 1996], [Ackerman et al., 2000], [Handzic et al., 2004], [Malhotra 2005] entre otros.

- Artículos de la investigación reconocidos en el área de Razonamiento Basado en Casos [Kolodner, 1993], [Aamodt et al., 1994], [Sankar et al., 2004].

- Reuniones entre investigadores del Grupo de Investigación de Desarrollo en Ingeniería de Software (GIDIS-Web)[5] de la Facultad de Ingeniería, Universidad Nacional de La Pampa, para alcanzar un acuerdo general para esta terminología.

- Algunos conceptos se tomaron de la ontología terminológica WordNet [Fellbaum, 1998].

En este punto es importante mencionar que hasta ahora no existe ningún estándar internacional (como ISO) en el área de Gestión del Conocimiento [Ferguson 2006].

[5] http://gidis.ing.unlpam.edu.ar/

6.3 Ontología de MOBC

6.3.1 Metodología Empleada

Para el desarrollo de la ontología se usó la metodología METHONTOLOGY (descripta en la sección 5.6.3), que provee guías para especificar ontologías en un nivel de conocimiento, particularmente provee técnicas para especificar la conceptualización. Esta metodología fue propuesta por Fernández [Fernández et al., 1997] e incluye un conjunto de pasos y estrategias denominados proceso de desarrollo de la ontología donde se representan las principales actividades de un ciclo de vida basado en la evolución de prototipos; y la metodología propiamente dicha, en la cual se especifican al menos los siguientes pasos:

Paso 1 Especificación: El objetivo de la especificación es considerar los principales requerimientos de la ontología, el alcance y el propósito.

Paso 2 Conceptualización: Cuando todo el conocimiento ha sido adquirido, el ontologista tiene una pila de conocimiento desestructurado que debe ser organizado. La conceptualización ayuda a organizar y estructurar el conocimiento usando un lenguaje externo de representación, que es independiente de los lenguajes de implementación. Dicho lenguaje de representación puede ser UML[6], y entonces en este paso se construyen diagramas UML que puedan interpretar los expertos y ayudar a la comunicación entre los participantes.

Paso 3 Implementación: Consiste en implementar el modelo conceptual en un lenguaje formal tal como RDF/S (Resource Description Language) u OWL (Ontology Web Language).

[6] *Unified Modelling Language*: Lenguaje unificado de modelado para la construcción de software.

6.3.2 Especificación

El principal objetivo de la ontología de la MOBC es contribuir a la integración de los conceptos básicos de Memoria Organizacional y Razonamiento Basado en Casos, de manera que sirva como base para el intercambio de conocimiento organizacional y aún más, que sirva como base para el sistema de recomendación, otorgándole potencia semántica al especificar formalmente el significado de los conceptos involucrados.

La ontología de Memoria Organizacional es una ontología de nivel genérico, a partir de la cual se pueden formular otras representaciones para ser aplicadas a dominios específicos (ver Fig. 6.1). Es decir, la ontología de MOBC debe estar relacionada con otras ontologías que operan en distintos niveles de abstracción (en particular, dos niveles según se observa en dicha figura).

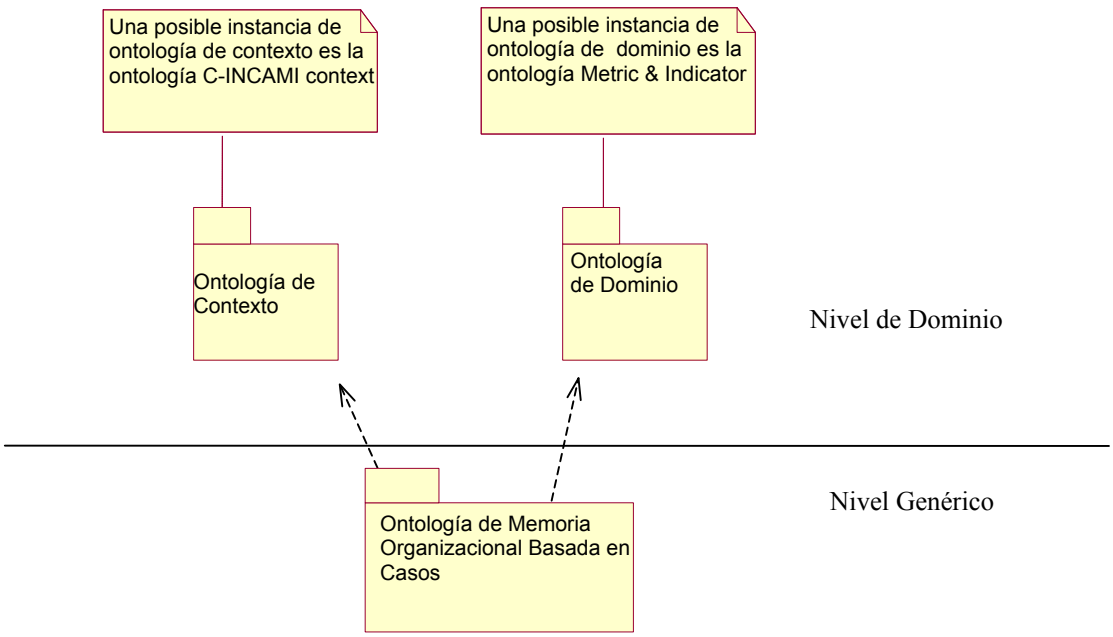


Figura 6.1. Relación entre ontologías en el nivel de dominio específico y la ontología de nivel genérico de Memoria Organizacional.

Por un lado se define la ontología genérica de Memoria Organizacional Basada en Casos (MOBC) y por otro lado, para caracterizar los casos de acuerdo al conocimiento de un dominio específico y su contexto [Molina et al., 2007], se deberán definir las ontologías de dominio y de contexto respectivamente. Por ejemplo, se puede

usar la ontología de métricas e indicadores definida por Martín y colaboradores [Martín et al., 2003] para instanciar la Memoria Organizacional, de manera que pueda se aplicada a la gestión de conocimiento en el área de medición y evaluación de productos software.

6.3.3 Conceptualización

En esta sección se muestra la conceptualización del dominio de Memoria Organizacional Basada en Casos, presentando primero estrategias y técnicas usadas para tal fin, y una descripción detallada del dominio para facilitar su entendimiento.

6.3.3.1 Técnicas Empleadas en la Etapa de Conceptualización

Para la etapa de conceptualización se emplearon estrategias y técnicas como las tablas sugeridas por la metodología METHONTOLOGY [Gomez-Perez 1996] y diagramas UML.

El uso de UML como lenguaje de representación de ontologías ha sido propuesto por diversos autores [Cranefield et al., 1999] [Cranefield, 2001a] [Kogut et al., 2003]. Algunos de ellos también proponen transformaciones de UML a lenguajes formales para la implementación de ontologías modeladas con UML [Cranefield, 2001b], [Baclawski, 2002].

Si bien UML se utilizó originalmente para la comunicación entre seres humanos de modelos para la construcción de sistemas con un enfoque orientado a objetos, actualmente se está usando para modelar artefactos de Ingeniería de Software más declarativos tal como son las ontologías [Kogut et al., 2003], los esquemas XML [Carlson, 2001], esquemas RDF y esquemas de bases de datos. Además existen otras buenas razones para usar el lenguaje UML como notación para la conceptualización de una ontología, a saber:

- Las ontologías incluyen taxonomías de conceptos (jerarquías clase/subclase), relaciones entre conceptos (asociaciones) y definición de

atributos de conceptos. Esta información de las ontologías puede ser modelada con diagramas de clases del lenguaje UML.

- UML es una notación gráfica que surgió con el aporte de la experiencia de muchos años en análisis y diseño de software, por parte de una variedad de compañías en un amplio espectro de industrias y dominios.

- Es un estándar abierto administrado por el Object Management Group (OMG).

- Provee mecanismos para definir extensiones para aplicaciones específicas como es el modelado de ontologías.

- Está soportado por herramientas CASE ampliamente difundidas, y más accesibles a los profesionales del software que herramientas específicas para ontologías como Ontolingua y Protegè, las cuales requieren experticia en la representación del conocimiento [Kogut et al., 2008].

La conceptualización o elaboración de la ontología de Memoria Organizacional Basada en Casos propiamente dicha ha sido realizada teniendo en cuenta los siguientes pasos en forma iterativa e incremental:

1. Definir el glosario de conceptos a partir de las fuentes de conocimiento citadas en la sección 6.2, teniendo en cuenta los objetivos y el alcance de la ontología.

2. Definir las interrelaciones semánticas entre dichos conceptos representándolas mediante un diagrama de clases UML, a la vez que se elabora una tabla de clases y de relaciones con las distintas clases que se van identificando.

3. Analizar los conceptos relacionados para identificar las partes comunes a dos o más conceptos. Decidir si estas partes son a su vez conceptos y, en su caso, incluirlos en el glosario de conceptos.

4. Identificar los atributos de cada concepto y construir la tabla de atributos, incluyéndolos a la vez al diagrama de clases UML.

5. Comprobar la completitud y corrección de todas las tablas, analizando el modelo conceptual UML, los objetivos y alcance de la ontología, la correspondencia con las fuentes de información citadas y además analizando y resolviendo conflictos de interpretación conceptual.

6.3.3.2 Descripción Conceptual de la Memoria Organizacional Basada en Casos

En esta sección presentamos una descripción de los principales conceptos de la ontología cuyo diagrama UML puede observarse en la figura 6.2. Debido a que muchas de las fuentes de información para la construcción de la ontología están escritas en idioma inglés, hemos considerado más ilustrativo (en el texto que sigue) indicar los términos de la ontología en inglés, entre paréntesis, al lado de su correspondiente término en español.

Una Memoria Organizacional (*OrganisationalMemory*) es un repositorio que almacena la totalidad del conocimiento formal e informal presente en una organización [Conklin, 1996] y por lo tanto tiene una o más bases de conocimiento (*KnowledgeBase*). Un tipo de base de conocimiento es la base de conocimiento basada en casos (*CaseKnowledgeBase*) que almacena el conocimiento adquirido en experiencias pasadas, buenas prácticas, lecciones aprendidas, heurísticas, etc. de distintos dominios; es decir, almacena casos. Los casos son un tipo de ítem de conocimiento (*KnowledgeItem*).

Un caso es una pieza contextualizada de conocimiento que representa una experiencia, contiene la experiencia pasada que es el contenido del caso y el contexto en el cual la lección puede ser utilizada [Kolodner, 1993]. La experiencia (*Caso*) se compone de un problema (*Problem*) y su solución (*Solution*). Teniendo en cuenta esta definición, es fundamental en toda Memoria Organizacional guardar la información del contexto (*Context*) donde ocurre cada caso. Una forma de modelar contexto ha sido desarrollada por Molina y colaboradores [Molina et al., 2007].

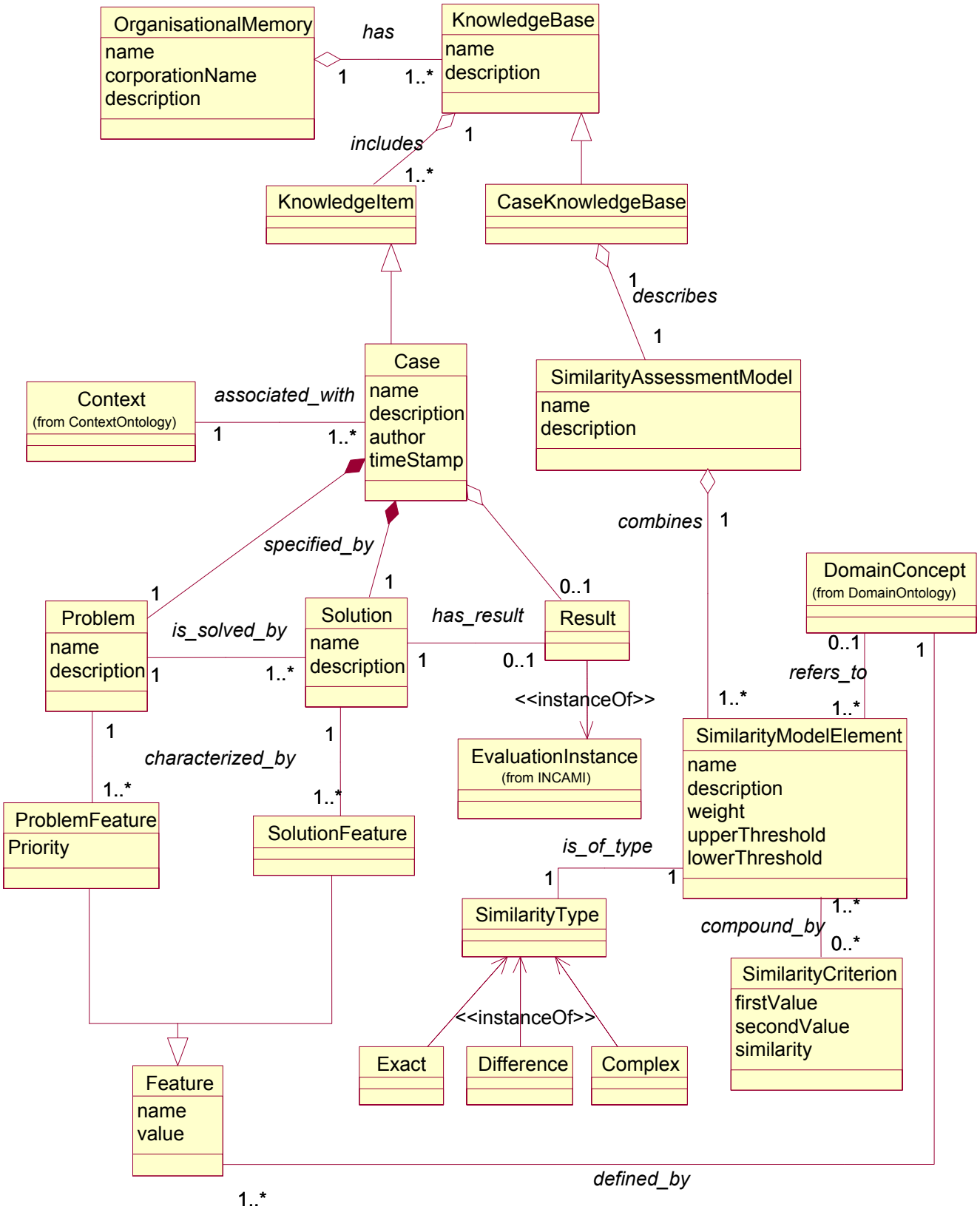


Figura 6.2. Modelo conceptual de la ontología de Memoria Organizacional Basada en Casos (tomada de [Martin et al., 2009])

La representación del conocimiento a través de casos, facilita el reuso del conocimiento adquirido en situaciones de problemas similares pasados para ser aplicado a un nuevo problema [Aamodt et al., 1994]. Tradicionalmente, existen varios métodos para representar casos, que van desde los no estructurados hasta representaciones completamente estructuradas y automáticamente procesables [Chen et al., 2003]. Para proveer a la Memoria Organizacional la máxima capacidad de procesamiento, se optó por la última estrategia, donde se especifican formalmente los conceptos involucrados y, por lo tanto, son automáticamente procesables.

En una definición formal un caso es un par ordenado <P, S>, donde: P es el espacio del problema; y S es el espacio de la solución. Como un mismo problema, puede tener varias soluciones, no todas con el mismo resultado, para una mejor elección de una solución en la toma de decisiones, se necesita guardar también el resultado (*Result*) obtenido en la aplicación de cada solución.

Los problemas y las soluciones se describen a través de variables de características del problema (*ProblemFeature*) y variables de características de la solución (*SolutionFeature*) respectivamente, y ambas son variables de características (*Feature*). Debido a que los casos almacenados en la memoria se refieren a conocimiento de un dominio específico, estos atributos (*Feature*) están formalmente definidos por un concepto de dominio (*DomainConcept*) que constituye un término en la ontología de dominio relacionada a la Memoria Organizacional.

En esta tesis, se ilustrará la MOBC, aplicándola a un sistema de soporte a la recomendación en aseguramiento de calidad, por lo tanto, usaremos la ontología de métricas e indicadores, especificada en [Martin et al., 2003], [Olsina et al., 2004]. Esta definición formal de los términos en la ontología de dominio asociada, hace que la definición de los casos se realice de manera no ambigua, asegurando un intercambio de conocimiento eficaz. La especificación de la ontología de dominio proporciona la terminología (*Domain Concept*) que provee los tipos de las variables que caracterizan al problema y a la solución.

Con respecto a la representación de información de contexto ocurre algo similar, según la definición de Dey: *"Contexto es cualquier información que puede ser usada para caracterizar la situación de una entidad, siendo una entidad una persona, lugar, u*

objeto que se considera relevante en la interacción entre un usuario y una aplicación, incluyendo también a ellos mismos, usuario y aplicación" [Dey, 2001]. En términos generales, un sistema es consciente del contexto, si lo usa para proporcionar información y/o servicios, donde la relevancia de dicha información depende del contexto actual. De igual modo, en las memorias organizacionales basada en casos, la relevancia de la información de experiencias similares, está muy ligada al contexto en que es válido aplicar dicha experiencia. En este caso, y volviendo a la definición de Dey, la entidad relevante en la interacción entre el usuario y la aplicación es el caso. Por los tanto, debemos definir cuál es la información que puede ser usada para caracterizar la situación de un caso, es decir su contexto (*Context*) de aplicación. Los parámetros para caracterizar dicho contexto, están definidos en la ontología de contexto, por lo tanto, una Memoria Organizacional, tiene asociada una ontología de contexto, que permitirá definir en forma no ambigua el contexto de cada caso que almacena.

El proceso de Razonamiento Basado en Casos consiste en asignar valores a las variables de características del problema (caracterizar el problema) y encontrar los valores adecuados para las instancias de la solución, a través de criterios de evaluación de similitud de casos.

Muchas de las aplicaciones de CBR se han enfocado en dominios de problemas específicos. Sin embargo, para que un sistema CBR sea útil a una organización, debería ajustarse a las principales fuentes de conocimiento que puede ser de distintos dominios y, por lo tanto, necesitan funciones de similitud apropiadas a cada base de casos. Así, para cada base de conocimiento basado en casos, se debe especificar un modelo de evaluación de similitud (*SimilarityAssessmentModel*), que a su vez combina varios elementos de similitud (*SimilarityModelElement*), uno para evaluar cada característica constituyente del caso.

Debe existir entonces un modelo de similitud global que especifique de qué manera se realiza la comparación global de los casos, que se compone a su vez de modelos de similitud elementales que calcula la similitud de cada variable de característica del problema que constituye cada caso. Los modelos de similitud elementales están referidos a un concepto de la ontología de dominio que define la característica a comparar y tendrán atributos que indiquen la manera de ser comparados tales como el peso de relevancia relativa (*weight*) y el tipo de función de similitud

(*SimilarityType*) que permitirá obtener un valor para la característica de un problema tras ser evaluada contra la misma característica de otro problema. A continuación se explica con más detalle el el modelo de similitud.

Tradicionalmente, la similitud entre un caso recuperado R y un nuevo caso C, se define como la suma de las similitudes entre los valores de sus características constituyentes multiplicados por sus pesos de relevancia relativa:

$$Similitud\ (R, C) = \sum_{f \in F} w_f \times sim_f (f_R, f_C) \qquad \text{(Ecuación 6.1)}$$

donde w_f es el peso de relevancia de la característica f, y sim_f es la función de medida de similitud de una característica específica f.

Por lo tanto, para proveer una representación adecuada de la similitud, es necesario representar tanto los pesos de relevancia como la descripción de la función de similitud para una característica específica. Los pesos se representan como un atributo dentro de cada modelo de similitud elemental (*SimilarityModelElement*), y la función de similitud se restringe a tres tipos generales de funciones de similitud: Exacta (*Exact*), Diferencia (*Difference*) y Compleja (*Complex*). La elección de estos tres tipos de funciones se basa en el análisis de numerosos trabajos de investigación en el área [Coyle et al., 2004], por considerar que cubre las necesidades de representación de similitud de la mayoría de los casos aplicados al área de Ingeniería de Software/Web.

La **función de similitud Exact**, devuelve 1 si los valores de característica son iguales, y 0 en otro caso.

La **función de similitud Difference**, es inversamente proporcional a la diferencia entre los valores de las características. Esta función solamente se puede aplicar cuando es posible definir la diferencia entre los valores, por ejemplo entre valores numéricos la diferencia es la diferencia matemática (distancia euclidiana).

La **función de similitud Complex**, resuelve la similitud para todas aquellas situaciones donde las dos funciones de similitud anteriores no son aplicables. Por ejemplo, la diferencia semántica entre dos términos sinónimos, que no necesariamente son totalmente iguales ni totalmente distintos. Si el número de valores posibles de la característica es finito, es posible guardar la medida de similitud para cada combinación

de valores posibles. En nuestro modelo, estos parámetros están representados en la clase Criterio de Similitud (*SimilarityCriterion*).

6.3.3.3 Descripción Práctica a través de un Ejemplo

Para ilustrar los principales conceptos, atributos y relaciones de la ontología, en esta sección se muestra en un ejemplo de base de conocimiento basada en casos y su modelo de evaluación de similitud para un dominio específico: el dominio de medición y evaluación para los productos del software [Olsina et al., 2007]. Esta base de casos almacena conocimiento relacionado a proyectos de medición y evaluación de manera que sirva como la base para un sistema de la recomendación que de soporte al proceso de diseño de un nuevo proyecto similar a otro del pasado. La manera en que los casos se caracterizan y evalúan con respecto a sus similitudes depende del dominio de aplicación específico; por consiguiente, es necesario establecer una ontología del dominio que proporcione los tipos del atributos (*Feature*) que caracterizan al problema y su solución. En este ejemplo, se usa la ontología de métricas e indicadores como la ontología de dominio (vea el Fig. 6.1) usada en el marco de medición y evaluación INCAMI [Olsina et al., 2007].

6.3.3.3.1 Qué es INCAMI?

El marco INCAMI (acrónimo de Information Need, Concept model, Attribute, Metric and Indicator [Olsina et al., 2005] [Olsina et al., 2007]) se fundamenta en un conjunto de componentes y conceptos involucrados en la medición y evaluación de requerimientos non-funcionales en proyectos software y Web, como parte de actividades de aseguramiento de calidad de una organización. Está basado en una ontología de métricas e indicadores [Martin et al., 2003] [Olsina et al., 2004] que explícitamente define esos conceptos, y también en el proceso subyacente en la metodología WebQEM (Web Quality Evaluation Method) [Olsina et al., 2002]. Los principales componentes del marco INCAMI son, a saber:

- El componente de especificación de requerimientos no funcionales, que trata la definición de la Necesidad de Información y la especificación

requerimientos por medio de uno o más Modelos de Concepto (ConceptModel) (ver Fig. 6.3) (Notar que un modelo de concepto puede ser instanciado, por ejemplo. en calidad externa, la calidad en uso, entre muchos otros).

- El componente de diseño y ejecución de la medición, que trata de la especificación de entidades concretas a ser medidas, la selección de métricas para cuantificar los atributos del modelo de calidad y el registro de las medidas recogidas; este componente se centra en el concepto "Métrica".

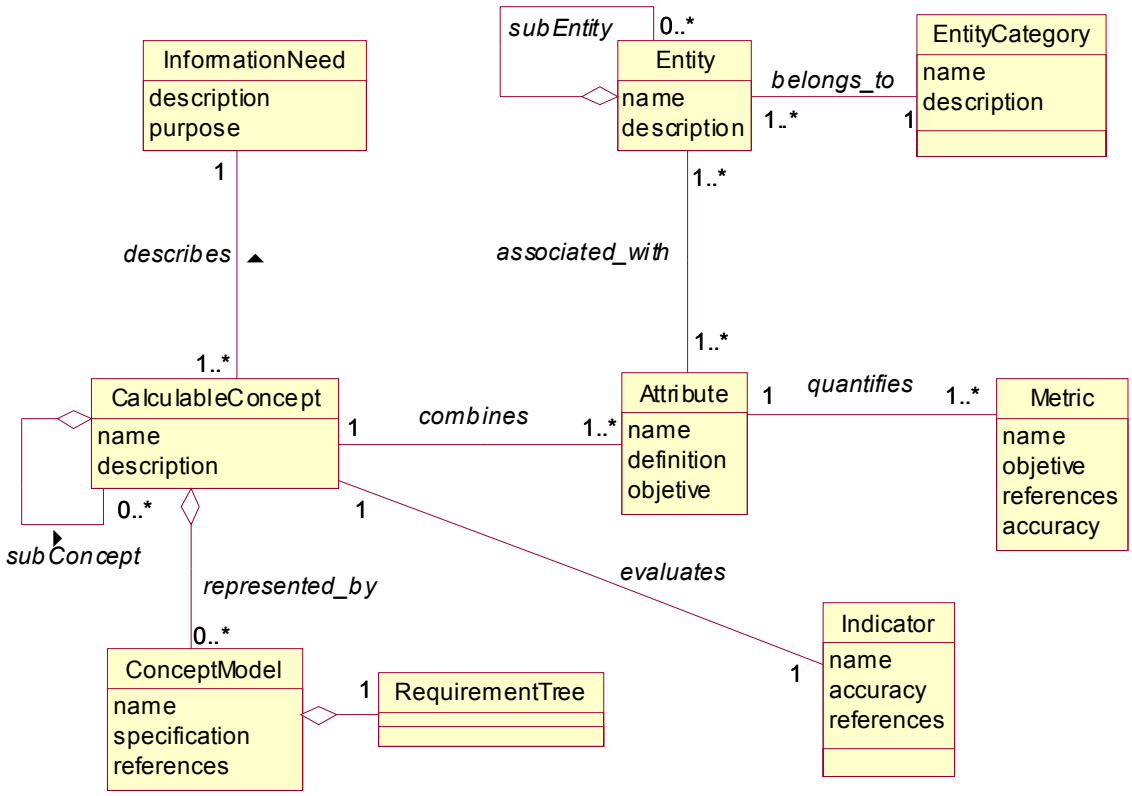


Figura. 6.3 Componentes de requerimientos no funcionales de la ontología de dominio de métricas e indicadores

- El componente de diseño y ejecución de la evaluación, que trata de la definición de indicadores elementales y globales, criterios de decisión y modelos de agregación que ayudarán a interpretar al modelo de concepto seleccionado; este componente se centra en el concepto de "Indicador" (ver [Olsina et al., 2007] para más detalles).

En la Fig. 6.3 se muestra un diagrama resumido de algunos conceptos del componente de requerimientos de INCAMI para ilustrar el próximo caso práctico.

6.3.3.3.2 Base de Conocimiento para la Proyectos de Medición y Evaluación

Para ilustrar una aplicación de la ontología de Memoria Organizacional Basada en Casos, se presenta la definición de una base de conocimientos para proyectos de medición y evaluación. Esta memoria guarda conocimiento sobre proyectos de medición y evaluación (como problemas) y sus respectivos árboles de requerimientos no funcionales (como las soluciones). La tabla 6.1 muestra un posible modelo de similitud para este ejemplo. Esta base de conocimientos caracteriza los proyectos de medición y evaluación a través de cuatro atributos: *InformationNeed.purpose*, *CalculableConcept.name*, *EntityCategory.name* y *CalculableConcept.indicator Accuracy* que se definen en la ontología de dominio de métricas e indicadores cuyos términos se mostraron en la figura 6.3.

Para cada atributo que caracteriza un caso, nosotros debemos establecer su peso (*Weight*) y su tipo de función de similitud. Estas decisiones de diseño de la base de conocimiento deben ser hechas por un especialista teniendo en cuenta qué atributos se consideran más relevantes desde el punto de vista de la similitud para evaluar finalmente la similitud global de dos casos.

Tabla 6.1 Ejemplo de modelo de valoración de similitud para una base de casos

Feature	**Description**	**similarity Type**	**Weight**
InformationNeed.purpose	Proposito de la evaluación	Complex	0.2
CalculableConcept.name	El concepto calculable a evaluar, por ejemplo: calidad, costo, etc.	Exact	0.25
EntityCategory.name	El tipo de la entidad a evaluar	Exact	0.35
CalculableConcept.indicatorAccuracy	Nivel de exactitud en la evaluación	Difference	0.2

Como se puede observar en Tabla 6.1, los atributos CalculableConcept.name, EntityCategory.name y CalculableConcept.indicatorAccuracy tienen tipos de similitud

Exacta y *Diferencia*; por otro lado, para el atributo InformationNeed.purpose el cálculo de similitud es menos trivial y deben establecerse las medidas de similitud de cada par de posibles valores.

Supongamos que los posibles valores por el atributo InformationNeed.purpose son tres: Evaluar (*Evaluate*), Estimar (*Estimate*) y Mejorar (*Improve*). Entonces un experto determina que hay mayor similitud entre los valores *Evaluate* y *Estimate* (similitud 0.8) que entre los valores *Estimate* e *Improve* (0.3), y entre *Evaluate* e *Improve* (0.4). La tabla 6.2 muestra esta valoración de similitud compleja donde se puede ver que la mayor similitud semántica es entre *Evaluate* y *Estimate*.

Tabla 6.2. ***Ejemplo de valoraciones de similitud complejas para tres valores de un atributo***

	Evaluate	**Estimate**	**Improve**
Evaluate	1	0.8	0.4
Estimate	0.8	1	0.3
Improve	0.4	0.3	1

Una vez definida la estructura del caso y su modelo de evaluación de similitud, cada caso se guarda con todos los valores de los atributos que lo caracterizan y su solución respectiva. En la Fig. 6.4 se muestran dos ejemplos de casos. Estos casos representan problemas de obtener un árbol de requerimientos para una necesidad de información específica y la solución aplicada, que consiste en el árbol de requerimientos diseñado para tal fin.

Además, los dos casos anteriores se muestran en la tabla 6.3 con sus valores de atributos, tipos de función de similitud, y similitud calculada para cada atributo.

<table>
<tr><td colspan="3">CASO 1: Obtener el árbol de requerimientos para evaluar la eficacia de desarrollo de una persona.</td></tr>
<tr><td rowspan="4">PROBLEMA:</td><td>InformationNeed.purpose</td><td>Evaluate</td></tr>
<tr><td>CalculableConcept.name</td><td>Efficiency</td></tr>
<tr><td>EntityCategory.name</td><td>Person</td></tr>
<tr><td>Indicator.accuracy</td><td>0.7</td></tr>
<tr><td>SOLUCIÓN:</td><td colspan="2">1. Development Efficiency
1.1. Development Completeness
Associated attributes</td></tr>
<tr><td colspan="3">CASO 2: Obtener el árbol de requerimientos para mejorar la eficacia de desarrollo de un grupo.</td></tr>
<tr><td rowspan="4">PROBLEMA:</td><td>InformationNeed.purpose</td><td>Improve</td></tr>
<tr><td>CalculableConcept.name</td><td>Efficiency</td></tr>
<tr><td>EntityCategory.name</td><td>Group</td></tr>
<tr><td>Indicator.accuracy</td><td>0.5</td></tr>
<tr><td>SOLUCIÓN:</td><td colspan="2">1. Development Efficiency
1.1. Development Productivity
Associated attributes
1.2. Development Completeness
Associated attributes</td></tr>
</table>

Figura. 6.4 Ejemplo de representación de dos casos guardados para necesidades de información específicas.

Tabla 6.3 Ejemplo de cálculo de similitud entre los valores de atributos de dos casos.

Atributo	**Tipo de Similitud**	**Caso 1**	**Caso 2**	**Similitud**
InformationNeed.purpose	Compleja	Evaluate	Improve	0.4
CalculableConcept.name	Exacta	Efficiency	Efficiency	1
EntityCategory.name	Exacta	Person	Group	0
CalculableConcept.indicatorAccuracy	Diferencia	0.7	0.5	0.2

Para calcular la similitud global entre los dos casos, se aplica la ecuación 6.1. De manera que, reemplazando los pesos definidos en el modelo de similitud (Tabla 6.1) y

los valores de similitud local calculada para cada atributo (tabla 6.3) los resultados obtenidos son:

$$\text{Similitud (Caso 1, Caso 2)} = 0.2 \times 0.4 + 0.25 \times 1 + 0.35 \times 0 + 0.2 \times 0.2 = 0.37$$

Un nuevo proyecto de medición y evaluación se puede beneficiar de la MOBC recuperando la información de diseño del proyecto más similar y adaptándolo al nuevo caso. Supongamos que se necesita desarrollar un nuevo proyecto de medición y evaluación para evaluar la eficacia de desarrollo de un grupo, y que la base de conocimiento alberga los dos casos mostrados en la figura 6.4, entre otros. En lugar de diseñar el nuevo árbol de requerimiento completamente desde el principio, se puede tomar ventaja del conocimiento guardado, recuperando y reusando el árbol de requerimiento más similar basándose en la similitud de casos. Por consiguiente, la tabla 6.4 muestra el cálculo de similitud de cada atributo del nuevo caso (Nuevo) comparándolo con los anteriores, es decir, los casos Caso1 y Caso2.

Tabla 6.4 Ejemplo de cálculo de similitud entre los dos casos anteriores y el nuevo.

Atributo	**Caso 1**	**Caso 2**	**Nuevo Caso**	**similitud Caso1/Nuevo**	**Similitud Caso2/Nuevo**
InformationNeed.purpose	Evaluate	Improve	Evaluate	1	0.4
CalculableConcept.name	Efficiency	Efficiency	Efficiency	1	1
EntityCategory.name	Person	Group	Group	0	1
CalculableConcept.indicatorAccuracy	0.7	0.5	0.8	0.1	0.3

Nuevamente se calculan las similitudes globales entre cada uno de los dos casos anteriores y el nuevo caso, que dan como el resultado:

$$\text{Similitud (Caso 1, Nuevo)} = 0.2 \times 1 + 0.25 \times 1 + 0.35 \times 0 + 0.2 \times 0.1 = 0.47$$

$$\text{Similitud (Caso 2, Nuevo)} = 0.2 \times 0.4 + 0.25 \times 1 + 0.35 \times 1 + 0.2 \times 0.3 = 0.74$$

Como se observa, el Caso 2 resulta ser el más similar al nuevo caso y por consiguiente su árbol de requerimiento se recomienda para el reuso en el nuevo caso.

6.3.3.4 Definición de Términos de la Ontología y sus Relaciones

En esta sección definimos todos los términos de la ontología de Memoria Organizacional Basada en Casos que luego se presentan en una tabla de conceptos (ver

tabla 6.5) ilustrando cada concepto con un ejemplo y mostrando cómo se relaciona con otros conceptos.

6.3.3.4.1 Caso (Case)

Definición

Ítem de conocimiento contextualizado que representa una experiencia por medio de un problema y su solución.

Relaciones

- Uno o más casos se asocian a un contexto.
- Un caso contiene un problema.
- Un caso contiene una solución.
- Un caso contiene ningún o varios resultados.

