

DAWIS¹: Enfoques preliminares sobre la arquitectura de referencia para la integración de Archivos Digitales en Web

C. Costilla, S. Eibe, E. Menansalvas, J. Sáenz
Grupo de Bases de Datos
Universidad Politécnica de Madrid
costilla@dit.upm.es

E. Marcos, P. Cáceres, J. M. Cavero, B. Vela
Grupo KYBELE
Universidad Rey Juan Carlos
e.marcos@escet.urjc.es

Resumen

DAWIS es un proyecto coordinado que ahora comienza a realizar el grupo Kybele de la URJC y el grupo de Bases de Datos de la UPM. El grupo de la UPM cuenta con dos profesores más de la Universidad Carlos III, expertos en materia archivística y documental. Además, como terceras partes, participan ICM de la Comunidad de Madrid y la empresa CRC Information Technologies. El proyecto tiene como objetivo la construcción de un entorno que facilite, en la medida de lo posible, el desarrollo sistemático de portales Web para el acceso integrado, mediante una arquitectura de referencia, a múltiples Archivos Digitales (en adelante, ADs). Presentamos aquí los primeros estudios realizados en torno a ontologías y la arquitectura de referencia de DAWIS.

1. Introducción

DAWIS [11] es un proyecto recién iniciado cuyo objetivo global es *sistematizar y automatizar, en lo posible, la construcción de portales Web para el acceso integrado a múltiples Archivos Digitales mediante una arquitectura de referencia*. Por un lado, el equipo de la URJC liderará los problemas del desarrollo sistemático y metodológico y, por otro lado, desde la UPM se abordarán los problemas relativos a la arquitectura de referencia, modelo del archivo digital (en adelante, AD) y semántica ontológica de la integración. Este trabajo describe algunos enfoques previos de DAWIS sobre ontologías y arquitecturas. Esta parte tiene diversos aspectos a considerar, como son los referentes a: arquitecturas web, modelo del AD, tratamiento de datos semi-estructurados, técnicas de recuperación de información, ontologías para la gestión de la globalidad, enriquecimiento semántico de la Web, etc. Este trabajo tiene en cuenta algunos enfoques preliminares necesarios para abordar la arquitectura de referencia para la integración de los ADs y la semántica de las ontologías.

DAWIS definirá una **arquitectura de referencia** (véase fig. 1) para el acceso consultivo integrado a diversos ADs en la web, conforme a los modelos de arquitectura centradas en la web (J2EE). Nos basaremos en el AD que opera con éxito en la Asamblea de Madrid desde 1999. El AD vendrá expresado en términos de un metamodelo a establecer en este proyecto, y se constituirá, en cada caso, en un modelo específico definido para cada archivo original o nativo y para cada integración en la que participe dicho archivo.

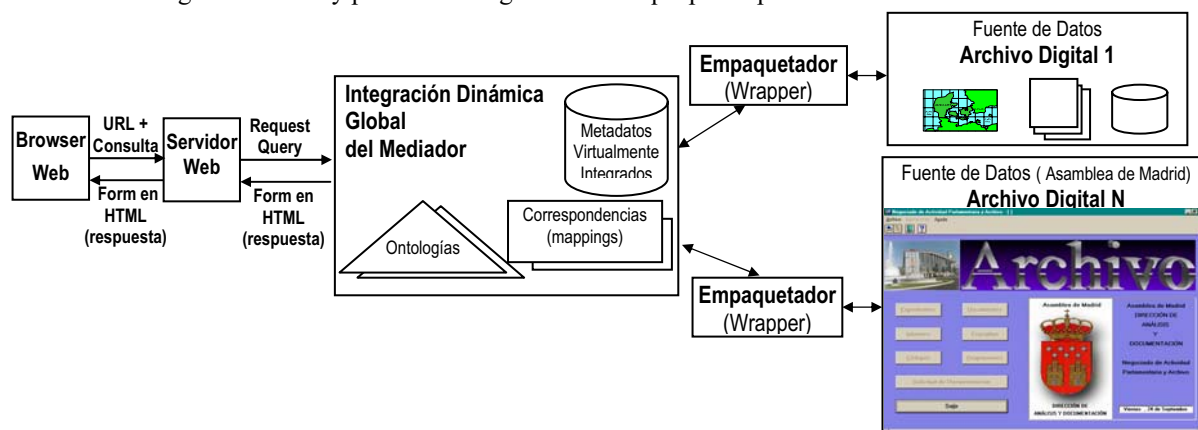


Fig. 1. Arquitectura de integración de Archivos Digitales con Mediator y Empaquetador

Llamamos globalidad al resultado de cada integración que agrupa diversos ADs de diferentes localidades. En la globalidad contaremos con un servidor que alberga y gestiona cada integración, donde reside el mediator (véase fig.1). La tecnología que gestiona cada archivo nativo puede ser diversa y los documentos multimedia pueden estar almacenados con diversos formatos.

