IX Congreso de Ingeniería de Organización
Gijón, 8 y 9 de septiembre de 2005

Modelado flexible de asignación de atributos a entidades empresariales

Miguel Gutiérrez[1], Pedro Cocho[2], Alfonso Durán[1]

[1] Área de Ingeniería de Organización. Escuela Politécnica Superior. Universidad Carlos III de Madrid. Avenida de la Universidad 30, 28911 Leganés (Madrid). mgfernan@ing.uc3m.es, duran@ing.uc3m.es
[2] Adalid MyO. c/Berlín 3F, of. 1.16, 30395 Cartagena (Murcia). pcocho@adalidmyo.com

Resumen

En esta comunicación se presenta un metamodelo de entidades empresariales, cuya implementación permite la definición de entidades con asignación de atributos en tiempo de ejecución. Se hace un análisis del modelo y se tratan en detalle sus implicaciones en cuanto a la definición de un lenguaje de modelado empresarial y la necesidad de establecer unas librerías básicas de entidades o diccionarios básicos de ese lenguaje. Del análisis se desprende que una aplicación de software desarrollada de acuerdo con el planteamiento del modelo flexible, frente al enfoque tradicional de la definición previa rígida de un modelo de entidades y atributos, permite un salto importante en la flexibilidad de las aplicaciones de software empresariales. En particular, se detectan potenciales ventajas económicas en las sucesivas adaptaciones que demandan las empresas usuarias del software tras la fase de implantación. El análisis teórico muestra que estas adaptaciones, que derivan en el enfoque tradicional en importantes costes de desarrollo a medida de las modificaciones requeridas, se pueden afrontar de un modo mucho más asequible en un software que implemente el enfoque flexible planteado.

Palabras clave: Modelado empresarial, entidad-relación, UML, desarrollo software

1. Introducción

El desarrollo de software empresarial se basa de forma mayoritaria en tomar como punto de partida el modelado genérico de las entidades que están presentes en una empresa o área de una empresa y sus relaciones. Es necesario por tanto, previamente a la codificación del software, identificar de forma precisa dichas entidades —departamentos, recursos humanos y de equipo, productos, etc.—, y definir sus posibles atributos o características (por ejemplo, en el caso de la entidad *Empleado*: *Nombre*, *Apellidos*, *DNI, Dirección*, *eMail*, *Teléfono de contacto*, *Foto*, etc.). Típicamente, en la fase de implantación del software en una empresa concreta, se define, a partir del modelo genérico inicial referido, el submodelo que mejor se ajusta a la realidad de dicha empresa. Ello se hace seleccionando un subconjunto de entidades y de atributos de dichas entidades. La flexibilidad en la fase de implantación depende en esencia de la riqueza del modelo inicial. Un ejemplo clásico de este enfoque son los modelos de Scheer (1994). Actualmente, la empresa líder en sistemas ERP (*Enterprise Resource Planning*, planificación de recursos de empresa), SAP, tiene una utilidad para componer una solución específica a partir de una solución genérica prediseñada, que consta de un conjunto de módulos entrelazados (SAP, 2005a).

Sin embargo, la flexibilidad posterior a la implantación, es decir, transcurrido un cierto plazo y con el software ya en uso, es mucho más limitada. Esto es debido a que el funcionamiento del software está ligado al submodelo específico. En tiempo de ejecución no se definen nuevas

entidades ni se asignan nuevos atributos, sino que, según tienen lugar los múltiples procesos empresariales y en función de la interacción con los usuarios, el software genera ejemplares concretos de las entidades previamente modeladas y modifica los valores de los atributos respectivos de las entidades, también previamente definidos. Se pueden generar tantos ejemplares como se quiera, pero los objetos que constituyen esos ejemplares están pues restringidos a la definición inicial de la entidad (o clase en desarrollo orientado a objetos) correspondiente. Si la empresa usuaria requiere algún tipo de adaptación o modificación que altera el submodelo concreto, se incurre habitualmente en unos costes de desarrollo de la adaptación elevados. Así ocurre en la mayoría de los sistemas ERP y las aplicaciones software específicas, pues está desarrollada siguiendo este enfoque tradicional.