Ejemplos

Ejemplo de un caso para diseño de testing de un módulo con dos variables de entrada X, Y

Problema: Diseñar Casos de testing para un módulo con dos variables de entradas X e Y .

Solución : Los casos de prueba diseñados fueron 9:

1. X=valor válido ; Y= valor válido
2. X=valor válido ; Y= valor en el límite de validez
3. X=valor válido ; Y= valor inválido
4. X=valor en el límite de validez ; Y= valor válido
5. X=valor en el límite de validez; Y= valor en el límite de validez
6. X=valor en el límite de validez; Y= valor inválido
7. X=valor inválido ; Y= valor válido
8. X=valor inválido ; Y= valor en el límite de validez

9. X=valor inválido ; Y= valor inválido

Resultado: Todos los defectos existentes fueron detectados.

Nota

La definición más difundida en la literatura, es la de Kolodner: "*A case is a contextualised piece of knowledge representing an experience*" [Kolodner, 1993]. Esta definición fue enriquecida para destacar las partes componentes de un caso.

6.3.3.4.2 Base de Conocimiento Basada en Casos (Case Knowledge Base)

Definición

Una base de conocimiento específica dónde cada ítem de conocimiento se representa a través de un caso.

Relaciones

- Una base de conocimientos basada en casos es descripta por un modelo de evaluación de similitud.

Ejemplos

Base de conocimiento de casos de testing, de casos de diseño arquitectónico, de casos de implementación de algoritmos.

6.3.3.4.3 Compleja (Complex)

Definición

Un tipo de función de similitud que guarda la medida de similitud para cada par de posibles valores de una característica.

Ejemplos

La siguiente tabla muestra una función de similitud compleja entre distintos colores de piel de un paciente:

	Pálido	**Amarillo**	**Rosado**	**Rojo**
Pálido	1	0.82	0.43	0.1
Amarillo	0.82	1	0.65	0.32
Rosado	0.43	0.65	1	0.85
Rojo	0.1	0.32	0.85	1

Nota

La función de similitud compleja hace referencia a una similitud semántica entre valores de características que son difíciles de medir automáticamente. Estas similitudes generalmente las determina un experto.

6.3.3.4.4 Diferencia (Difference)

Definición

Un tipo de función de similitud que es inversamente proporcional a la diferencia entre valores de características.

Ejemplos

La similitud de características que hacen referencia a fechas o cantidades se pueden modelar con una función de tipo diferencia.

6.3.3.4.5 Exacta (Exact)

Definición

Un tipo de función de similitud que devuelve 1 si dos valores de características son iguales, y 0 en otro caso.

Ejemplos

La similitud de características como *Lenguaje de programación usado, Plataforma*, etc.

6.3.3.4.6 Característica (Feature)

Definición

Una propiedad mensurable, física o abstracta, de una entidad.

Relaciones

- Una o más características están definidas por un concepto de dominio de una ontología de dominio.

Ejemplos

Ejemplos de características del caso "*Diseño de evaluación*", pueden ser: *Concepto Calculable a Evaluar, árbol de requerimientos*.

Nota

Esta definición coincide con la de **attribute** (*atributo*) definido en el estándar ISO/IEC 14598-1:1999. A los fines de esta tesis, Feature y attribute se consideran sinónimos.

6.3.3.4.7 Base de Conocimiento (Knowledge Base)

Definición

Un cuerpo organizado de conocimiento relacionado.

Relaciones

- Una o más bases de conocimiento están contenidas en una Memoria Organizacional.
- Una base de conocimiento contiene uno o más items de conocimiento.

Ejemplos

Bases de conocimiento de procedimientos, normas y manuales, experiencias y lecciones aprendidas.

Nota

Esta definición coincide con la definida en el estándar australiano AS 5037.

6.3.3.4.8 Ítem de Conocimiento (Knowledge Item)

Definición

Una pieza atómica de conocimiento.

Relaciones

- Uno o más items de conocimiento están incluidos en una base de conocimiento.

Ejemplos

Ejemplos de items de conocimiento pueden ser: Una norma de la empresa, una fórmula para calcular un costo, la descripción de un procedimiento, una experiencia.

6.3.3.4.9 Memoria Organizacional (Organisational Memory)

Definición

La manera en que una organización guarda y lleva cuenta de lo que sabe.

Relaciones

- Una Memoria Organizacional está compuesta (tiene) una o varias bases de conocimiento.

Ejemplos

Memoria Organizacional Basada en Casos.

6.3.3.4.10 Problema (Problem)

Definición

Un estado de dificultad que necesita ser resuelto.

Relaciones

- Un problema es parte componente de un caso.
- Un problema se resuelve con una o más soluciones
- Un problema está caracterizado por una o más características de problema.

Ejemplos

Ejemplos de problemas son: "*Cómo diseñar una evaluación para un sitio Web*", "*Cómo diseñar casos de testing para un módulo de dos entradas"*.

Nota

Esta definición coincide con la definida para el término "*Problem***"** en la ontología WordNet [Fellbaum 1998].

6.3.3.4.11 Característica de Problema (Problem Feature)

Definición

Rasgo que caracteriza a un problema.

Relaciones

- Una o más características de problemas caracterizan a un problema

Ejemplos

Ejemplos de características de un problema de diseño de evaluación pueden ser: *Propósito de Evaluación, Entidad a Evaluar, Nivel de precisión requerido*.

6.3.3.4.12 Resultado (Result)

Definición

Nivel de satisfacción logrado por la aplicación de una solución a un problema.

Relaciones

- Ninguno o un resultado es parte componente de un caso.

6.3.3.4.13 Modelo de Evaluación de Similitud (Similarity Assessment Model)

Definición

Algoritmo o función con elementos de similitud asociados que modela la valoración de similitud de casos.

Relaciones

- Un modelo de evaluación de similitud describe una base de conocimiento basada en casos.
- Un modelo de evaluación de similitud combina uno o más elementos de modelo de similitud.

Ejemplos

El modelo de similitud que describe la base de conocimientos basada en casos para diseño de proyectos de medición y evaluación, puede ser el que se describe en la tabla 6.1.

6.3.3.4.14 Criterio de Similitud (Similarity Criterion)

Definición

El patrón de valoración usado para determinar la similitud semántica entre dos valores del rasgo.

Relaciones

- Ninguno o varios criterios de similitud computan uno o varios elementos de modelo de similitud.

Ejemplos

Ejemplos: Similitud (Evaluar, Estimar)= 0.8; Similitud (Estimar, Mejorar)=0.4.

6.3.3.4.15 Elemento de Modelo de Similitud (Similarity Model Element)

Definición

Algoritmo o función con criterios de similitud asociados que modela la valoración de similitud de una característica.

Relaciones

- Uno o varios elementos de modelo de similitud son combinados en un modelo de evaluación de similitud.
- Uno o varios elementos de modelo de similitud se refieren a un concepto de dominio de una ontología de dominio.
- Uno o varios elementos de modelo de similitud están compuestos por ninguno o varios criterios de similitud.

Ejemplos

Ejemplos de elementos del modelo de similitud pueden ser: (*Entidad a evaluar*, tipo de similitud: *Exacta*, peso: *0.35*); (*Precisión*, tipo de similitud: *Diferencia*, peso: *0.2*).

6.3.3.4.16 Tipo de Similitud (Similarity Type)

Definición

Especifica el tipo de función de similitud que puede ser exacta, diferencia o compleja.

Relaciones

- Un tipo de similitud especifica la similitud de un elemento del modelo de similitud

Ejemplos

Son instancias de tipo de similitud: *Exacta, diferencia y compleja*.

6.3.3.4.17 Solución (Solution)

Definición

Una declaración que resuelve un problema o explica cómo resolver un problema

Relaciones

- Una solución es parte componente de un caso.
- Una o más soluciones resuelve un problema
- Una solución está caracterizada por una o más características de solución.

Ejemplos

Ejemplos de soluciones son: "*El árbol de requerimientos de diseño de una evaluación* ", "*El diseño de casos de testing para un módulo de dos entradas"*.

Nota

Esta definición coincide con la definida para el término "*Solution***"** en la ontología WordNet [Fellbaum 1998].

6.3.3.4.18 Característica de Solución (Solution Feature)

Definición

Rasgo que caracteriza a una solución.

Relaciones

- Una o más características de soluciones caracterizan a una solución

Ejemplos

Ejemplo de característica de una solución de diseño de evaluación puede ser el árbol de requerimientos obtenido, por ejemplo el siguiente:

1. Development Efficiency
1.1. Development Productivity
Associated attributes
1.2. Development Completeness
Associated attributes

Nota

Tanto las características de problemas como las características de solución, pueden ser de tipo simple (como por ejemplo enteros o caracteres), o de tipo estructurado (como el árbol de requerimientos mostrado en el ejemplo).

6.3.3.5 Representación de la Ontología

Luego de haber definido los conceptos y relaciones de la ontología, se identificaron los atributos de cada concepto y se construyó una tabla de atributos, incluyéndolos a la vez al diagrama de clases UML (ver figura 6.2).

De esta manera y como resultado de la etapa de conceptualización, se obtuvo una primera versión de la ontología de Memoria Organizacional Basada en Casos que fue semi-formalizada usando algunas representaciones intermedias, independientes del dominio y de su implementación, como son el Diccionario de Términos, la Tabla de

Relaciones y la ya mencionada Tabla de Atributos que mostramos a continuación en las tablas 6.5, 6.6 y 6.7 respectivamente.

Table 6.5. Diccionario de términos de la ontología de Memoria Organizacional Basada en Casos [Martin et al., 2009].

Concepto	**Descripción**
Case	Ítem de conocimiento contextualizado que representa una experiencia por medio de un problema y su solución.
Case Knowledge Base	Una base de conocimiento específica dónde cada ítem de conocimiento se representa a través de un caso.
Complex	Un tipo de función de similitud que guarda la medida de similitud para cada par de posibles valores de una característica.
Difference	Un tipo de función de similitud que es inversamente proporcional a la diferencia entre valores de características.
Exact	Un tipo de función de similitud que devuelve 1 si dos valores de características son iguales, y 0 en otro caso.
Feature	Una propiedad mensurable, física o abstracta, de una entidad. (sinómimo del término *Attribute* definido en ISO 14598-1).
Knowledge Base	Un cuerpo organizado de conocimiento relacionado. [AS 5037, 2005].
Knowledge Item	Una pieza atómica de conocimiento. Comentario: Conocimiento es una pieza de entendimiento y/o lecciones aprendidas de las habilidades y experiencias construidas por las personas.
Organisational Memory	La manera en que una organización guarda y lleva cuenta de lo que sabe.

Concepto	Descripción
Problem	Un estado de dificultad que necesita ser resuelto (WordNet).
Problem Feature	Rasgo que caracteriza a un problema.
Result	Nivel de satisfacción logrado por la aplicación de una solución a un problema.
Similarity Assessment Model	Algoritmo o función con elementos de similitud asociados que modela la valoración de similitud de casos.
Similarity Criterion	El patrón de valoración usado para determinar la similitud semántica entre dos valores del rasgo.
Similarity Model Element	Algoritmo o función con criterios de similitud asociados que modela la valoración de similitud de una característica.
Similarity Type	Especifica el tipo de función de similitud que puede ser exacta, diferencia o compleja.
Solution	Una declaración que resuelve un problema o explica cómo resolver un problema. (WordNet) [Fellbaum, 1998].
Solution Feature	Rasgo que caracteriza a una solución

Tabla 6.6 Ontología de Memoria Organizacional Basada en Casos: Descripción de Atributos.

Concepto	Atributo	Descripción
Case	name	El nombre de un caso para ser identificado.
	description	Una descripción no ambigua del significado del caso.

Concepto	**Atributo**	**Descripción**
	author	El nombre de la persona, grupo o agente responsable de crear un caso.
	timestamp	Instante del tiempo en que se crea el caso.
Feature	name	El nombre de una característica para ser identificada.
	value	Dato numérico o categórico asignado a la característica.
Knowledge Base	name	El nombre de una base de conocimiento para ser identificada.
	description	Una descripción no ambigua del contenido de la base de conocimiento.
Organisational Memory	name	El nombre de una Memoria Organizacional para ser identificada.
	corporationName	Nombre de la corporación que es propietaria de la Memoria Organizacional.
	description	Una descripción no ambigua del contenido de la Memoria Organizacional.
Problem	name	El nombre de un problema para ser identificado.
	description	Una descripción no ambigua del problema.
Problem Feature	priority	Valor numérico que representa la prioridad de una característica de problema.
Similarity Assessment Model	name	El nombre de un modelo de evaluación de similitud para ser identificado..

Concepto	**Atributo**	**Descripción**
	description	Una descripción no ambigua del significado del modelo de evaluación de similitud.
Similarity Criterion	firstValue	Primer dato valor numérico o categórico para el cual la similitud semántica se compara.
	secondValue	Segundo dato valor numérico o categórico para el cual la similitud semántica se compara.
	similarity	Dato numérico (entre 0 y 1) que representa la similitud semántica entre dos valores de características.
Similarity Model Element	name	El nombre de un elemento de modelo de similitud para ser identificado.
	description	Una descripción no ambigua del significado del elemento del modelo de similitud .
	weight	Valor numérico que representa la importancia relativa de un elemento de característica en un modelo de evaluación de similitud dado.
	upperThreshold	Constante numérica que especifica el umbral superior para un elemento del modelo de similitud dado.
	lowerThreshold	Constante numérica que especifica el umbral inferior para un elemento del modelo de similitud dado.

Concepto	**Atributo**	**Descripción**
Solution	name	El nombre de una solución para ser identificada.
	description	Una descripción no ambigua de la solución.

Tabla 6.7. Ontología de Memoria Organizacional Basada en Casos: Descripción de Relaciones.

Name	**Description**
associated_with	Uno o más casos están asociados con un contexto.
characterised_by	Un problema o una solución se caracterizan por uno o más rasgos (características).
combines	Un modelo de evaluación de similitud combina (asocia) uno o más elementos del modelo de similitud.
compound_by	Uno o más elementos del modelo de similitud se componen de cero o más criterios de similitud.
defined_by	Una o más características están definidas por un concepto de dominio en una ontología del dominio.
describes	Un modelo de evaluación de similitud se define para especificar a la estructura del caso y su modelo de similitud. Por lo tanto, un modelo de evaluación de similitud describe una base de conocimiento basada en casos concreta.
has	Una Memoria Organizacional tiene una o más bases de conocimiento.
has_result	Una solución tiene cero o un resultado

includes	Una base de conocimientos incluye uno o más ítems de conocimiento.
is_of_type	Un elemento del modelo de similitud es de un tipo específico (Exacto, Diferencia o Complejo).
is_solved_by	Un problema es resuelto por una o más soluciones.
refers_to	Uno o más elementos del modelo de similitud se refieren a uno o más conceptos del dominio.
specified_by	Un caso está especificado por un problema, una solución, y un resultado determinados.

6.3.4 Implementación

6.3.4.1 Tecnología Empleada

En la actualidad se cuenta con una gran variedad de aplicaciones y lenguajes para el desarrollo e implementación de ontologías [Corcho et al., 2003] entre las cuales se cuenta con herramientas que permiten la construcción de ontologías de cero, no sólo con funciones de edición sino también, entre otras, de documentación y exportación/ importación a diferentes formatos y lenguajes. Ejemplos de tales herramientas son Protégé [Noy, 2000], WebODE [Corcho et al., 2002], y OntoEdit [Sure et al., 2000], que tienen la ventaja de ser independientes del lenguaje y se caracterizan por su fácil extensibilidad e integración con otras aplicaciones.

Respecto a los lenguajes para la implementación de ontologías se pueden encontrar lenguajes fundados en paradigmas de representación del conocimiento basados en lógica de primer orden (por ejemplo KIF), basados en frames combinados con lógica de primer orden (por ejemplo Cycl, Ontolingua, OCML, Flogic) o basados en lógica descriptiva (por ejemplo LOOM). Estos lenguajes se utilizan para la

representación formal de los componentes de modelado basado en paradigmas tradicionales de representación del conocimiento.

Con el auge de Internet y la Web Semántica [Berners-Lee et al., 2001], se impulsó la creación de lenguajes para la implementación de ontologías basados en la Web o lenguajes de marcado, que además de permitir la representación formal del conocimiento, facilitan su reuso, diseminación e interoperabilidad, al estar basados en estándares para la Web, como son HTML y XML.

En particular, el lenguaje RDF (Resource Description Framework) es una recomendación del W3C (WWW Consortium [W3C]) para la estandarización de la definición y uso de metadatos. Utiliza la sintaxis de XML y soluciona las carencias de XML agregando semántica a la estructura de los datos.

RDF proporciona un modelo para describir aserciones sobre recursos Web equivalente a los formalismos de las redes semánticas que consiste en tres tipos de componentes: recursos, propiedades y sentencias. Los recursos también se denominan objetos y las propiedades son las relaciones entre objetos. Una sentencia RDF puede ser también un recurso en sí mismo, pero esta recursividad, no es interpretada por RDF, pues no provee primitivas de modelado para definir las relaciones entre recursos y propiedades a nivel de esquema. Como consecuencia de esto surgió la extensión RDF Schema (RDFS), que es una extensión semántica de RDF basada en XML Schema.

En la siguiente sección se describirá con más detalle los aspectos básicos de RDF y RDFS mostrando sus principales características a través de ejemplos de aplicación al dominio métricas y mediciones.

Teniendo en cuenta que uno de los objetivos de la ontología es constituir la base conceptual consensuada para la construcción de una Memoria Organizacional Basada en Casos con capacidades para el procesamiento semántico y automático del conocimiento presente en experiencias y lecciones aprendidas; y además, dado que es importante que el conocimiento capturado en la ontología pueda ser reusado con otros fines, o en otros sistemas, se eligió el lenguaje RDFS, para la implementación de la ontología, ya que es uno de los lenguajes estándar recomendados por el W3C para tal fin. Y para el desarrollo y edición de la ontología se usó la herramienta Protégé, por ser una

herramienta de amplia utilización, simple de usar y adecuada para el desarrollo de la ontología.

6.3.4.2 El Modelo Básico RDF/RDFS

6.3.4.2.1 El Modelo RDF

El modelo RDF se compone de tres conceptos principales:

Recursos: Todas las cosas descriptas por expresiones RDF se denominan recursos. Los recursos, representan cualquier entidad (lugares, personas, objetos) del mundo real y están identificados por un URI (Universal Resource Identificator).

Propiedades: Una propiedad es un aspecto específico, característica, atributo, o relación utilizado para describir un recurso. Cada propiedad tiene un significado específico, define sus valores permitidos, y sus relaciones con otras propiedades. Las características de las propiedades se definen con RDFS (RDF Schema).

Sentencias (declaraciones, enunciados): Un recurso específico junto con una propiedad denominada, más el valor de dicha propiedad para ese recurso es una sentencia RDF. Estas tres partes individuales de una sentencia se denominan, respectivamente, sujeto, predicado y objeto. Una propiedad es a su vez un recurso. El objeto es el valor asignado a dicha propiedad, y puede ser una cadena simple de caracteres (string), u otro recurso, es decir, un recurso especificado por un URI.

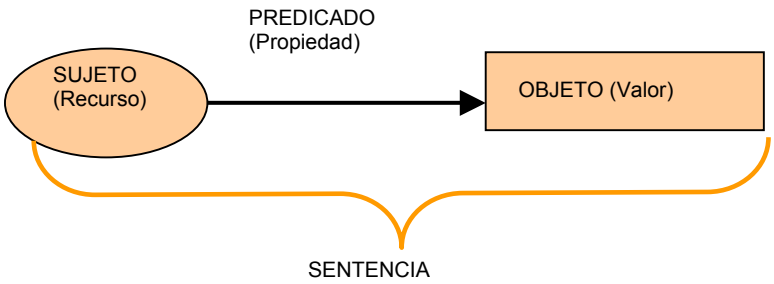


Figura 6.5 Representación gráfica de las sentencias RDF.

Las sentencias constituyen la construcción básica que establece el modelo de datos RDF. Es decir, el significado de los datos se expresa mediante un conjunto de

sentencias RDF que formalmente son representadas por tri-uplas (sujeto, predicado, objeto). Otra forma de representar las declaraciones (sentencias) RDF es a través de un grafo rotulado como se muestra en la Fig. 6.5.

La figura 6.6 ilustra la construcción de dos sentencias que describen un **recurso**, en este caso el recurso es una métrica que está definida en el URI http://gidis.edu.ar/DatosMetricas#ErroresPorMódulo. Las dos sentencias del ejemplo son:

"El 20-04-2003 fue catalogada 'http://gidis.edu.ar/DatosMetricas#ErroresPorMódulo'"
"El atributo *Densidad De Errores* de URI http://gidis.edu.ar/DatosMetricas# DensidadDeErrores es cuantificado por la métrica http://gidis.edu.ar/DatosMetricas# ErroresPorMódulo".

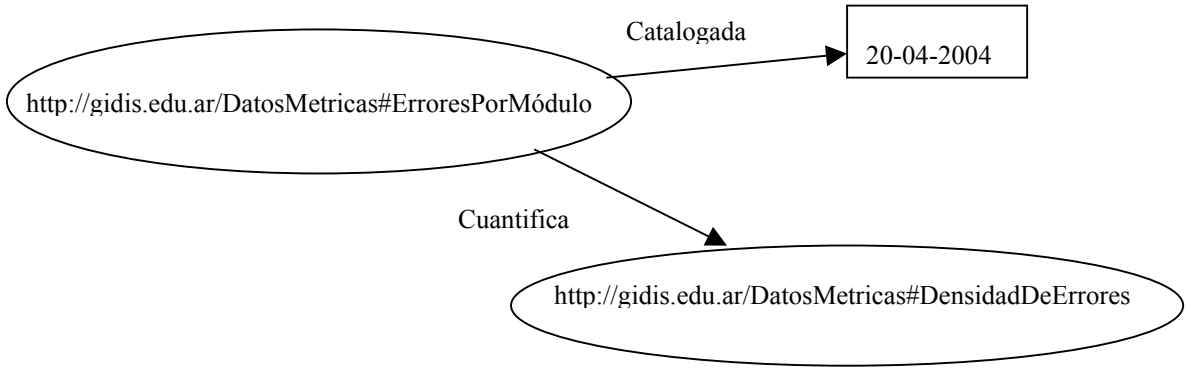


Figura 6.6 Representación gráfica las sentencias RDF.

En estos gráficos (también denominados "diagramas de nodos y arcos"), los nodos (dibujados como óvalos) representan recursos y los arcos representan propiedades etiquetadas. Los nodos que representan cadenas de literales pueden dibujarse como rectángulos.

La tabla 6.8 muestra las partes correspondientes a las sentencias mostradas en la Fig. 6.6. Las sentencias representadas formalmente por triuplas (predicado, sujeto, objeto), le confieren al modelo de datos de RDF la cualidad de ser entendido tanto por seres humanos, como por máquinas, ya que tienen acceso a la representación formal de conjunto de triuplas de este modelo.

Tabla 6.8 Representación mediante triuplas de las sentencias RDF.

Predicado (Propiedad)	Sujeto (Recurso)	Objeto (Valor)
Catalogada	http://gidis.edu.ar/DatosMetricas#ErroresPorMódulo	20-04-2004
Cuantifica	http://gidis.edu.ar/DatosMetricas#ErroresPorMódulo	http://gidis.edu.ar/DatosMetricas#DensidadDeErrores

6.3.4.2.2 XML como Lenguaje de Especificación de la Sintaxis RDF

Uno de los principales aspectos que han contribuido al éxito de la tecnología RDF en el contexto de la Web, es la posibilidad de poder representar e intercambiar datos (sentencias) RDF usando XML [Bray et al. 1998]. Además toda la sintaxis RDF está definida en XMLS en el documento http://www.w3.org/1999/02/22-rdf-syntax-ns.

Sin embargo, es importante remarcar que el modelo de datos RDF no es serializado en un árbol sintáctico XML, donde se debe respetar la jeraquía especificada por el esquema XML relacionado. El modelo de datos RDF (a diferencia del modelo de árbol XML) es un grafo dirigido, lo que nos permite una mayor potencia para representar sentencias (podemos aseverar cualquier cosa sobre cualquier objeto), y esto es totalmente independiente de su representación serializada XML.

La figura 6.7 muestra cómo las sentencias de la tabla 6.8 pueden ser serializadas usando XML. La segunda línea del código indica el trozo que usa sintaxis RDF, e identifica con los prefijos “rdf” y “m” la ubicación de los documentos que contienen las definiciones de los elementos usados, es decir los espacios de nombres. Las demás líneas representan una descripción RDF del recurso identificado por el URI http://gidis.edu.ar/DatosMetricas#ErroresPorMódulo, que en este caso representa una métrica, la marca <m:catalogada> indica que la métrica que se está describiendo tiene una propiedad llamada “catalogada” cuyo valor es “20/04/2004” y cuya semántica está definida en el vocabulario asociado al prefijo m:. Además, la marca <m:cuantifica> indica que la métrica también tiene una propiedad llamada “cuantifica” cuyo valor es el

objeto (que en este caso es un recurso, no un literal) identificado por el URI "http://gidis.edu.ar/DatosMetricas#DdensidadDeErrores".

```
<?xml version="1.0" >
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:m="http://gidis.edu.ar/Esquema#">
  <rdf:Descritpion about="http://gidis.edu.ar/DatosMetricas#ErroresPorMódulo">
    <m:catalogada>20/04/2004</m:catalogada>
    <m:cuantifica
rdf:resource="http://gidis.edu.ar/DatosMetricas#DensidadDeErrores"/>
  </rdf:Descritpion>
</rdf:RDF>
```

Figura 6.7 Serialización en XML de sentencias RDF.

Cuando escribimos una frase en lenguaje natural utilizamos palabras que, en lo posible, encierran la intención de transmitir un significado inequívoco. En el caso de descripciones RDF se debe establecer el significado preciso de cada propiedad para que el procesamiento automático de las sentencias dé el resultado esperado. Es muy importante que tanto el escritor como el lector de una sentencia (declaración) entiendan el mismo significado para los términos utilizados, tales como catalogada, cuantifica, etc.

En RDF el significado se expresa a través de un esquema. Podemos pensar en un esquema como en una especie de diccionario. Un esquema define los términos que se utilizarán en una declaración RDF y le otorgará significados específicos. Para evitar confusiones entre definiciones independientes (y posiblemente conflictivas) del mismo término, RDF utiliza la facilidad de los namespace de XML. Los namespaces ("espacios de nombre") son simplemente una forma de asociar el uso específico de una palabra en el contexto del diccionario (esquema) en que se puede encontrar la definición. En RDF, cada predicado utilizado en una sentencia debe ser identificado con un sólo namespace, o esquema. Sin embargo una descripción de recurso puede contener sentencias con predicados de varios esquemas.

6.3.4.2.3 Definición de Tipos

El modelo RDF prevé algunas primitivas importantes para una mejor descripción de los recursos, de ellas podemos destacar la primitiva rdf:type, para definir

tipos (instancias) de recursos, y las primitivas para definir contenedores, que explicaremos en la siguiente sección.

De hecho rdf:type indica una relación binaria entre dos elementos, estableciendo un mecanismo de instanciación, o sea especifica que un elemento es una instancia de otro. De esta manera se incluyen en una misma descripción datos y metadatos.

6.3.4.2.4 RDF Shema

RDF no ofrece ningún mecanismo para especificar el vocabulario usado en las descripciones (sentencias) tal como el término "documentada" usado en el ejemplo anterior. RDF Schema (RDFS) [Brickley et al., 2000], es el lenguaje que completa el modelo RDF básico en la función de alcanzar interoperabilidad semántica en el contexto de la Web. Esta propuesta del W3C provee un sistema de tipos básicos conjuntamente con un mecanismo para definir nuevos tipos, conformando distintos espacios de nombres, permitiendo de esta manera, que comunidades de usuarios de distintos dominios puedan compartir sus propios vocabularios. RDFS es una extensión de RDF que permite: i) describir los conceptos usados en aplicaciones RDF y ii) especificar restricciones de tipos para los sujetos y objetos de las triuplas.

RDFS se puede ver como una forma de modelado orientado a objetos (OO) en la Web. Con los atributos predefinidos rdfs:Class y rdfs:subClass es posible definir una jerarquía de clases, y las instancias pueden referirse a ellas a través del atributo rdf:type de RDF.

El mecanismo para la definición de tipos en un esquema RDF es ligeramente diferente a la definición de tipos de las metodologías tradicionales de modelado OO. En ésta última, la principal preocupación es la identificación de las entidades que serán representadas como clases, subclases y sus correspondientes atributos; en RDFS, en cambio, las propiedades (atributos) son definidas en términos de las clases de recursos a las cuales ellas se aplican. Este enfoque centrado en las propiedades facilita la descripción semántica de los recursos existentes en la Web (enumerando sus propiedades y valores asociados), principal objetivo de la arquitectura RDF.

A continuación se describen las principales primitivas de modelado RDFS que están agrupadas en clases, propiedades y restricciones para facilitar su explicación. La figura 6.8 (extraída del reporte técnico del W3C [Brickley et al., 2000]), muestra la relación existente entre estas primitivas y las primitivas del modelo RDF básico, sobre el cual está construido RDFS.

En dicha figura, los rectángulos con puntas redondeadas indican clases y las flechas señalan cuál es la clase que define al recurso. La relación subclase se indica con un rectángulo (la superclase) que contiene a otro rectángulo (la subclase). Por último, se muestra que todo objeto en RDF es un recurso (Resource).

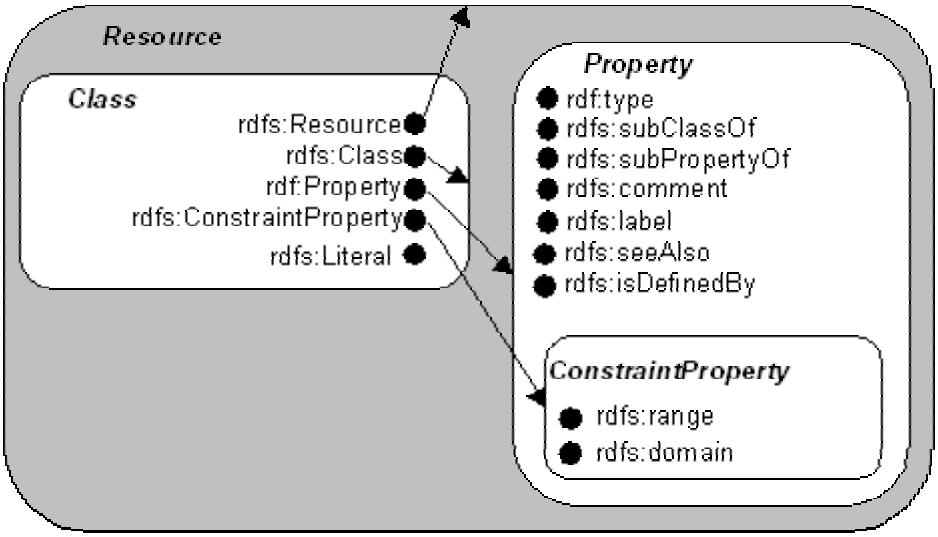


Figura 6.8 Jerarquía de clases del modelo RDFS (Extraída de [Brickley et al., 2000])

6.3.4.2.5 Clases

RDFS permite la definición de subclases que heredan las definiciones de una o más clases ascendentes, permitiendo la implementación de herencia múltiple. Esta característica incorpora gran flexibilidad al modelo RDF, porque las clases que se están definiendo pueden heredar de clases ya definidas en esquemas previos, especializando los metadatos, promoviendo de esta manera compartir y reusar estos esquemas. Para este propósito, las principales primitivas que ofrece RDFS son:

rdfs:Resource - representa la clase genérica en el modelo RDF Schema. Todo objeto descrito por expresiones RDF es un recurso.

rdfs:Class – es subclase de rdfs:Resource y representa el concepto genérico de tipo o categoría, similar a la noción de clase en orientación a objetos.

rdf:Property – es subclase de rdfs:Resource y representa un aspecto del recurso que se está describiendo, similar a la noción de atributo en orientación a objetos.

6.3.4.2.6 Propiedades

Las propiedades posibilitan expresar relaciones entre instancias y sus clases o clases y sus superclases. Relaciones entre propiedades también son permitidas, obteniéndose así una jerarquía de propiedades. Las principales propiedades disponibles en RDFS son:

rdf:type – es subclase de rdf:Prorperty y denota que un recurso es instancia de una clase, poseyendo todas sus características. Un recurso puede ser instancia de más de una clase.

rdfs:subClassOf - es subclase de rdf:Property y denota la relación de subclase/superclase entre dos clases. Esta propiedad es la base para la especificación de la herencia múltiple, y tiene característica de transitividad.

rdfs:subPropertyOf - es subclase de rdf:Property y denota la relación de especialización entre dos propiedades, posibilitando la definición de una jerarquía de propiedades.

6.3.4.2.7 Restricciones

Este mecanismo permite asociar restricciones a las propiedades de un recurso. Las principales primitivas de restricciones son:

rdfs:domain – es una instancia de la clase rdfs:ConstraintProperty y especifica a qué clase se aplica una propiedad.

rdfs:range – es una instancia de la clase rdfs:ConstraintProperty y restringe los valores (u objetos) que una propiedad puede asumir.

Para ilustrar mejor todos estos conceptos, en la siguiente figura (Fig. 6.9) se muestra un ejemplo de aplicación de esquemas RDF al dominio de métricas.

La parte superior de la figura muestra la definición del esquema RDF. La parte inferior muestra la descripción de algunas instancias de métricas en sentencias RDF. El tipo de gráfico que se usó está basado en la propuesta de Staab y colaboradores [Staab et al., 2000] que sugiere un diseño preliminar para modelar ontologías en RDF(S).

En la parte superior de la figura se ilustra cómo los conceptos y relaciones de un dominio se pueden modelar con una jerarquía de clases RDFS, y propiedades respectivamente. Los rectángulos representan clases en el dominio de métricas que se traducen en clases en RDFS, usando la primitiva rdfs:Class excepto la clase Literal, que es una clase predefinida del lenguaje RDFS.

Las flechas con línea simple representan relaciones en el modelo conceptual, que se traducen en propiedades RDFS, usando la primitiva **rdf:Property**. Por último, las flechas de línea doble representan la relación de herencia, proporcionada por la propiedad predefinida **rdfs:subClassOf.**

En las definiciones RDFS las primitivas **Class, subClassOf**, etc., le dan a las sentencias RDF una semántica cuya interpretación es común al modelado orientado a objetos. Esta semántica nos dice cómo los conceptos se relacionan entre sí jerárquicamente y qué restricciones presentan sus atributos.