La arquitectura de referencia se construirá mediante: a) la especificación del modelo del AD -aplicado al archivo nativo y local-, b) 'wrappers' y c) 'mediators' con ontologías, repositorios y mappings; utilizando tecnología de ficheros y Objeto-Relacional.

¹ DAWIS: Digital Archive Web Information Systems. DAWIS es un proyecto coordinado (MCYT- TIC2002-04050-C02).

Esta arquitectura se implementará mediante una biblioteca de clases basada en XML, y muy especialmente en XMI de CWM. Y se utilizará para cualquier integración de ADs en la web, ya que el ‘mediator’ y los ‘wrappers’ serán generados dinámicamente (en XML, DTDs) a partir de la consulta del usuario web y de las fuentes concretas² de ADs que participan en cada integración particular.

2. Arquitecturas WEB

Con la Web, se ha multiplicado el número de servicios, usuarios y dispositivos para acceder a los sistemas y, con ello, el grado de heterogeneidad. Como consecuencia, el problema se traslada al usuario, a su conocimiento del uso y de las posibilidades de tales sistemas.

La situación descrita se manifiesta muy claramente en el caso de DAWIS que busca posibilitar el acceso flexible, virtual, dinámico e integrado a múltiples ADs a través de la Web. Sobre la base de la tecnología existente, DAWIS presenta la problemática asociada con la producción de una arquitectura que sirva de solución genérica para el acceso integrado de ADs en la Web. El reto principal no es ya posibilitar dicho acceso (transparencia) sino, además, asistir al usuario para que pueda sacar el máximo provecho del mismo.

En estos inicios de DAWIS, nuestro primer propósito es producir un modelo de referencia que defina una *arquitectura genérica* con las siguientes características que consideramos claves:

- *Flexibilidad*, que resulta de una concepción basada en servicios, es decir, arquitecturas dirigidas por los servicios que se ofrecen y soportan. Tales servicios -no definidos exhaustivamente- van surgiendo de manera dinámica y progresiva, a partir del uso de los sistemas. Con la suficiente flexibilidad para definir y ofrecer servicios altamente diferenciados y lo más autocontenidos posibles, esto es, con la interacción mínima necesaria.
- *Integración*, aspecto primordial en DAWIS, cuya dificultad estriba en que la integración de datos es un problema semántico. A pesar de contar en Bases de Datos con estándares y modelos de referencia para la integración estática de esquemas; una solución sistemática de integración dinámica en web, requiere estándares que todavía no existen (las soluciones son a menudo estáticas y *ad hoc*). Sin embargo, el problema de la inexistencia de un estándar es antiguo. Desde las primeras tentativas para regularizar el intercambio de datos (EDI) hasta la proliferación propiciada por la Web (http, html, xml, obi, cxml, etc) no se ha conseguido aún modelar adecuadamente las interacciones entre los componentes de un sistema.
La aproximación basada en servicios Web tampoco resuelve este problema. Los servicios Web permiten intercambiar datos entre aplicaciones remotas, pero no garantizan que la aplicación receptora de servicios sea capaz de entenderlos. Tal y como se señala en [14], el problema es la falta de una ontología global.
- *Interoperabilidad y cooperación*. En un entorno de múltiples componentes distribuidos, los datos y la lógica están diseminados por la red, cuya sinergia ha de determinar el buen funcionamiento del sistema completo.
- *Escalabilidad*. Los usuarios de un sistema requieren un número creciente de capacidades y servicios de muy diversa índole. Curiosamente, esta tendencia es tanto más acusada cuanto mejor sea el funcionamiento del sistema en cuestión (si el AD resulta ser útil, será usado masivamente). Por tanto, el aumento gradual de funcionalidad no es algo probable sino completamente seguro; y la arquitectura del sistema deberá contar con ello escalando adecuadamente.
- *Eficiencia*. La arquitectura del sistema necesita contemplar la eficiencia como un axioma básico de diseño. Se trata de ofrecer soluciones prácticas en las que incluso un rendimiento óptimo puede ser insuficiente. Por ejemplo; si la optimización de algunas consultas es máxima, puede que sus tiempos de ejecución resulten inaceptables y deban ser calculados a priori.