Hace unos años, en respuesta a esta problemática de la falta de flexibilidad del software en la adaptación a los sucesivos nuevos requerimientos específicos de las empresas usuarias, la empresa Adalid MyO (cuyo actual director es coautor de la presente comunicación) comenzó a desarrollar un software empresarial en el que se buscaba dicha flexibilidad siguiendo un enfoque alternativo consistente en admitir, en tiempo de ejecución, no sólo la creación de ejemplares de las entidades empresariales previamente modeladas, sino la definición de nuevas entidades y la asignación y modificación de los atributos de las entidades. El desarrollo tuvo una primera materialización en un sistema software que pasó a comercializar. Este sistema, siguiendo el nuevo enfoque, presenta una flexibilidad mucho mayor en la adaptación a las necesidades cambiantes de los clientes. Sin embargo, las exigencias derivadas de poner en marcha el software y entrar en fase de comercialización, llevó a acotar la implementación de las posibilidades dinámicas del mencionado enfoque alternativo.

Los resultados y la realimentación con comentarios positivos provenientes de los usuarios fueron afianzando la potencialidad e interés del nuevo enfoque, y llevaron a Adalid MyO junto con el Área de Ingeniería de Organización de la Universidad Carlos III de Madrid a promover un proyecto de investigación con el objeto de definir e implementar un modelo conceptual de este enfoque flexible, que permitiese la explotación de toda su potencialidad. Incluido en los objetivos del proyecto, y ligado a dicha explotación, se explicitaba en la definición el interés de desarrollar un lenguaje pseudo-natural orientado a facilitar la creación dinámica de entidades. En este marco, y como resultado parcial del proyecto actualmente en curso, el trabajo que se presenta a continuación recoge la propuesta de un modelo flexible que posibilita la asignación dinámica de atributos a entidades empresariales definidas en tiempo de ejecución.

En el siguiente apartado se describe dicho modelo, del cual se derivan implicaciones relativas a dos aspectos particularmente interesantes y relacionados: por un lado la posibilidad de crear un lenguaje pseudo-natural derivado del modelo, según se desarrolla en el tercer apartado; y por otro, la necesidad de una librería básica de entidades junto con la utilidad de contar con librerías adicionales para una explotación efectiva, según se comenta en el cuarto apartado. En el quinto apartado se discute acerca de las facilidades que ofrece del modelo para hacer modificaciones y adaptarse a requerimientos cambiantes de las empresas usuarias. Seguidamente se comenta brevemente el estado actual de desarrollo del proyecto del cual surge este trabajo y la línea de continuación, para pasar finalmente al apartado de conclusiones donde se comentan de forma resumida las principales ventajas del modelo propuesto, arrojadas por el análisis del modelo y de sus implicaciones.

2. Descripción del modelo

Para lograr el grado de flexibilidad requerido, el enfoque propuesto se basa en la implementación directa de un metamodelo de entidades y objetos, que podría considerarse una particularización específica para los requerimientos del software empresarial de los metamodelos de los lenguajes de modelado habituales en el desarrollo software, como el del lenguaje estándar UML (OMG, 2003). De esta manera, en lugar de seguir el enfoque tradicional de crear un modelo concreto siguiendo la especificación del metamodelo —por ejemplo crear un modelo UML que recoja la actividad empresarial objetivo, ajustándose a la especificación dada por el metamodelo UML—, lo que se hace es desarrollar un modelo genérico en dos capas que incluye, por un lado, el propio *metamodelo de entidades*, y por otro y ligado al anterior, el *metamodelo de objetos* que resultan al tomar realidad esas entidades. Además, se ligan ambos metamodelos a un *metamodelo de tipos de dato* y valores, según muestra el modelo UML de la Figura 1. La clave para lograr la flexibilidad surge de la implementación de este modelo genérico, dejando la definición de las entidades específicas, con la correspondiente asignación de atributos, a la fase de ejecución, a diferencia de lo que se hace en el enfoque habitual.