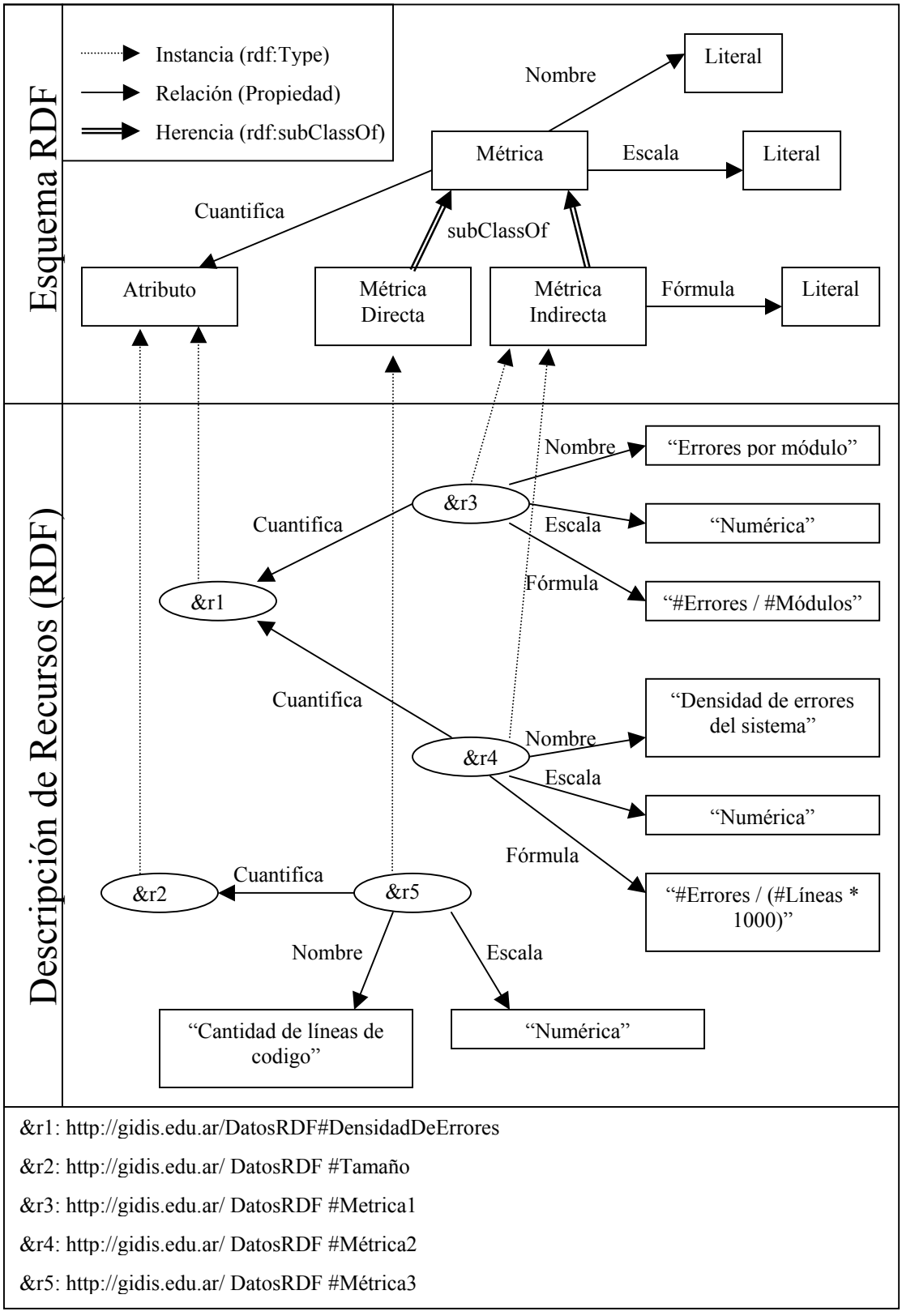


Figura 6.9 Un ejemplo de modelo RDFS para el dominio de Métricas y algunas instancias RDF.

6.3.4.3 Codificación

Como Chang expone en una nota publicada por el W3C [Chang, 1998], el modelo RDFS es equivalente a un subconjunto del modelo de clases en UML. RDFS usa un DLG (Directed Labeled Graph, Grafo dirigido etiquetado) para describir los esquemas. Los esquemas de clases expresados en UML también pueden verse como DLGs. Si se usa un DLG para un esquema del modelo de clases de UML, puede demostrarse que el DLG de RDFS es isomórfico a un subgrafo del DLG del esquema de clases de UML [Chang, 1998].

```
<rdfs:Class rdf:ID="Case">
   <rdfs:label xml:lang="en">Case</rdfs:label>
   <rdfs:comment> Ítem de conocimiento contextualizado que representa una experiencia por medio de un problema y su solución.</rdfs:comment>
   <rdfs:subClassOf rdf:resource="#KnowledgeItem"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Problem">
   <rdfs:label xml:lang="en">Problem</rdfs:label>
   <rdfs:comment> Un estado de dificultad que necesita ser resuelto (WordNet)
  </rdfs:comment>
   <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Solution">
   <rdfs:label xml:lang="en">Solution</rdfs:label>
   <rdfs:comment> > Una declaración que resuelve un problema o explica cómo resolver un
  problema. (WordNet)</rdfs:comment>
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdfs:Class>

<rdf:Property rdf:ID="is_solved_by">
  <rdfs:label xml:lang="en">is_solved_by </rdfs:label>
  <rdfs:domain rdf:resource="#Problem"/>
  <rdfs:range rdf:resource="#Solution"/>
</rdf:Property>
```

Figura 6.10 Trozo de código RDFS que implementa parte la ontología de Memoria Organizacional Basada en Casos (algunos conceptos y relaciones)

Las estructuras y elementos en el modelo de clases de UML y de RDFS se corresponden unos con otros. El diagrama de clases UML puede transformarse a una representación equivalente en un esquema RDFS y viceversa. La citada nota de Chang también describe la relación entre los elementos de los diagramas de clase para UML y RDFS, que fue tenida en cuenta para la implementación de la ontología.

En las figuras 6.10 y 6.11 se muestran, a modo de ilustración, trozos del código RDFS que implementan la ontología de MOBC. El código completo puede consultarse en el apéndice A.

Si observamos el modelo conceptual de la figura 6.2, para su implementación se define una clase en RDF para cada término (ver Fig. 6.10), y se define una propiedad para cada relación, con restricciones de rango y dominio sobre las clases origen y destino respectivamente(ver Fig. 6.11).

Un problema que necesitó ser analizado, fue la implementación de los atributos de clases, que pueden ser vistos como propiedades RDFS que tienen como rango la clase a la que pertenecen y como dominio un valor literal. Debido a que las propiedades RDFS son en sí mismas una clase, y por lo tanto no pueden definirse en relación a una clase, no se puede definir una propiedad dada, con un rango particular cuando se aplica a objetos de una clase, y otro rango cuando se aplica a objetos de otra clase distinta.

```
 <!-- Atributos de Case -->
<rdf:Property rdf:ID="caseName">
    <rdfs:label xml:lang="en">caseName</rdfs:label>
    <rdfs:domain rdf:resource="#Case"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>

<rdf:Property rdf:ID="caseDescription">
    <rdfs:label xml:lang="en">caseDescription</rdfs:label>
    <rdfs:domain rdf:resource="#Case"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>

<rdf:Property rdf:ID="caseAuthor">
    <rdfs:label xml:lang="en">caseAuthor</rdfs:label>
```

```
    <rdfs:domain rdf:resource="#Case"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>

<!-- Atributos de Problem -->

<rdf:Property rdf:ID="problemName">
    <rdfs:label xml:lang="en">problemName</rdfs:label>
    <rdfs:domain rdf:resource="#Problem"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>

<rdf:Property rdf:ID=" problemDescription ">
     <rdfs:label xml:lang="en"> problemDescription </rdfs:label>
     <rdfs:domain rdf:resource="#Problem"/>
     <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
```

Figura 6.11 Trozo de código RDFS que implementa parte de la ontología de MOBC (algunos atributos)

Se han discutido varias soluciones a este problema y la opción que se eligió en esta implementación es crear propiedades con nombres de la forma: claseAtributo, de esta manera, cada atributo es único para cada clase. En la figura 6.11 mostramos algunos ejemplos.

6.4 Conclusiones

En este capítulo hemos definido la ontología de MOBC, resultante de distintas fuentes de conocimiento citadas en la sección 6.2 y un arduo trabajo de consenso entre investigadores y profesionales del área de Ingeniería de Software de la Facultad de Ingeniería. Cabe destacar la importancia que tiene, contar con una ontología para el ámbito de Memoria Organizacional Basada en Casos, ya que provee una base conceptual consensuada, para compartir, reusar y organizar el conocimiento relacionado a experiencias y lecciones aprendidas en una organización, y en particular en una empresa dedicada al desarrollo de software.

También es importante señalar, que si bien existe un estándar de gran utilidad relacionado a Gestión del Conocimiento [AS5037, 2005], y numerosos trabajos de

investigación en el área de Gestión del Conocimiento, Memoria Organizacional, y CBR, donde se definen marcos conceptuales, y la terminología usada, estos no constituyen en sí mismos una ontología formal o semiformal.

Desde el punto de vista de la implementación, es importante destacar que las ontologías implementadas con lenguajes para la Web Semántica, representan una tecnología muy útil al propósito de proveer conocimiento "entendible" en forma automática por computadoras.

En el caso de nuestra ontología de MOBC, sirve además como base conceptual para compartir y organizar el conocimiento, de una forma extensible, interoperable y automáticamente procesable. La ontología propuesta, se usa como base para el diseño y construcción de un sistema de Memoria Organizacional Basada en Casos, que permitirá explorar, reusar y compartir conocimiento relacionado a distintos dominios, si se instancia dicho sistema asociándole la ontología de dominio correspondiente, como se ilustrará en el próximo capítulo. Específicamente, en esta tesis se usará dicha MOBC para su aplicación en un sistema de recomendación en el diseño de proyectos de medición y evaluación para el área de Ingeniería de Software.

Capítulo 7- Memoria Organizacional Basada en Casos y Ontologías

7.1 Introducción

La Memoria Organizacional de una empresa favorece la aplicación de prácticas de Gestión del Conocimiento dentro de las organizaciones ayudando a las personas a "recordar lo que ellos saben". Es más, algunos autores [Dieng et al. 1999] afirman que la construcción de una Memoria Organizacional debe ser considerada como el primer paso en la implementación de un KMS.

En este capítulo se mostrará el diseño un sistema de Memoria Organizacional Basada en Casos y basada en ontologías (SMOBC). El sistema propuesto es genérico, basado en componentes y desarrollado con tecnologías de Web Semántica, lo que lo hace consistentemente interoperable para ser usado en aplicaciones distribuidas en la Web. El SMOBC permite el almacenamiento persistente del conocimiento informal obtenido en experiencias y problemas resueltos para su posterior consulta y aplicación en recomendación, toma de decisiones y/o resolución de nuevos problemas.

La contribución de este sistema de Memoria Organizacional respecto a otros ya propuestos en la literatura, es que al estar basado en ontologías, puede ser fácilmente instanciado a través del aporte de una ontología de dominio que provea un lenguaje específico del ámbito de aplicación del conocimiento, donde los tipos de atributos que caracterizan a un problema y a una solución estén formalmente definidos. Otra característica que le da flexibilidad al diseño, es la representación del modelo de similitud (detallado en el capítulo 6), que debe ser especificado para cada base de casos, permitiendo adecuar el motor de CBR al tipo de conocimiento que almacena la base.

7.2 Características Deseables de una MOBC

A partir del análisis de la situación actual en Memoria Organizacional y Gestión del Conocimiento realizado en el capítulo 2, se han identificado un conjunto de requisitos que una Memoria Organizacional debería satisfacer para que sea de utilidad y sirva de apoyo a los procesos de Gestión del Conocimiento en una empresa. Los mismos se discuten en las secciones siguientes.

7.2.1 Interoperabilidad

Las distintas actividades de una organización pueden estar soportadas y/o automatizadas por herramientas de software heterogéneas (en cuanto a plataformas, lenguajes, estructuración de información, codificación de datos, etc). Sin embargo, es importante proveer mecanismos para que compartan y diseminen las experiencias y lecciones aprendidas generadas en cada una de ellas, de manera de tomar ventaja del conocimiento organizacional compartido.

Por lo tanto, un requerimiento deseable que debería cumplir una Memoria Organizacional es que esté desarrollada con tecnologías que faciliten el intercambio de información y conocimiento entre distintas aplicaciones de usuario o herramientas, aún cuando estén implementadas en plataformas heterogéneas. Para garantizar la interoperabilidad en el sistema de Memoria Organizacional, se propone su implementación bajo estándares de marcado (como XML/XMLS), estándares de Web Semántica (como RDF/RDFS), Web Services y arquitecturas de ambientes distribuidos de amplia difusión.

7.2.2 Estructuración del Conocimiento

Evaluando la importancia de contar con sistema de Memoria Organizacional que resguarde el conocimiento adquirido en experiencias previas para su posterior uso en distintos procesos y actividades de la organización, observamos la necesidad de que dicho sistema esté robustamente estructurado para la explotación automática de dicho conocimiento.

Es decir, para facilitar el procesamiento del conocimiento y su integración a los sistemas de información de la empresa es necesario guardar el conocimiento de manera altamente estructurada, como es la representación del conocimiento a través de casos. La estructuración del conocimiento informal en casos, facilita la captura en forma automática, la recuperación, transferencia y reuso.

Esta característica permite que el conocimiento pueda ser procesado no sólo por seres humanos sino también por agentes, herramientas automáticas u otro software de computadoras para potenciar su explotación. El Razonamiento Basado en Casos se ha aplicado exitosamente como una forma de promover y gestionar el conocimiento adquirido en experiencias y lecciones aprendidas almacenadas a lo largo del tiempo en una organización, es decir, una manera de proveer persistencia (memoria) al conocimiento. Además, los sistemas basados en RBC se pueden beneficiar de una representación formal y compartida de una conceptualización (ontología) del dominio de conocimiento, que permita estructurar y especificar los casos mediante características cuya semántica y terminología esté claramente definida en dicha ontología.

7.2.3 Generalidad

Otra característica deseable que debería tener el SMOBC es su generalidad, que proporcione un marco común de trabajo en la estructuración y codificación del conocimiento de manera que pueda ser aprovechado en distintos ámbitos y desde distintas perspectivas. Para este fin, se debe usar un modelo de diseño que no esté comprometido con un dominio de conocimiento específico.

Para este fin, se ha diseñado el SMOBC sobre la base de dos niveles distintos de ontologías (como se mostrará en la sección 7.3). Por un lado, en un nivel alto de abstracción se define la ontología genérica de Memoria Organizacional Basada en Casos que fue especificada en el capítulo anterior, y por otro lado, a un nivel más específico, para caracterizar los casos de acuerdo al conocimiento de un dominio determinado y su contexto, se deberán definir las ontologías de dominio y de contexto respectivamente. De esta manera se facilita la adaptación del sistema para ser aplicado en distintas áreas de conocimiento.

7.2.4 Extensibilidad

Teniendo en cuenta el dinamismo del crecimiento y cambios en el capital intelectual de las empresas modernas, donde los recursos de conocimiento colectivo continuamente evolucionan y se amplían, es importante contar con una Memoria Organizacional que facilite la gestión de nuevas bases de conocimiento con un mínimo costo de implementación. Para lograr este objetivo, el SMOBC deberá proveer mecanismos que facilite la creación y el mantenimiento de bases de conocimiento que pueden estar geográficamente dispersas (distribuidas) y relacionadas a distinto dominio de conocimiento. Para facilitar este objetivo, tanto la estructura como la semántica del conocimiento (los metadatos) deberán formar parte del sistema de Memoria Organizacional.

7.2.5 No Ambigüedad

Para que el conocimiento almacenado en la MOBC pueda ser claramente interpretado y, como consecuencia de ello, su aplicación en distintos sistemas de recomendación y toma de decisiones sea efectiva y comparable, se debe establecer una terminología común, consensuada y formalmente especificada; es decir, el SMOBC debe estar basado en Ontologías.

Entre los beneficios de basar el sistema en ontologías, se destaca la posibilidad de compartir el conocimiento entre sub-sistemas heterogéneos de la misma empresa, y tener una interpretación unificada de dicho conocimiento.

7.2.6 Gestión Distribuida

Existen varias razones que fundamentan la importancia de gestionar el conocimiento en forma distribuida. Por un lado, si la administración y el accionar de la organización no está centralizada, existen dificultades para el desarrollo de una única Memoria Organizacional que resguarde todo el conocimiento presente en la misma.. Por otro lado, existen ventajas en relación a la gestión distribuida que tienen que ver con la

posibilidad de manejar múltiples bases del conocimiento locales de forma efectiva, que luego podrán ser compartidas y difundidas dentro de la organización.

Muchas veces una parte de la organización repite el trabajo y los errores de otra parte, simplemente porque es difícil llevar el rastro y hacer uso del conocimiento generado en lugares geográficamente alejados y a veces en distintos contextos. Las organizaciones necesitan contar con sistemas que se adapten a entornos de trabajo distribuidos, para gestionar la totalidad de sus bienes de conocimiento corporativos de manera que estén disponibles en tiempo y forma donde se las necesiten.

Para que la gestión de este conocimiento sea efectiva, el sistema deberá estar basado en redes de nodos de Memoria Organizacional interconectadas que faciliten la transferencia de conocimiento entre las distintas dependencias de la organización de manera que éste pueda ser transmitido y compartido eficazmente.

7.3 Integración de Distintos Niveles de Ontologías en la Memoria Organizacional

Como ya fue analizado en el capítulo 3, existen varias definiciones y puntos de vista en la literatura, referentes al contenido y la organización de una Memoria Organizacional, pero en términos generales, se puede ver una MO como un sistema de resguardo de conocimiento que combina descripciones de las entidades del dominio (conocimiento formal) y conocimiento relacionado a experiencias o lecciones aprendidas que las involucran (conocimiento informal), que puede ser estructurado en casos [Martín et al., 2009].

Típicamente, las organizaciones acumulan conjuntos de experiencias que son ricas en conocimiento para ser explotadas. Por ejemplo, un grupo de aseguramiento de calidad en ingeniería de software, guarda una colección de diseños de proyectos de medición, e informes de resultados de mediciones y evaluaciones desarrolladas durante las últimos años. Por otro lado, estas experiencias relacionan conceptos cuya definición debe estar formalmente especificada para el área de aplicación y es conocimiento común al dominio y no específico de una experiencia particular (por ejemplo, definiciones de *Arbol de requerimiento, Métrica, Entidad, Atributo*, etc) . Por lo tanto,

para una organización más conveniente (y teniendo en cuenta las características deseables expuestas en la sección anterior) de la Memoria Organizacional, es importante separar estructuralmente el conocimiento de las experiencias o lecciones aprendidas (casos), de la definición formal de la terminología relacionada al dominio de aplicación (ontología de dominio) y la definición formal de la terminología relacionada a las variables de contexto del conocimiento (ontología de contexto).

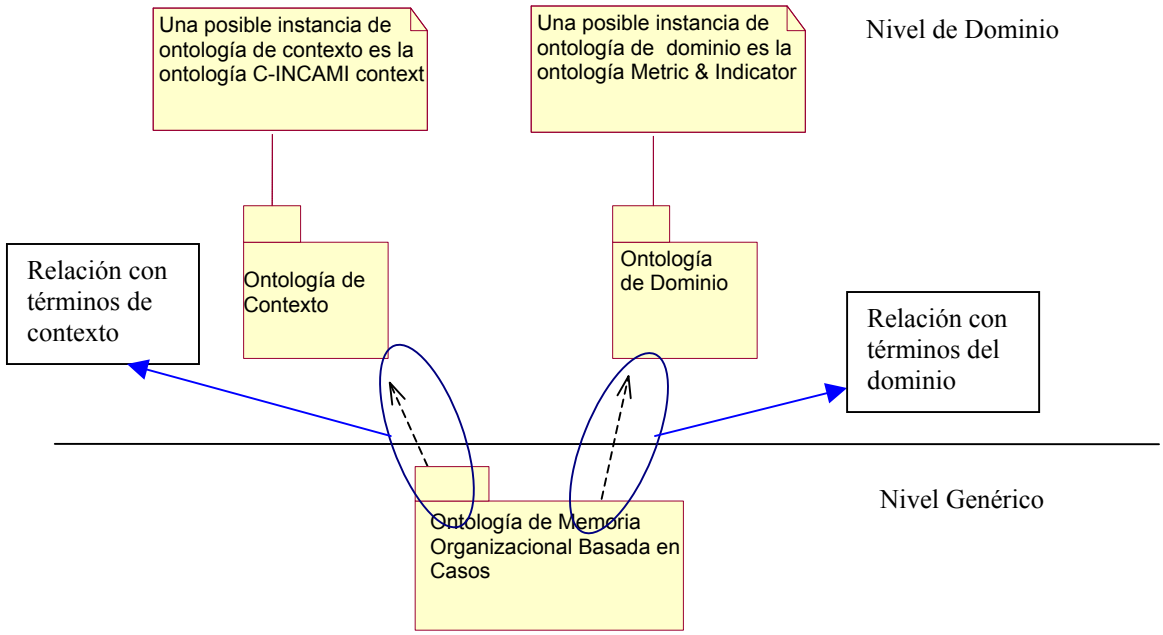


Figura 7.1. Relaciones con términos del dominio y con términos de contexto de la ontología de MOBC.

Para estructurar el conocimiento en casos, el núcleo de la Memoria Organizacional se basa en la ontología de MOBC descripta en el capítulo anterior. Dicha ontología es de nivel genérico, describe la estructura y terminología de una Memoria Organizacional Basada en Casos y constituye un modelo conceptual que se puede aplicar a la gestión de conocimiento en distintas áreas. A partir de ella, se pueden formular otras representaciones para ser aplicadas a dominios específicos completando la formalización con una ontología de dominio.

Es decir, la ontología de MOBC debe estar relacionada con otras ontologías que operan en distintos niveles de abstracción (en particular, dos niveles según se observa en la figura 7.1. Por un lado se define la ontología genérica de Memoria Organizacional Basada en Casos y por otro lado, para caracterizar los casos de acuerdo al conocimiento

de un dominio específico y su contexto [Molina et al., 2007], se deberán definir las ontologías de dominio y de contexto respectivamente. Por ejemplo, se puede usar la ontología de métricas e indicadores definida por Martín y colaboradores [Martín et al., 2003] para instanciar la Memoria Organizacional, de manera que pueda se aplicada a la gestión de conocimiento en el área de medición y evaluación de productos software.

Esta separación de las ontologías en niveles, genera dos relaciones entre ontologías: La *Relación con Términos del Dominio* y la *Relación con Términos del Contexto* como se ilustra en la figura 7.1. En las siguientes dos secciones se explica con mas detalle dichas relaciones.

7.3.1 Relación con Términos del Dominio

Como fue analizado en el capítulo 2, frecuentemente los sistemas de RBC utilizan conocimiento adicional sobre el dominio en forma fuertemente ligada al motor de RBC. Desde el punto de vista de flexibilidad y adaptación, es mejor considerarlos separadamente ya que se trata de conocimientos de distintos orígenes.

El SMOBC que se propone en esta tesis se basa en la separación entre los principales componentes de conocimiento de la Memoria Organizacional que provienen de distintas fuentes, a saber: el conocimiento adquirido por las experiencias (casos) y el conocimiento del dominio. En concreto, la terminología de RBC y Memoria Organizacional que se usa como elemento integrador proviene de la ontología de MOBC y el conocimiento del dominio proviene de la reutilización de una ontología producto de la conceptualización y formalización del dominio específico de su aplicación.

La ontología de MOBC y el sistema de Memoria Organizacional propuesto en conjunto, permiten que el esfuerzo de desarrollo de un nuevo sistema de Gestión del Conocimiento para un área específica, se centre solamente en adquirir el conocimiento especializado del dominio y en incorporar la estructura (metadatos) de la nueva base de conocimiento basada en casos.

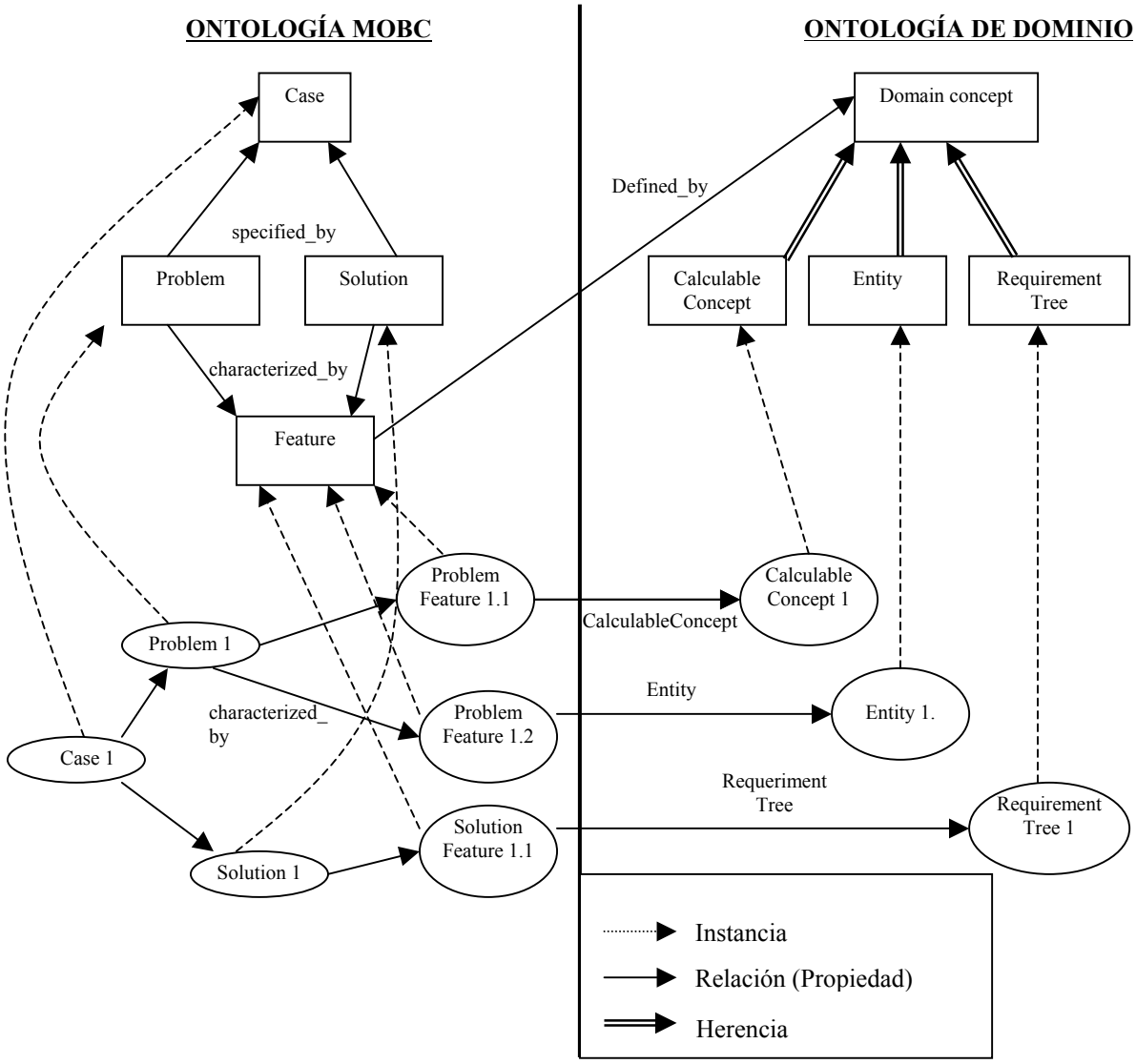


Figura 7.2. Relación entre la ontología de MOBC y los términos del dominio.

En el capítulo 4 hemos introducido distintas posibilidades para representar casos que están presentes en la literatura de CBR [Chen et al., 2003] y que van desde representaciones totalmente formales hasta descripciones informales como textos, gráficos, etc. Nuestra propuesta utiliza una representación formal y estructurada para guardar el conocimiento a través de casos. En la figura 7.2 se muestra un ejemplo de representación parcial de un caso relacionado a experiencias de proyectos de medición y evaluación de software. En la figura, los rectángulos representan conceptos de la ontología (clases), las elipses representan instancias y las flechas distintos tipos de relaciones, tal cual se indica en las leyendas de dicha figura.

Cada caso es un individuo, instancia del concepto *Case* de la ontología de MOBC (ver Fig. 7.2). El caso se compone de un problema y una solución que están descriptos por un conjunto de características (instancias de *Feature*), cada instancia de *Feature* está relacionada directamente a instancias de conceptos de la ontología de dominio. Las relaciones entre las instancias de *Feature* y las instancias de conceptos del dominio heredan las restricciones de la relación *defined_by* que asocia cada *Feature* un concepto del dominio (*Domain concept*). Una de las ventajas de utilizar un lenguaje de especificación de ontologías expresivo, como RDF/RDFS, es que permite definir jerarquías de relaciones. Por ejemplo, en la figura 7.2, las relaciones *CalculableConcept*, *Entity* y *RequerimentTree* son un subtipo específico de la relación *defined_by*.

Esta representación permite que los casos se compongan de otros individuos (instancias), que a su vez pueden ser simples (números, cadenas, etc) o estructurados (objetos complejos) y cuya estructura y semántica está formalmente definida en la ontología de dominio. De esta manera se provee un mecanismo de estructuración del conocimiento que además de ser potente en cuanto a las posibilidades de representación, permite un procesamiento automático, debido a su formalización.

Es importante destacar que la base de conocimiento basada en casos, se define tanto por extensión (ya que incluye todas las instancias de casos para un determinado dominio de aplicación), como por su estructura, ya que asociada a cada base de conocimientos se define un modelo de evaluación de similitud que describe la estructura, composición y función de evaluación de similitud de los casos componentes de la base. Es decir, cada base de conocimiento basada en casos, guarda también información de cómo se estructura el conocimiento (metadatos).

En el momento de crear una nueva base de conocimiento basada en casos para un dominio específico, el diseñador debe definir cuáles son los conceptos de la ontología de dominio que intervienen en la conformación de los casos, cual es el peso de relevancia de cada concepto y el tipo de función de similitud que se aplica para dicho concepto. Es decir, a cada base de conocimientos basada en casos (consistente en todas las experiencias o casos relacionados al mismo dominio) se le asocia un modelo de evaluación de similitud y una ontología de dominio.

7.3.2 Relación con Términos de Contexto

En la bibliografía más referenciada de esta área [Nonaka et al., 1995] [Wiig, 1993], el conocimiento es conceptualizado como una representación útil, y significativa de hechos en un *Contexto*. Por lo tanto, para poder gestionar adecuadamente el conocimiento en una empresa, las Memorias Organizacionales deben guardar también información contextual de cada ítem de conocimiento, ya que para poder reusar y explotar dicho conocimiento en forma eficiente, éste debe ser interpretado y analizado en su contexto.

Particularmente, en el SMOBC que se propone en esta tesis donde los items de conocimiento son representados a través de casos, la información de contexto debe estar asociada a cada caso. Según Kolodner, "Un caso es una pieza *contextualizada* de conocimiento que representa una experiencia. Contiene la lección pasada que es el contenido del caso y el contexto en el cual la lección puede ser utilizada" [Kolodner, 1993]. Teniendo en cuenta esta definición, y que una de las principales beneficios de la Gestión del Conocimiento es su reuso como una ventaja competitiva, surge la importancia de considerar el contexto a la hora de su aplicación en la resolución de nuevos problemas de una forma adecuada y consistente.

Según la definición de Dey: "*Contexto es cualquier información que puede ser usada para caracterizar la situación de una entidad, siendo una entidad una persona, lugar, u objeto que se considera relevante en la interacción entre un usuario y una aplicación, incluyendo también a ellos mismos, usuario y aplicación*" [Dey, 2001]. En términos generales, un sistema es consciente del contexto, si lo usa para proporcionar información y/o servicios, donde la relevancia de dicha información depende del contexto actual. De igual modo, en las memorias organizacionales basada en casos, la relevancia de la información de experiencias similares, está muy ligada al contexto en que es válido aplicar dicha experiencia. En este caso, la entidad relevante en la interacción entre el usuario y la aplicación es el caso. Debemos definir cuál es la información que puede ser usada para caracterizar la situación de un caso, es decir su contexto de aplicación.

Los parámetros usados para caracterizar el contexto (*Context property*) son términos definidos en la ontología de contexto, por lo tanto, una Memoria Organizacional, tiene asociada una ontología de contexto, que proporcionará los conceptos que permitirán analizar el contexto de cada caso que almacena.

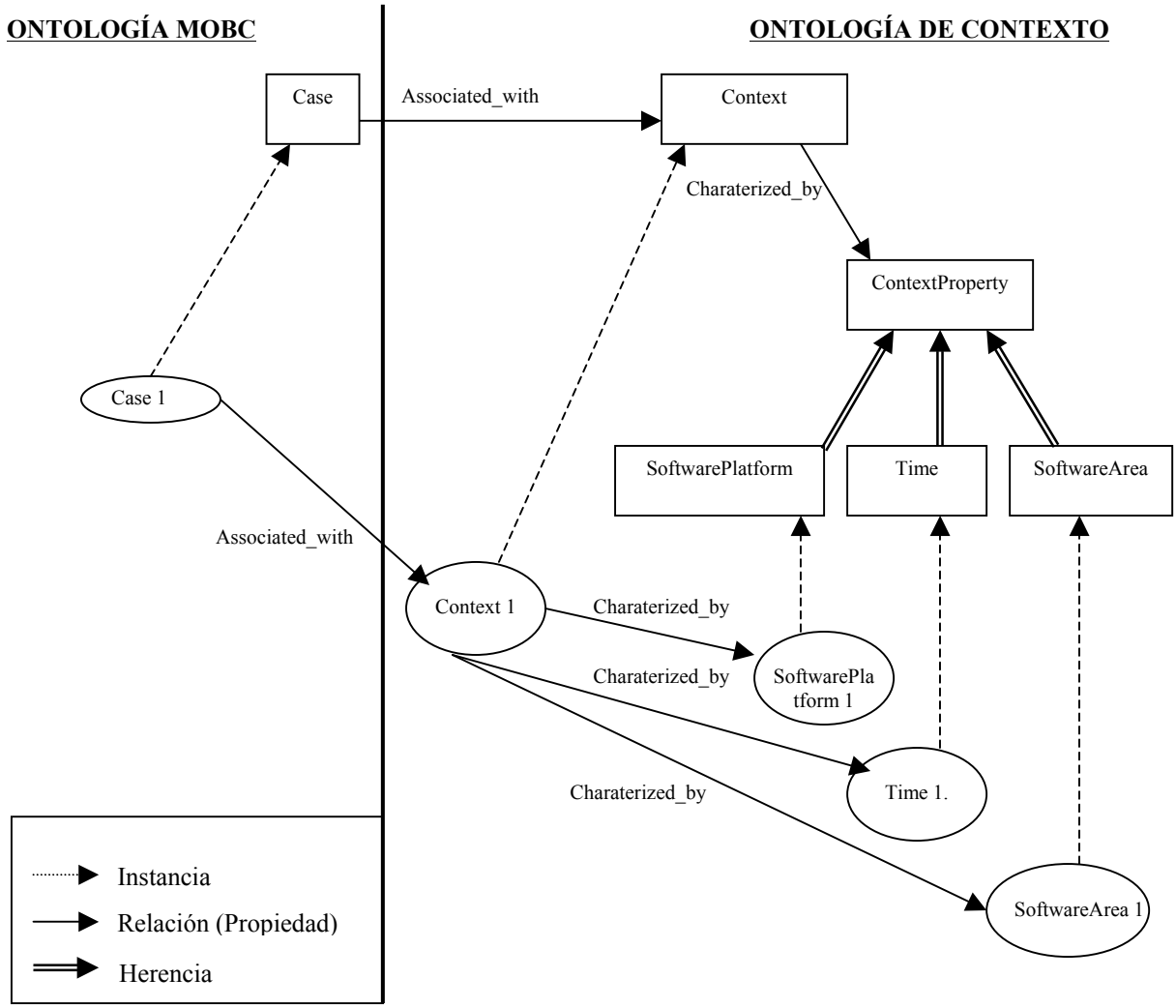


Figura 7.3. Relación entre la ontología de MOBC y los términos de contexto.

