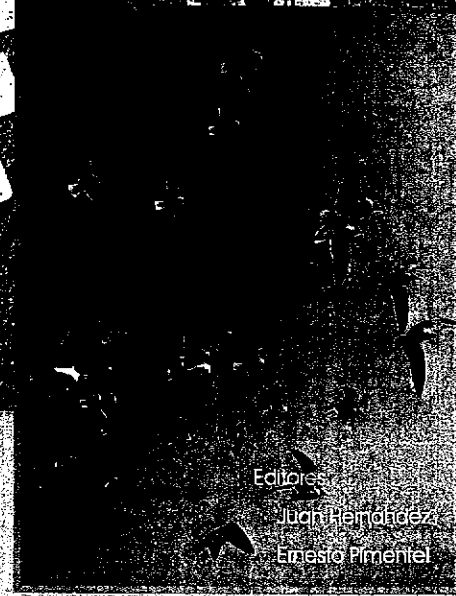
JISBD 2004

Jornadas de Ingeniería del Software y Bases de Datos

Málaga, 10 - 12 Noviembre 2004

http://jisbd2004.lcc.uma.es



Editores:
Juan Hernández
Ernesto Pimentel

ACTAS

IX Jornadas de Ingeniería del Software y Bases de Datos

Málaga, 9 al 12 de Noviembre de 2004

I.S.B.N. 84-688-8983-0
D.L.: MA-1468-2004

Imprime: AltaGrafics

Prólogo

Las IX Jornadas de Ingeniería del Software y Bases de Datos han constituido, de nuevo, un punto de encuentro en el que profesionales y académicos de España y Portugal de ambos campos han podido compartir experiencias y resultados en un ambiente de colaboración entre distintos grupos de investigación, desarrollo e innovación tecnológica. Estas jornadas vieron su nacimiento en Sevilla en 1996, cubriendo inicialmente el área de la Ingeniería del Software, e incorporando en la edición que se celebró en Cáceres en 1999 el campo de las Bases de Datos. Desde entonces, se han consolidado como uno de los eventos celebrados en España de mayor prestigio en estas áreas.

Las IX Jornadas de Ingeniería del Software y Bases de Datos representan una edición especial, con la existencia de un Comité de Programa único formado por investigadores expertos en las materias de interés de las Jornadas.

El presente volumen contiene los trabajos seleccionados por el Comité de Programa para su presentación en la novena edición de estas Jornadas, ***JISBD'2004***, celebradas en Málaga durante los días 10, 11 y 12 de Noviembre de 2004. Se recibieron un total de 121 contribuciones, y cada una de ellas fue revisada por al menos tres miembros del Comité de Programa, quienes tras una ardua labor, dado el número y calidad de los artículos recibidos, seleccionaron 38 trabajos. Adicionalmente, se consideraron 14 trabajos para su presentación en la modalidad de artículos cortos. El programa científico se ha completado con dos excelentes conferencias desarrolladas por investigadores de gran prestigio a nivel internacional. La conferencia de apertura, *Towards Active Software*, fue dictada por el Dr. Ivar Jacobson, quien actualmente trabaja para Rational y Jaczone AB. Las contribuciones del Dr. Ivar Jacobson en múltiples campos de la Ingeniería del Software son ampliamente conocidas, entre las que merece la pena destacar el diseño del Lenguaje Unificado de Modelado (UML) y el proceso para el desarrollo de software basado en componentes (RUP). La conferencia en el campo de las Bases de Datos, *Enhancing the Web with DB Technology*, fue impartida por el Prof. Timos Sellis, Profesor de la *National Technical University* de Atenas, Grecia. El Profesor Timos Sellis, uno de los investigadores europeos más importantes en el campo de las Bases de Datos, recibió el premio Presidencial Jóvenes Investigadores del Presidente de los Estados Unidos, así como el premio de VLDB 1997, por sus trabajos en el área de las Bases de Datos.