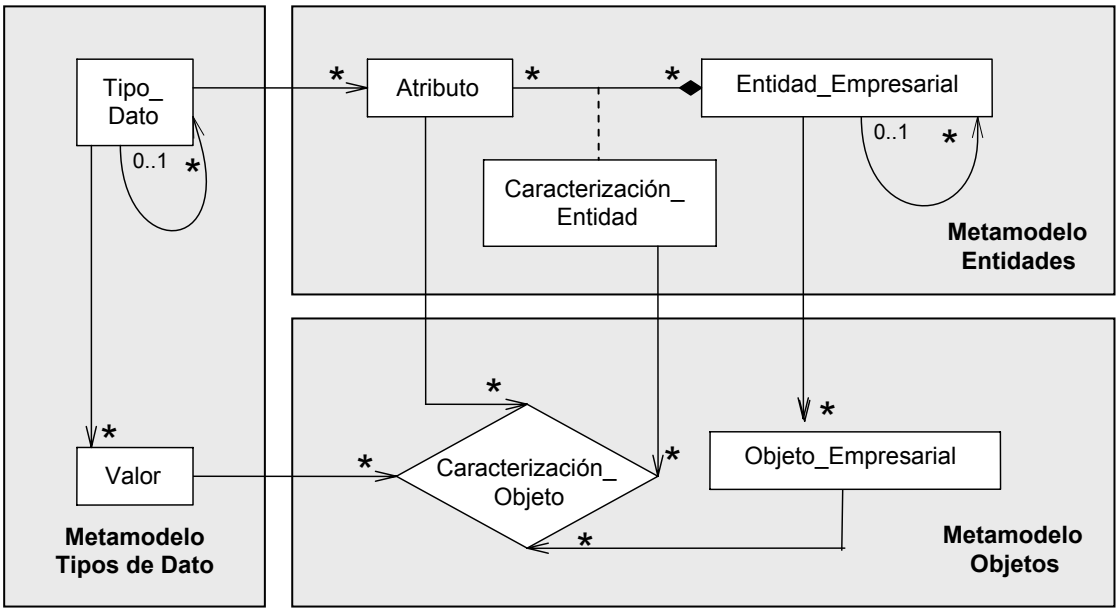

Figura 1. Modelo UML de asignación flexible de atributos a entidades empresariales

El *metamodelo de entidades* consiste en una jerarquía de *entidades empresariales* (jerarquía de clases), cada una de las cuales caracterizada a través de la asignación de un conjunto de *atributos*. La jerarquía permite la herencia de atributos de las entidades padre a las entidades hijo. Una entidad heredará todos los atributos de sus entidades jerárquicamente superiores, y además podrá tener asignados sus propios atributos específicos. La herencia se implementa mediante un método propio de la clase *entidad_empresarial*. Por razones de claridad, en el modelo no se especifican los métodos propios de cada clase, que han de implementarse según la metodología estándar.

Según muestra el *metamodelo de objetos*, de cada ejemplar de *entidad empresarial* se producirán múltiples ejemplares de *objetos empresariales*, que se caracterizarán mediante la asignación de valores concretos de los atributos que les corresponden. De este modo, los atributos pueden existir conceptualmente de forma independiente a las entidades —en el

metamodelo de entidades—, pero, como se representa en la Figura 1, los ejemplares de atributos sólo existen ligados a un ejemplar de objeto empresarial concreto y nunca de forma autónoma.

Para lograr la flexibilidad derivada de la asignación dinámica de atributos a entidades empresariales, es necesario que, cuando en tiempo de ejecución se defina un atributo genérico —el cual podrá a continuación relacionarse con una entidad—, sea posible especificar el tipo de dato asociado a dicho atributo. Coherentemente, cuando un objeto empresarial se caracterice mediante la asignación de valores concretos de los atributos, estos valores deberán pertenecer a los tipos de dato respectivos. Según se comentó al principio del apartado, esto se logra mediante la relación de los dos metamodelos descritos con el *metamodelo de tipos de dato*. Los tipos de dato también guardan una relación jerárquica entre ellos. Se relacionan con el *metamodelo de entidades* en la definición de los atributos, en la que es necesario asignarles un tipo de dato. De cada tipo habrá posteriormente un conjunto de *valores* (ejemplares), que servirán para dar realidad a los atributos y caracterizar los objetos en el *metamodelo de objetos*.