Con respecto a la representación de la información de contexto ocurre algo similar al conocimiento del dominio (ver figura 7.3). El contexto en el que se aplica cada caso se describe a través de un conjunto de propiedades de contexto que son instancias de conceptos de la ontología de contexto [Molina et al., 2007].

Este modelado de contexto, brinda importantes capacidades de adaptación de la Memoria Organizacional, a distintos dominios, donde también la terminología para describir el contexto es distinta en cada área de aplicación. En el momento de crear una nueva base de conocimiento basada en casos para un dominio específico, el diseñador además de proporcionar una ontología de dominio, debe definir cuáles son los conceptos de la ontología de contexto que intervienen en la descripción del mismo para su análisis y comparación en forma consistente.

7.4 Arquitectura del SMOBC

El sistema de Memoria Organizacional Basada en Casos, sigue el modelo arquitectónico distribuido en capas, constituyendo una estructura clara, modular y comprensible de la aplicación, además de ser fácilmente modificable y escalable. Es una arquitectura genérica, basada en la Web [Martín et al., 2009] y pensada para ser desarrollada con tecnologías de Web Semántica [Berners-Lee et al., 2001] [W3Csw, 2010], como son los lenguajes de etiquetado semántico (RDF/RDFS), ontologías y Web Services. Permite el almacenamiento persistente de conocimiento representado con casos y su contexto. Además los estructura en distintas bases de conocimiento basada en casos, y facilita la búsqueda y recuperación de casos previos más similares con su contexto.

El sistema propuesto está pensado para ser usado como núcleo de aplicaciones de Gestión del Conocimiento para distintos dominios y fue desarrollado teniendo en cuenta cuatro aspectos principales:

1. El conocimiento guardado en la Memoria Organizacional propuesta, se limita al conocimiento informal adquirido a través de experiencias, lecciones aprendidas, buenas prácticas, etc, que pueda ser estructurado en casos. El conocimiento explícito, como pueden ser los conceptos básicos, reglas del negocio o guías de procedimiento, normalmente se encuentran bien estructurados y formalizados en toda organización, es estable y por lo tanto pueden ser fácilmente gestionados con sistemas de información tradicionales. En cambio, el conocimiento que es de interés gestionar en

la Memoria Organizacional, es el innovativo, es el conocimiento en evolución, más dinámico y complejo por naturaleza.

2. El sistema de MOBC de la organización como un todo, está distribuida en distintos nodos de Memoria Organizacional. Cada nodo contiene conocimiento local, correspondiente a una dependencia de la organización, aunque puede estar relacionado a varios dominios de aplicación. El conocimiento de cada nodo puede ser difundido y compartido a través de una red de nodos de MOBC distribuidos interconectados.

3. La MOBC integra dos niveles de conocimiento a saber: las ontologías de dominio y de contexto, que permite guardar conocimiento del dominio por un lado y el conocimiento innovador de las experiencias adquiridas (casos) por otro.

4. La recuperación del conocimiento contenido en casos previos similares está basada en el acceso semántico a través de una meta-descripción o caracterización de los items de conocimiento almacenados. Dicha caracterización debe ser especificada por un experto y es distinta para cada base de conocimiento basada en casos.

Cada nodo de Memoria Organizacional está implementado como un servicio Web llamado "servicio MOBC" que implementa los servicios básicos de la Memoria Organizacional y es deseable que se encuentre replicado en distintos servidores; de manera de permitir que el conocimiento se halle distribuido geográficamente y pueda ser accedido y compartido a través de la red. El servicio *MOBC* es el encargado de administrar la base de casos, crear nuevas bases de casos, importar bases de casos de otros nodos o de archivos RDF/RDFS y resolver consultas de casos similares.

En la figura 7.4 mostramos una vista general de la arquitectura del servicio MOBC, y cómo se puede usar como componente principal de herramientas, agentes o aplicaciones de Gestión del Conocimiento. Para el diseño de la arquitectura del servicio MOBC, se ha elegido un estilo arquitectónico multi-nivel o de n-capas. Este estilo

arquitectónico proporciona al servicio una estructura clara y comprensible, siendo más fácilmente modificable y escalable. Las capas de la arquitectura son basicamente:

- **Capa de Lógica de Negocio:** Implementa la funcionalidad que el sistema debe proporcionar a los usuarios, agentes o herramientas. Esta funcionalidad ofrece métodos específicos y los traduce en invocaciones a distintos métodos de los DBMS apropiados. Dos ventajas importantes de contar con esta capa separada son, por un lado la posibilidad de implementar la Memoria Organizacional arriba de una gran variedad de repositorios en distintos DBMS, sin tener que cambiar ninguna componente; y por otro, que la funcionalidad provista por la arquitectura, puede ser accedida tanto por agentes y herramientas como por otras aplicaciones de usuario. Las funcionalidades de esta capa se agrupan en tres módulos a saber:

    - *Módulo de Recuperación de Casos*: Este módulo evalúa consultas sobre casos similares (invocando el motor de Razonamiento Basado en Casos). Recibe un nuevo caso pasado por parámetro desde el cliente y, teniendo en cuenta el tipo de base de conocimiento y modelo de similitud para ese caso, realiza la comparación del nuevo caso contra aquellos almacenados en el repositorio de casos. El servicio devolverá como solución aquella que más se asemeje al nuevo caso y el contexto donde debe ser aplicada.

    - *Módulo de Administración de Casos*: Este módulo implementa la administración total de las bases de conocimiento basada en casos, permitiendo la actualización tanto de datos como de metadatos, es decir, provee tanto funcionalidad para la creación y mantenimiento de nuevas bases de conocimiento basadas en casos, como la actualización de instancias de casos del repositorio.

    - *Módulo de Importación*: Este módulo permite la importación de casos tanto desde otras memorias organizacionales distribuidas como de archivos codificados en RDF/RDFS.

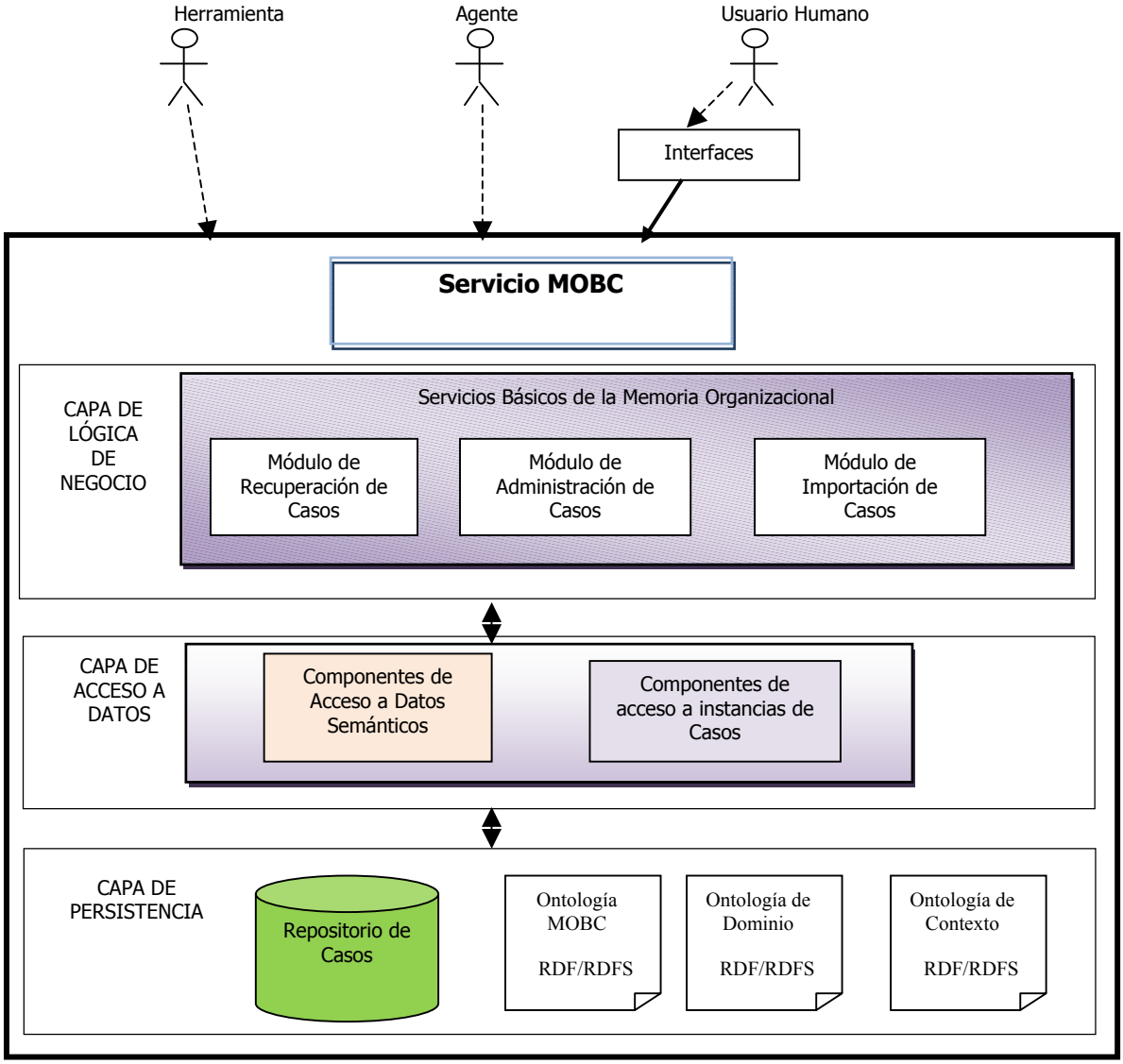


Figura 7.4. Una vista general de la arquitectura de la MOBC

- **Capa de Acceso a Datos:** Se integra de un conjunto de componentes de software que independizan y facilitan a las capas superiores el acceso a los datos del repositorio almacenados en la capa de persistencia. Está integrada por dos tipos de componentes:

  - *Componentes de acceso a instancia de casos*
  - *Componentes de actualización de casos*

- **Capa de persistencia:** almacena de manera persistente tanto las bases de conocimiento basada en casos como las ontologías asociadas y sus esquemas.

7.5 Sistema de Memoria Organizacional Distribuida

Existen básicamente dos tendencias relacionadas con el enfoque de distribución de los sistemas de memorias organizacionales: el enfoque de repositorio y el enfoque de red [Abidi et al., 2005]. El enfoque de repositorio apunta principalmente a la codificación del conocimiento, es decir, la creación y mantenimiento de almacenes de conocimiento visto como un objeto que puede ser recolectado, almacenado y organizado. Como tales, estos sistemas se orientan primariamente a los aspectos de almacenamiento y recuperación en la Gestión del Conocimiento organizacional. En este enfoque se establece una ubicación central para almacenar el conocimiento codificado.

Las experiencias y habilidades colectivas de las personas son el marco del capital intelectual en cualquier organización. El enfoque de repositorio de conocimiento puede aliviar o incluso eliminar la pérdida de capital intelectual. Comparada con la memoria humana que puede ser limitada y transitoria, un repositorio de conocimiento bien diseñado puede ser extenso y confiable. El costo decreciente de los dispositivos de almacenamiento permite que una sustancial cantidad de información pueda acumularse en repositorios de conocimiento digitales. Las computadoras permiten, al mismo tiempo, el procesamiento de grandes cantidades de información considerablemente más rápido de lo que las personas pueden hacerlo, reduciendo el impacto de la sobrecarga de información.

La otra tendencia en la implementación de memorias organizacionales, es el enfoque de red, que apunta a usar el poder de las tecnologías de información y comunicación para dar soporte al flujo de conocimiento en el ámbito organizacional entre distintos sectores de la empresa. En este enfoque el conocimiento permanece distribuido a través de nodos correspondientes a departamentos, sucursales o dependencias. Debido a que el conocimiento no está almacenado en una ubicación central, el énfasis en este enfoque está en las relaciones entre los actores involucrados,

que pueden ser personas, grupos, organizaciones, comunidades o incluso sociedades [Seufert et al., 1999].

De acuerdo a lo anterior, es evidente que lo más importante en la implementación de un modelo basado en el enfoque de red es la estructuración de la distribución del conocimiento a través de la red, el establecimiento de los medios para accederlo y la gestión de algunos datos sobre el conocimiento (metadatos) para propósitos estructurales, es decir, para crear un directorio que establezca qué conocimiento detenta cada nodo. Otra consideración importante para la gestión en un enfoque de red es la integración de diferentes tipos de conocimiento. Debido a que la mayoría del conocimiento no está almacenado en un repositorio central, se tendrá que combinar conocimiento individual con tipos más grandes (grupal, organizacional) como así también combinar las diferentes áreas de especialización (por ejemplo, desarrollo, testing, calidad, etc.) para cada uno de esos niveles.El enfoque de red enfatiza el uso de las tecnologías de comunicación. Con la creciente ubicuidad de Internet y de los dispositivos para acceder a ella, las redes de conocimiento ya no están confinadas a las intranets organizacionales. Una organización puede incorporar y combinar de forma más fácil conocimiento entrante desde otras dependencias localizadas en zonas geográficas muy distantes o de fuentes externas a la organización. De esta forma se potencia la habilidad para usar, compartir y explotar el conocimiento, y las capacidades y recursos distribuidos en redes, tanto dentro como fuera de la organización. Además, le da a dicha organización una ventaja competitiva distintiva sobre aquellas que sólo están basadas en un enfoque de repositorio.

En esta tesis se ha considerado, que los enfoques de repositorio y de red descriptos en los párrafos anteriores no son mutuamente excluyentes y que se puede aprovechar los beneficios de cada uno, que son importantes para una implementación eficiente de un sistema de Memoria Organizacional. Por lo tanto, para el diseño del sistema de la Memoria Organizacional total de una empresa se tuvo en cuenta que el mismo debe dar soporte a dos procesos cualitativamente diferentes: la gestión autónoma del conocimiento local a un nodo (que será diseñado con el enfoque de repositorio), y la coordinación para su interacción y la distribución del conocimiento con otros nodos (que será diseñado con un enfoque de red). Estos son los lineamientos principales del diseño del Sistema de Memoria Organizacional Distribuida que se propone.

En la figura 7.5 se esquematiza el modelo de la Memoria Organizacional distribuída propuesta donde se pueden observar tres tipos de elementos interconectados a través de una red, a saber: los nodos de MOBC, el catálogo de bases de conocimiento basada en casos, y los agentes externos, los cuales se describen a continuación:

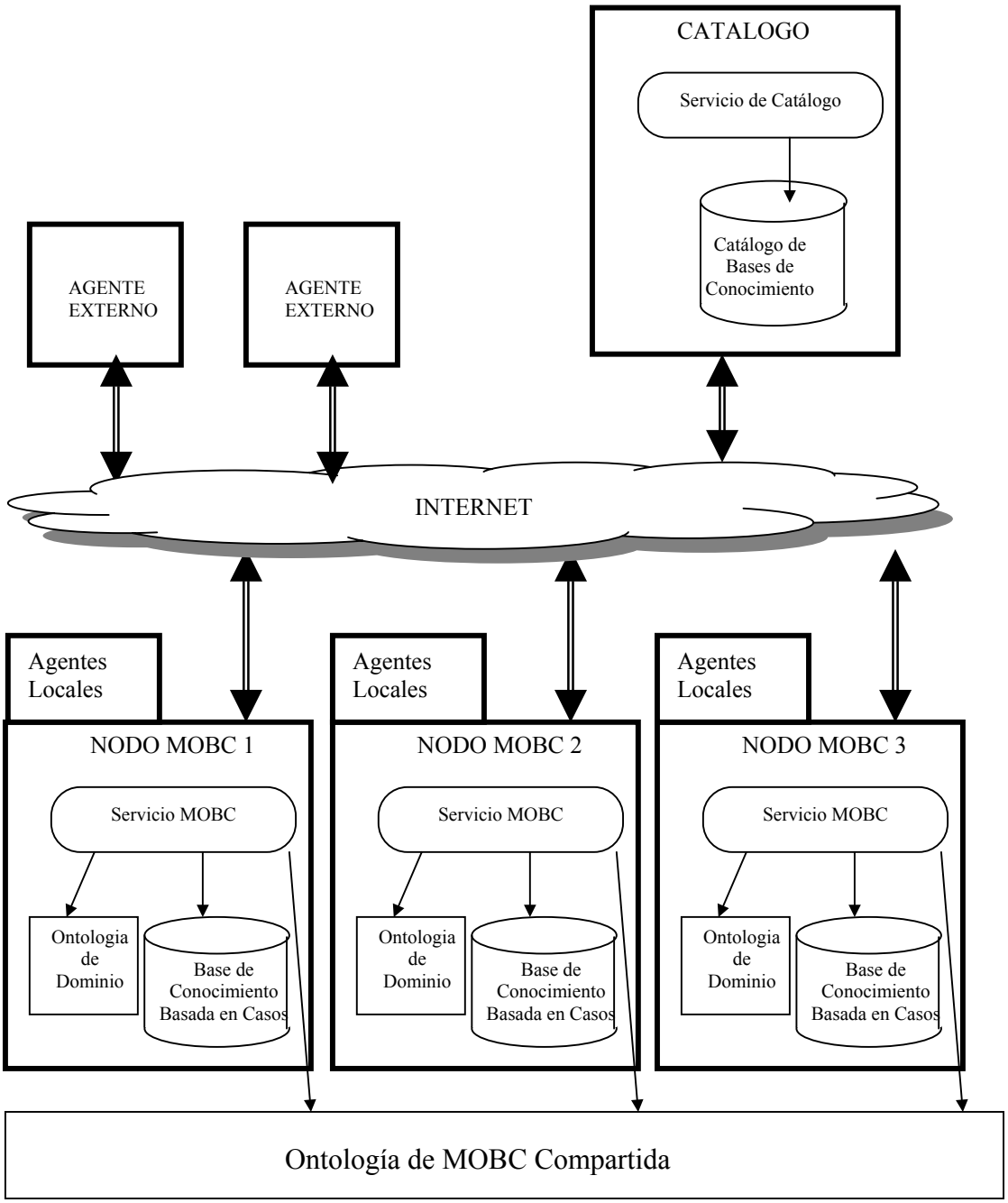


Figura 7.5. Modelo de la MOBC Distribuída

Nodos de MOBC: consiste de los servicios y componentes informáticos usados para almacenar, buscar y recuperar conocimiento estructurado en casos, relacionado a

determinados dominios. Este componente fue descrito en la sección 7.3. y está basado en el principio de un modelo de repositorio. La interpretación semántica del conocimiento que administra es llevado a cabo valiéndose de las ontologías de dominio y contexto asociadas. Este conocimiento puede ser usado y explotado tanto por agentes locales, que a su vez pueden ser tanto automáticos (como aplicaciones o herramientas) como humanos o por agentes externos a través de la red.

Es conveniente que estos nodos sean replicados en distintas dependencias de la organización, instanciados con las respectivas ontologías de dominio y contexto para gestionar el conocimiento de la comunidad local y conectados a la red para compartir conocimiento con otros nodos.

Catálogo de bases de conocimiento basada en casos: consiste de los servicios y componentes informáticos usados para administrar un catálogo de bases de conocimiento basadas en casos. El catálogo registra la descripción y los URL (Localizador de Recurso Universal) de todas las bases de conocimiento basada en casos distribuidas en distintos nodos. Las dos funcionalidades principales definidas para este servicio son:

Registrar: permite a un nodo de MOBC publicar (registrar) una base de conocimiento basado en casos, para lo cual deberá proveer como entrada al servicio, el nombre de la base, el dominio, el contexto y el URL donde esté corriendo el servicio MOBC correspondiente.

Recuperar: devuelve la meta-información de cuáles son las bases de casos disponibles y las URLs de los servicios que las implementan. La consulta puede ser filtrada por dominio de conocimiento o por nodo proveedor.

Agentes Externos: Son clientes que acceden a los distintos nodos de MOBC para consultar, usar, o compartir conocimiento de las distintas bases de conocimiento distribuidas. Un agente local, puede adquirir el rol de agente externo, accediendo a otros nodos de la red, distinto al de su nodo, para integrar conocimiento de otras comunidades.

Para que un agente pueda acceder al conocimiento almacenado en los distintos nodos, además de tener los permisos correspondientes, deberá conocer la dirección URL del servicio, que puede ser enlazada en forma estática, o dinámicamente consultado el servicio de catálogo. Esto es posible porque la interconexión de componentes del SMOBC está basada en la tecnología SOA (Arquitectura orientada a servicios). En la figura 7.6 se describe brevemente este proceso.

Cada vez que se crea una nueva base de conocimientos basada en casos para ser compartida, el nodo de MOBC crea un registro que publica en el catálogo para su consulta por parte de otros agentes (operación publicar). Los agentes de las distintas comunidades de la organización, pueden localizar las bases de conocimientos relacionadas a determinados dominios, de distintos nodos, realizando una operación de búsqueda solicitada al servicio del catálogo (operación recuperar). El catálogo devuelve la descripción solicitada con el correspondiente URL, que es usado por el agente para invocar al servicio de MOBC correspondiente (operación enlazar). Una vez que se encontró el servicio buscado, las interacciones se llevan a cabo directamente entre el agente y el servicio del nodo de MOBC.

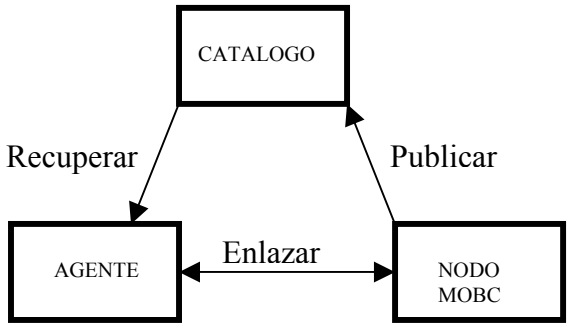


Figura 7.6. Modelo SOA de SMOBC Distribuido.

El sistema de Memoria Organizacional Basada en Casos distribuida descripta en esta sección, ofrece por un lado la facilidad de gestionar el conocimiento local con un alto grado de cohesión y el establecimiento de una red de conocimiento mediante la implementación de una Memoria Organizacional en cada una de las comunidades de conocimiento identificadas. Para lograr esto es necesario identificar dentro de la organización los dominios de conocimiento existentes como así también las comunidades de práctica y de conocimiento que los integran, conectándolos en una red

de conocimiento que potencie su papel en las actividades de creación y distribución de conocimiento.

7.6 El SMOBC Distribuido como soporte a la Gestión del Conocimiento

El sistema de Memoria Organizacional Basada en Casos distribuida descripta en esta tesis fue diseñado con el propósito de ser usado como componente central de sistemas de Gestión del Conocimiento organizacional. En tal sentido, en esta sección se analizan las capacidades potenciales que brinda el SMOBC al desarrollo de tales sistemas, tanto desde el punto de vista de su *dimensión semántica* (es decir capacidad de razonamiento) como desde el punto de vista de su *dimensión web* (es decir su capacidad de compartir y diseminar el conocimiento distribuido en la Web). El análisis que se realiza, se basa en el "*marco de clasificación en Web Semántica para las aplicaciones de Gestión del Conocimiento basados en ontologías*" definido por Mika y Akkermans [Mika et al., 2004].

Dicho marco, propone la matriz *Web Semántica*, una novedosa guía de clasificación de aplicaciones para la Gestión del Conocimiento basados en ontologías. El primer eje de esta matriz es el escenario de uso y funcionalidad de la ontología. El segundo eje de clasificación está relacionado con la dimensión Web, con el objetivo de representar los varios estamentos de descentralización en las aplicaciones de KM.

Los autores afirman que la comprensión compartida representada por las ontologías puede usarse como sostén de tres procesos claves en la Gestión del Conocimiento a saber: la comunicación, la integración y el razonamiento (ver figura 7.7). Como será explicado en la próxima subsección, estos procesos construyen niveles crecientes de formalidad de las ontologías involucradas a medida que la complejidad del conocimiento requerido aumenta. La dimensión semántica formará el eje vertical de la matriz.

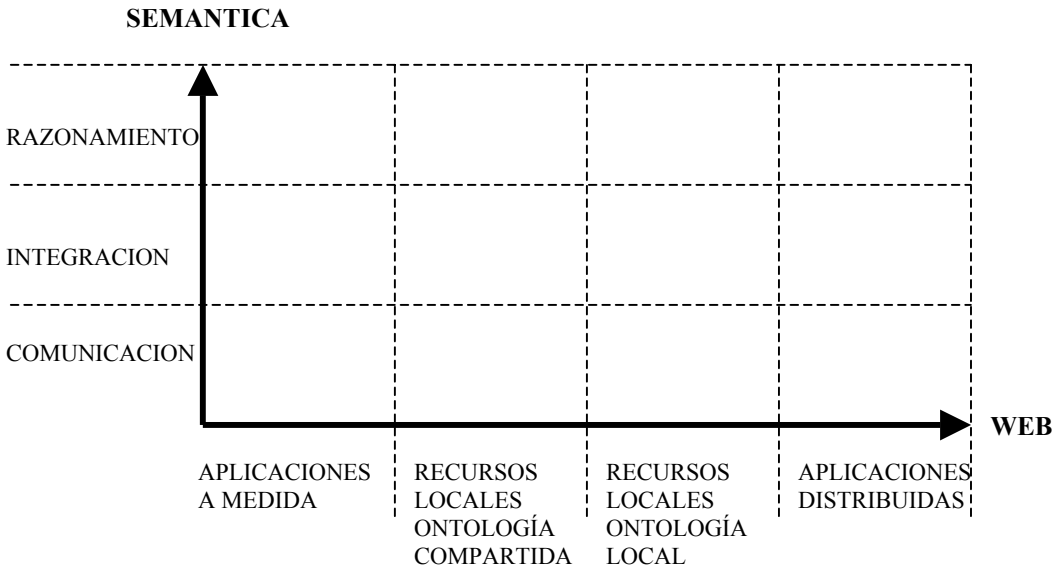


Figura 7.7. Matriz de clasificación en Web Semántica para las aplicaciones de Gestión del Conocimiento basados en ontologías (adaptada de [Mika et al., 2004]).

El segundo eje se refiere al alcance de la aplicación de las ontologías. En el extremo negativo, la aplicación de ontologías se limita al conocimiento explícito. No obstante, una vez que se tiene algún conocimiento descrito de una manera explícita y formal, se puede confiar en las computadoras para aprender con este conocimiento. Esto no sólo significa un aumento en el tamaño del conocimiento que se gestiona, sino también, y no menos importante, nos da la oportunidad de usar la computadora conectada a una red (intranets, extranets e Internet) para extender significativamente el alcance de este conocimiento. Esto significa la comunicación a través de los grupos autónomos, integración de su conocimiento y razonando, agregando información de fuentes dispares. Este análisis constituye la segunda dimensión de la matriz (la dimensión web).

7.6.1 La Dimensión Semántica

En esta subsección se analiza el potencial de procesamiento semántico que el SMOBC propuesto le confiere a los sistemas de Gestión del Conocimiento basados en él. Para ello se consideran tres procesos de conocimiento claves que las ontologías apoyan en las aplicaciones de Gestión del Conocimiento [Mika et al., 2004]. Estos

procesos se disponen sobre el eje de la dimensión semántica (ver figura 7.7) en forma creciente del nivel de comprensión y razonamiento hacia la parte superior del eje. Estos procesos son comunicación, integración y razonamiento, que se detallan a continuación:

Comunicación: Las ontologías facilitan la comunicación proporcionando un entendimiento compartido sobre los casos en el dominio, a través de una terminología no ambigua y formal, que asegura una interpretación única y un procesamiento automático de los mensajes.

En el caso del SMOBC propuesto en esta tesis, la ontología de MOBC se aplica en forma consistente como un lenguaje formal y compartido que habilita a los grupos activos y comunidades a compartir el conocimiento de una forma más eficiente evitando malos entendidos.

Integración: Las ontologías pueden ser más que un simple vocabulario de términos formalmente definidos. Su verdadero potencial se observa en la descripción de las relaciones entre entidades del dominio. Las ontologías ricas en relaciones son más potentes (en cuanto a su capacidad de integrar conocimiento) que las simples jerarquías taxonómicas que se usan frecuentemente en soluciones actuales de Gestión del Conocimiento.

En el caso del SMOBC propuesto en esta tesis, el potencial de integración del conocimiento está asegurado con la asociación de ontologías de dominio y de contexto a cada MOBC, como fue detallado en las secciones 7.3.1 y 7.3.2 respectivamente. Al agregar las relaciones entre los conceptos de la ontología de MOBC y la ontología de dominio, como parte constitutiva de la MO (ver figura 7.2), una aplicación puede integrar los distintos items de conocimiento en forma automática valiéndose de los metadatos a través de un procesamiento semántico. De esta manera, agregando descripciones basadas en ontologías de dominio (metadatos) que describen la estructura (relaciones entre los items de conocimiento) de los casos se contribuye a una integración más inteligente y automática del conocimiento, que pueda tener origen en dominios o contextos diversos.

Razonamiento: Las aplicaciones de razonamiento terminológico representan el uso más complejo para las ontologías. Mientras que las dos capas de abajo en la figura

7.7 están construidas sobre el conocimiento de *qué* tipos de objetos están en nuestro dominio (la comunicación) y *cómo* ellos están relacionados (la integración), este nivel requiere el conocimiento de *por qué* ellos están relacionados. En otros términos, este nivel involucra las reglas y principios detrás de una cierta conceptualización.

En el caso del SMOBC propuesto, la capacidad de razonamiento no está basado en la ontología (es decir en la incorporación de reglas lógicas a la ontología), sino en estructurar el conocimiento en casos para poder aplicar técnicas de RBC. Como ya se mostró en el capítulo 4 (sección 4.7), el Razonamiento Basado en Casos ofrece numerosas ventajas sobre el razonamiento basado en reglas, cuando se aplica a una Memoria Organizacional de experiencias y lecciones aprendidas como es el caso del SMOBC. Además, la capacidad de razonamiento es inherente al RBC, por lo tanto es una capacidad provista por nuestra propuesta.

Como conclusión, podemos decir que las aplicaciones de Gestión del Conocimiento que usen el SMOBC propuesto en esta tesis como componente central, se ubicarán al tope de la matriz Web Semántica en relación al eje semántico, es decir, se ubican en un nivel de razonamiento.

7.6.2 La Dimensión Web

La dimensión Web de la matriz de clasificación de aplicaciones de Gestión del Conocimiento permite identificar las ventajas y desventajas de mover un KMS hacia escenarios cada vez más descentralizados (ver figura 7.7). Se describen cuatro niveles de descentralización en aplicaciones basadas en ontologías [Mika et al., 2004] que se describen a continuación:

Aplicaciones a medida. Estas son aplicaciones basadas en ontologías que operan en un ambiente cerrado donde tanto los recursos de conocimiento como la lógica de la aplicación está bajo control de una sola entidad. Notar que esta definición no se refiere a la implementación técnica de la aplicación que puede tomar la forma de un sistema distribuido, como sería el caso de una aplicación cliente/servidor punto a punto, satisfecha dentro de una red organizacional.

Recursos locales, ontología compartida. En este nivel se ubican las aplicaciones que usan una ontología central compartida pero que gestionan recursos de conocimiento locales controlados en forma independiente. Note que este nivel de descentralización presupone la existencia de alguna forma de una red que conecta estos recursos a la aplicación.

Recursos locales, ontología local. Las aplicaciones en este nivel de descentralización están construidas sobre ontologías locales y gestionan recursos de conocimiento locales. No hay ninguna ontología global o compartida como en el caso anterior.

Aplicaciones distribuidas. Estas aplicaciones se caracterizan por la descentralización de la propia lógica de la aplicación. Los nodos de la red no sólo controlan recursos y ontologías, sino que crean sus propias aplicaciones de componentes independientes (servicios y agentes) distribuídos encima de la red. Este nivel donde tanto recursos de conocimiento como la lógica están descentralizados, representa el máximo nivel de descentralización .

En el caso del SMOBC propuesto, el nivel de descentralización está determinado por la aplicación de Gestión del Conocimiento en sí misma que lo utiliza como componente central, y no por el SMOBC en sí mismo, ya que el componente SMOBC no le imprime ninguna restricción en cuanto a la distribución a los sistema que lo usen.

Por ejemplo, utilizando el SMOBC, se puede construir una aplicación que administre conocimiento local de varios nodos de MOBC que comparten la misma ontología de dominio. Esta aplicación sería clasificada dentro de un nivel *Recursos locales, ontología compartida,* (marcada con el número 1 en la figura 7.8). A este cuadrante pertenece la aplicación del sistema de recomendación en proyectos de medición y evaluación de calidad que se describirá en el próximo capítulo.

Otro caso sería construir una aplicación que administre conocimiento local en varios nodos de MOBC cada uno con ontología de dominio distinta. Esta aplicación sería clasificada dentro de un nivel *Recursos locales, ontología local,* (marcada con el número 2 en la figura 7.8).