Otra actividad que, con cada edición, va adquiriendo mayor relevancia, y que complementan el atractivo de las jornadas es la organización de talleres y tutoriales sobre temas muy específicos y/o de especial interés. Tuvieron lugar, en su mayor parte el día 9 de Noviembre, precediendo a la conferencia principal, un total de 9 talleres y 4 tutoriales con una nutrida y diversa participación.

La celebración de un evento de las características de JISBD, con una participación cada vez más numerosa y consolidada, y con unas exigencias de calidad que se van incrementando en cada edición, no podría realizarse sin la dedicación de los miembros del Comité Organi-

zador del Grupo de Ingeniería del Software de la Universidad de Málaga, a los que queremos agradecer el trabajo desarrollado en beneficio del éxito de las jornadas. Asimismo, queremos agradecer la labor del Grupo Quercus de Ingeniería del Software de la Universidad de Extremadura, quienes han estado a cargo de todo el sistema de recepción y revisión de artículos. Del mismo modo, no podemos olvidar que el objetivo último de este congreso es hacer posible que los investigadores y desarrolladores compartan y debatan sus ideas, y para ello hemos de agradecer y reconocer el trabajo desarrollado por los miembros del Comité de Programa y los revisores adicionales, que han contribuido a conformar un programa científico atractivo y de gran calidad. También queremos agradecer el soporte recibido por las entidades patrocinadoras y colaboradoras.

Deseamos, finalmente, que la próxima edición de estas jornadas, que se celebrarán en Granada en el 2005 como parte del I Congreso Español De Informática (CEDI), mantenga la tendencia de estas últimas ediciones en el nivel de participación .

Málaga, Noviembre 2004

Juan Hernández
Presidente del Comité de Programa
JISBD'2004

Ernesto Pimentel
Presidente del Comité de Organización
JISBD'2004

Índice

Artículos cortos

PRESIDENCIA DEL COMITÉ PROGRAMA

Juan Hernández	Universidad de Extremadura

MIEMBROS COMITÉ DE PROGRAMA

José Aldana	Universidad de Málaga
María José Aramburu	Universidad de Castellón
Luís Arriaga da Cunha	Universidad do Évora
Orlando Belo	Universidad do Minho
Pere Botella	Universidad de Politécnica de Cataluña
Nieves Brisaboa	Universidad de La Coruña
Matilde Celma	Universidad Politécnica de Valencia
Rafael Corchuelo	Universidad de Sevilla
Carmen Costilla	Universidad Politécnica de Madrid
Yania Crespo	Universidad de Valladolid
Carlos Delgado Kloos	Universidad Carlos III
Óscar Díaz	Universidad del País Vasco
João Falcão e Cunha	Universidad do Porto
Xavier Franch	Universidad Politécnica de Cataluña
Mario J. Gaspar da Silva	Universidad de Lisboa
Jaime Gómez	Universidad de Alicante
Alfredo Goñi	Universidad del País Vasco
Natalia Juristo	Universidad Politécnica de Madrid
Antonia Lopes	Universidad de Lisboa
Henrique Madeira	Universidad de Coimbra
Esperanza Marcos	Universidad Rey Juan Carlos
Ana Moreira	Universidad. Nova de Lisboa
Juan José Moreno	Universidad Politécnica de Madrid
Juan Manuel Murillo	Universidad Extremadura
Óscar Pastor	Universidad Politécnica de Valencia
Mario Piattini	Universidad de Castilla la Mancha
Antonio Polo	Universidad de Extremadura
Celia Ramos	Universidad do Algarve
Isidro Ramos	Universidad Politécnica de Valencia
José C. Riquelme	Universidad de Sevilla
Antonio Rito Silva	Universidad Técnica de Lisboa
Francisco Ruiz	Universidad de Castilla la Mancha
José Samos	Universidad de Granada
Isabel Sofía Sousa Brito	Instituto Politécnico Beja
Miguel Toro	Universidad de Sevilla
Ambrosio Toval	Universidad de Murcia
José M. Troya	Universidad. de Málaga
Toni Urpi	Universidad Politécnica de Cataluña
Antonio Vallecillo	Universidad de Málaga

interesante será la definición y aplicación de otros criterios de cobertura que permitan reducir el número de nodos del árbol.