En definitiva, el enfoque planteado se basa en un modelo a implementar que consta de dos capas: una de clases, en la cual se definen las entidades empresariales genéricas y los atributos que las caracterizan, teniendo cada atributo un tipo de dato genérico; y otra de objetos, en la que las entidades y los atributos toman realidad respectivamente en los objetos empresariales correspondientes y su caracterización mediante la asignación de valores concretos. De esta estructura de dos capas se derivan dos cuestiones relacionadas y de particular interés: por un lado, la posibilidad ya contemplada desde los objetivos del proyecto de definir un lenguaje de modelado empresarial con unos elementos constitutivos básicos que permitan especificar el modelo concreto de una empresa sin necesidad de compilación del software; y por otro, la necesidad práctica de contar con una librería básica inicial de entidades, que se deberá precargar antes de comenzar una implantación del software, y que, en términos del lenguaje, no será sino un diccionario básico a partir del cual comenzar la definición de la empresa y el enriquecimiento con nuevas palabras —surgidas de entrada como consecuencia de la definición de la empresa y posteriormente de la gestión del día a día—.

3. Implicaciones para la definición de un lenguaje de software empresarial

Al dejar para la fase de ejecución la definición de entidades y la asignación flexible de atributos, surge de forma lógica el interés en desarrollar un lenguaje básico cercano al natural, que permita explotar esa funcionalidad a un usuario no experto. En el enfoque tradicional, la definición de una nueva entidad o la asignación de nuevos atributos a entidades existentes, implicaría, siempre que no formasen parte del modelo genérico que se comentó al inicio de la introducción, una necesidad de intervención experta en el software, y, en general, una necesidad de compilación para satisfacer la funcionalidad asociada a la nueva entidad/atributo.

Con el enfoque basado en el modelo de dos capas del apartado anterior, se abre la posibilidad de añadir y modificar entidades sin tener que actuar sobre el código o la definición de la base de datos que soporta el software. Si por ejemplo se implementa el modelo de dos capas de la Figura 1 en una base de datos, y se hace corresponder una tabla a la *entidad empresarial*, otra al *atributo* y otra la *caracterización de la entidad*, la creación de una entidad o un atributo se traduce en una instrucción de adición de un nuevo registro en las tablas respectivas, indicando, en el caso de la entidad, la entidad padre (si la tiene), y, en el caso del atributo, el

tipo de dato correspondiente. Igualmente, la asignación de un atributo a una entidad, una vez que ambos están registrados, pasa a ser la adición de un registro en la tabla de *caracterización de entidad*.

Es interesante observar que todas las acciones anteriores recaen en la capa genérica del modelo. De este modo, se infiere del modelo un primer lenguaje construido a partir de las acciones básicas de interacción con una base de datos, que ya cuentan con un lenguaje pseudo-natural estándar, el SQL (*Structured Query Language*, lenguaje estructurado de consultas) (Date y Darwen, 1997), más las partículas correspondientes a los elementos de la capa genérica: *entidad empresarial* y *atributo*, junto con la relación que los une, la *caracterización de la entidad*. Así, las acciones básicas ligadas al software empresarial pueden expresarse adaptando las acciones básicas del lenguaje SQL. Por ejemplo, la creación de entidades y de caracterizaciones se puede plasmar en las siguientes expresiones en lenguaje de alto nivel, en las que los corchetes significan elementos opcionales de la sintaxis:

> CREAR ENTIDAD <Nombre de la Entidad> [DESCENDIENTE DE <Nombre de la Entidad Padre>]
> CREAR ATRIBUTO <Nombre del Atributo> DE TIPO <Nombre del Tipo de Dato>
> CARACTERIZAR <Nombre de la Entidad> CON <Nombre del Atributo1> [, <Nombre del Atributo 2>, ...]