2.1.- Estado del Arte

Las actuales arquitecturas más conocidas se basan en los modelos de tres o cuatro capas. Evitando hacer una reseña histórica exhaustiva de los acontecimientos, tomaremos como origen de referencia temporal el momento en que se acuña el término de aplicación Web (en adelante, WebApp) [13]. Una WebApp es aquella donde la capa de presentación se basa en una interfaz Web y de algún modo también los servicios que oferta la aplicación. Aunque a menudo no se presta la atención suficiente, esta última consideración es trascendente. La cuestión es que, conforme se han ido incorporando posibilidades en las WebApp, se han descubierto carencias en la infraestructura subyacente y, consecuentemente, se ha ido incorporando también la tecnología que las minimiza. Por ejemplo, inicialmente las WebApp mostraban sólo contenido estático, mientras que ahora -en el marco de J2EE- la tecnología que permite la generación dinámica de contenidos es la combinación de Servlets y JSPs.

En el contexto de DAWIS, los Servlets son de especial interés -pues tienen una cierta orientación semántica-. Los Servlets son clases Java dedicadas al procesamiento de solicitudes y a la consiguiente generación de respuestas de modo que, aunque explícitamente, soportan parte de la lógica de aplicación.

² Las actualizaciones y digitalización de documentos deberán realizarse desde cada archivo nativo y local, por tanto, no forman parte de DAWIS.

Otros estándares, como CORBA, también manifiestan la necesidad de un amplio abanico de servicios que sean comunes a la mayoría de las WebApp: seguridad, identificación, persistencia, transacciones, mensajería, conectividad, caching, etc. Estos y otros servicios permiten, en conjunto, diseñar y desarrollar los sistemas distribuidos disponibles actualmente. Sin embargo, el puzzle aún no está completo. De nuevo, la parte más débil en todo esto es *la componente semántica de los sistemas*, la lógica de negocio que se materializa en la lógica de control de cada una de las aplicaciones. No es de extrañar, por tanto, que sea en el ámbito de los servidores de aplicaciones donde primero ha saltado la alarma. Para la totalidad de los servidores de aplicaciones existentes en el mercado, *J2EE es el modelo de referencia*.

La tecnología J2EE de los servidores de aplicaciones son los EJBs (Enterprise Java Beans), los componentes J2EE. Evitando entrar en una descripción detallada de esta tecnología, interesa -a pesar del gran éxito de esta tecnología-, citar a continuación algunos de los inconvenientes que plantea:

- Sólo maneja objetos de primera clase (first class, que incluyen un identificador o clave primaria del objeto).
- No hay concepto de dependencia por lo que, entre otros, no es posible determinar si los identificadores se generan externamente por la aplicación, por el contenedor o automáticamente por la base de datos.
- El lenguaje de consulta, EJB-QL, no es sencillo y no soporta muchos de los operadores SQL. Además, es muy transaccional, aún cuando la aplicación no lo necesite.
- Es muy ineficiente y, consecuentemente, no escala adecuadamente.
- No hay una separación clara entre los componentes y la lógica precisa para proveer persistencia a los mismos.
- Los EJBs no consideran metadatos ni descripciones de negocio sino que son código únicamente. Por tanto, codifican acciones, pero nada más en lo relativo a semántica.

En DAWIS, el desarrollo de una arquitectura para la integración de ADs en la Web, que provea un modelo de referencia en base al cual construir WebApps, plantea una problemática análoga a la ya descrita. Una aplicación de este estilo es paradigma de integración de datos en diferentes sitios así como de componentes distribuidos que ejecutan funciones muy diversas. Como resultado, la interoperabilidad entre tales componentes es una necesidad característica de estos sistemas. La funcionalidad del conjunto depende de estos intercambios y de la agilidad con la que consiga plasmar la semántica que cada acceso al AD requiera. Sin embargo, la tecnología actualmente disponible (J2EE o CORBA) sólo permite implementar servicios a través de la Web (WebApp) lo cual es muy distinto a soportar el desarrollo, despliegue y aprovechamiento de tales servicios. Para esto último, las WebApp deben construirse conforme a un paradigma diferente de los existentes en la actualidad, con los requisitos básicos existentes que son comunes a todas las WebApp.

Por tanto, entendemos que el paradigma de referencia que contribuya a resolver la problemática del acceso integrado a ADs debe contemplar como característica importante la **semántica de los sistemas**, esto es: a) considerar las interacciones entre los diferentes componentes que constituyen -o pueden constituir en el futuro- una WebApp; y b) debe basarse en el servicio que se ofrece, esto es, dirigido por servicio. Opinamos que las ontologías pueden contribuir en este aspecto semántico de forma notable. Esto supone una diferencia trascendental con los sistemas existentes que únicamente contemplan las interacciones estáticas entre los componentes (las definidas durante la compilación) y las interacciones entre el programa y el usuario.