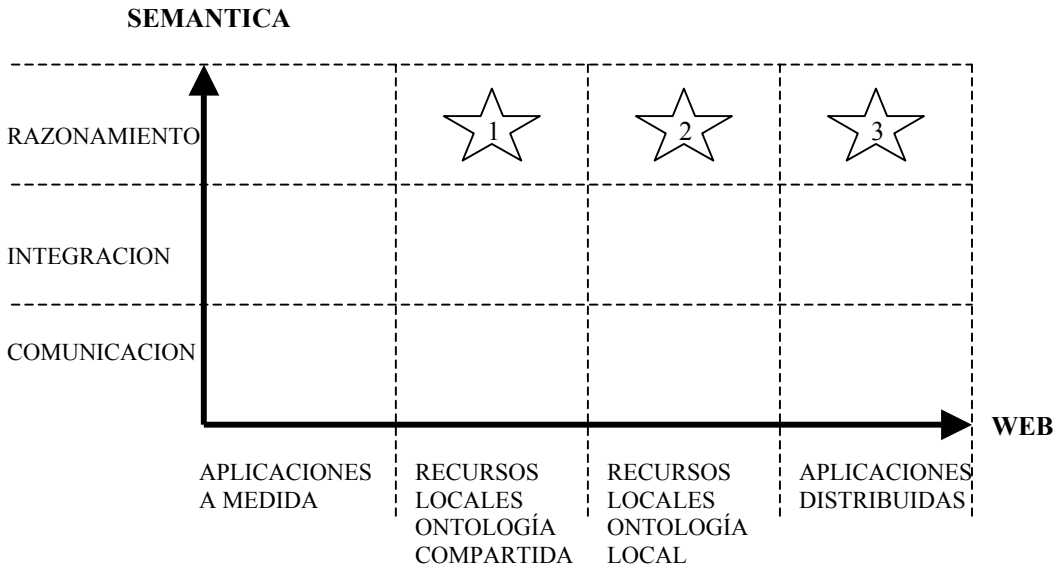


Figura 7.8. Ejemplo de clasificación de aplicaciones de Gestión del Conocimiento basados en ontologías (adaptada de [Mika et al., 2004]).

Finalmente, una aplicación que administre recursos de conocimiento distribuidos en distintos nodos de MOBC cada uno con una ontología de dominio distinta, sería clasificada dentro de un nivel *Aplicación distribuida,* (marcada con el número 3 en la figura 7.8).

7.7 Conclusiones

La problemática identificada en el capítulo 2 del estado del arte, planteó la necesidad de contar con un sistema robustamente estructurado para explotar el valioso conocimiento presente en una organización El sistema de MOBC propuesto en este capítulo tiene como objetivo cubrir esta necesidad.

La distinción de los dos niveles de ontologías como fue expresado, facilita la tarea de estructurar la MO, y le confiere flexibilidad, adaptabilidad y mayor capacidad de procesamiento automático. Estas características se fundamentan en la relación establecida entre los conceptos de la MO y los conceptos del dominio, que se define a un nivel de metadatos, aprovechando las tecnologías que se han desarrollado para la Web Semántica.

El diseño de la distribución del sistema de MOBC permite que el conocimiento esté siempre disponible y sea de fácil acceso, aún cuando se haya generado en distintos lugares, distribuidos geográficamente. Para facilitar el desarrollo de herramientas que automaticen los procesos de captura, consulta y reuso de casos, de una manera interoperable aún cuando estén implementadas en plataformas distintas, se propone su implementación bajo estándares de marcado (como XML/XMLS), estándares de Web Semántica (como RDF/RDFS), Web Services y arquitecturas de ambientes distribuidos de amplia difusión.

Es posible concluir que el sistema de MOBC propuesto es una alternativa superadora, pudiendo servir como componente central para la implementación de sistemas de Gestión de Conocimiento diversos y distribuidos. En particular, en el próximo capítulo mostramos la construcción de un prototipo de sistema de recomendación para proyectos de medición y evaluación en el área de aseguramiento de calidad en Ingeniería de Software.

Capítulo 8 - Aplicación de la MOBC a un Sistema de Recomendación en Proyectos de Medición y Evaluación

8.1 Introducción

El estándar IEEE 610.12 define Ingeniería de Software como "*La aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software, es decir la aplicación de ingeniería al software*". [IEEE 1990]. Según la opinión de Ye, aunque el desarrollo del software comparte muchas características con otras disciplinas de la ingeniería, presenta, sin embargo, nuevos desafíos a la investigación porque es una disciplina fuertemente dependiente del conocimiento y la creatividad de los individuos de desarrollo del software [Ye, 2006].

Varios autores han observado la conveniencia de fortalecer las relaciones entre las disciplinas de Ingeniería de Software y Gestión del Conocimiento debido a que la Ingeniería de Software se caracteriza, precisamente, por ser una actividad intensiva en conocimiento [Kukko et al., 2008]. Una opinión similar fue expresada por Aurum y colaboradores[Aurum et al., 2008], quienes mencionan que para ubicar a las compañías de software en forma competitiva en el mercado, es importante desarrollar maneras eficaces de manejar el conocimiento que es de interés a desarrolladores de software.

Un área muy importante de la Ingeniería de Software y Web actual, es el de aseguramiento de calidad. En la implementación de proyectos de desarrollo de software, los procesos de medición suponen, junto con los de evaluación, una de las actividades de soporte principales previstas en los estándares para el control y el aseguramiento de la calidad[7]. Esta es una de las áreas donde la Gestión del Conocimiento se vuelve muy importante. Particularmente en los proyectos de medición y evaluación, se puede usar el conocimiento adquirido en experiencias con proyectos anteriores para reusar y mejorar

[7] Algunos estándares ISO tratan sobre calidad, procesos y evaluación: ISO 9126-1:2001 relacionada a la calidad del producto software; ISO 15504:2003 (SPICE, *Software Process Improvement and Capability Determination*) especifica el proceso de calidad; ISO 14598:1998 establece el proceso de evaluación; ISO15939:2002 define el proceso de medición.

las prácticas en proyectos futuros. El aprendizaje organizacional basado en estas experiencias es un factor clave para mejorar la productividad y corregir errores.

Si bien la literatura general en Gestión del Conocimiento contiene muchos ejemplos de sistemas exitosos usados en compañías relacionadas a tecnologías de la información, muy pocos se relacionan al área de Ingeniería de Software y hasta donde conocemos, ninguno de esos ejemplos son específicos para proyectos de medición y evaluación en aseguramiento de calidad de software. Además, según lo expresa Bjornson, a pesar de que hay una comunidad de investigadores en Gestión del Conocimiento en Ingeniería de Software, sus trabajos están lejos de la principal tendencia en Gestión del conocimiento, ya que tratan principalmente del almacenamiento y recuperación del conocimiento, mientras que los temas como la creación, transferencia y aplicación necesitarían mayor atención [Bjornson, 2007].

Las formas usuales de gestionar el conocimiento en Ingeniería de Software están principalmente basadas en técnicas como entrevistas semi-estructuradas [Komi-Sirvio et al., 2002], el análisis de proyectos postmortem (después de finalizados) [Birk et al., 2002], [Harrison, 2003] o informes de experiencias y revisiones [Cooper, 2007], [Dingsoyr et al., 2001]. El inconveniente principal de estas formas de capturar el conocimiento es que requiere que los participantes de estas experiencias, es decir los miembros de equipos del proyecto, dispongan del tiempo necesario para participar en esta actividad lo que no es posible en la mayoría de los casos [Basili et al., 2001], [Cooper, 2007]. Además, si hay una diferencia de tiempo entre la experiencia y su captura, existe el riesgo de que esa experiencia finalmente se pierda.

Con el fin de brindar alguna solución a estos problemas, en este capítulo se muestra la aplicación del SMOBC desarrollado en el capítulo anterior a un sistema de recomendación en el área de aseguramiento de calidad. Dicho sistema de recomendación, permite a las empresas desarrolladoras de software capturar y administrar el conocimiento obtenido de experiencias en proyectos de medición y evaluación, en el mismo momento en que ellas ocurren durante el ciclo de vida del proyecto. De esta manera, se integran procesos y artefactos específicos de Gestión del Conocimiento a las herramientas de apoyo a los procesos de desarrollo del software. En particular, en este caso se integra gestión de conocimiento para la recomendación en

proyectos de medición y evaluación gestionados con la herramienta C-INCAMI-Tool [Olsina et al., 2007]. A los fines de una mejor comprensión de la aplicación, en las próximas dos secciónes presentamos una introducción a los principales conceptos de medición y evaluación en aseguramiento de calidad y del marco de medición y evaluación C-INCAMI respectivamente.

8.2 Medición y Evaluación en Aseguramiento de Calidad

En los proyectos de desarrollo de software, los procesos de medición y evaluación, son una de las actividades de soporte principales para el control y el aseguramiento de la calidad. Los procesos de medición son fundamentales dado que permiten cuantificar un conjunto de características deseadas acerca de un aspecto específico de algún ente en particular, proveyendo una visión más o menos detallada de su estado o condición. Por su parte, la evaluación interpreta los valores obtenidos en la medición. Para dichos procesos es necesario obtener datos cuantitativos, a partir de métricas de atributos de entes y la posterior interpretación de la medida a partir de indicadores [Olsina et al., 2007].

Es numerosa la información existente referida a la definición de medición y evaluación, pero sin un claro consenso en cuanto a la terminología utilizada. Afortunadamente, existen trabajos de investigación que proporcionan una ontología de métricas e indicadores que especifica un vocabulario común, con una semántica bien definida para el área de medición y evaluación [Martin, 2005]. Dicha ontología constituye una importante propuesta en el área de gestión de calidad ya que facilita la interpretación de la información de métricas e indicadores en forma uniforme, y permite que los resultados obtenidos en distintos proyectos puedan ser comparados en forma consistente.

Además, las organizaciones decididas a encarar un proyecto de medición y evaluación deben contar con un marco que permita definir cada una de las actividades involucradas y los conceptos que ellas implican. Afortunadamente, existen trabajos que proveen una especificación clara y detallada en este sentido. Particularmente,

C-INCAMI es un marco conceptual de medición y evaluación orientado a propósitos y centrado en la organización [Olsina et al., 2007].

C-INCAMI se fundamenta en la metodología de evaluación WebQEM (Web Quality Evaluation Method) [Olsina et al., 2002]. WebQEM está basada en un modelo jerárquico de requerimientos de calidad, partiendo de las características de alto nivel prescriptas en la norma ISO-9126 [ISO9126-1, 2001], pudiendo también utilizarse otras.

De esta manera, C-INCAMI puede ser utilizado en el diseño de requerimientos no funcionales, en el diseño de métricas para cuantificar los atributos de las entidades involucradas y en la interpretación de los valores correspondientes mediante indicadores, conforme a una necesidad de información concreta a nivel de proyecto en una organización.

8.3 Introducción al Marco C-INCAMI

El marco C-INCAMI define un conjunto de componentes y conceptos relacionados con la medición y evaluación de requerimientos no funcionales en proyectos de software y Web, como parte de las actividades de aseguramiento de calidad de una organización. Está preparado no sólo para guardar valores finales obtenidos de las mediciones y evaluaciones sino que además, permite que se registren todos los metadatos asociados para garantizar consistencia, repetitividad y replicabilidad.

C-INCAMI está estructurado en base a las actividades o fases a realizar en el proceso de medición y evaluación. Se basa en el principio de que una organización que busque medir y evaluar un proyecto de software de un modo eficaz, debe hacerlo orientado a un propósito, por lo cual debe especificar requerimientos no funcionales a partir de la identificación de una necesidad de información. Además se debe especificar el contexto de información relevante al proyecto. Luego, se debe diseñar y seleccionar un conjunto específico de métricas útiles para el propósito definido. Por último, se deben interpretar los valores obtenidos por medio de indicadores, para evaluar el grado de satisfacción que proporciona la entidad evaluada respecto a los requerimientos. Este marco conceptual está formado por cinco etapas principales que son:

1. Definición y Especificación de Requerimientos no Funcionales

2. Especificación del Contexto del Proyecto

3. Diseño y Ejecución de la Medición

4. Diseño y Ejecución de la Evaluación

5. Análisis y Recomendación

El marco C-INCAMI está soportado por la herramienta C-INCAMI_Tool que automatiza las actividades mencionadas. A continuación se explican brevemente cada uno de los componentes principales, remarcando con itálica aquellos términos que pertenecen a la ontología de métricas e indicadores [Martin et al., 2003] [Martin, 2005].

8.3.1 Definición y Especificación de Requerimientos no Funcionales

La primera etapa corresponde a la definición y especificación de requerimientos. Este módulo trata con la definición de la necesidad de información (*InformationNeed*) (es decir, el objetivo de la evaluación y el perfil del usuario) y el diseño de los requerimientos no funcionales, que servirán como guías para las actividades posteriores de medición y evaluación. En la figura 8.1 se puede visualizar los términos, atributos y relaciones que forman parte del componente Definición y Especificación de Requerimientos no Funcionales.

La necesidad de información es descripta por un concepto calculable (*CalculableConcept)*. Un concepto calculable se define como una relación abstracta entre atributos de un ente y una necesidad de información. Ejemplos de instancias de conceptos calculables son calidad, calidad en uso, calidad externa, etc. Un concepto calculable puede ser representado por un modelo de concepto (*ConceptModel*), como por ejemplo los modelos de calidad de software ISO 9126-1. Un modelo de concepto permite diseñar los requerimientos a evaluar para una determinada categoría de entidad enfocándose en un concepto calculable. Dicho modelo puede tener conceptos de menor nivel de abstracción que el concepto calculable. En el nivel más bajo se encuentra los

atributos del ente a medir. Tanto los conceptos de más alto nivel como aquellos de bajo nivel conforman lo que se denomina árbol de requerimientos.

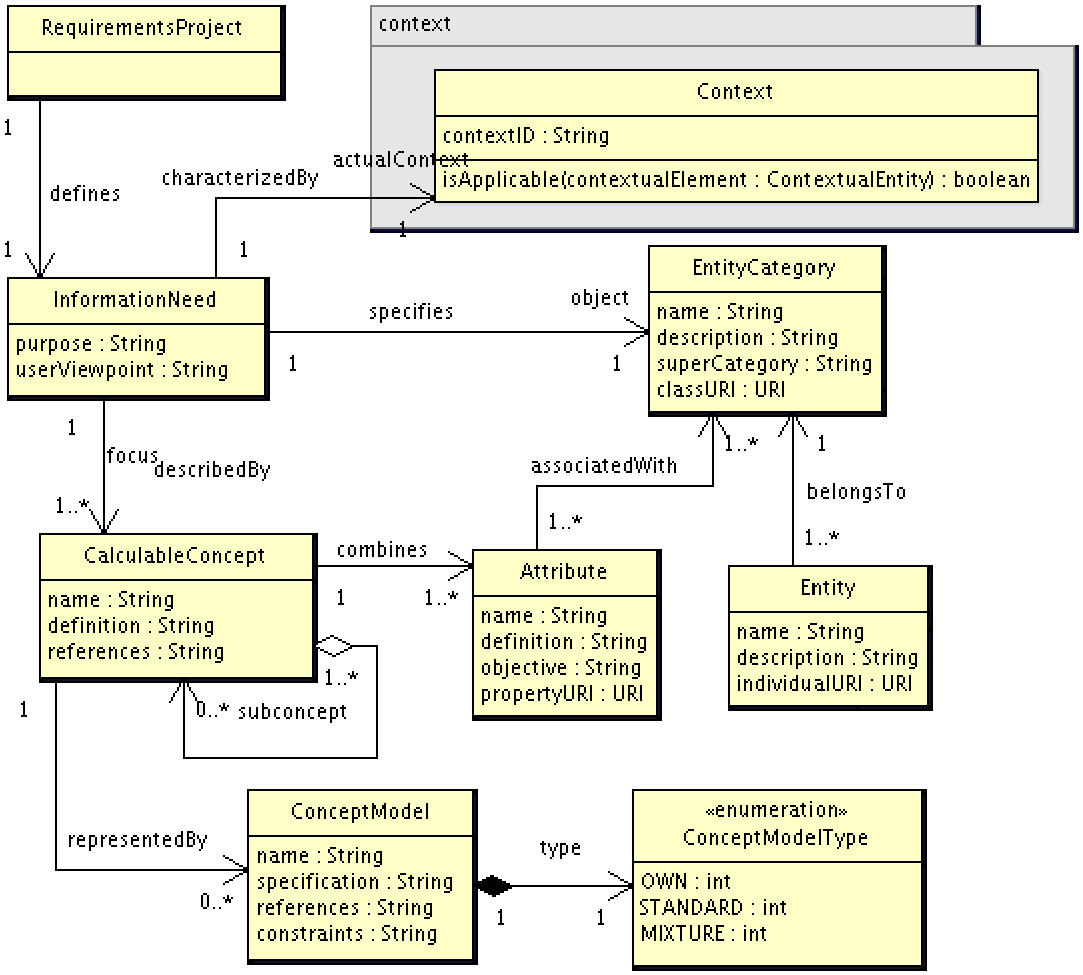


Figura 8.1. Modelo conceptual de Requerimientos No Funcionales

El término atributo (*Attribute*) se define como una propiedad mensurable de una categoría de entidad. Los atributos conforman el árbol de requerimientos que se diseña en esta etapa. Los atributos pertenecen a una categoría de entidad (*EntityCategory*), y son medidos para una entidad (*Entity*) particular que pertenece a dicha categoría. Como ejemplos de categorías de entidad de interés para la Ingeniería de Software y Web pueden mencionarse: "Grupo de Desarrollo", "Producto Software", "Código Fuente". Para dichas categorías, pueden asociarse como ejemplo las siguientes entidades respectivamente: "Grupo de desarrollo Gidis_Web", "sitio Web de la UNLPam", "holaMundo.c".

8.3.2 Especificación del Contexto del Proyecto

Este componente permite describir el contexto (*Context*) por medio de propiedades contextuales (*ContextProperty*) que sean relevantes para una necesidad de información (ver figura 8.2), como por ejemplo, considerar el ciclo de vida utilizado para el desarrollo de algún proyecto de software que se desea medir.

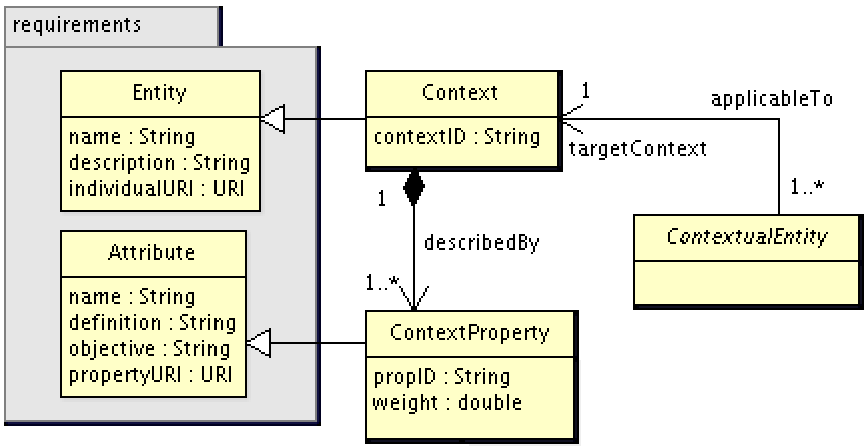


Figura 8.2. Modelo Conceptual de Contexto

Contar con información de contexto permite posteriormente realizar análisis más consistentes y objetivo de los resultados [Molina et al., 2008]. Además, permite realizar recomendaciones en proyectos futuros basados en casos similares y contextos compatibles. Una vez definidas las propiedades que caracterizan el contexto, se debe proceder a cuantificar cada una de ellas haciendo uso de alguna métrica asociada. De esta manera, se obtendrán las propiedades de contexto relevantes y sus valores.

8.3.3 Diseño y Ejecución de la medición

Este componente corresponde al diseño e implementación de la medición. Trata con la definición de las métricas que serán útiles para cuantificar los atributos, que se identificaron como parte de la especificación de requerimientos, y que son de especial interés en el proyecto, dado que constituyen las características que se medirán para el ente a evaluar, considerando la necesidad de información establecida (es decir, el objetivo final de la evaluación). La figura 8.3 ilustra este módulo.

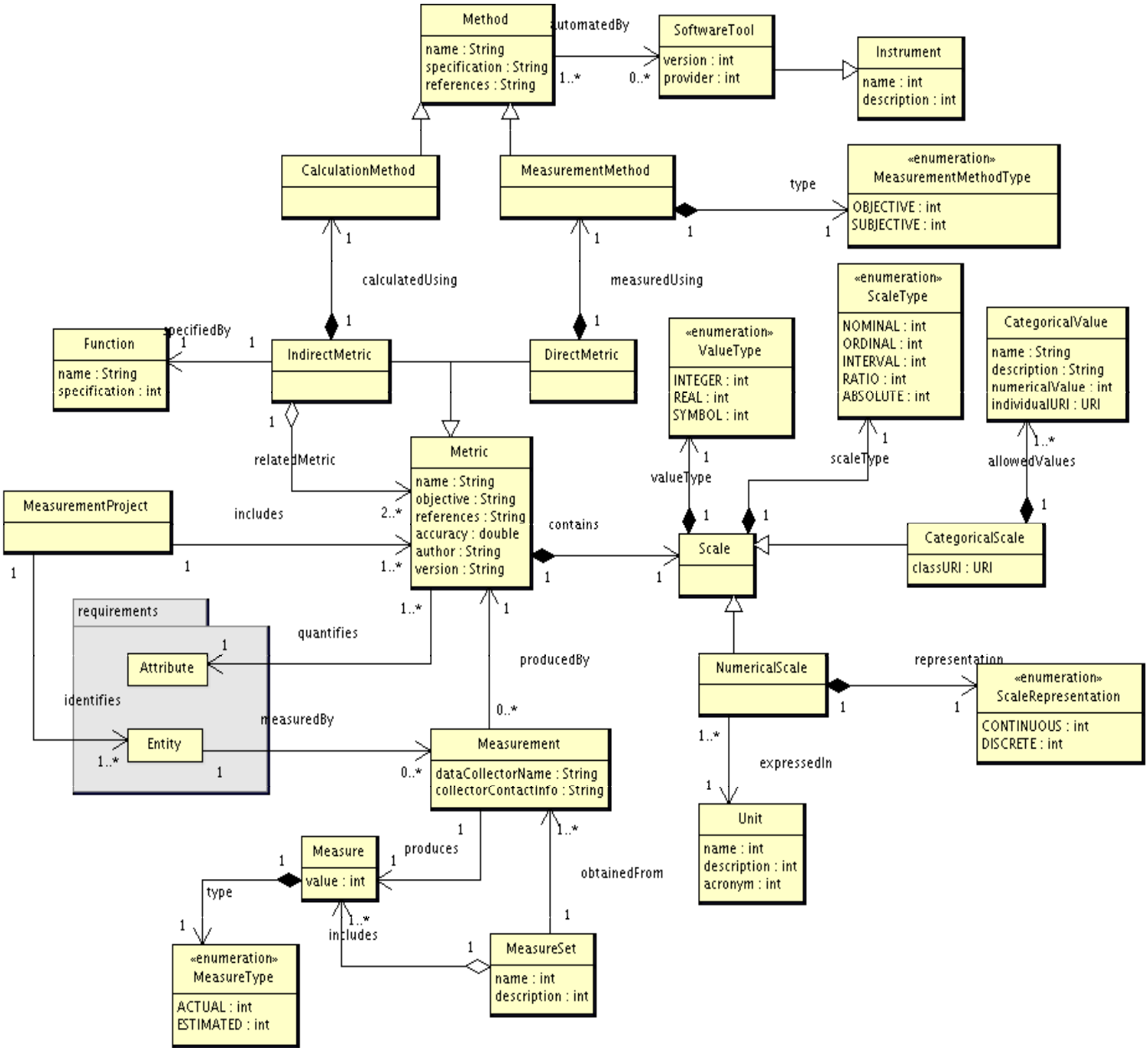


Figura 8.3. Modelo Conceptual de Medición

Cada atributo del árbol de requerimientos diseñado puede ser cuantificado por muchas métricas, pero para un proyecto específico, sólo una métrica (*Metric*) debería ser seleccionada para cuantificar cada atributo. La métrica define cómo se llevará a cabo la medición (*Measurement*). Las métricas contienen una escala (*Scale*), que puede ser

numérica (*NumericalScale*) o bien categórica (*CategoricalScale*). Si es numérica, contiene una unidad (*Unit*).

Las métricas pueden ser directas (*DirectMetric*) o indirectas (*IndirectMetric*) y debe seleccionarse para cada una, el método (*Method*). Una métrica directa tiene un método de medición (*MeasurementMethod*), que puede ser objetivo, cuando sólo se basa en métodos numéricos, o bien subjetivo, cuando supone la intervención del juicio humano. Para la métrica indirecta, se utiliza un método de cálculo (*CalculationMethod*), y adicionalmente especifica una fórmula o función (*Function*).

Posteriormente se lleva a cabo la medición de los atributos a partir de las métricas seleccionadas y se obtiene un valor de una medida (*measure*). Este valor no representa el nivel de satisfacción de un atributo, por lo que entonces se hace necesario definir una nueva correspondencia que producirá un valor de un indicador elemental.

8.3.4 Diseño y Ejecución de la Evaluación

En la etapa de evaluación, las métricas deben ser interpretadas a través de indicadores (*Indicator*) con el objetivo de evaluar o estimar el grado de conformidad que los requerimientos propuestos alcanzaron.

Los indicadores son la base para la interpretación de atributos y conceptos calculables. Se distinguen dos tipos de indicadores: los elementales y los globales. Un indicador elemental (*ElementaryIndicator*) es un indicador que no depende de otros indicadores para evaluar o estimar un atributo. Un indicador global (*GlobalIndicator*) es el que se deriva de otros indicadores para evaluar o estimar un concepto calculable.

Cada indicador elemental interpretará las métricas que cuantifican a cada atributo correspondiente en el árbol de requerimientos. Los indicadores contienen también una escala y una función o algoritmo a través del cual será posible interpretar el valor de la métrica, con ayuda también de un criterio de decisión (*DecisionCriteria*), que establecerá umbrales de aceptabilidad al valor obtenido. El valor del indicador global final es el que representa el grado de satisfacción global de los requerimientos para la necesidad de información establecida para el proyecto. La figura 8.4 corresponde

al modelo completo del componente de evaluación (sólo se describieron los elementos principales a los fines del alcance de esta tesis).

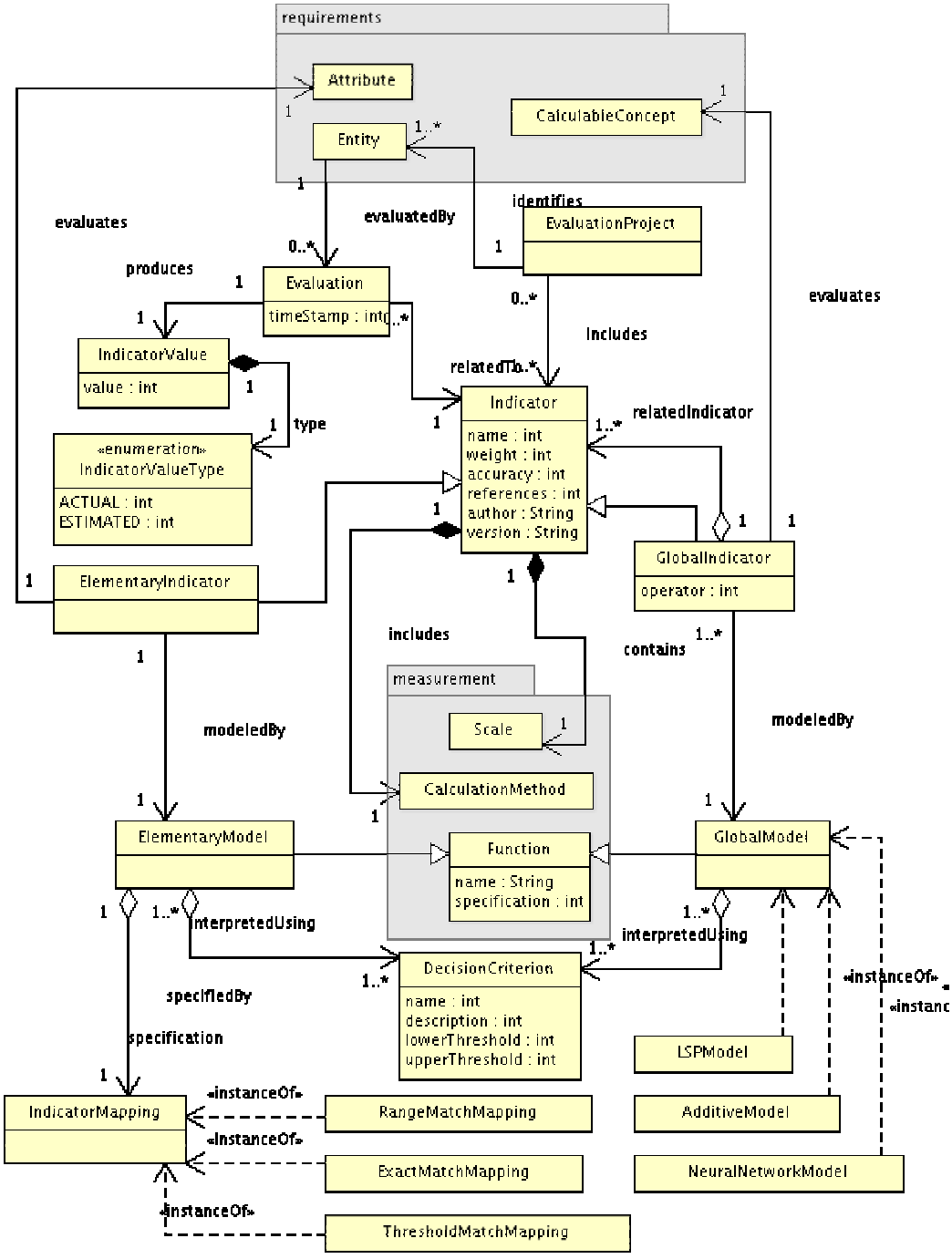


Figura 8.4. Modelo Conceptual de Evaluación

8.3.5 Análisis y Recomendación

Este componente (y su proceso asociado [Becker et al., 2009]) permite a los evaluadores realizar actividades de análisis de datos y comparación de las preferencias de calidad elementales y globales. Se generarán recomendaciones en base a los resultados obtenidos que sirvan para comprender y tomar decisiones de mejora.

8.4 Sistemas de Recomendación

Un sistema de recomendación es "*Aquel sistema que tiene como principal tarea seleccionar ciertos objetos de acuerdo a los requerimientos del usuario, dado que estos objetos están almacenados y caracterizados por sus atributos*" [Wang 1998]. Los sistemas de recomendación surgen a partir de la necesidad de poder suministrar a sus usuarios información relevante y personalizada sobre determinados objetos que pudieran ser de interés para ellos.

Tradicionalmente, los seres humanos han solucionado el problema de la búsqueda de información para resolver un problema mediante la petición de recomendaciones a quienes han considerado más expertos en la materia en cuestión, o bien lo han resuelto confiándose de aquellos objetos que a su juicio más se parecían a los que ellos buscaban, o bien, mediante la combinación de ambos métodos.

Los enfoques de recomendación se dividen en dos grandes categorías de acuerdo a si basan sus predicciones en un perfil de usuario construido en forma individual o en las opiniones de múltiples usuarios. Existe además, un tercer enfoque, fruto de la combinación de los anteriores. El primer caso se conoce como enfoque de recomendación basado en contenido. La segunda forma usual de recomendación se la conoce como filtrado colaborativo o social. Finalmente, la combinación de ambas se denomina enfoque híbrido [Adomavicius et al., 2004].

Un sistema de recomendación basado en el contenido es aquél en el que al usuario le será recomendado aquellos ítems similares a los que prefirió en el pasado. Es decir, prevé que para un usuario serán de interés ciertos objetos, si éstos son muy parecidos en su contenido a otros objetos que se sabe son del agrado del usuario en

cuestión. Son basados en contenido ya que el contenido de un ítem (por ejemplo una página Web) determina el interés potencial que un usuario puede tener en el mismo. Puede ejemplificarse dicha noción en un sistema que recomienda películas a un usuario. El sistema de recomendación basado en contenido tratará de entender los aspectos comunes entre las películas que el usuario ha calificado en el pasado (actores específicos, directores, géneros, tema, etc.). Luego, sólo las películas que tienen un alto grado de similitud con aquello que el usuario ha preferido antes, le serán recomendadas.

Un sistema de recomendación colaborativo es aquél en el que el usuario le será recomendado ítems que personas con preferencias similares optaron en el pasado. Es decir, se basa en el grado de similitud entre diferentes usuarios del sistema para realizar el proceso de recomendación. Por ejemplo, en el sistema que recomienda películas, para recomendar alguna a un usuario, el sistema de recomendación colaborativa trata de hallar los usuarios que tengan mayor similitud en la preferencia de películas. Luego, sólo aquellas películas que son más parecidas a las del usuario le serán recomendadas. Como ejemplo concreto de un sistema de recomendación colaborativo puede citarse el sistema de recomendación de libros de *Amazon*[8] *y Barnes&Nobles*[9].

Enfoques de recomendación híbridos son usualmente utilizados para paliar los problemas de los enfoques basados en contenidos y colaborativos. Los perfiles de usuario se mantienen según el análisis de los contenidos de los ítems, y directamente se comparan esos perfiles para determinar las similaridad entre usuarios para una recomendación colaborativa.

El sistema de recomendación presentado en esta tesis, se encuadra en esta última categoría. Por un lado, para realizar la recomendación tiene en cuenta el contenido de los casos pasados similares, es decir recomienda según el análisis de los contenidos de los ítems. No obstante eso, además se utilizan la colección entera de casos pasados de distintos usuarios y opiniones disponibles al momento de generar las predicciones para un usuario determinado, es decir, es a su vez un sistema de recomendación colaborativo.

El dominio de aplicación de los sistemas de recomendación es amplio y diverso: música, películas, libros, noticias, comercio electrónico, entre otros. En la siguiente

[8] www.amazon.com

[9] www.barnesandnoble.com

sección se muestra su aplicación al área de Ingeniería de Software y Web, a través de la realización del sistema de recomendación basado en casos para proyectos de medición y evaluación. Como ejemplo de aplicación de la MOBC, a dicho sistema de recomendación lo llamaremos C-INCAMI *Recommender*.

8.5 Sistema de Recomendación en Proyectos de Medición y Evaluación

Este sistema de recomendación C-INCAMI *Recommender*[10] tiene como objetivo servir de soporte a la herramienta *C-INCAMI_Tool* mediante la incorporación de un mecanismo de recomendación en forma proactiva usando el conocimiento almacenado en una MOBC.

Dicho sistema robustece a C-INCAMI_Tool principalmente en las primeras etapas de un proyecto de medición y evaluación, donde se requiere de la experticia en el diseño, y donde la disponibilidad de recomendaciones desde una base de conocimientos son de mayor utilidad. Nos estamos refiriendo a las etapas de especificación de requerimientos no funcionales y diseño de la medición de la metodología descripta en la sección 8.3.