Una cuestión importante en relación con este enfoque, es que, si bien sería posible también crear *tipos de dato* del mismo modo, en la práctica, y dado que cada tipo de dato llevará una funcionalidad asociada a sus operaciones, se hace necesaria la definición previa de los tipos de dato con la codificación de sus operaciones. Por otra parte, para que un software que implemente este enfoque tenga utilidad práctica, es preciso tener un diccionario básico de entidades y atributos, además de los tipos de dato. Esto tiene una serie de implicaciones que se pasan a discutir en el siguiente apartado.

4. Implicaciones para la definición de librerías básicas

Según se desprende de lo comentado, si no se cuenta al comenzar la implantación del software en una empresa con un conjunto de entidades y atributos básico, se emplearía mucho tiempo en ponerlo en funcionamiento. Habría que definir y caracterizar toda la jerarquía de entidades empresariales. Sin embargo, gran parte de esa jerarquía es común a todas las empresas. En términos del lenguaje de modelado, se puede decir que hay un diccionario básico de términos con el que comenzar a hacer frases. El diccionario básico incluiría, por un lado, los tipos de dato, y por otro, un conjunto primario de entidades empresariales posibles (como *cliente*, *suministrador*, *producto*, *pedido*, etc.) y atributos (como *DNI*, *dirección*, *teléfono*, *precio*, etc.). Eso sí, al implantar el software en una empresa, se podrían particularizar estas entidades estándar sin necesidad de codificar ni compilar, asignando o quitando atributos, modificando los tipos de dato de los atributos, y cambiando la jerarquía de entidades o ampliándola añadiendo niveles más detallados de acuerdo con criterios específicos.

Por tanto, si los elementos básicos de la capa genérica del modelo —junto con las adaptaciones de los términos del lenguaje SQL— determinan los términos reservados del lenguaje de modelado empresarial (ENTIDAD… DESCENDIENTE DE, ATRIBUTO… DE TIPO, CARACTERIZAR… CON, etc.), las entidades ejemplares de esos elementos básicos constituyen el diccionario del lenguaje. El lenguaje esbozado en el apartado anterior sería pues una porción de un auténtico lenguaje de modelado empresarial, ya que precisamente

estos lenguajes y pseudo-lenguajes, al igual que las utilidades de modelado empresarial gráficas, tienen como objetivo la definición de la capa genérica de un modelo planteado de acuerdo a la filosofía de este trabajo.

El mismo papel que juegan en los sistemas ERP los diferentes modelos de partida para diferentes sectores y tipos de empresa —empresa farmacéutica, de automoción, química, de seguros, etc.— con el objetivo de reducir el esfuerzo de definición del modelo específico (SAP, 2005b), (SASGlobal, 2005), (Oracle, 2005), etc., lo jugaría en el enfoque descrito una serie de diccionarios para estos entornos de la capa genérica del modelo. Al ir haciendo implantaciones, se iría igualmente enriqueciendo el conjunto de diccionarios empresariales de partida.

La ayuda que algunos sistemas tienen para la definición del modelo específico a partir de unas configuraciones y módulos básicos, como el configurador de SAP ya referido en la introducción (SAP, 2005a), podría también tener su equivalente en un software diseñado con el enfoque de este trabajo. En particular, para evitar tener que definir desde cero toda la empresa, parece factible contar con una herramienta gráfica en la que se presentase al usuario uno o varios árboles desplegables de entidades jerárquicas, al modo de los programas de navegación de ficheros (como el Explorador de MS Windows), en los que, al señalar con el ratón una entidad, se presentase en un marco paralelo un conjunto básico de atributos predefinidos con sus tipos de dato. El usuario iría modificando la lista de atributos, eliminando algunos, incorporando otros —bien de una lista pre-existente, bien definidos a medida—, y también cambiando según las necesidades los tipos de dato asignados por defecto. El árbol de entidades se modificaría igualmente, añadiendo y eliminando entidades o cambiando la jerarquía.

Al contar con una librería básica, la asignación de conjuntos de atributos, que tendría el inconveniente de llegar a ser una tarea muy tediosa, se aceleraría drásticamente. En definitiva, una de las funciones de la librería sería contar precisamente con jerarquías de entidades, teniendo asociada cada entidad un conjunto coherente de atributos del cual una empresa usuaria pudiera aprovecharse para la definición de sus propias entidades. Con acciones gráficas como arrastrar esos conjuntos o subconjuntos a las entidades deseadas, el proceso de modelado de la empresa sería suficientemente ágil y más aún a medida que se contara con ese conjunto de diccionarios adaptados a los diferentes sectores comentado antes.