2.2.- Servicios Web

Los servicios Web (WS) [9] son un nuevo paradigma con dos características esenciales: autocontenidos y autodescriptivos; que son las que permiten publicarlos, localizarlos e invocarlos a través de la red. De tal forma, que una WebApp pasa a ser un conjunto de servicios Web, con lo cual está inherentemente orientada al servicio. Esto coincide perfectamente con los requisitos de una arquitectura de referencia para WebApps dedicadas a ofrecer un acceso integrado a ADs disponibles en Internet. A continuación se muestran los principios de este paradigma que suscitan nuestro interés.

Un servicio Web es una interfaz que describe una colección de operaciones tan complejas como se quiera y a las cuales se puede acceder a través de la red pues, más allá del problema de conectividad, están descritas de una forma estándar [3], [10]. Las abstracciones fundamentales de este modelo son el *servicio* y la *descripción del servicio* que incluye todos los detalles necesarios para interactuar con el mismo (formato de los mensajes intercambiados, detalle de las operaciones que se efectúan, información de conexión y localización, etc). Junto a esto, se define un *conjunto de operaciones* y otro *de roles*. En todo ello, los detalles de la implementación del servicio quedan ocultos, garantizando así la portabilidad e interoperabilidad entre diferentes plataformas.

Las operaciones son de tres tipos: 1) *publish*: publicar cualquier descripción de servicio para que pueda ser accedida (las características del servicio determinan dónde publicar las descripciones). 2) *find*: el solicitante recupera una descripción del servicio o servicios de un determinado tipo o subconjunto. 3) *bind*: las ligaduras ocurren siempre en tiempo de ejecución, y permiten la invocación concreta de un servicio, a partir de los detalles de enlace obtenidos del registro.

Adicionalmente, los roles del servicio también son de tres tipos: el de proveedor (*service provider*), solicitante (*service requestor*) y registro del servicio (*service registry*).

3. Ontologías para dotar de semántica a la web y XML

La web, según su creador Tim Berners-Lee, evolucionará inevitablemente hacia la web semántica [2]. Esto quiere decir que los ordenadores encontrarán el significado de los datos siguiendo los *links* a las definiciones de los términos y reglas clave. Como consecuencia, será más fácil el diseño de servicios automáticos en la web. El problema que tiene este desarrollo es si la información dada por los usuarios no es lo suficientemente rica o las máquinas no la pueden procesar. El modelado de la información parece ser el cuello de botella para la construcción de la web semántica. Para evitar el problema y poder codificar la información que sea necesaria, entre los candidatos, parecen destacar XML y las ontologías.

Una ontología [7] es una especificación formal explícita de una conceptualización compartida. En este contexto, la conceptualización se refiere a un modelo abstracto de cómo piensa la gente sobre las cosas del mundo, generalmente restringido a un área particular o dominio de interés. Y la especificación explícita significa que los conceptos y relaciones del modelo abstracto se dan en definiciones y términos explícitos. La razón más típica para construir una ontología es la de facilitar un lenguaje común para compartir y reutilizar conocimiento de algún fenómeno del dominio de interés. En DAWIS, el dominio de interés es la integración virtual y dinámica de Archivos Digitales en la Web.

Diseñar una ontología no es tarea trivial; y si, además, ésta ha de servir para una audiencia amplia, entonces el reto es mucho mayor. Recientemente se vienen realizando muchas ontologías utilizando distintos enfoques [4] (inspiración, inducción, deducción, síntesis y colaboración), métodos y técnicas. Sin embargo, son escasos los trabajos sobre propuestas metodológicas para construir y utilizar ontologías. Si se contara con una metodología sistemática, con métodos bien definidos para la construcción de la ontología, entonces se podría hablar de Ingeniería de las Ontologías. En este sentido, existen, entre otras, las propuestas incipientes de *Methontology* [3], *Ontoclean* [2] y *Soalar* de IBM.

Para la web semántica, la ontología debe expresarse en un lenguaje formal, sin ambigüedad en su interpretación. De esta manera, las ontologías -entendidas como representaciones explícitas de conocimiento compartido- se pueden utilizar para codificar la semántica de la terminología [7]. Junto a esto, contamos con el hecho de que XML es una tecnología mucho más madura que las ontologías en lo que a tamaño de la comunidad de usuarios y disponibilidad de herramientas se refiere. Algunos autores [8] aseguran que, las ontologías se pueden adoptar en situaciones donde la capacidad de representar la semántica sea tan importante como para olvidar las ventajas de madurez que ya ofrece XML. No obstante, también cabe destacar que las ontologías se adoptarán para la web semántica si se dispone de herramientas de desarrollo que se puedan utilizar por los ingenieros del conocimiento. Y posiblemente, el primer paso para la evolución hacia la web semántica sea el desarrollo de ontologías descentralizadas y adaptativas para la especificación del software.