Por otro lado, este trabajo podría servir de base para la automatización de la búsqueda de nuevos casos de prueba en la base de datos o para los parámetros de las consultas así como para facilitar al usuario la tarea de añadirlos manualmente.

Además, será necesario llevar a cabo la evaluación empírica de la eficacia del criterio de cobertura establecido para detectar defectos.

Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Ciencia y Tecnología (España) bajo el Plan Nacional de investigación, Desarrollo e Innovación, proyecto TIC2001-1143-C03-03 (IDAS), subproyecto del coordinado TIC2001-1143-C03 (ARGO).

Referencias

1. Chan, M.Y. and Cheung, S.C. *Applying white box testing to database applications*. CSTR, Hong Kong University of Science and Technology, HKUST-CS99-01. 1999.
2. Chays D., Deng, Y., Frankl, P.G., Frankl, P.G., Dan S., Vokolos, F.I. and Weyuker, E.J. *An AGENDA for testing database applications*. Software Testing, Verification and Reliability, 14: 17-44. 2004.
3. Davies, R.A., Beynon, R.J.A. and Jones, B.F. Automating the testing of databases. 1st International Workshop of Automated Program Analysis, Testing and Verification. 2000.
4. Encontre, V. *Empowering the developer to be a tester tool*. Int. Symposium on Software Testing and Analysis, Industry panel. ACM SIGSOFT. 2002.
5. Hartman, A. *Is ISSTA research relevant to industry?* Int. Symposium on Software Testing and Analysis, Industry panel. ACM SIGSOFT. 2002.
6. Kapfhammer, G.M. and Soffa, M.L. *A family of test adequacy criteria for database-driven applications*. ESEC/FSE. ACM. 2003.
7. Suárez-Cabal, M.J., Tuya, J. *Mejora de casos de prueba midiendo la cobertura de sentencias SQL*. VIII Jornadas de Ingeniería del Software y Bases de datos. 2003.
8. Suárez-Cabal, M.J., Tuya, J. *Improvement of test data by measuring SQL statement coverage*. 11th International Workshop on Software Technology and Engineering Practice (STEP 2003). IEEE Computer Society . 2004.
9. Suárez-Cabal, M.J., Tuya, J. *Using an SQL coverage measurement for testing database applications*. ACM SIGSOFT/FSE. 2004.
10. Tassey, G. *The economic impacts of inadequate infrastructure for software testing*. National Institute of Standards and Technology. Planning Report 02-3. 2002.
11. Zang, J., Xu, C. and Cheung, S. C. *Automatic generation of database instances for white-box testing*. 25th International Computer Software and Applications Conference. 2001.
12. Zhu, H., Hall, P. A. V., May, J. H. R. *Software Unit Test Coverage and Adequacy*. ACM Computing Surveys, 49(4) 366-427. 1997.

Un *profile* de UML para diseñar almacenes de datos seguros

Rodolfo Villarroel[1], Eduardo Fernández-Medina[2], Juan Trujillo[3], y Mario Piattini[2]

(1) Departamento de Computación e Informática. Universidad Católica del Maule (Chile)
rvillarr@spock.ucm.cl
(2) Departamento de Informática. Universidad de Castilla-La Mancha (España)
{Eduardo.FdezMedina, Mario.Piattini}@uclm.es
(3) Departamento de Lenguajes y Sistemas Informáticos. Universidad de Alicante (España)
jtrujillo@dlsi.ua.es