Ligado a este proceso, surge un aspecto importante a considerar. Al seleccionar conjuntos y subconjuntos de atributos, podría detectarse la necesidad de que una entidad precisara una herencia múltiple. En el modelo de la Figura 1, como se indicó en el segundo apartado, se hace la hipótesis de que las entidades guardan una jerarquía arborescente. Una entidad hijo sólo hereda de la entidad padre y sus ascendientes. No se contempla en el modelo la posibilidad de heredar directamente de varios padres (que sería una jerarquía de grafo, no arborescente), que puede representar situaciones reales empresariales, como que una empresa sea a la vez cliente y suministradora. Ante esta situación caben dos alternativas, cada una con sus ventajas y sus inconvenientes: bien modificar el modelo para que permita la herencia múltiple, o bien mantener la herencia simple y prestar alguna ayuda para gestionar estas situaciones.

Si se permite la herencia múltiple, que significaría que una entidad heredaría atributos provenientes de dos padres y sus respectivos ascendentes, se complica el modelado de la empresa a partir de las librerías básicas. Es mucho más intuitivo para la empresa ir

determinando una herencia en árbol, donde las entidades van adquiriendo una especialización progresiva a lo largo de una rama. Al permitir la multiplicidad de padres directos, si en un momento dado se quiere añadir un conjunto de atributos que se localiza en alguna entidad de las aportadas por las librerías, se entraría en la duda de si hacer que la entidad tuviera dos padres o de si en realidad lo que se quiere es simplemente añadir los atributos. A esto se suma que desde el punto de vista del software se complica la programación. Por otra parte, las ventajas vendrían de poder caracterizar de una forma precisa los casos como el comentado cliente-suministrador.

Si en el compromiso anterior pesan los inconvenientes, como parece intuirse, podría programarse la herramienta de modelado de manera que asistiera para casos en los que se detecta la dualidad de una entidad. La salida sería crear una entidad especializada, dejando al usuario la elección sobre de cuál de las ramas colgar la nueva entidad, y atribuirle a la nueva entidad mixta todos los atributos de ambas entidades. En el caso de cliente-suministrador, se crearía una nueva entidad, bien un tipo especial de cliente o bien un tipo especial de suministrador, con las atribuciones de ambas.

Esta herramienta no sería en realidad sólo una herramienta de configuración, sino auténticamente una herramienta de modelado empresarial. Cada acción del usuario tendría un equivalente en el lenguaje comentado en el apartado anterior. La herramienta gráfica supondría por tanto una alternativa equivalente a dicho lenguaje. Una ventaja inmediata de este enfoque es que, de una manera muy asequible, la empresa cliente podría tener una participación más activa y por una vía más intuitiva que la actual. Se mejoraría la transmisión de requerimientos, y la fase de implantación, especialmente crítica en los sistemas ERP (Davenport, 1998), (Barker y Frolick, 2003), (Umble *et al*., 2003), se afrontaría con mayores garantías.

Por otra parte, la creación de los ejemplares iniciales de la capa de objetos se haría de igual modo a como ocurre en el enfoque tradicional. Es decir, una vez definido el modelo específico, y aún en la fase de implementación, se rellenaría la base de datos correspondiente con los objetos concretos de la empresa, mediante interfaz gráfica y/o migrando de otras aplicaciones existentes. Posteriormente, en régimen permanente, seguirían creándose y modificándose los ejemplares de la capa de objetos, como consecuencia del funcionamiento de la empresa en el día a día. Sin embargo, con el modelo flexible aparecerían diversas cuestiones relacionadas con estos ejemplares. En particular, ¿cómo se podría aprovechar la flexibilidad del planteamiento para introducir cambios en el modelo, ya en el régimen permanente, y de manera que el conjunto numeroso de objetos creado pudiese actualizarse tras el cambio sin un desarrollo específico? En el siguiente apartado se incluye una discusión a este respecto.