En DAWIS, adoptaremos las ontologías como paradigma para reducir la complejidad o incertidumbre donde sea preciso (la semántica reduce la complejidad), para su uso en la web semántica; y XML será la tecnología a utilizar en cualquier caso para la compartición de datos, principalmente la parte XMI de CWM (www.omg.org).

El amplio consenso de XML (o de html) se enfoca mayoritariamente a la expresión sintáctica de la comunicación, y hace resurgir el problema de la semántica como aquello que trasciende a la integración sintáctica de términos. Dichos términos pueden, incluso, venir expresados con iguales estructuras sintácticas y poseer significados diferentes (y viceversa) [1].

En el proyecto DAWIS, que ahora iniciamos, trataremos de diseñar e implementar ontologías para permitir el acceso integrado, flexible y dinámico a múltiples ADs a través de la web, como si de uno solo se tratara. Tendremos muy en cuenta recientes iniciativas como *Observer* [12] para: a) mejorar el procesamiento de consultas en la globalidad descrita, b) la forma de proporcionar descripciones semánticas a los datos del repositorio global utilizando las ontologías y c) para el diseño de estrategias que permitan un enriquecimiento incremental de respuestas a partir de las ontologías previas que existan en cada momento.

4. Conclusiones

DAWIS es un proyecto coordinado cuyo objetivo es posibilitar el acceso flexible, virtual, dinámico e integrado a múltiples archivos digitales a través de la Web. Para ello, se plantean los siguientes sub-objetivos: a) el desarrollo sistemático y (semi-) automático de ADs, así como de portales Web que permitan el acceso integrado a los mismos; y b) la integración de ADs basada en una arquitectura de referencia, cuya especificación y construcción es una parte central de este proyecto. En este momento, el proyecto inicia su andadura. Este trabajo y el descrito en [11] resumen los aspectos más relevantes que pensamos acometer durante los tres próximos años.

5. Referencias

- [1] Bermudez, J. *Una lógica de descripciones en un nivel meta-ontológico para la gestión de sistemas de información globales*. Tesis Doctoral, Dpto. de Lenguajes y Sistemas Informáticos, Universidad del País Vasco, Nov. 2001
- [2] Berners-Lee, T., Hendler J., Lassila O. *The semantic Web*. Scientific American 284, 5, pp. 34-43, May 2001.
- [3] Brittenham, P. *Web Services Development Concepts (WSDC 1.0)*, IBM Software Group, May 2001.

- [4] Clyde W. Hossapple and K. D. Joshi. *A collaborative approach to Ontology design*. Communications of the ACM. Vol. 45 n. 2, pp.42-47, Feb. 2002.
- [5] Fernández, M., Gomez-Perez A. and Jurista N. *Methondology: From ontological art towards ontological Engeneering*. In Spring Symposium Series, Standford University, 1997.
- [6] Guarino, N. and Welty, Ch. *Evaluating Ontological Decision with ONTOCLEAN*. Communications of the ACM. vol. 45 n. 2, pp. 42-47, Feb. 2002.
- [7] Gruber, T. R. *Towards principles for the design of ontologies used for knowledge sharing*. Int. Workshop on Formal Ontology, 1993.
- [8] Kim, H. *Predicting how ontologies for the semantic web will evolve*. Communications of the ACM. Vol. 45, n. 2, pp. 47-51, Feb. 2002.
- [9] Kreger, H. *Web Services: Conceptual Architecture (WSCA 1.0)*, IBM Software Group, May 2001.
- [10] Leymann, F. *Web Services Flow Language (WSFL 1.0)*, IBM Software Group, May 2001.
- [11] Marcos E., Cáceres P., Cavero JM, Vela B., Costilla C., Eibe S., Menasalvas E. y Sáenz J. *DAWIS:Sistematización del Desarrollo de Portales para el Acceso Integrado a Archivos Digitales en la Web*, Taller RedBD dentro de JISBD 2002, El Escorial, Madrid, Noviembre 2002.
- [12] Mena, E. and Illarramendi, A. *Ontology-Based Query Processing for Global Information Systems*. Kluwer Academic Publishers, July 2001.
- [13] Mohan, C. *Dynamic e-Business. Trends in Web Services*. IBM Almadén Research Center, 2002.
- [14] Shirky, C. *Web Services and Context Horizons*. IEEE Computer, pp. 98-100, September 2002.