Resumen. Los Almacenes de Datos (*Data Warehouses, DW*), Bases de Datos Multidimensionales y aplicaciones de Procesamiento Analítico En-Línea (*On-Line Analytical Processing, OLAP*) son usados como un mecanismo muy poderoso para descubrir información de negocio crucial en la toma de decisiones estratégicas de las empresas. Considerando de extrema importancia la información manejada por este tipo de aplicaciones, es esencial definir y hacer cumplir medidas de seguridad desde las etapas tempranas del diseño del DW. Además, la confidencialidad es un requisito especialmente importante para aplicaciones basadas en modelos multidimensionales (MD), ya que la información de negocio, que es muy sensible, puede ser descubierta ejecutando simples consultas. En los últimos años, se han propuesto una serie de aproximaciones para realizar el diseño conceptual de los DW's siguiendo el paradigma de modelado MD. Sin embargo, ninguna de estas propuestas considera la seguridad como un elemento importante en sus modelos, y por tanto, no permiten especificar restricciones de confidencialidad para ser cumplidas por las aplicaciones que usarán estos modelos MD. En este artículo, analizamos los problemas de confidencialidad de los DW's y presentamos un *profile* del Lenguaje de Modelado Unificado (UML) que nos permite especificar los principales aspectos de seguridad en el modelado conceptual MD, permitiendo diseñar DW's seguros. Adicionalmente, mostramos los beneficios de nuestro *profile* aplicándolo a un ejemplo.

1. Introducción

La seguridad de la información es un serio requisito que debe ser cuidadosamente considerado, no como un aspecto aislado, sino como un elemento que esté presente en todas las etapas del ciclo de vida del desarrollo, desde el análisis de requisitos hasta la implementación y mantenimiento [3, 5]. Se han propuesto diferentes ideas para la integración de la seguridad en el proceso de desarrollo de sistemas, pero sólo consideran la seguridad de la información desde un punto de vista criptográfico [8]. Chung et al. también enfatizan la integración de los requisitos de seguridad en el diseño, ofreciendo a los diseñadores modelos que especifican aspectos de seguridad, pero sin abordar temas específicos de bases de datos y DW's [1]. MOMT [13] es una

propuesta que trata de integrar la seguridad en el modelado conceptual, pero no cubre el proceso de desarrollo completo y no ha tenido mucho reconocimiento. Otra propuesta interesante es UMLsec [9], pero sólo trata con sistemas de información en general, mientras que el diseño conceptual de bases de datos y DW no es considerado. Existe, también, una propuesta metodológica y un conjunto de modelos para diseñar bases de datos seguras [4] para ser implementadas con Oracle9i Label Security (OLS). Unida a la anterior metodología, la propuesta de un Lenguaje de Restricciones de Seguridad Orientado a Objetos (Object Security Constraint Language, OSCL) [14], permite especificar restricciones de seguridad en el proceso de diseño conceptual y lógico de bases de datos. Sin embargo, las propuestas anteriores no consideran el diseño de modelos MD seguros para DW's.

En la literatura encontrar diversas iniciativas para incluir seguridad en DW's. [10, 11, 15, 16]. Muchas de ellas están enfocadas a aspectos específicos relacionados con el control de acceso, la seguridad multinivel, sus aplicaciones en bases de datos federadas, aplicaciones con herramientas comerciales, etc. Sin embargo, ninguna de ellas considera los aspectos de seguridad en todas las etapas del ciclo de desarrollo ni la introducción de la seguridad en el diseño conceptual MD.

En cuanto al diseño de DW's, veremos que varios enfoques se han propuesto para representar las principales propiedades MD a nivel conceptual [7, 17, 19]. Estas propuestas proporcionan sus propias notaciones gráficas no estándares, y ninguna de ellas ha sido ampliamente aceptada como un modelo conceptual estándar para el modelado MD. Recientemente otro enfoque [12, 18] ha sido propuesto para realizar el modelado conceptual MD orientado a objetos (OO). Esta propuesta es una extensión de UML (*profile*), que se define en base a los mecanismos de extensión estándar (estereotipos, valores etiquetados y restricciones) provistos por UML. Sin embargo, ninguno de estos enfoques de modelado MD considera la seguridad como un aspecto importante de sus modelos conceptuales, por lo que no resuelven el problema de la seguridad en estos tipos de sistemas.