5. Flexibilidad en las modificaciones del modelo

Resulta muy habitual que, una vez implantado un software de gestión empresarial, a la empresa usuaria le surjan cada cierto tiempo nuevos requerimientos y/o necesidad de adaptaciones. Una participación más activa en la fase de implantación ya ayuda a que las necesidades derivadas de unos requerimientos incompletos disminuyan. Pero, por un lado, es difícil en todo caso captar todos los requerimientos de entrada, y por otro, el paso del tiempo genera nuevas necesidades. Por ello, una vez entrado en régimen permanente, aparece frecuentemente el interés de alterar el modelo de la empresa plasmado en el sistema.

El modelo flexible que se explora en esta comunicación permitiría plantear estas alteraciones, con el software ya en uso, y resolverlas de una forma poco costosa. Cada tipo de alteración tendría sus implicaciones y problemas a resolver, derivados de la existencia de objetos empresariales creados bajo un modelo que pasaría a ser sometido a modificaciones. Algunos de los problemas derivados de las modificaciones serían los habituales ante un cambio de modelo de este tipo:

- Una adición de un atributo a una entidad empresarial plantearía la cuestión de cómo asignar los valores de ese atributo a todos los objetos existentes, ejemplares de esa entidad. Podría asignárseles un valor por defecto, dejar el atributo sin valor para los objetos de esa entidad anteriores al cambio, etc.
- La supresión de un atributo de una entidad podría o no implicar la supresión de los valores de ese atributo en los objetos creados previamente. Borrar esos valores implicaría borrar cierta historia de la empresa. Una posibilidad sería mantener ese atributo para los objetos ya creados, poniéndole una marca de eliminación a partir del cambio, que evitase que los objetos posteriores de esa entidad lo tuvieran asignado.

La diferencia que presenta el enfoque flexible de la Figura 1, consiste en que, de una manera muy sencilla, permitiría implementar una solución a las cuestiones anteriores: incorporar un periodo (o periodos) de validez a la asignación de atributos a entidades empresariales, que se registraría/n asociados a la clase *caracterización_entidad* del modelo de la Figura 1. De este modo, de forma natural, se contaría con un conjunto consistente de datos, y además estaría registrada la información de los cambios realizados. No haría falta ningún tipo de intervención sobre el código del software ni sobre la base de datos, y los cambios podría realizarlos un usuario no experto.

La solución anterior también facilitaría algo que en un software tradicional supondría una gran complicación: alterar la jerarquía de entidades, esto es, añadir entidades por ejemplo por especialización de las existentes (querer afinar en la definición de los recursos de la empresa, por ejemplo) o variar las dependencias jerárquicas. Definir el nuevo árbol sería sencillo de acuerdo con la herramienta gráfica de modelado. Surgiría entonces de nuevo el problema del cambio de atributos, que podría resolverse de forma automatizada y de manera similar a lo descrito en el párrafo anterior.

Más complicado sería afrontar una situación que no se contempla en los modelos tradicionales y que podría tener cabida en el enfoque flexible. Como consecuencia de esta flexibilidad, cabría incluso plantear una alteración del modelo, consistente en la posibilidad de que un objeto, creado como un ejemplar de una entidad concreta, cambiara de entidad. Si un recurso de la empresa pasase a estar en venta, se habría convertido en un bien comerciable. Sería muy normal que el software, para un recurso, no tuviera preparado el soporte para su venta ni, por ejemplo, los consiguientes apuntes contables. Podría solucionarse teniendo dos objetos diferentes, pero se perdería la historia asociada al recurso, que podría interesar conservar. Igualmente, podría ocurrir que un recurso de un tipo pasase a ser un recurso de otro tipo, o un cambio organizativo, etc.

En este caso, no serviría el histórico de asociación de atributos a entidades antes descrito. Pero podría dársele solución, ampliando el modelo de la Figura 1 para permitir un histórico de la relación entre los objetos empresariales y las entidades empresariales. Un objeto sería una entidad de un tipo en un intervalo dado, y pasaría a convertirse en un objeto de otro tipo de entidad en otro intervalo. Nuevamente, el modelo flexible proporcionaría una solución al

cambio en el modelo de la empresa que podría automatizarse y realizarse incluso sin ningún coste (aparte del derivado de la implementación de la solución en el desarrollo del software, lógicamente).