En la etapa de especificación de requerimientos no funcionales, el sistema recomendará un árbol de requerimiento apropiado para las características del proyecto, basado en la similitud de casos anteriores guardados en la Memoria Organizacional, (en la subsección 8.5.2 se muestra con más detalle el mecanismo de CBR usado para atender a tal recomendación). En la etapa de diseño de la medición, el sistema recomienda las métricas apropiadas para la medición de cada atributo, también usando casos anteriores guardados. La funcionalidad de recomendación está basada en el mecanismo de comparación de casos, usando un SMOBC.

Luego de que la recomendación haya sido brindada al usuario, puede optar entre dos alternativas: o bien acordar la recomendación y por lo tanto, se guarda el proyecto en cuestión, o bien, introducir cierta modificación en la recomendación (por ejemplo, si considera que faltan atributos en el árbol de requerimientos) y entonces, modifica a su criterio los ítems deseados. Esta facilidad de modificación se contempla debido a que la

[10] Esta herramienta fue objetivo de la tesis para obtener el titulo de Ingeniera en Sistemas de María Belén Rivera [Rivera, 2009]

idea es que el sistema no intenta imponer una solución, sino más bien, y como su nombre lo indica, dar recomendaciones, que sirvan para mejores soluciones y enriquecer la base de conocimientos. El nuevo proyecto modificado, constituye en sí un nuevo caso y es guardado en la Memoria Organizacional para su uso futuro.

El sistema de recomendación, por otra parte, mantiene un catálogo de los nombres y URL de bases de conocimiento. Esto permitirá al sistema no sólo recomendar soluciones relacionadas a casos de proyectos de medición y evaluación, sino que también será posible utilizar el servicio para recomendar otros tipos de soluciones de casos diversos. La funcionalidad de este catálogo está representada por dos operaciones principales: el *Registro* de una nueva base de conocimiento basado en casos, y la *Búsqueda* de la meta-datos de cuáles son las bases de casos disponibles y los URL de los servicios que implementan.

8.5.1 Arquitectura del Sistema de Recomendación

En esta sección se muestra el diseño arquitectónico del sistema de recomendación *C-INCAMI Recommender*. La arquitectura propuesta es genérica, basada en servicios Web y desarrollada con tecnologías de Web semántica. Sigue el modelo arquitectónico distribuido en capas y tiene como componente principal el Sistema de MOBC descrito en el capítulo anterior. Por lo tanto, la arquitectura se estructura en tres grandes componentes que son el *Módulo de Recomendación,* el *Módulo de Catálogo de Bases de Conocimiento* y el *Módulo de la Memoria Organizacional Basada en Casos* (ver figura 8.5)

En primer lugar, el *Módulo de Recomendación* en sí, denominado *C-INCAMI Recommender*, el cual da soporte a las primeras etapas de un proyectos de medición y evaluación a saber: (i) la definición y especificación de requerimientos no funcionales; (ii) la especificación del contexto del proyecto; y (iii) el diseño de la medición. Durante estas etapas, el sistema en forma proactiva recomienda al usuario posibles árboles de requerimientos y posibles métricas, de acuerdo a las características del proyecto y su contexto. Para cumplir estas recomendaciones el módulo usa los servicios de la Memoria Organizacional Basada en Casos.

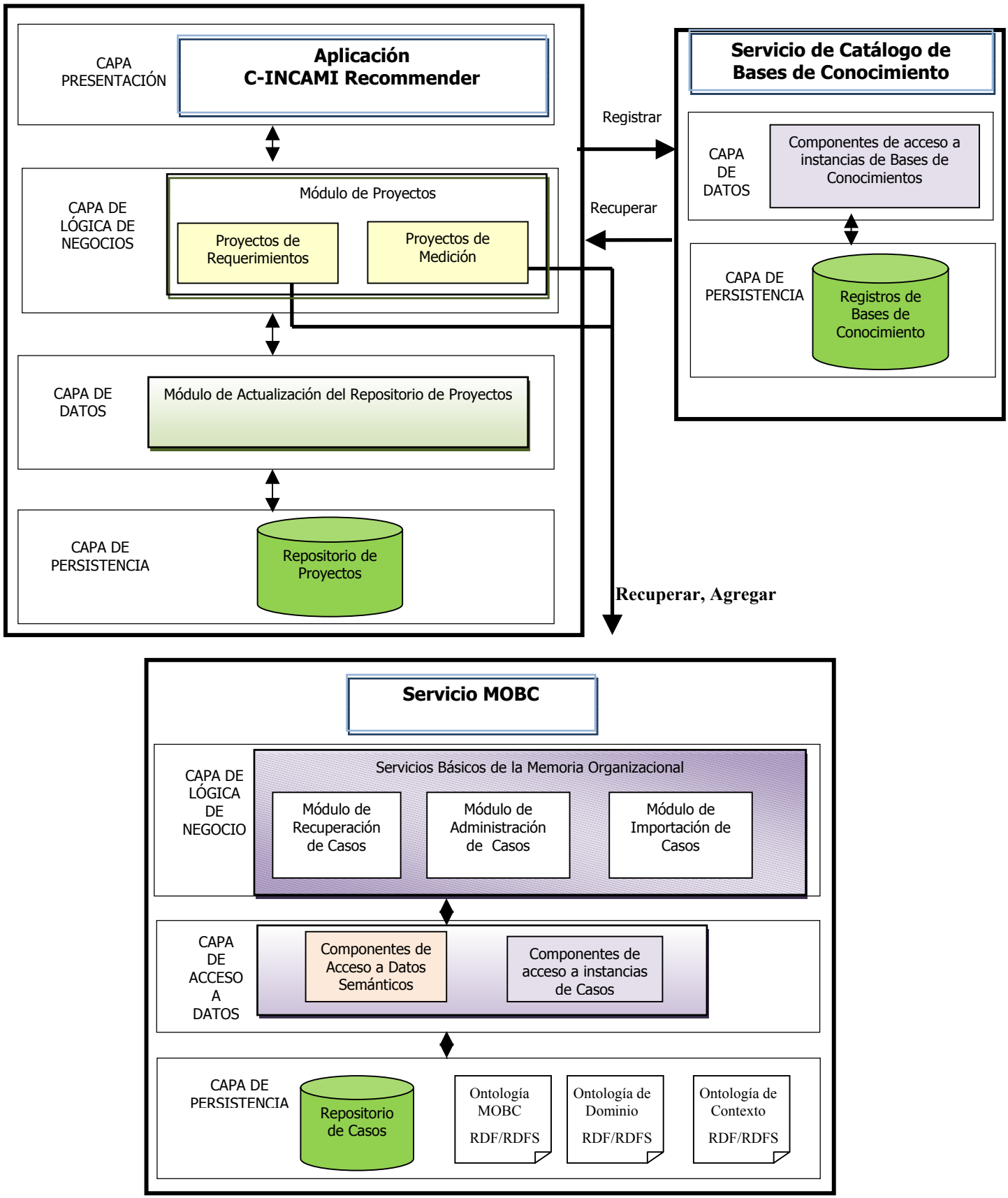


Figura 8.5. Arquitectura del Sistema de Recomendación para C-INCAMI

Segundo, el *Módulo de la Memoria Organizacional Basada en Casos*, que se encuentra distribuida en varias bases de conocimiento basadas en casos, cada una implementada como un servicio Web llamado Servicio MOBC. En este caso, se usan sólo dos bases de conocimientos, una base de conocimientos de árboles de requerimientos y otra base de conocimientos de métricas.

Por último, otro servicio Web, denominado *Servicio de Catálogo de Bases de Conocimiento,* mantiene un catálogo de los nombres y las URLs de todas las bases de conocimiento basadas en casos distribuidas.

8.5.1.1 Módulo de Recomendación

El Módulo de Recomendación es el principal componente desde el punto de vista de control. Constituye la aplicación que gestiona proyectos de requerimientos y de medición guiando al usuario en las primeras etapas del proceso de medición y evaluación en aseguramiento de calidad. Este módulo está organizado en cuatro capas, a seber:

- *Capa de Presentación*: interfaz de usuario. Permite al usuario evaluador crear proyectos de requerimientos o medición. Durante esta actividad, el usuario es asistido proactivamente con recomendaciones relacionadas a árboles de requerimientos y métricas.

- *Capa lógica de negocios*: recibe las peticiones de la capa de presentación, procesa la lógica de negocio basada en las peticiones y media en los accesos a los recursos tanto de la capa de datos, como del Servicio de MOBC. Esta capa, por lo tanto asiste a la capa de presentación en dos tipos de funcionalidades.

  Por un lado la funcionalidad de recomendación, para lo cuál usa los servicios de la MOBC. Cuando la capa de presentación detecta que el usuario está ingresando un nuevo proyecto de requerimientos, solicita recomendación para el diseño del árbol de requerimientos, que es resuelta por la capa de lógica de negocios, haciendo uso de la MOBC, lo mismo ocurre con la recomendación de métricas de los proyectos de medición.

También esta capa es responsable de la lógica para capturar el nuevo conocimiento generado en los proyectos de requerimientos y/o medición que se gestionan.

Por otro lado las funcionalidades propias de la aplicación en sí, relacionadas a la gestión de proyectos de requerimientos y de medición, para lo cuál usa los servicios ofrecidos por la Capa de Datos.

Para cumplir con todas estas funcionalidades, esta capa cuenta con dos módulos a saber:

- *Módulo de Proyectos de Requerimientos:* crea proyectos de requerimientos para un proyecto de medición y evaluación en particular. Se comunica con el módulo de recomendación a fin de solicitar recomendación de un árbol de requerimiento para el proyecto actual.

- *Módulo de Proyectos de Medición:* crea proyectos de medición para un proyecto de requerimientos seleccionado. Se comunica con el módulo de recomendación para solicitar recomendación de métricas para los atributos que conforman el árbol de requerimientos del proyecto en cuestión.

- *Capa de datos:* compuesta por los componentes que sirven para acceder a los datos de la aplicación.

  - *Componente de Actualización del repositorio de proyectos:* incorpora nuevos proyectos de requerimientos y medición al servidor de proyectos que utiliza la herramienta C-INCAMI_Tool. De esta manera, dicha aplicación puede continuar con la medición y evaluación de los proyectos creados en la herramienta C-INCAMI Recommender.

- *Capa de persistencia:* compuesta por los sistemas gestores de bases de datos.

- *Repositorio de proyectos*: mantiene información de proyectos de requerimientos y medición de una organización.

8.5.1.2 Módulo de Memoria Organizacional Basada en Casos

El Servicio de MOBC implementa los servicios básicos de la Memoria Organizacional Basada en Casos, que se describió en detalle en el capítulo anterior. En el caso particular de esta aplicación, administra dos bases de conocimiento basadas en casos a saber: una base de conocimiento relacionada a árboles de requerimientos, para proyectos de medición y evaluación, y otra base de conocimientos relacionada a métricas a usar para medir determinados atributos.

8.5.1.3 Módulo de Catálogo de Bases de Conocimientos

El *Servicio de Catálogo de Bases de Conocimiento* tiene como funcionalidad administrar un catálogo de bases de conocimiento basadas en casos. El catálogo registra la descripción y los URL (Localizador de recurso universal) de todas las bases de conocimiento basada en casos distribuidas en distintos nodos. Las dos funcionalidades principales definidas para este servicio son:

- *Registrar:* que permite a un proveedor (cliente del servicio) publicar una base de conocimiento basado en casos, para lo cual deberá proveer como entrada al servicio, el nombre de la base y el URL donde esté corriendo el servicio MOBC corespondiente.

- *Recuperar:* devuelve la meta-información de cuáles son las bases de casos disponibles y las URLs de los servicios que las implementan.

Para cumplir con todas estas funcionalidades, este componente está estructurado en dos capas a saber:

- *Capa de Datos:* componentes de accesos a instancias de bases de conocimientos.

- *Capa de Persistencia:* se almacenan los distintos tipos de bases de conocimiento.

8.5.2 Recomendación en la Etapa de Especificación de Requerimientos

El sistema de recomendación se basa en una MOBC que administra casos relacionados a proyectos de medición y evaluación, que están caracterizados con términos la ontología citada de Métricas e Indicadores [Martin et al., 2003] [Olsina et al., 2004], esta ontología incluye la definición de los términos *ProjectPurpose*, *CalculableConcept*, *ConceptModelType*, *Userview*, *EntityCategory*, *Entity* entre sus principales conceptos.

Para adecuar el motor de RBC de la MOBC a esta aplicación, se debió definir el modelo de similitud que relaciona los *Features* (Características) de los casos con términos del dominio, y cómo estos serán evaluados en relación a su similitud (como fue detallado en el capítulo 7, sección 7.3.1).

Tabla 8.1. Modelo de similitud para la Base de Conocimiento de Árboles de Requerimiento

Descripción	**Característica (*Feature*)**	**Función Similitud**	**Peso**
Propósito de la evaluación	*Purpose*	Compleja	0.1
Concepto calculable a evaluar, por ejemplo: calidad, costo, etc.	*CalculableConcept*	Exacta	0.3
Tipo de modelo de concepto, por ejemplo ISO o Propio	*ConceptModelType*	Exacta	0.05
Punto de vista del usuario	*Userview*	Exacta	0.15
Tipo de categoría de entidad a evaluar	*EntityCategory*	Exacta	0.25
Tipo de entidad a evaluar	*Entity*	Exacta	0.15

En la tabla 8.1 se muestra el modelo de similitud usado. Los nombres de las características aparecen en inglés porque deben coincidir con los términos de la ontología. Para cada *feature* que caracteriza a un caso, debemos establecer su peso de preponderancia relativa, y el tipo de función de similitud. Estas decisiones de diseño, las debe tomar un experto, teniendo en cuenta qué características se consideran más relevantes desde el punto de vista de la similitud, para evaluar en definitiva la similitud global de dos casos.

Las características *CalculableConcep*t, *ConceptModelType*, *Userview*, *EntityCategory* y *Entity*, tienen funciones de similitud tipo *Exact* cuyo cálculo es trivial, en cambio, para la característica *Purpose*, el cálculo de similitud no es trivial debiendo proporcionarse las medidas de similitud para cada par de valores posibles, como podemos observar en la tabla 8.2.

Tabla 8.2. Definición de similitud compleja para la característica *Purpose*

	Evaluate	**Estimate**	**Improve**
Evaluate	1	0.8	0.4
Estimate	0.8	1	0.3
Improve	0.4	0.3	1

Supongamos que un usuario necesita desarrollar un nuevo proyecto de medición y evaluación para *evaluar la eficiencia en desarrollo de cierto grupo de trabajo*. Debido a que no desea tener que diseñar completamente (desde cero) el árbol de requerimientos que identifique los atributos mensurables de la entidad, decide utilizar la herramienta *C-INCAMI Recommender*, para gestionar la creación del proyecto tomando ventajas tanto de la recomendación de un árbol de requerimientos como de las métricas para cuantificar cada atributo del mismo.

El primer paso que se debe realizar para un proyecto de medición y evaluación según el enfoque detallado anteriormente (sección 8.3) es el de definir los requerimientos no funcionales. Para esto, se deben realizar todas las actividades que intervienen en este proceso principal, a saber: establecer la necesidad de información,

especificar el contexto y seleccionar un modelo de concepto. La figura 8.6 muestra el flujo de estas actividades.

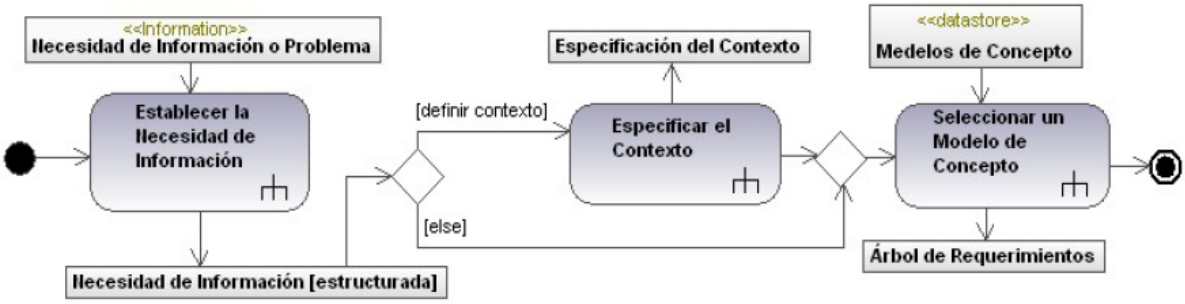


Figura 8.6. Flujo de actividades del proceso de Definición de Requerimientos no Funcionales (Fuente [Becker et al., 2009])

El establecimiento de la Necesidad de Información implica determinar el porqué de la medición y evaluación, identificando qué se quiere comprender, predecir o mejorar. Por lo tanto, al crear un nuevo proyecto se deberán especificar las características que hacen a la necesidad de información, como se muestra en el ejemplo que sigue :

- Propósito*: evaluar*
- Punto de vista*: usuario evaluador*
- Categoría de entidad del ente a evaluar*: grupo de personas*
- Ente en concreto que se va a evaluar*: grupo de desarrollo Motorola*
- Foco de la evaluación (determinar cuál será el concepto de más alto nivel que se desee evaluar): *eficiencia en desarrollo.*

De esta manera se completa la actividad de *Establecer la Necesidad de Información*, teniendo como resultado el propósito y el punto de vista de usuario, como así también el foco para la necesidad de información acordada, y la categoría de la entidad que será objeto de estudio. La figura 8.7 ilustra esta actividad en la herramienta presentada.

Finalizada la actividad de *Establecer la Necesidad de Información*, y según el modelo de la figura 8.6 se puede optar por realizar la actividad *Especificar el contexto* del proyecto en la cual se seleccionan propiedades que son relevantes al proyecto en cuestión. Si bien esta actividad es opcional, la especificación del contexto es importante, a la hora de obtener una mejor recomendación.

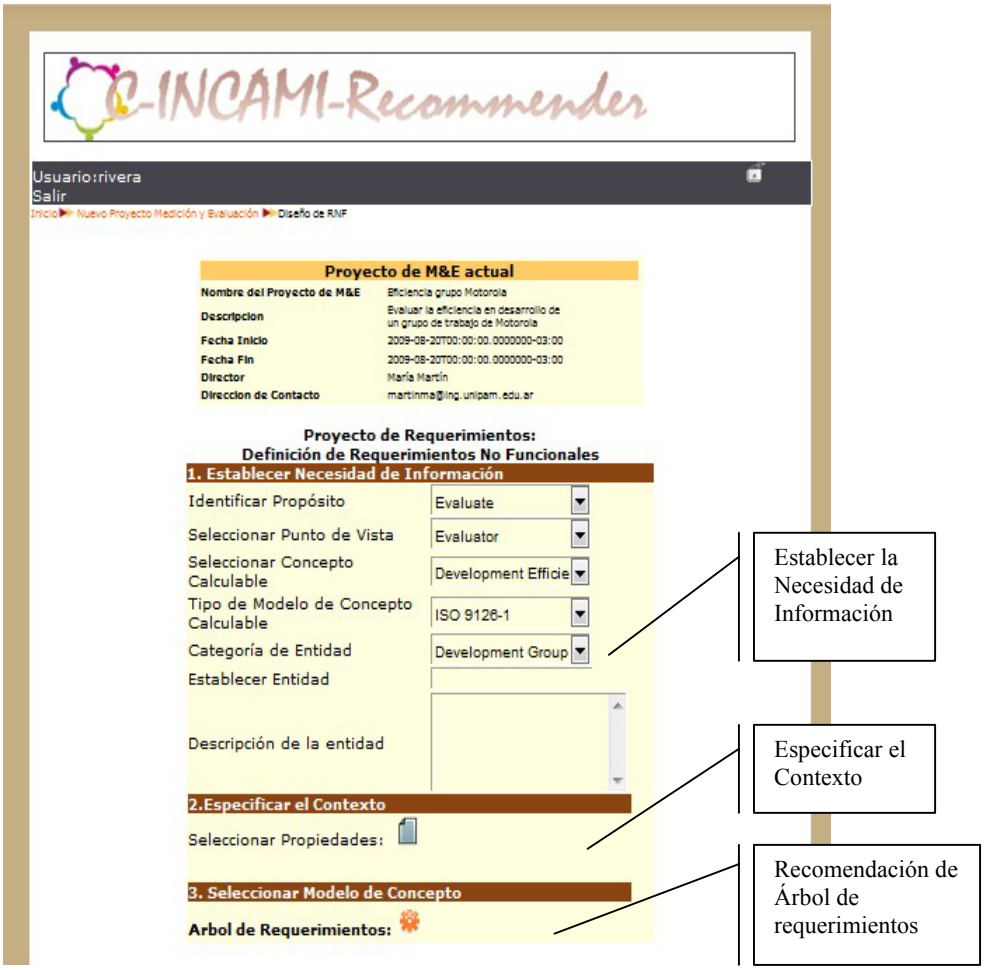


Figura 8.7 Creación de un proyecto de requerimientos con la herramienta *C-INCAMI Recommender*

La última actividad de esta etapa es de *Seleccionar un Modelo de Concepto* en la cual se debe especificar un árbol de requerimientos que relacione los conceptos y subconceptos establecidos para el foco identificado. En C-INCAMI *Recommender*, ésta última actividad es asistida por el sistema, el cuál recomienda un árbol de requerimientos sin tener la necesidad de diseñarlo completamente. Para llevar a cabo la recomendación, la aplicación interactúa con el servicio de MOBC, que basándose en la similitud con los casos anteriores, recupera el caso más similar, y como consecuencia la solución asociada (árbol de requerimientos) para el proyecto especificado. Esta solución

puede no conformar al usuario, el cuál puede editar tal árbol de requerimientos, proporcionando de esta manera nuevo conocimiento, que será capturado como un nuevo caso.

CASO 1: Obtención del Árbol de requerimientos para estimar la eficiencia en desarrollo de un programador

PROBLEMA:

Purpose	*Estimar*
CalculableConcept	*Eficiencia*
ConceptModelType	*Propio*
UserView	*Evaluador*
EntityCategory	*Persona*
Entity	*Programador X*

SOLUCIÓN:

1. *Eficiencia en Desarrollo*
   - 1.1. *Productividad de Desarrollo*
   - 1.2 *Cantidad de Módulos desarrollados*

Figura 8.8. Caso 1 recuperado de la base de conocimiento de árboles de requerimiento

CASO 2: Obtención del Árbol de requerimientos para evaluar la eficiencia en desarrollo del grupo de trabajo Gidis_Web

PROBLEMA:

Purpose	*Evaluar*
CalculableConcept	*Eficiencia*
ConceptModelType	*ISO*
UserView	*Evaluador*
EntityCategory	*Grupo de Desarrollo*
Entity	*Grupo de trabajo Gidis_Web*

SOLUCIÓN:

1. *Eficiencia en Desarrollo*
   - 1.1. *Productividad de Desarrollo*
   - 1.2 *Completitud del Desarrollo*
   - 1.3 *Duración del Desarrollo*

Figura 8.9. Caso 2 recuperado de la base de conocimiento de árboles de requerimiento

La descripción inicial del problema (los ítems de la necesidad de información, y del contexto) define la composición del nuevo caso que se usa para recuperar casos existentes en la base de conocimiento adecuada. Supongamos que la base de

conocimientos guarda los dos casos que se ilustran en las figuras 8.8 y figura 8.9 entre muchos otros. Las figuras de los casos muestran las variables de características del problema y la solución.

El servicio de MOBC, tan pronto recibe la petición, dirige su operatoria según los cuatro pasos del CBR ya explicados en la sección 4.5 del capítulo 4 (conocidos como "*las cuatro R*": *Recuperar, Reusar, Revisar y Registrar)*, a saber:

- *Recuperar* los casos almacenados que pertenecen a la misma base de conocimiento que el caso actual (base de conocimiento de árboles de requerimientos). Junto con los casos se recuperan los *ProblemFeatures* asociado a cada caso para la comparación. Cada *Feature* tiene establecido un peso de preponderancia relativa, y el tipo de función de similitud (ver tablas 8.1 y 8.2).

Tabla 8.3 Cálculo de similitud entre los casos almacenados y el nuevo caso

Característica (*Feature*)	**Caso 1**	**Caso 2**	**Caso Nuevo**	**Similitud Caso1-Nuevo**	**Similitud Caso 2-Nuevo**
Purpose	Estimar	Evaluar	Evaluar	0.8	1
CalculableConcept	Eficiencia	Eficiencia	Eficiencia	1	1
ConceptModelType	Propio	ISO	ISO	0	1
Userview	Evaluador	Evaluador	Evaluador	1	1
EntityCategory	Persona	Grupo de desarrollo	Grupo de desarrollo	0	1
Entity	Programador X	Grupo_Gidis	Grupo "Motorola"	0	0

- *Reusar* la información y conocimiento en los casos recuperados para resolver el problema. El caso que se recupera se compara con el nuevo caso, a través del reuso de información, basándose en la similitud entre ambos. La tabla 8.3 muestra el cálculo de similitud de cada característica del nuevo caso,

comparado a los recuperados (en esta situación, sólo se ejemplifica la recuperación de los dos casos de las figuras 8.7 y 8.8). También se calculan las funciones de similitud global que dan como resultado los valores de similitud que se presentan en la tabla 8.4, usando la ecuación 6.1.

Tabla 8.4 Resultados de la comparación de los dos casos

Similitud (Caso 1, Caso Nuevo)	0.1x0.8+0.3x1+0.05x0+0.15x1+0.25x0+0.15x0=0.53
Similitud (Caso 2, Caso Nuevo)	**0.1x0.1+0.3x1+0.05x1+0.15x1+0.25x1+0.15x0=0.85**

- *Revisar* la solución propuesta. Aquel caso cuyo resultado de la función de similitud haya sido el mayor, significa que es el más parecido al caso nuevo y por lo tanto se debe recomendar el árbol de requerimientos solución de dicho caso. Sin embargo, esta solución se revisa para confirmar que puede ser aplicada, es decir, que en situaciones anteriores el resultado de aplicar dicha solución fue exitoso. Es decir, la herramienta no sólo analiza la solución del caso más semejante para recomendar, sino que también tiene en cuenta la valoración del resultado obtenido al aplicarla, de manera de no repetir errores si el resultado no fue positivo. A tales efecto se considera que el resultado es exitoso si su valoración es superior o igual a 8 en un rango de 1-10 (1 fracaso, 10 éxito total). Para la situación planteada, y según se observa en la tabla 8.4, el caso más semejante es el caso 2 (puntaje final 0.85) y suponiendo que el valor de resultado recuperado de la memoria para dicho caso, es mayor que 8, se devuelve la solución del caso 2.

- *Registrar* o almacenar la solución. Luego que la solución haya sido analizada por el usuario, y tan pronto éste la confirme como parte de su proyecto o le introduzca mejoras, el servicio de recomendación guarda el caso nuevo, con su problema y solución de manera que pueda ser útil para resolver problemas similares futuros.

8.5.3 Recomendación en la Etapa de Diseño de la Medición

Una vez especificado el árbol de requerimientos quedan finalizadas todas las actividades del proceso de *Definición de Requerimientos no Funcionales y Especificación del Contexto del Proyecto*. La siguiente actividad en el proceso de medición y evaluación es la de *Diseño de la Medición*, en la cual se identifican las métricas que se utilizarán en el proceso *Implementación de la Medición* y obtener así, las medidas de los atributos de la entidad a evaluar.

La herramienta *C-INCAMI Recommender* habilita al usuario a identificar y asignar las métricas más apropiadas para cada atributo del árbol de requerimientos seleccionándolas de un repositorio de métricas. En esta etapa la aplicación guía al usuario sugiriéndole una métrica para cada atributo. Nuevamente la aplicación interactúa con el servicio de MOBC para llevar a cabo la recomendación. Al igual de lo que ocurre con recomendaciones de árboles de requerimientos, se llevan a cabo los mismos pasos, sólo que ahora la base de conocimientos que se consulta es la de métricas, por lo tanto se recuperarán casos cuyas soluciones son métricas.

Tabla 8.5 Modelo de similitud para la base de casos de métricas

Descripción	**Característica(Feature)**	**Función Similitud**	**Peso**
Propósito de la evaluación	Purpose	Compleja	0.1
Concepto calculable a evaluar, por ejemplo: calidad, costo, etc.	CalculableConcept	Exacta	0.03
Tipo de modelo de concepto, por ejemplo ISO o Propio	ConceptModelType	Exacta	0.02
Tipo de categoría de entidad a evaluar	EntityCategory	Exacta	0.05
Tipo de entidad a evaluar	Entity	Exacta	0.1
Atributo	Attribute	Exacta	0.7

La tabla 8.5 muestra las características a comparar para la base de casos de métricas, con sus respectivos pesos y funciones de similitud. La característica *Attribute* tiene un peso significativamente mayor que el resto de las características debido a que las métricas que se recomiendan deben ser aquellas que cuantifican al atributo en la ontología de métricas e indicadores, por lo tanto, deben coincidir.

Supongamos que la base de conocimientos basada en casos relacionada a métricas tiene almacenados, entre otros, los casos que muestran las figuras 8.10, 8.11 y 8.12

CASO 1: Obtención de una métrica para el atributo *Productividad de Desarrollo*

PROBLEMA:

Purpose	*Evaluar*
CalculableConcept	*Eficiencia*
ConceptModelType	*ISO*
EntityCategory	*Grupo de Desarrollo*
Entity	*Grupo de trabajo Gidis_Web*
Attribute	*Productividad de Desarrollo*

SOLUCIÓN: ***Cantidad de horas de desarrollo***

Figura 8.10 Caso 1 recuperado de base de conocimiento de métricas

CASO 2: Obtención de una métrica para el atributo *Completitud de Desarrollo*

PROBLEMA:

Purpose	*Evaluar*
CalculableConcept	*Eficiencia*
ConceptModelType	*ISO*
EntityCategory	*Grupo de Desarrollo*
Entity	*Grupo de trabajo Gidis_Web*
Attribute	*Completitud de Desarrollo*

SOLUCIÓN: ***Cantidad de módulos completados***

Figura 8.11 Caso 2 recuperado de base de conocimiento de métricas

CASO 3: Obtención de una métrica para el atributo *Enlaces Internos Rotos*

PROBLEMA:

Purpose	*Evaluar*
CalculableConcept	*Confiabilidad de enlaces*
ConceptModelType	*ISO*
EntityCategory	*Producto Software*
Entity	*Sitio web unlpam*
Attribute	*Enlaces Internos Rotos*

SOLUCIÓN: ***Cantidad de Enlaces Internos Rotos***

Figura 8.12 Caso 3 recuperado de base de conocimiento de métricas

Consideremos que el usuario desea establecer una métrica para el atributo *Productividad de Desarrollo.* La composición del caso para este ejemplo es la siguiente:

- Propósito: *Evaluar*
- Concepto Calculable: *Eficiencia*
- Tipo de Modelo de Concepto: *ISO*
- Categoría de Entidad: *Grupo de desarrollo*
- Entidad: Grupo de Trabajo *"Grupo Motorola"*
- Atributo: *Productividad de Desarrollo*

Tabla 8.6. Similitud entre los casos almacenados de métricas y el nuevo caso

Característica	Caso 1	Caso 2	Caso 3	Caso Nuevo	Similitud Caso1-Nuevo	Similitud Caso 2-Nuevo	Similitud Caso 3-Nuevo
Purpose	Evaluar	Evaluar	Evaluar	Evaluar	1	1	1
CalculableConcept	Eficiencia	Eficiencia	Confiabilidad de enlaces	Eficiencia	1	1	0
ConceptModelType	ISO	ISO	ISO	ISO	1	1	1
EntityCategory	Grupo de desarrollo	Grupo de desarrollo	Producto Software	Grupo de desarrollo	1	1	0
Entity	Grupo de trabajo "Gidis Web"	Grupo de trabajo "GidisWeb"	Sitio Web "UNLPam"	Grupo de trabajo "Motorola"	0	0	0
Attribute	Productividad de desarrollo	Completitud de desarrollo	Enlaces Internos Rotos	Productividad de desarollo	1	0	0

En esta situación, la aplicación invocará al servicio de MOBC para recomendar al usuario la métrica adecuada. El servicio de MOBC llevará a cabo la comparación que se muestra en la tabla 8.6. Luego, computa el resultado obtenido de la aplicación de la función de similitud a todas las características a evaluar, obteniendo los resultados globales que detallan la tabla 8.7.

Tabla 8.7. Resultados de la comparación para casos de métricas

Similitud (Caso 1, Caso Nuevo)	**0.1x1+0.03x1+0.02x1+0.05x1+0.1x0+0.7x1=0.9**
Similitud (Caso 2, Caso Nuevo)	0.1x1+0.03x1+0.02x1+0.05x1+0.1x0+0.7x0=0.2
Similitud (Caso 3, Caso Nuevo)	0.1x1+0.03x0+0.02x1+0.05x0+0.1x0+0.7x0=0.12

Se observa en la misma que el caso 1 es el más semejante al nuevo caso. Luego de revisar el valor del resultado de dicho caso, corroborando un valor adecuado, se devuelve como solución, la métrica del caso 1.

Cuando finaliza el proceso del diseño de métricas, se debe continuar con la ejecución de la medición y la evaluación. Estas actividades aún no están implementadas en C-INCAMI Recommender, por lo que, al finalizar con el diseño de la medición, el usuario tiene la posibilidad de continuar el proceso de medición y evaluación con la herramienta C-INCAMI_Tool.

8.6 Conclusiones

Administrar el conocimiento organizacional representa una ventaja clave en toda empresa de software. Para que dicha administración sea eficiente es importante contar con una Memoria Organizacional que de soporte a la captura, estructuración, reuso, procesamiento y diseminación del conocimiento creado por sus empleados. La utilidad de una Memoria Organizacional radica en que el conocimiento que almacena pueda ser fácilmente compartido entre sistemas dentro de la organización o entre organizaciones colaborativas.

Particularmente en el área de aseguramiento de la calidad y como consecuencia la medición y evaluación, la administración en forma consistente y eficiente del conocimiento relacionado a proyectos de medición y evaluación es muy importante para la toma de decisiones y facilita la recomendación en nuevos proyectos, tomando ventaja de las experiencias y lecciones aprendidas.

En este capítulo se mostró cómo se puede implementar una herramienta de recomendación basada en casos, como soporte para C-INCAMI. El prototipo creado ha permitido poner en práctica los desarrollos teóricos sobre la ontología de MOBC definida en el Capítulo 6 y la Memoria Organizacional Basada en Casos descripta en el Capítulo 7. Además, contribuye con la difusión de la importancia de administrar en forma consistente y eficiente el conocimiento presente en cualquier área de una organización, permitiendo la recuperación de experiencias pasadas exitosas y la recomendación de soluciones que sirvan para la toma de decisiones y también, para el aprendizaje y la mejora continua.

Capítulo 9 - Conclusiones y Trabajos Futuros

9.1 Conclusiones

En esta tesis se presentó el desarrollo de una Memoria Organizacional Basada en Ontologías y Casos, que fue diseñada y utilizada para su aplicación a un sistema de recomendación en proyectos de medición y evaluación en aseguramiento de calidad, como prueba de los resultados de investigación.

Para el diseño de dicha Memoria Organizacional y sistema de recomendación se debieron integrar conocimientos y tecnologías de distintas áreas de investigación, como son el área de Ontologías, de Gestión del Conocimiento, Razonamiento Basado en Casos, Ingeniería de Software, Web Semántica, entre otras. Como consecuencia, los aportes realizados son interdisciplinarios.

Para una mejor síntesis de las conclusiones de esta tesis, a continuación se analizan los resultados obtenidos teniendo en cuenta las principales contribuciones realizadas en las distintas áreas.

9.1.1 Ontología de Memoria Organizacional Basada en Casos

La gestión eficaz del conocimiento en cualquier empresa, necesita de una Memoria Organizacional bien estructurada y formalmente especificada para poder administrar en forma consistente y automatizada todo el conocimiento presente en la organización.

Observando el volumen cada vez mayor y heterogéneo de información relacionada a Memoria Organizacional, y teniendo en cuenta su importancia como base fundamental para las actividades de Gestión del Conocimiento y apoyo a la recomendación, se puede ver la necesidad de disponer de una Ontología de Memoria Organizacional Basada en Casos que proporcione una base conceptual consistente en dicho dominio.

En el capítulo 6 se especificó la Ontología de MOBC, resultante de un arduo trabajo de consenso entre profesionales e investigadores del área de Ingeniería de Software y Gestión del Conocimiento. Destacamos, además, la importancia que tiene contar con una ontología para el ámbito de Memoria Organizacional Basada en Casos, ya que provee una base conceptual consensuada, para compartir, reusar y organizar el conocimiento relacionado a experiencias y lecciones aprendidas en una organización, y en particular en empresas dedicadas al desarrollo de software.

La ontología de MOBC, combina aspectos de Memoria Organizacional con Razonamiento Basado en Casos para representar cada ítem de conocimiento informal. La estructuración del conocimiento informal en casos, facilita la captura, recuperación, transferencia, y reuso del conocimiento a través de su procesamiento automático. Además, evita la dispersión de la experticia de los miembros de una organización, guardando el conocimiento de todos los expertos en diversos casos, y permite la continua evolución de la memoria gracias a la adición progresiva de nuevos conocimientos.

Si bien la idea de aplicar métodos de Razonamiento Basado en Casos para representar conocimiento de experiencias exitosas no es novedosa [Weber et al., 2001], y los beneficios derivados de la aplicación de dichos métodos son bien conocidos, es también notoria la falta de consenso en los conceptos y terminología empleados tanto sea en Gestión del Conocimiento, como en áreas de RBC.

También es importante señalar, que si bien existe un estándar de gran utilidad relacionado a Gestión del Conocimiento [AS5037, 2005], y numerosos trabajos de investigación en las áreas de Gestión del Conocimiento, Memoria Organizacional, y RBC, donde se definen marcos conceptuales, y la terminología usada, éstos no constituyen en sí mismos una ontología formal o semiformal.

La ontología propuesta, se usó como base para el diseño y construcción de un sistema de MOBC, que permite explorar, reusar y compartir conocimiento relacionado a distintas áreas, si se instancia dicho sistema asociándole la ontología de dominio correspondiente.

9.1.2 Sistema de Memoria Organizacional Basada en Casos Distribuida

El Sistema de MOBC distribuida que se propuso en esta tesis (capítulo 7), permite el almacenamiento persistente del conocimiento informal obtenido en experiencias y problemas resueltos para su posterior consulta y aplicación en recomendación, toma de decisiones y/o resolución de nuevos problemas.

La contribución de este sistema de Memoria Organizacional respecto a otros ya propuestos en la literatura, es que al estar basado en ontologías, puede ser fácilmente instanciado a través del aporte de una ontología de dominio que provea un lenguaje específico del ámbito de aplicación del conocimiento, donde los tipos de atributos que caracterizan a un problema y a una solución estén formalmente definidos. Otra característica que le da flexibilidad al diseño, es la representación del modelo de similitud, que debe ser especificado para cada base de casos, permitiendo adecuar el motor de RBC al tipo de conocimiento que almacena la base.

Además, para garantizar la generalidad (es decir que el SMOBC se pueda aplicar a distintos dominios de conocimiento), ha sido diseñado sobre la base de dos niveles distintos de ontologías. Por un lado, en un alto nivel de abstracción la ontología genérica de MOBC, y por otro lado, a un nivel más específico, para caracterizar los casos de acuerdo al conocimiento de un dominio determinado y su contexto, se deberán definir las ontologías de dominio y de contexto respectivamente.

Generalmente la administración y el accionar de una organización no está centralizada. Como consecuencia, existen dificultades para el desarrollo de una única Memoria Organizacional que resguarde todo el conocimiento presente en la misma, dada la dispersión geográfica que caracteriza a las empresas modernas. Por estas razones, el sistema de MOBC está basado en redes de nodos de Memoria Organizacional interconectadas, que facilitan la transferencia de conocimiento entre las distintas dependencias de la organización de manera que éste pueda ser transmitido y compartido eficazmente. Además existen ventajas en relación a la gestión distribuida que tienen que ver con la posibilidad de manejar múltiples bases del conocimiento locales de forma efectiva, que luego podrán ser compartidas y difundidas dentro de la organización.

Finalmente, pero no menos importante, en el capítulo 7 (sección 7.6) se analizaron las capacidades potenciales que brinda el SMOBC propuesto en el desarrollo de sistemas de Gestión del Conocimiento organizacional, tanto desde el punto de vista de su *dimensión semántica* (es decir capacidad de razonamiento) como desde el punto de vista de su *dimensión web* (es decir su capacidad de compartir y diseminar el conocimiento distribuido en la Web). El análisis que se realizó, estuvo basado en el "*marco de clasificación en Web Semántica para las aplicaciones de Gestión del Conocimiento basados en ontologías*" definido por Mika y colaboradores [Mika et al., 2004]. Se mostró que el SMOBC propuesto en esta tesis provee nivel de Razonamiento teniendo en cuenta la dimensión **Semántica** y nivel de *Aplicaciones Distribuidas* teniendo en cuenta la dimensión **Web.**

9.1.3 Sistema de Recomendación en Proyectos de Medición y Evaluación en Aseguramiento de Calidad

Para mostrar los beneficios de la MOBC en la Gestión del Conocimiento (buenas prácticas y lecciones aprendidas), hemos presentado un caso de estudio: esto es, su aplicación a un sistema de recomendación en etapas de diseño de proyectos de medición y evaluación de calidad de productos software y Web. Mediante un prototipo implementado realizado para tal fin, llamado *C-INCAMI Recomender,* se muestra su flexible integración con herramientas existentes (en este caso la integración al marco de medición y evaluación C-INCAMI y su herramienta C-INCAMI-Tool). Se muestra además, la explotación de las bases de conocimiento por parte de herramientas automáticas y cómo este enfoque puede dar soporte a las características de interoperabilidad, generalidad y procesamiento automático tan importantes a la hora de reusar el conocimiento organizacional.

Si bien la literatura general en Gestión del Conocimiento contiene muchos ejemplos de sistemas exitosos usados en compañías relacionadas a tecnologías de la información, muy pocos se relacionan al área de Ingeniería de Software y hasta donde conocemos, ninguno de esos ejemplos son específicos para proyectos de medición y evaluación en aseguramiento de calidad de software. Además, según lo expresa Bjornson, a pesar de que hay una comunidad de investigadores en Gestión del

Conocimiento en Ingeniería de Software, sus trabajos están lejos de la principal tendencia en Gestión del Conocimiento, ya que tratan principalmente del almacenamiento y recuperación del conocimiento, mientras que los temas como la creación, transferencia y aplicación necesitarían mayor atención [Bjornson, 2007].

El capítulo 8 muestra cómo el sistema de recomendación implementado puede dar soporte a las principales actividades de Gestión del Conocimiento como son la adquisición, resguardo, aplicación y distribución de forma consistente y automatizada para el área de medición y evaluación en aseguramiento de calidad. Además, contribuye con la difusión de la importancia de administrar en forma consistente y eficiente el conocimiento presente en cualquier área de una organización, permitiendo la recuperación de experiencias pasadas exitosas y la recomendación de soluciones que sirvan para la toma de decisiones y también, para el aprendizaje y la mejora continua.

9.2 Trabajos Futuros

Una línea de avance es la especificación de una ontología de contexto para el área de medición y evaluación en Ingeniería de Software. Dicha ontología podrá ser usada en conjunto con el sistema de MOBC para caracterizar el contexto de cada ítem de conocimiento en dicho dominio. Hasta el momento, en las pruebas de laboratorio realizadas con la herramienta *C-INCAMI Recomender*, no se especifica el contexto de cada proyecto de medición y evaluación. Por lo tanto tampoco es tenido en cuenta por el motor de RBC al momento de reusar el conocimiento para realizar una recomendación. En las memorias organizacionales basada en casos, la relevancia de la información de experiencias similares, está muy ligada al contexto en que es válido aplicar dicha experiencia, de ahí la importancia de representar consistentemente el contexto de los items de conocimiento.

Actualmente se está avanzando en una línea de investigación de modelado de información de contexto y algoritmos para el cálculo de similitud de contexto, llevada a cabo por otro integrante de nuestro grupo [Molina et al., 2010]. Un trabajo fututo en esta dirección será aprovechar los resultados de esta investigación con el fin de incorporarlos a la MOBC.

Otro trabajo futuro tendiente a mejorar la MOBC es dotarla de algoritmos de RBC distribuidos. Actualmente, si bien la MOBC está diseñada para ser usada en forma distribuida, la distribución está relacionada a las bases de conocimiento, que pueden estar dispersas sobre varios nodos de MOBC distribuidos en una red. Sin embargo, en cada nodo de MOBC el motor de RBC funciona localmente.

Referencias

[Aamodt et al., 1994] Aamodt, A.; Plaza E. (1994) “Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Comm. IOS Press, Vol. 7:1, pp.39-59.

[Abasolo et al., 2002] Abasolo, C.; Plaza, E.; Arcos, J.L. (2002). “Components for Case-based Reasoning Systems”, In Topics in Artificial Intelligence, Lecture Notes in Artificial Intelligence, Vol. 2504, p. 1-12.

[Abdrabou et al., 2008] Abdrabou, E.; Salem, A. (2008). “Case-based reasoning tools from shells to object-oriented frameworks”, Proceedings of the 12th WSEAS international conference on Computers, p.781-786, July 23-25, Heraklion, Greece.

[Abecker et al., 1998] Abecker, A.; Bernardi, A.; Hinkelmann, K.; Kühn, O.; Sintek, M. (1998). “Toward a Technology for Organizational Memories”. IEEE Intelligent Systems 13(3): 40-48.

[Abecker et al., 1999] Abecker, A.; Bernardi, A.; Sintek, M.. (1999). “Developing a knowledge management technology: An en-compassing view on know more”. In Proceedings of the 8th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enter-prises, pages 216–222.

[Abecker et al., 2000] Abecker, A., Bernardi, A., Hinkelmann, K., K¨uhn, O., and Sintek, M.. (2000). “Context-aware, proactive delivery of task-specific knowledge: The knowmore project”. Int. Journal on Information Systems Frontiers, 2(3/4):139–162.

[Abidi et al., 2005] Abidi, S.; Cheah, Y.; Curran, J. (2005) “A Knowledge Creation Info-Structure to Acquire and Crystallize the Tacit Knowledge of Health-Care Experts”. IEEE Transactions on Information Technology in Biomedicine., vol. 9, núm. 2, p.193-204.

[Abril et al., 2009] Abril, R.; Müller, R.. (2009). “Lessons Learned as Organizational Project Memories”. Chapter’s book in Building Organizational Memories: Will You Know What You Knew?. John Girard (Ed). IGI Global.

[Ackerman, 1994] Ackerman, M. (1994). “Definitional and Contextual Issues in Organizational and Group Memories”. Proceedings of the Twenty-seventh IEEE Hawaii International Conference of System Sciences (HICSS 94) : 191-200.

[Ackerman et al., 1990] Ackerman, M.; Malone, T. (1990), "Answer Garden: a tool for growing organisational memory", Proceedings of the Conference on Office Information Systems, ACM Press. Cambridge, M.A. p 31-39.

[Ackerman et al., 2000] Ackerman, M.; Halverson, C.. (2000). "Reexamining Organizational Memory", Communications of the ACM, vol. 43, no. 1, pp. 59-64.

[Alavi, 1997]Alavi, M. (1997). KPMG Peat Marwick U.S.: one giant brain, 9-397-108, Harvard Business School (Case), Boston, MA.

[Alavi et al., 2001]Alavi, M.; Leidner, D. (2001). "Review: knowledge management and knowledge management systems: conceptual foundations and research issues". MIS Quarterly, v. 25 n. 1, p. 107-136.

[Alexaki, 2001] Alexaki, S.; Christophides, V.; Karvounarakis; Plexousakis, D.; Tolle K. (2001). "The RDFSuite: Managing Voluminous RDF Description Bases". Technical report, Institute of Computer Science, FORTH, Heraklion, Greece.

[Ale et al., 2006] Ale, M. A.; Chiotti, O.; Galli, M.. (2006). "Gestión del Conocimiento Empresarial en Organizaciones Emergentes: Un Enfoque basado en Ontologías". En los anales del SSI 2006. 35 Jornadas Argentinas de Informática e Investigación Operativa (JAIIO). Mendoza, Argentina.

[Ale et al., 2007] Ale, M., Gerarduzzi, C., Chiotti, O., & Galli, M. (2007). "Onto-Dom: A Question-Answering Ontology-Based Strategy for Heterogeneous Knowledge Sources". In Proceedings of the 6th Conference on Software Engineering and Knowledge Engineering (JIISIC'07), Lima: Peru. pp. 79-86.

[Ale, 2009] Ale, A. "Modelo Conceptual De Gestión Del Conocimiento Empresarial". (2009). Tesis doctoral, Universidad Tecnológica Nacional, Santa Fé. Argentina.

[Althoff et al., 1995]. Althoff, K.; Auriol, E.; Barletta, R.; Manago, M.. (1995). "A review of industrial case-based reasoning tools,". In Goodall, A. (Ed.), An AI Perspectives Report. Al intelligence, Oxford, UK, Tech. Rep.

[Althoff et al., 2005] Althoff, K.; Weber, R..(2005). "Knowledge management in case-based reasoning", The Knowledge Engineering Review, v.20 n.3, p.305-310.

[Aparicio et al., 2005] Aparicio, A.; Farias, O.; dos Santos, N. (2005). "Applying Ontologies in the Integration of Heterogeneous Relational Databases". En Conferences in Research and Practice in Information Tecnology. Disponible en <http://crpit.com/confpapers /CRPITV58Aparicio.pdf>. Último acceso: Noviembre 2008.

[Arcos, 1997] J.L.Arcos. "The Noos representation language". (1997). PhD thesis, Universitat Politecnica de Catalunya.

[AS5037, 2005] Australian Standard (2005) “Knowledge Management - A guide”.

[AskMe, 2009] ASKME REALCOM CORPORATION. (2009). Disponible en URL: <http://www. realcom-inc.com/>. Último acceso en: Septiembre de 2009.

[Aurum, 2008] Aurum, A., Daneshgar, F., Ward, J. (2008). “Investigating Knowledge Management practices in software development organizations - An Australian experience”. Information and Software Technology 50, pp. 511–533

[Baclawski, 2002] Baclawski, K.; Kokar, M.; Kogut, P.; Hart, L.; Smith, J.; Holmes, W.; Letkowski, J.; Aronson, M. and Emery, P. (2002). "Extending the UML for Ontology Development", SOSYM 2002, Sofware System Model. 1: 1-15, Springer-Verlag.

[Bareiss, 1989] Bareiss, R. (1989). “Exemplar-Based Knowledge Acquisition: A Unified Approach to Concept Representation, Classification, and Learning”, Academic Press, San Diego, CA.

[Bareiss, 1991] Bareiss R. Editor. Proceedings from the Case-Based Reasoning Workshop, Washington D.C., May 8-10, (1991). Sponsored by DARPA. Morgan Kaufmann.

[Bartsch, 2004] Bartsch, S. (2004). “Knowledge Representation Language and Conceptualization”. Knowledge Representation Handouts. Disponible en <http:// schwicky.net/projects/2004/language_and_conceptualization/20040208-Knowledge _Representation-Handouts.pdf>. Último acceso: Novimbre de 2009.

[Basili et al., 1994] Basili, V., Caldeira, G., Rombach, H. (1994). The experience factory, in J. Marciniak (Ed.) Encyclopedia of software engineering, New York, J. Wiley & Sons, pp. 469-476.

[Basili et al., 1997] Basili, V.; McGarry, F.. (1997). “The Experience Factory: How to Build and Run One”. Proceedings 19th Int’l Conference on Software Engineering, (ICSE'97), Boston, Massachusetts. Pp. 17-23.

[Basili et al., 2001] Basili, V., Lindvall, M., Costa, P. (2001). “Implementing the Experience Factory as a set of experience bases”, Proceedings of the 13th International Conference on Software Engineering and Knowledge Engineering, , pp. 102-109.

[Bateman et al., 1995] Bateman, J. A.; Henschel R.; Rinaldi F. (1995). “Generalized Upper Model 2.0: documentation”,Technical Report, GMD/IPSI, Darmstadt, Germany.

[Beck et al., 2002] Beck, H.; Pinto, S. (2002). "Overview of Approach, Methodologies, Standards, and Tools for Ontologies". Disponible en http://it.ifas.ufl.edu/AOS /BackgroundPaper/.

[Bechhofer et al., 2004] Bechhofer S., van Harmelen F., Hendler J., Horrocks I., McGuinness D., Patel-Schneider P. and Stein L. (2004). "OWL Web Ontology Language Reference", W3C Recommendation, 10 February 2004, http://www.w3.org/TR/2004/ REC-owl-ref-20040210/.

[Becker et al., 2009] Becker, P; Molina, H; Olsina, L. (2009). "Integrando Proceso y Marco de Medición y Evaluación", En Actas del XII Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software (CIbSE former IDEAS09), Medellín, Colombia, pp. 259-266, ISBN 978-958-44-5028-9.

[Bellinger, 2004] Bellinger, G.. (2004). "Knowledge Management—Emerging Perspectives". Disponible en http://www.systems-thinking.org/kmgmt/kmgmt.htm. Último acceso: Novimbre de 2009.

[Bello-Tomás et al., 2004] Bello-Tomás, J.; González-Calero, P.; Díaz-Agudo, B.. (2004). "JColibri: An Object-Oriented Framework for Building CBR Systems". In Advances in Case-Based Reasoning, Lecture Notes in Computer Science. Springer Berlin/ Heidelberg, Vol. 3155/2004, p. 32-46.

[Benjamins et al., 1996] Benjamins, V. R.; Gomez-Perez A. (1996). "Knowledge-System Technology: Ontologies and Problem-Solving Methods". Disponible en http://www.swi.psy.uva.nl/ usr/richard/pdf/kais.pdf. Último acceso: Marzo de 2008.

[Benjamins et al., 1998] Benjamins, V. R.; Fensel, D.; Gomez-Perez, A. (1998). 'Knowledge Management through Ontologies', in Proceedings of the 2th International Conference on Practical Aspects of Knowledge Management (PAKM'98), Basel, Switzerland.

[Berners-Lee et al., 2001] Berners-Lee, T.; Hendler, J.; Lassila, O. (2001). "The Semantic Web". Scientific American, http://www.scientificamerican.com/2001/0501issue/ 0501berners-lee.html.

[Bjornson, 2007] Bjornson, F. (2007). "Knowledge management in software process improvement". Doctoral thesis, Department of Computer and Information Science, Norwegian University of Science and Technology.

[Blázquez et al., 1998] Blazquez, M.; Fernandez, M.; Garcia-Pinar, J.; Gomez-Perez, A. (1998). "Building Ontologies at the Knowledge Level using the Ontology Design Environment". Knowledge Acquisition Workshop (KAW'98).

[Bogaerts et al., 2005] Bogaerts, S.; Leake, D..(2005). "*IUCBRF: A Framework For Rapid And Modular Case-Based Reasoning System Development"*. Report Version 1.0. Technical report 617, Computer Science Department Lindley Hall, Indiana University. URL: http://www.cs.indiana.edu/~sbogaert/CBR/IUCBRF.pdf . Último acceso: 10/12/09.

[Borst, 1997] Borst, W. N., (1997). "Construction of Engineering Ontologies". PhD thesis, University of Twente, Enschede, 1997.

[Bray et al. 1998] Bray T.; Paoli J. and Sperberg-Mcqueen C. (1998). "Extensible Markup Language (XML) 1.0.". W3C (World-Wide Web Consortium) Recommendation 10 February 1998. http://www.w3.org/TR/REC-xml .

[Birk et al., 2002] Birk, A.; Dingsoyr, T.; Stalhane, T.(2002). "Postmortem: never leave a project without it", IEEE Software, 19 (3), (2002) 43-45

[Brickley et al., 2000] Brickley, D.; Guha, R.(2000). "Resource Description Framework (RDF) Schema Specification 1.0.", Technical report, W3C, 27 March 2000. http://www.w3.org/TR/2000/CR-rdf-schema-20000327.

[Bogaerts et al., 2005a] Bogaerts, S.; Leake, D.. (2005). "IUCBRF: A Framework For Rapid And Modular Case-Based Reasoning System Development". Technical Report 617, Indiana University http://www.cs.indiana.edu/ sbo-gaert/CBR/IUCBRF.pdf.

[Bogaerts et al., 2005b] Bogaerts, S.; Leake, D.. (2005). "Increasing ai project effectiveness with reusable code frameworks: A case study using IUCBRF". In Proceedings of the Eighteenth International Florida Artificial Intelligence Research Society Conference. AAAI Press.

[Borst, 1997] Borst, W. N., (1997). "Construction of Engineering Ontologies". PhD thesis, University of Twente, Enschede, 1997.

[Carlson, 2001] Carlson, D.; (2001). "Modeling XML Applications with UML", Addison-Wesley.

[Chang, 1998] Chang, W.; (1998). "A Discussion of the Relationship Between RDF-Schema and UML", http://www.w3.org/TR/1998/NOTE-rdf-uml-19980804/.

[Chandrasekaran, 1999] Chandrasekaran, B. (1999). "What Are Ontologies, and Why Do We Need Them?. *IEEE Inteligent Systems*, Janeiro 7 Fevereiro 99, EUA.

[Chau et al., 2005] Chau, T.; Maurer, F.. (2005). A case study of wiki-based experience repository at a medium-sized software company, University of Calgary, Department of Computer Science.

[Chen et al., 2003] Chen, H.; Wu, Z.(2003). "On Case-Based Knowledge Sharing in Semantic Web"; in 15 IEEE Conference on Tools with Artificial Intelligence (ICTAI'03).

[Chen et al., 2006] Chen A. P.; Chen, M.Y. (2006). "Knowledge Management Performance Evaluation: A Decade Review from 1995 to 2004", Journal of Information Science, vol. 32, pp. 15-36.

[Chu et al., 2008] Chu, P.; Huang, C.; Lin, L.; (2008). "Understanding Knowledge Management Performance: A Test of an Integrated Model," Advanced Management of Information for Globalized Enterprises, 2008. AMIGE 2008. IEEE Symposium on , vol., no., pp.1-5, 28-29.

[Conklin et al., 1987] Conklin, J.; Begeman, M.. (1987). "gIBIS: A Hypertext Tool for Team Design Deliberation". In Proceedings of the 1st ACM International Hypertext Conference (Hypertext-87), November, Chapel Hill, NC, 247-252.

[Conklin, 1996] Conklin, J.. (1996). "Designing Organizational Memory: Preserving Intelectual Assets in a Knowledge Economy. Group Decision Support Systems. Disponible en <http:\\www.gdss.com \DOM.htm> . Último acceso: 10/11/2006.

[Conklin, 2003] Conklin, J.. (2003). "Dialog Mapping: Reflections on an Industrial Strength Case Study", capítulo del libro: *Visualizing Argumentation – Tools for Collaborative and Educational Sense-Making*, P. Kirschner, S.J.B Shum,C.S. Carr (Eds), Springer-Verlag, London.

[Cooper, 2007] Cooper, L. (2007). "Converting project team experience to organizational learning". Proceedings of the 40th Hawaii International Conference on System Sciences, pp. 195

[Corcho et al., 2001] Corcho, O.; Fernández-López, M. and Gomez-Perez A. (2001). "Technical Roadmap v 1.0".Deliverable 1.1.1 del Consorsio OntoWeb , disponible en: www.lsi.us.es/iberamia2002/tutoriales/d11_v1_0.pdf. Último acceso: 03/2010.

[Corcho et al., 2002] Corcho, O.; Fernández-López, M; Gómez-Pérez, A.; Vicente, O (2002). "WebODE: an Integrated Workbench for Ontology Representation, Reasoning and Exchange". In: Gómez-Pérez A, Benjamín VR editores, XIII International Conference on Knowledge Engineering and Knowledge Management. Spain. Lecture Notes in Artificial Intelligence LNAI 2473. Springer-Verlang, pp 138-153.

[Corcho et al., 2003] Corcho, O.; Fernández-López, M.; Gómez-Pérez, A. (2003). "Methodologies, tools and languages for building ontologies. Where is their meeting point?". Data & Knowledge Engineering. 2003. Vol. 46, núm 1, p.41-64.

[Coyle et al., 2004] Coyle, L.; Doyle, D.; Cunningham, P. (2004) "Similarity for CBR", Technical Report TCD-CS-2004-25, Trinity College, Dublin.

[Cranefield et al., 1999] Cranefield, S.; Purvis, M. (1999). "UML as an ontology modelling language". In Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99).

[Cranefield, 2001a] Cranefield, S. (2001). "Networked knowledge representation and exchange using UML and RDF". *Journal of Digital Information*, 1(8), . http://jodi.ecs.soton.ac.uk/.

[Cranefield, 2001b] Cranefield, S. (2001) "UML and the Semantic Web". Discussion Paper 2001/04, Department of Information Science, University of Otago, New Zealand http://www.otago.ac.nz/informationscience/publctns/complete/papers/dp2001-04.pdf.gz

[Curbera et al., 2002] Curbera, F.; Duftler, M.; Khalaf, R.; Nagy, W.; Mukhi, N. and Weerawarana, S.; 2002. "Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI". *IEEE Internet Computing* V.6, N 2, pp. 86-93.

[Cyc 2002] OpenCyc Selected Vocabulary and Upper Ontology site, http://www.cyc.com/cycdoc/vocab/vocab-toc.html. Último acceso: Enero 2010

[Davies et al., 2003] Davies, J.; Fensel, D. and Van Harmelen, F.;(2003). "Towards the Semantic Web: Ontology-driven Knowledge Management", John Willey & Sons.

[Davenport et al., 1998a] Davenport, T.H.; Prusak, L. (1998). "*Working Knowledge: How Organiazations Manage What They Know*", Hardvard Business School Press, Boston, MA; 1998.

[Davenport et al., 1998b] Davenport, T. H; De Long, D. W.; Beers, M. C. (1998). "Successful Knowledge Management Projects," Sloan Management Review, vol. 39, pp. 43-57.

[Davenport, 2006] Davenport, T.. (2006). "Some Principles of Knowledge Managment"; White paper en *Information Technology Management WEB*. Disponible en: www.itmweb .com/essay538.htm. Último acceso: Septiembre de 2009.

[Despres, 2000] Despres, S. (2000); "Multiple Views on Consensual Categories: A Contribution for Corporate Memory Management", Proceedings of the ECAI'2000 Workshop on Knowledge Management and Organizational Memories, 2000.

[Dey, 2001] Dey, A. (2001) "Understanding and Using Context". Personal and Ubiquitous Computing 5(1).

[Dieng et al. 1999] Dieng, R.; Corby, O.; Giboin, A. y Ribière, M. "Methods and Tools for Corporate Knowledge Management". International Journal of Human-Computer Studies. 1999. (Eds.) S. Decker and F. Maurer. Special Nro. on Knowledge Management. Vol. 51, p.567-598.

[Dingsoyr et al., 2001] Dingsoyr, T.; Moe, N.; Nytro, O. (2001). "Augmenting experience reports with lightweight postmortem reviews", in F. Bomarius, S. Komi.Sirvio (Eds.): PROFES 2001, LNCS 2188, 2001, pp. 167-181.

[Dogson, 1993] Dogson, M. (1993) "Organizational Learning: A Review of Some Literatures". Organization Studies, Vol. 14:3, pp. 375-394.

[Drucker, 1998] Drucker, P. (1998). The Coming of the New Organization. Harvard Business Review. Reimpresión de 1998.

[Ermine, 2000] Ermine, J.L. (2000) "Challenges and Approaches for Knowledge Management in Companies" en Workshop in Knowledge Management: Theory and Applications, Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases. Lyon, France.

[Falbo et al. 2003] Falbo, R.; Natali, A.; Mian, P.; Bertollo, G.; Ruy, F.. (2003)."ODE: Ontology-based software Development Environment", en: Memórias del IX Congreso Argentino de Ciencias de la Computación (CACIC 2003), p. 1124-1135, La Plata, Argentina.

[Fellbaum, 1998] Fellbaum, C. (ed.) (1998). WordNet: An Electronic Lexical Database. Cambridge: MIT Press.

[Fensel et al 2003] Fensel, D.; Motta, E.; Benjamins, V.; Crubezy, M.; Decker, S.; Gaspari, M.; Groenboom, R.; Grosso, W.; Van Harmelen, F.; Musen, M.; Plaza, E.; Schreiber, G.; Studer, R.; Wielinga, B.. (2003). "The Unified Problem-solving Method Development Language UPML". Knowledge and Information Systems, 5(1).

[Ferguson 2006] Ferguson, S.. (2006). "AS 5037-2005: knowledge management blueprint for Australian organisations?", Australian Library Journal, V. 55:3, pp.196-209.

[Fernández et al., 1997] Fernández López, M.; Gómez-Pérez, A.; Juristo, N. (1997). "METHONTOLOGY: From Ontological Art Towards Ontological Engineering".

Proceed. of the AAAI Symposium. University of Stanford; P.A., California, US, pp. 33-40.

[Fernández, 1999a] Fernández, M. (1999). "Overview Of Methodologies for Building Ontologies". In Proceedings of the IJCAI-99 workshop on Ontologies and Problem-Solving Methods (KRR5) Stockholm, Sweden.

[Fernández et al., 1999b] Fernández-López, M.; Gómez-Pérez, A. and Pazos-Sierra, J. (1999). "Building a Chemical Ontology Using METHONTOLOGY and the Ontology Design Environment". *IEEE Intelligent Systems & their applications*. January/February, pp. 37-46.

[Fischer, 2002] Fischer, W.J. (2002). "*Knowledge Reuse: The Roles of Human* and Technical Intermediaries". Thesis (Master). Faculty of the Graduate School of Arts and Sciences of Georgetown University. Washington, DC.

[van Harmelen, 1999] Frank van Harmelen, Dieter Fensel. "Practical Knowledge Representation for the Web", Proceedings of the IJCAI-99 workshop on Intelligent Information Integration Stockholm, Sweden,1999.

[Gomez-Perez et al., 1995] Gómez-Pérez, A.; Juristo, N.; Pazos, J. (1995). Evaluation and Assessment of the Knowledge Sharing Technology Proceedings of the Second International Conference on Building and Sharing Very Large-Scale Knowledge Bases. Editorial IOS Press. Pags: 289-296.

[Gomez-Perez 1996] Gomez-Perez, A.; Fernandez, M. and De Vicente, A.. (1996). "Towards a Method to Conceptualize Domain Ontologies". Working notes of the workshop Ontological Engeniering. ECAI'96 pp. 41-52.

[Gómez-Pérez, 1999] Gómez-Pérez, A., (1999). "Ontological Engineering: A State Of The Art". Expert Update. British Computer Society. Autumn. Vol. 2. Nº 3.

[Gómez-Pérez et al., 2004] Gómez Pérez, A., Fernández López, M. Corcho, O. (2004). "Ontological Engineering". Springer-Verlag, London.

[Gruber 1993] Gruber T. R. (1993). "A translation approach to portable ontology specifications". Knowledge Acquisition, 5:199–220, 1993.

[Gruber et al. 1994a] Gruber T. R.; Olsen G. R. (1994). "An Ontology for Engineering Mathematics", Knowledge Systems Laboratory, Stanford University disponible en: http://www.ksl.stanford.edu/knowledge-sharing/papers/engmath.html.

[Gruber et al., 1994b] Gruber T.; Gerbaux F., (1994). "Frame ontology". ftp://ftp.ksl.stanford.edu/pub/knowledge-sharing/ontologies/html/frame-ontology/index.html. Último acceso Julio de 2007.

[Gruber, 1995] Gruber, T. (1995). "Towards Principles for the Design of Ontologies used for Knowledge Sharing". *International Journal of Human-Computer Studies*, 43(5/6), pp. 907-928.

[Grüninger, 1995] Grüninger, M. and Fox, M.S. 1995. "Methodology for the design and evaluation of ontologies". Workshop on Basic Ontological Issues in Knowledge Sharing. Montreal, Canada.

[Grupta et al., 2000] Gupta, A.; Govindarajan, V. (2000). "Knowledge management's social dimension: lessons from Nucor Steel". Sloan Management Review, 42(1), 71-80.

[Guarino, 1995] Guarino, N. 1995. "Formal Ontology, Conceptual Analysis and Knowledge Representation: The Role of Formal Ontology in the Information Technology", International Journal of Human-Computer Studies, Vol. 43, Nos. 5/6, 1995, pp. 625-640.

[Guarino et al., 1999] Guarino, N.; Masolo, C.; Vetere, G. (1999). "Ontoseek: Content-based Access to the Web". IEEE Intelligent Systems. Vol. 14, núm. 3, p.70-80.

[Hammond, 1990] Hammond, K.. (1990). "Case-based planning: A frame-work for planning from experience". Cognitive Science. CHEF, 1990. URL: http://eprints.kfupm.edu.sa/29317/. último acceso: 28/10/09.

[Handzic et al., 2004] Handzic, M.; Chaimungkalanont, M.. (2004), 'Enhancing organisational creativity through socialisation', The Electronic Journal of Knowledge Management, vol. 2, issue 1, pp. 57-64 .

[Harrison, 2003] Harrison, W.(2003). "A software engineering lessons learned repository", Proceedings of the 27th Annual NASA Goddard/IEEE Software Engineering Workshop, pp. 139.