6. Situación actual: trabajo en curso y perspectivas futuras

Como se comentó en la introducción, el presente trabajo es un resultado parcial de un proyecto de investigación en curso. Una vez formulado el modelo y explorada su utilidad práctica desde un punto de vista teórico, si bien siempre teniendo presentes sus implicaciones prácticas directas, actualmente el proyecto continúa con dos actividades principales:

- Por un lado, se está desarrollando un prototipo que implementa el modelo y la herramienta gráfica de modelado descritos. La idea es ir probando progresivamente su funcionalidad y encontrar las ventajas e inconvenientes del enfoque, alimentando de datos de procesos empresariales reales o que pudieran provenir de procesos empresariales reales.
- Por otro lado, se ha afrontado el modelado, siguiendo la misma filosofía, de toda la parte funcional de la empresa. El modelo de entidades será una parte del modelo completo.

Una vez terminado el modelo completo, el paso inmediato será la extensión del prototipo para poder tener una primera versión de un software de gestión empresarial desarrollado con esta filosofía flexible y con el que probar su potencialidad.

7. Conclusiones

En esta comunicación se ha presentado un modelo de entidades empresariales, cuya implementación permite la definición de entidades con asignación de atributos en tiempo de ejecución. Tras explorar las implicaciones del modelo, se observa que este enfoque, frente al tradicional de la definición previa de un modelo genérico de entidades de la empresa, permite, al menos desde un plano teórico, un salto importante en la flexibilidad de las aplicaciones empresariales.

Se detecta una ventaja en la mayor facilidad de la implementación, al permitir implicar mucho más al usuario del software en el modelado de su empresa, que se hace de una forma muy intuitiva, lo que redunda en una mejoría en la captación de requerimientos. La ventaja se incrementa en relación con las posibilidades que ofrece el enfoque flexible en cuanto a las posibilidades que surgen para modificar el modelo inicial de la empresa y adaptarse a la habitual sucesión de nuevas necesidades, con sus consiguientes nuevos requerimientos. Este punto aparece como el aspecto más ventajoso, en cuanto a que los costes de adaptar el software a unas necesidades cambiantes y específicas disminuirían drásticamente respecto al enfoque tradicional de los sistemas ERP.

Agradecimientos

Este trabajo se realiza en el marco de *ARMORsystem*, proyecto de colaboración de ADALID y UC3M con financiación del FEDER, del CDTI y del Instituto de Fomento de Murcia.

Referencias

Barker, T.; Frolick, M.N. (2003). ERP Implementation Failure: A Case Study. Information Systems Management, Vol. 20, No. 4, pp. 43-49.

Date, C.J.; Darwen, H. (1997). *A Guide to the SQL Standard*. 4th ed. Addison-Wesley.

Davenport, T.H. (1998). Putting the Enterprise into de Enterprise System. *Harvard Business Review*, Vol. 76, No. 4, pp. 121-131.

OMG (2003). *OMG Unified Modeling Language Specification, Ver. 1.5*. http://www.uml.org.

Oracle (2005). Oracle Solutions for Industries. http://www.oracle.com/industries/index.html. Consultado en mayo 2005.

SAP (2005a). *Solution Composer: Quick Guide*. http//www.sap.com/businessmaps. Descargado en mayo 2005.

SAP (2005b). *SAP – Industries*. http://www.sap.com/industries/index.epx. Consultado en mayo 2005.

Scheer, A.-W. (1994). *Business Process Engineering: Reference Models for Industrial Enterprises*. 2nd ed. Springer-Verlag.

SSAGlobal (2005). *SSA Global Enterprise Solutions*. http://www.ssaglobal.com/industries/index.aspx. Consultado en mayo 2005.

Umble, E.J.; Haft, R.R.; Umble, M.M. (2003). Enterprise Resource Planning: Implementation Procedures and Critical Success Factors. *European Journal of Operations Research*, Vol. 146, No. 2, pp. 241-257.