[Haustein, 2000] Haustein, S.; (2000). "RDF for object serialization". Start of thread on www-rdf-interest mailing list. http://lists.w3.org/Archives/Public/www-rdf-interest/2000Feb/0157.html .

[Hayes, 2004] Hayes P. (editor). (2004). WWW Consortium, "RDF Semantics",W3C Recommendation 10 February 2004, http://www.w3.org/TR/rdf-mt/.

[Hinrichs, 1992] Hinrichs, T.. (1992). *Problem solving in open worlds*. Lawrence Erlbaum Associates.

[Huber, 1991] Huber, G. (1991). "Organizational learning: the contributing processes and the literatures", Organizational Science, Vol. 2 No. 1, pp. 88-115.

[IEEE 1990] IEEE 610.12:1990. (1990). "International Standard, Glossary of Software Engineering Terminology".

[INRECA] INRECA Proyect. http://www.wi2.uni-trier.de/de/cms/projects/INRECA/.

[ISO12207, 1994] ISO/IEC 12207-1. (1994). "International Standard, Information technology - Software life cycle processes".

[ISO9126-1, 2001]. ISO/IEC 9126-1. (2001). "Software Engineering Product Quality - Part 1: Quality Model".

[Jaczynski et al., 1998] Jaczynski, M.; Trousse, B.. (1998). "An Object-Oriented Framework for the Design and the Implementation of Case-Based Reasoners". In Proceedings of the 6th German Workshop on Case-Based Reasoning, Berlin.

[Kaidara, 2009] Kaidara Software. http://www.kaidara.com/. último acceso: 28/10/09

[Karacapilidis, 2006] Karacapilidis, N.. (2006). "An Overview of Future Challenges of Decision Support Technologies". In Intelligent Decision-Making Support Systems: Foundations, Applications and Challenges, J. N. D. Gupta, G. A. Forgionne, M. Mora (Eds.), Springer-Verlag, London, 386-399.

[Kemp et al., 2001] Kemp, J.; Weber, F.; Pudlatz, M.; Wunram, M.; Bredehorst, B.. (2001) "Knowledge Management 'Standardisation'", European KM Forum. (IST Project No. 2000-26393).

[Klyne et al., 2004] Klyne G.; Reynolds F.; Woodrow C.; Ohto H.; Hjelm J.; Butler M. and Tran L., (2004). "Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0". W3C Recommendation 15 January 2004, http://www.w3.org/ TR/2004/REC-CCPP-struct-vocab-20040115/.

[Kogut et al., 2008] Kogut, P.; Cranefield, S.; Hart, L.; Dutra, M.; Baclawski, K.; Kokar, M. and Smith, J.; (2008). "UML for Ontology Development". Disponible en: http://www. sandsoft.com/docs/ker4.pdf. Último acceso: Junio de 2008.

[Kolodner, 1983a] Kolodner, J.; (1983). "Maintaining Organization in a Dynamic long-term Memory"; Cognitive Science; Vol 7, p 243-280.

[Kolodner, 1983b] Kolodner, J.; (1983). "Reconstructive memory, a Computer Model"; Cognitive Science, Vol 7, p 281-328.

[Kolodner, 1993] Kolodner J. (1993), "*Case-based Reasoning*", Edit. Morgan Kaufmann

[Komi-Sirvio et al., 2002] Komi-Sirvio, S.; Mantyniemi, A.; Seppanen, V. (2002). "Toward a practical solution for capturing knowledge for software projects". IEEE Software, 19 (3), 2002, pp. 60-62

[Koton, 1989] Koton, P.. (1989). "Using experience in learning and problem solving". Massachusetts Institute of Technology, Laboratory of Computer Science (Ph.D. diss, October 1988). MIT/LCS/TR-441.

[Kukko et al., 2008] Kukko, M.; Helander, N.; Virtanene, P. (2008). "Knowledge management in renewing software development process", Proceedings of the 41th Hawaii International Conference on System Sciences, pp. 332

[Lassila et al. 1999] Lassila O. and Swick R. R. (editores). (1999). "Resource description framework (RDF): Model and syntax specification", W3C Recommendation 1999-02-22, http://www.w3.org/TR/REC-rdf-syntax.

[Malhotra 1996] Malhotra,Y (1996) Organizational Learning & Learning Organizations: An Overview. [documento WWW]. URL http://www.brint.com/papers/orglrng.htm. último acceso: 05/10/08.

[Malhotra 2005] Malhotra, Y.. (2005), 'Integrating knowledge management technologies in organizational business processes: getting real time enterprises to deliver real business performance', Journal of Knowledge Management, Emerald Group Publishing Limited, vol. 9, no. 1 pp. 7-28

[Manola, 2002] Manola, F.; Miller E., WWW Consortium, (2002). "RDF Primer", http://www.w3.org/TR/2002/WD-rdf-primer-20020319/.

[Martin et al., 2003] Martín M. and Olsina, L.. (2003). "Towards An Ontology for Software Metrics and Indicators as the Foundation for a Cataloging Web System" . In IEEE C. Press Proceeding of the 1st Latin American Web Congress (LA-Web), Santiago de Chile, 10-12 Nov 2003, pp 103-113.

[Martin et al., 2003b] Martín, M.A.; Molina, H.; Papa, F.; Fons, J.; Pastor, O. and Olsina, L.. (2003). "Aspectos de Diseño Arquitectural y Semántico para un Sistema Web de

Catalogación de Métricas". in Proceedings del VI Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes Software (IDEAS), Asunción, Paraguay, pp. 224-236.

[Martin, 2005] Martín M. A., (2005). "Sistema de Catalogación de Métricas e Indicadores con Potencia de Web Semántica", Tesis de Magister, Facultad de Ciencias Exactas, UNLP, La Plata, Argentina.

[Martín et al., 2006] Martín M.; Olsina L.. (2006) "Memoria Organizacional Basada en Casos para dar Soporte al Proceso de Aseguramiento de Calidad", In CD-ROM proceed. of XII CACIC 2006, Workshop WISBD, Potrero de los Funes, San Luis.

[Martín et al., 2008] Martín, M., Olsina, L.. (2008). "Ontology for a Case-based Organizational Memory In Proceedings del XI Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes Software (IDEAS), Recife, Brasil. pp. 351-356, ISBN 97-8857084-13-46.

[Martin et al., 2009] Martín, M., Olsina, L.. (2009). "Added Value of Ontologies for Modeling an Organizational Memory". Chapter's book in Building Organizational Memories: Will You Know What You Knew?. John Girard (Ed). IGI Global.

[Mika et al., 2004] Mika, P.; Akkermans, H.. (2004). "Towards a new synthesis of ontology technology and knowledge management", Knowledge Eng. Rev. 19 (4) 317–345.

[Miyoshi et al., 1996] Miyoshi, H.; Sugiyama, K.; Kobayashi, M.; Ogino, T. (1996). "An Overview of the EDR Electronic Dictionary and the Current Status of Its Utilization". Proceedings of COLING-96, August 1996.

[Mizoguchi et al., 1995] Mizoguchi, R.; Vanwelkenhuysen, J. and Ikeda M. (1995). "Task ontology for reuse of problem solving knowledge". In 2nd International Conference on Very Large-Scale Knowledge Bases., pages 46–57. IOS Press, Amsterdam, NL.

[Molina et al., 2007] Molina H.; Olsina L.. (2007). "Towards the Support of Contextual Information to a Measurement and Evaluation Framework". In proceed. of the IEEE Computer Society, 6th Int'l Conference on the Quality of Information and Communication Technology (QUATIC'07) Lisbon, Portugal, pp. 154-163.

[Molina et al., 2008] Molina H.; Olsina L. (2008). "Assessing Web Applications Consistently: A Context Information Approach", In proceed. of IEEE Computer Society, 8th Int.l Congress on Web Engineering (ICWE08), NY, USA, pp. 224-230, ISBN 978-0-7695-3261-5.

[Natali et al., 2002] Natali A., Falbo R.. (2002). "Knowledge Management in Software Engineering Environments", In proceed. of the 16th Brazilian Symposium on Software Engineering, pp. 238-253.

[Newcomer, 2002] Newcomer, E.; (2002). "Understanding Web services: XML, WSDL, SOAP, and UDDI". Addison-Wesley, Boston, MA.

[Nonaka et al., 1995] Nonaka I.; Takeuchi H.. (1995). "Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation", Oxford University Press.

[Noy et al., 2000] Noy, N; Fergerson, R; Musen, A. (2000). "The knowledge model of Protégé-2000: combining interoperability and flexibility", In R. Dieng and O. Corby, editors, Proc. of the 12th EKAW, LNAI, pages 17—32. France. Springer.

[Öhgren et al., 2005] Öhgren, A., Sandkuhl, K. (2005). Towards a methodology for ontology development in small and medium-sized enterprises. In: IADIS Conference on Applied Computing, Algarve, Portugal. pp. 369-376

[O'Leary, 1998a] O'Leary, D. (1998). 'Knowledge Management Systems: Converting and Connecting', IEEE Intelligent Systems, 13(3), 30–33.

[O'Leary, 1998b] O'Leary, D. (1998), 'Using AI in Knowledge Management: Knowledge Bases and Ontologies', IEEE Intelligent Systems, 13(3), 34–39, (June 1998).

[O'Leary, 2002] O'Leary, D. (2002). "Knowledge Management in Accounting and Professional Services". En: V. Arnold y S. Sutton (eds.), Researching Accounting as an Information Systems Discipline, Sarasota, FL, American Accounting Association.

[Olsina et al., 2002] Olsina, L.; Rossi, G. (2002). "Measuring Web Application Quality with WebQEM", IEEE Multimedia, 9(4), pp. 20-29, 2002.

[Olsina et al., 2004] Olsina, L. and Martín, M.. (2004). "Ontology for Software Metrics and Indicators". in *Journal of Web Engineering*, Rinton Press, US, Vol 3 Nº X, 2004, ISSN 1540-9589.

[Olsina et al., 2007] Olsina, L; Papa, F.; Molina, H.. (2007). "How to Measure and Evaluate Web Applications in a Consistent Way". In: Springer, HCI Series, Rossi, Pastor, Schwabe & Olsina (Eds.) Web Engineering: Modelling and Implementing Web Applications. pp. 385–420.

[Pal et al., 2004] Pal, S.; Shiu, S.. (2004). "*Foundations of Soft Case-Based Reasoning"*, John Wiley: NY, 2004.

[Papoutsakis, 2009] Papoutsakis, H.. (2009). "Organizational Knowledge Sharing Networks". Chapter's book in Building Organizational Memories: Will You Know What You Knew?. John Girard (Ed). IGI Global.

[Price, 1999]. Price, C.. (1999). "Computer-Based Diagnostic Systems, University of Wales, berystwyth". Disponible en http://cadair.aber.ac.uk/dspace/bitstream/2160/67/2/ ComputerBasedDiagnosticSystemsBook.pdf, último acceso: 28/10/09.

[Prusak, 1996] Prusak, L.. (1996). "The Knowledge Advantage. Strategy & leadership" a publication of Strategic Leadership Forum, 1996, vol. 24, no. 2, p. 6.

[Rabarijaona et al., 1999] Rabarijaona, A.; Dieng, R.; Corby, O. (1999). "Building a XML-based Corporate Memory. IJCAI Workshop on Knowledge Management and Organisational Memory, 31st July, 1999, Stockholm.

[Rivera, 2009] Rivera, Belén (2010). "Sistema de Recomendación Basado en Casos para Proyectos de Medición y Evaluación en Ingeniería de Software". Tesis de grado, para obtención del título de Ingeniera en Sistema. Facultad de Ingeniería, UNLPam. General Pico, La Pampa, Argentina.

[Rodriguez et al., 2003] Rodriguez, M.A.; Egenhofer, M.J. (2003). "Determining Semantic Similarity among Entity Classes from Different Ontologies". IEEE Transactions on Knowledge and Data Engineering. Vol. 15, núm. 2.

[Sankar et al., 2004] Sankar, K.; Simon C.K. (2004). "Foundations of Soft Case-Based Reasoning". Wiley Series On Intelligent Systems. ISBN 0471086350.

[Schatten, 2003] Alexander Schatten, Stefan Biffl, A Min Tjoa. (2003). "Closing the gap, from nescience to knowledge management". In Proceedings of the Euromicro Conference. IEEE.

[Schank, 1982] Schank R. (1982) Dynamic memory; a theory of reminding and learning in computers and people. Cambridge University Press. 1982.

[Schwabe et al., 2004] Schwabe, G.; Hausmann, D.. (2004). "The Use of Software-Tools for Case-Based-Reasoning Methods". Department of Computer Science, Dresden University of Technology. Dresden, Germany. Disponible en www.informatik.uni-bremen.de/~hausmann/papers/use_of_tools.ps, último acceso: 28/10/09.

[Schulz, 1999] Schulz S.. (1999). "CBR-Works - A State-of-the-Art Shell for Case-Based Application Build". In: Proceedings of the 7th German Workshop on Case-Based Reasoning (GWCBR'99). LNAI. Springer.

[Seufert et al., 1999] Seufert, A.; von Krough, G.; Black, A. (1999). "Towards Knowledge Networking". Journal of Knowledge Management. núm 3.

[Simpson, 1985] Simpson R. L.(1985). "A computer model of case-based reasoning in problem solving: An investigation in the domain of dispute mediation". Technical Report GIT-ICS-85/18, Georgia Institute of Technology. 1985.

[Skyrme, 2003] Skyrme, D..(2003). "Knowledge Management: Making Sense of an Oxymoron". Management Insight. Disponible en http://www.skyrme.com/insights/ 22km.htm. Últimо acceso en Septiembre de 2009.

[Slade, 1991] Slade, S.; (1991)."Case-Based Reasoning: A Research Paradigm"; AI Magazine, Vol 12 Nº 1, p 42-55;

[Smith et al., 2000] Smith, R.G.; Farquhar, A. (2000). "The Road Ahead for Knowledge Management: An AI Perspective". AI Magazine. p.17-40.

[Szulanski, 1996] Szulanski, G.. (1996). "Exploring Internal Stickiness: Impediments to the Transfer of Best Practice within Firms", Strategic Management Journal, vol. 17, Special Issue Winter, pp. 27-44.

[Sowa, 2000] Sowa, J. F. (2000). "Knowledge Representation: Logical, Philosophical, and Computational Foundations", Brooks Cole Publishing Co., Pacific Grove, CA. Disponible parcialmente en http://www.jfsowa.com/ontology/.

[Staab et al., 2000] Staab S., Erdmann M., Maedche A.; Decker S. (2000). "An extensible approach for modeling ontologies in RDF(S)". First Workshop on the Semantic Web at the Fourth European Conference International Workshop on Research and Advanced Technology for Digital Libraries, Lisbon, Portugal 18-20.

[Stein et al., 2000] Stein, L. A.; Connolly, D.; McGuinness, D. (editors). (2000). "DAML Ontology language specification".http://www.daml.org/2000/10/ daml-ont.html.

[Studer et al., 1998] Studer, R.; Benjamins V. R. and Fensel D. (1998). "Knowledge engineering: principles and methods". Data and Knowledge Engineering, 25(1-2):161–197.

[Sure et al., 2000] Sure, Y.; Angele, J.; Staab, S. (2002). "OntoEdit: Guiding Ontology Development by Methodology and Inferencing". In: R. Meersman, Z. Tari et al. (eds.). Proceedings of the Confederated International Conferences CoopIS, DOA and ODBASE (2002) Springer, LNCS 2519, 1205-1222.

[Torres et al., 2004] Torres V.; Fons, J.; Asensi, O; Palechano V. , (2004). "Gettin Ready Web Engineering Methods for the Semantic Web Putting Ontologies into Pratice". In Proceedings of Fourth International Workshop on Web-Oriented Software Technologies (IWWOST'04), Munich, Germany.

[TOVE 2009] TOVE Ontologies site. http://www.eil.utoronto.ca/tove/toveont.html. Último acceso: Agosto 2009.

[Uschold et al., 1995] Uschold, M; King, M. (1995). "Towards a Methodology for Building Ontologies". Workshop on Basic Ontological Issues in Knowledge Sharing.

[Uschold et al., 1996a] Uschold, M. and Grüninger, M. (1996). "Ontologies: Principles Methods and Applications" Knowledge Engineering Review. Vol. 2.

[Uschold, 1996b] Uschold, M. (1996). "Building ontologies: Towards a unified methodology". In Expert Systems 96.

[Uschold et al., 1998] Uschold, M.; King, M.; Moralee, S. and Zorgios, Yannis, (1998). "The Enterprise Ontology". The Knowledge Engineering Review , Vol. 13, Special Issue on Putting Ontologies to Use (eds. Mike Uschold and Austin Tate).

[Uschold et al., 1999] Uschold, M.; Jasper, R. (1999). "A Framework for Understanding and Classifying Ontology Applications" en Proceedings of the IJCAI99 Workshop on Ontologies and Problem-Solving Methods. Stockholm.

[Van der Spek et al., 1997] Van der Spek, R.; Spijkervet, A. (1997), "Knowledge management: dealing intelligently with knowledge", In: Liebowitz, J. and Wilcox, L. (Eds), Knowledge Management and its Integrative Elements, CRC Press, New York.

[Van Heijst et al., 1996] Van Heijst, G.; Van der Spek, R. ; Kruizinga, E. (1996). "Organizing Corporate Memories" en Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW'96). Canada. p.42-1 42-17.

[Van Heijst et al., 1997] van Heijst, G.; Schreiber, A.Th. and Wielinga, B.J. , (1997). "Using Explicit Ontologies in KBS Development", International Journal of Human-Computer Studies, 46(2/3): pp. 183–292.

[Von Krogh, 1998] Von Krogh, G. (1998). "Care in Knowledge Creation". Special Nro. of Knowledge and the Firm. California Management Review. Vol. 40, núm. 3, p.133. ISSN: 0008-1256.

[W3C] W3C, WWW Consortium, http://www.w3.org

[W3Csw, 2010] W3C, WWW Consortium. (2010). Semantic Web, http://www.w3.org /standards/semanticweb/

[Wang et al., 2001] Wang, X. and Chan, C.W., (2001). "Ontology Modeling Using UML". 7th International Conference on Object Oriented Information Systems Conference (OOIS'2001), pp. 59-68.

[Watson, 2003] Watson, I. (2003). "Applying Knowledge Management: Techniques for Building Corporate Memories". San Francisco, CA: Morgan Kaufmann.

[Weber et al., 2001] Weber, R.; Aha, D.; Becerra-Fernandez, I.. (2001). " Intelligent Lessons Learned Systems". Int'l Journal of Expert Systems Research & Applications. 20 (1). pp. 17-34.

[Weber 2007] Weber, R.. (2007). "Addressing failure factors in knowledge management", The Electronic Journal of Knowledge Management, vol. 5, no. 3, pp. 333-46.

[WebOnt] Web Ontology Working Group of W3C, WWW Consortium, (2001), http://www.w3.org/2001/sw/WebOnt

[Wiig, 1993] Wiig, K.M. (1993). *Knowledge Management Foundations: Thinking About Thinking – How People and Organizations Create, Represent and Use Knowledge*. Schema Press, Arlington.

[Wiig, 1997] Wiig, K. (1997). "Knowledge management: where did it come from and where will it go? Expert Systems with Applications 13(1) 1–14.

[Xpert, 2009] XpertRule Software Ltd.. (2009). Disponible en URL:< http://www.xpertrule. com/>. Acesso en: 15 Septiembre de 2009.

[Yang et al., 2006] Yang, K.J;. Chen, Y.. (2006). "Ontology-based Knowledge Retrieval in Organizational Memory". In Proceedings of the First International Conference on Innovative Computing, Information and Control. IEEE Computer Society. pp. 566 – 569.

[Yates, 1990] Yates, J. (1990). "For the Record: The Embodiment of OrganizationalMemory, 1850-1920." Business And Economic History. Massachusetts. V 19, n. 2, p. 172-182.

[Ye, 2006] Ye, Y. (2006). "Supporting software development as knowledge intensive and collaborative activity". Proceedings of (WISER'06), pp. 15-22

[Zhu et al., 2007] Zhu, L.; Staples, M.; Gorton, I.. (2007). "An infrastructure for indexing and organizing best practices", Proceedings of the 2nd International Workshop on Realizing Evidence-based Software Engineering (REBSE '07), pp. 4

Apéndice

Listado del código RDF de la Ontología de Memoria Organizacional Basada en Casos

```
<!-- Esquema RDF de la Ontología de Memoria Organizacional Basada en Casos ->
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

 <!-- Conceptos -->

        <rdfs:Class rdf:ID="Case">
            <rdfs:label xml:lang="en">Case</rdfs:label>
            <rdfs:comment> Ítem de conocimiento contextualizado que representa  una
                    experiencia por medio de un problema y su solución.</rdfs:comment>
            <rdfs:subClassOf rdf:resource="#KnowledgeItem"/>
        </rdfs:Class>

        <rdfs:Class rdf:ID="CaseKnowledgeBase">
            <rdfs:label xml:lang="en"> CaseKnowledgeBase </rdfs:label>
             <rdfs:comment> Una base de conocimiento específica dónde cada ítem de
                    conocimiento se representa a través de un caso</rdfs:comment>
            <rdfs:subClassOf rdf:resource="#KnowledgeBase"/>
        </rdfs:Class>

        <rdfs:Class rdf:ID="Feature">
            <rdfs:label xml:lang="en">Feature</rdfs:label>
             <rdfs:comment> Una propiedad mensurable, física o abstracta, de una entidad.
                  (sinómimo del término Attribute definido en ISO 14598-).</rdfs:comment>
            <rdfs:subClassOfrdf:resource="http://www.w3.org/2000/01/rdf-hema#Resource"/>
        </rdfs:Class>

        <rdfs:Class rdf:ID="KnowledgeBase">
            <rdfs:label xml:lang="en">KnowledgeBase</rdfs:label>
             <rdfs:comment>Un cuerpo organizado de conocimiento relacionado. [AS
                  5037].</rdfs:comment>
            <rdfs:subClassOfrdf:resource="http://www.w3.org/2000/01/rdf-hema#Resource"/>
        </rdfs:Class>
```

```
<rdfs:Class rdf:ID="KnowledgeItem">
     <rdfs:label xml:lang="en">KnowledgeItem</rdfs:label>
      <rdfs:comment> Una pieza atómica de conocimiento. Comentario: Conocimiento
           es una pieza de entendimiento y/o lecciones aprendidas de las habilidades y
           experiencias construidas por las personas. </rdfs:comment>
     <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
           shema#Resource"/>
</rdfs:Class>

<rdfs:Class rdf:ID="OrganisationalMemory">
    <rdfs:label xml:lang="en"> OrganisationalMemory </rdfs:label>
     <rdfs:comment> La manera en que una organización guarda y lleva cuenta de lo que
           sabe. </rdfs:comment>
    <rdfs:subClassOfrdf:resource="http://www.w3.org/2000/01/rdf-chema#Resource"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Problem">
     <rdfs:label xml:lang="en"> Problem </rdfs:label>
      <rdfs:comment> Un estado de dificultad que necesita ser resuelto
           (WordNet)</rdfs:comment>
     <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
           schema#Resource"/>
</rdfs:Class>

<rdfs:Class rdf:ID="ProblemFeature">
     <rdfs:label xml:lang="en"> ProblemFeature </rdfs:label>
      <rdfs:comment> Rasgo que caracteriza a un problema</rdfs:comment>
     <rdfs:subClassOf rdf:resource="#Feature"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Result">
     <rdfs:label xml:lang="en"> Result </rdfs:label>
      <rdfs:comment> Nivel de satisfacción logrado por la aplicación de una solución a
           un problema</rdfs:comment>
     <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
           schema#Resource"/>
</rdfs:Class>
```

```
<rdfs:Class rdf:ID="SimilarityAssessmentModel">
    <rdfs:label xml:lang="en"> SimilarityAssessmentModel </rdfs:label>
     <rdfs:comment> Algoritmo o función con elementos de similitud asociados que
          modela la valoración de similitud de casos</rdfs:comment>
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
          schema#Resource"/>
</rdfs:Class>

<rdfs:Class rdf:ID="SimilarityCriterion">
    <rdfs:label xml:lang="en"> SimilarityCriterion </rdfs:label>
     <rdfs:comment> El patrón de valoración usado para determinar la similitud
          semántica entre dos valores del rasgo </rdfs:comment>
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
          schema#Resource"/>
</rdfs:Class>

<rdfs:Class rdf:ID="SimilarityModelElement">
    <rdfs:label xml:lang="en"> SimilarityModelElement </rdfs:label>
     <rdfs:comment> Algoritmo o función con criterios de similitud asociados que
          modela la valoración de similitud de una característica </rdfs:comment>
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
          schema#Resource"/>
</rdfs:Class>

<rdfs:Class rdf:ID="SimilarityType">
    <rdfs:label xml:lang="en"> SimilarityType </rdfs:label>
     <rdfs:comment> Especifica el tipo de función de similitud que puede ser exacta,
          diferencia o compleja </rdfs:comment>
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
          schema#Resource"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Solution">
    <rdfs:label xml:lang="en"> Solution </rdfs:label>
     <rdfs:comment> Una declaración que resuelve un problema o explica cómo
          resolver un problema. (WordNet)</rdfs:comment>
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
          schema#Resource"/>
</rdfs:Class>
```

```
        <rdfs:Class rdf:ID="SolutionFeature">
            <rdfs:label xml:lang="en"> SolutionFeature </rdfs:label>
             <rdfs:comment> Rasgo que caracteriza a una solución </rdfs:comment>
            <rdfs:subClassOf rdf:resource="#Feature"/>
        </rdfs:Class>


<!-- Atributos de Case -->

        <rdf:Property rdf:ID="caseName">
                <rdfs:label xml:lang="en"> caseName </rdfs:label>
                <rdfs:domain rdf:resource="#Case"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="caseDescription">
                <rdfs:label xml:lang="en">caseDescription</rdfs:label>
                <rdfs:domain rdf:resource="#Case"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="caseAuthor ">
                <rdfs:label xml:lang="en"> caseAuthor </rdfs:label>
                <rdfs:domain rdf:resource="#Case"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="caseTimestamp">
                <rdfs:label xml:lang="en"> caseTimestamp </rdfs:label>
                <rdfs:domain rdf:resource="#Case"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>


<!-- Atributos de Feature -->

        <rdf:Property rdf:ID="featureName">
                <rdfs:label xml:lang="en"> featureName </rdfs:label>
                <rdfs:domain rdf:resource="#Feature "/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>
```

```
        <rdf:Property rdf:ID="featureValue">
                <rdfs:label xml:lang="en"> featureValue </rdfs:label>
                <rdfs:domain rdf:resource="#Feature "/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>


<!-- Atributos de KnowledgeBase -->

        <rdf:Property rdf:ID="knowledgeBaseName">
                <rdfs:label xml:lang="en"> knowledgeBaseName </rdfs:label>
                <rdfs:domain rdf:resource="#KnowledgeBase"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="knowledgeBaseDescription">
                <rdfs:label xml:lang="en"> knowledgeBaseDescription </rdfs:label>
                <rdfs:domain rdf:resource="#KnowledgeBase"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

<!-- Atributos de Organisational Memory -->

        <rdf:Property rdf:ID="organisationalMemoryName">
                <rdfs:label xml:lang="en"> organisationalMemoryName </rdfs:label>
                <rdfs:domain rdf:resource="#OrganisationalMemory"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="organisationalMemoryCorporationName">
                <rdfs:label xml:lang="en"> organisationalMemoryCorporationName
                </rdfs:label>
                <rdfs:domain rdf:resource="#OrganisationalMemory"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="organisationalMemoryDescription">
                <rdfs:label xml:lang="en"> organisationalMemoryDescription </rdfs:label>
                <rdfs:domain rdf:resource="#OrganisationalMemory"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>
```

```
<!-- Atributos de Problem -->

        <rdf:Property rdf:ID="problemName">
                <rdfs:label xml:lang="en"> problemName </rdfs:label>
                <rdfs:domain rdf:resource="#Problem"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="problemDescription">
                <rdfs:label xml:lang="en"> problemDescription </rdfs:label>
                <rdfs:domain rdf:resource="#Problem"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

<!-- Atributos de ProblemFeature -->

        <rdf:Property rdf:ID="problemFeaturePriority">
                <rdfs:label xml:lang="en"> problemFeaturePriority </rdfs:label>
                <rdfs:domain rdf:resource="#ProblemFeature"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>


<!-- Atributos de Similarity Assessment Model -->

        <rdf:Property rdf:ID="similarityAssessmentModelName">
                <rdfs:label xml:lang="en"> similarityAssessmentModelName </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityAssessmentModel"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="similarityAssessmentModelDescription">
                <rdfs:label xml:lang="en">similarityAssessmentModelDescription
                </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityAssessmentModel"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>
```

```
<!-- Atributos de SimilarityCriterion -->

        <rdf:Property rdf:ID="similarityCriterionFirstValue">
                <rdfs:label xml:lang="en"> similarityCriterionFirstValue </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityCriterion"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="similarityCriterionSecondValue">
                <rdfs:label xml:lang="en"> similarityCriterionSecondValue </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityCriterion"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="similarityCriterionSimilarity">
                <rdfs:label xml:lang="en"> similarityCriterionSimilarity </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityCriterion"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>


<!-- Atributos de SimilarityModelElement -->

        <rdf:Property rdf:ID="similarityModelElementName">
                <rdfs:label xml:lang="en"> similarityModelElementName </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityModelElement"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="similarityModelElementDescription">
                <rdfs:label xml:lang="en"> similarityModelElementDescription </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityModelElement"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="similarityModelElementWeight">
                <rdfs:label xml:lang="en"> similarityModelElementWeight </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityModelElement"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>
```

```
        <rdf:Property rdf:ID="similarityModelElementUpperThreshold">
                <rdfs:label xml:lang="en"> similarityModelElementUpperThreshold
                </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityModelElement"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="similarityModelElementLowerThreshold">
                <rdfs:label xml:lang="en"> similarityModelElementLowerThreshold
                </rdfs:label>
                <rdfs:domain rdf:resource="#SimilarityModelElement"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>


<!-- Atributos de Solution -->

        <rdf:Property rdf:ID="solutionName">
                <rdfs:label xml:lang="en"> solutionName </rdfs:label>
                <rdfs:domain rdf:resource="#Solution"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

        <rdf:Property rdf:ID="solutionDescription">
                <rdfs:label xml:lang="en"> solutionDescription </rdfs:label>
                <rdfs:domain rdf:resource="#Solution"/>
                <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        </rdf:Property>

<!-- Relaciones -->

        <rdf:Property rdf:ID="associated_with">
          <rdfs:label xml:lang="en"> Is_solved_by </rdfs:label>
          <rdfs:domain rdf:resource="#Case"/>
          <rdfs:range rdf:resource="#Context"/>
        </rdf:Property>

        <rdf:Property rdf:ID="characterised_by">
          <rdfs:label xml:lang="en">characterised_by </rdfs:label>
          <rdfs:domain rdf:resource="#Problem"/>
          <rdfs:domain rdf:resource="#Solution"/>
          <rdfs:range rdf:resource="#Feature"/>
        </rdf:Property>
```

```
<rdf:Property rdf:ID="combines">
  <rdfs:label xml:lang="en"> combines </rdfs:label>
  <rdfs:domain rdf:resource="#SimilarityAssessmentModel"/>
  <rdfs:range rdf:resource="#SimilarityModelElement"/>
</rdf:Property>

<rdf:Property rdf:ID="compound_by">
  <rdfs:label xml:lang="en"> compound_by </rdfs:label>
  <rdfs:domain rdf:resource="#SimilarityModelElement"/>
  <rdfs:range rdf:resource="#SimilarityCriterion"/>
</rdf:Property>

<rdf:Property rdf:ID="defined_by">
  <rdfs:label xml:lang="en"> defined_by </rdfs:label>
  <rdfs:domain rdf:resource="#Feature"/>
  <rdfs:range rdf:resource="#DomainConcept"/>
</rdf:Property>

<rdf:Property rdf:ID="describes">
  <rdfs:label xml:lang="en"> describes </rdfs:label>
  <rdfs:domain rdf:resource="#SimilarityAssessmentModel"/>
  <rdfs:range rdf:resource="#CaseKnowledgeBase"/>
</rdf:Property>

<rdf:Property rdf:ID="has">
  <rdfs:label xml:lang="en">has</rdfs:label>
  <rdfs:domain rdf:resource="#OrganisationalMemory"/>
  <rdfs:range rdf:resource="#KnowledgeBase"/>
</rdf:Property>

<rdf:Property rdf:ID="defined_by">
  <rdfs:label xml:lang="en"> defined_by </rdfs:label>
  <rdfs:domain rdf:resource="#Feature"/>
  <rdfs:range rdf:resource="#DomainConcept"/>
</rdf:Property>

<rdf:Property rdf:ID="has_result">
  <rdfs:label xml:lang="en"> has_result </rdfs:label>
  <rdfs:domain rdf:resource="#Solution"/>
  <rdfs:range rdf:resource="#Result"/>
</rdf:Property>
```

```
<rdf:Property rdf:ID="includes">
  <rdfs:label xml:lang="en"> includes </rdfs:label>
  <rdfs:domain rdf:resource="#KnowledgeBase"/>
<rdfs:range rdf:resource="#KnowledgeItem"/>
</rdf:Property>

<rdf:Property rdf:ID="is_of_type">
  <rdfs:label xml:lang="en"> is_of_type </rdfs:label>
  <rdfs:domain rdf:resource="#SimilarityModelElement"/>
  <rdfs:range rdf:resource="#SimilarityType"/>
</rdf:Property>

<rdf:Property rdf:ID="is_solved_by">
  <rdfs:label xml:lang="en"> is_solved_by </rdfs:label>
  <rdfs:domain rdf:resource="#Problem"/>
  <rdfs:range rdf:resource="#Solution"/>
</rdf:Property>

<rdf:Property rdf:ID="is_solved_by">
  <rdfs:label xml:lang="en"> is_solved_by </rdfs:label>
  <rdfs:domain rdf:resource="#Problem"/>
  <rdfs:range rdf:resource="#Solution"/>
</rdf:Property>

<rdf:Property rdf:ID="refers_to">
  <rdfs:label xml:lang="en"> refers_to </rdfs:label>
  <rdfs:domain rdf:resource="#SimilarityModelElement"/>
  <rdfs:range rdf:resource="#DomainConcept"/>
</rdf:Property>

<rdf:Property rdf:ID="specified_by">
  <rdfs:label xml:lang="en">specified_by</rdfs:label>
  <rdfs:domain rdf:resource="#Case"/>
  <rdfs:range rdf:resource="#Problem"/>
  <rdfs:range rdf:resource="#Solution"/>
  <rdfs:range rdf:resource="#Result"/>
</rdf:Property>
```