En este artículo, presentamos un *profile* que nos permite representar la información de seguridad de los datos y sus restricciones de los DW's en el modelado MD a nivel conceptual. Esta propuesta se basa en el modelo de seguridad multinivel, considerando únicamente la operación de lectura (*read*) ya que ésta es la operación más común en aplicaciones de usuario final. Este modelo nos permite clasificar tanto la información como al usuario en clases de seguridad, y así hacer cumplir el control de acceso obligatorio. El uso de este enfoque, permite implementar los modelos MD seguros con cualquiera de los SGBD que son capaces de implementar bases de datos multinivel, tales como OLS y DB2 Universal Database (UDB).

El resto de este artículo se estructura así: En la sección 2 proponemos la nueva extensión UML para modelado MD seguro. La sección 3 ilustra un caso de estudio aplicando nuestra extensión UML. Finalmente, en la sección 4 presentamos las principales conclusiones e introducimos el trabajo futuro.

2. Extensión de UML para el Modelado Multidimensional Seguro

En este trabajo, nos basamos en un enfoque de modelado MD que utiliza UML para representar las propiedades estructurales de los modelos MD [12, 18]. El objetivo de nuestra extensión de UML es permitir que sea posible diseñar modelos conceptuales MD, pero clasificando la información para definir qué propiedades tiene que poseer el usuario para tener derecho a acceder a la información. Por tanto, consideramos tres etapas principales:

1. Definición precisa de la organización de usuarios que tendrá acceso al sistema MD. Podemos definir un nivel preciso de granularidad considerando tres formas de organización de usuarios: Niveles de seguridad (que indican el nivel de acreditación del usuario), Categorías de usuario (que indican una clasificación horizontal de usuarios atendiendo a ciertos criterios como de separación territorial, departamentos, etc.), y Roles de usuario (que indican una organización jerárquica de usuarios de acuerdo a sus roles o responsabilidades dentro de la organización).
2. Clasificación de la información del modelo MD. Podemos definir para cada elemento del modelo (clase de hecho, clase de dimensión, atributo de hecho, etc.) su información de seguridad, usando una tupla que esté compuesta de un intervalo de niveles de seguridad, un conjunto de categorías de usuario, y un conjunto de roles de usuario.
3. Cumplimiento del control de acceso obligatorio. Las operaciones típicas que los usuarios pueden ejecutar en este tipo de sistemas son operaciones de consulta. La regla de control de acceso para operaciones de lectura es la siguiente: Un usuario puede acceder a una información sólo si, a) el nivel de seguridad del usuario es mayor o igual que el nivel de seguridad de la información, b) todas las categorías de usuario que han sido definidas para la información están asignadas al usuario, y c) el usuario juega al menos uno de los roles que han sido definidos para la información.

Este artículo está enfocado en la segunda etapa, definiendo una extensión de UML que nos permite clasificar los elementos de seguridad en un modelo conceptual MD y especificar restricciones de seguridad. Debemos precisar también que la primera etapa se relaciona con aspectos de políticas de seguridad definidas en la organización por los administradores, lo que está fuera del alcance de este artículo.

De acuerdo a [2], una extensión de UML comienza con una breve descripción y a continuación se describen todos los estereotipos, valores etiquetados, y restricciones de la extensión. Además de estos elementos, una extensión contiene un conjunto de reglas bien formadas. Estas reglas ayudan a definir la consistencia semántica (desde un punto de vista estático) de la extensión. Por lo tanto, definimos nuestra extensión de UML siguiendo el esquema compuesto de los siguientes elementos: descripción, prerrequisitos de la extensión, nuevos tipos de datos, estereotipos / valores etiquetados, reglas bien formadas (definidas tanto en lenguaje natural como mediante un conjunto de invariantes definidas por medio de expresiones OCL), y comentarios. Para la definición de los estereotipos, seguimos la estructura que es sugerida en [6], la que está compuesta de un nombre, la metaclase base, la descripción, los valores etiquetados y una lista de restricciones definidas por medio de OCL. Para la definición de valores etiquetados, son definidos el tipo de valores etiquetados, la multiplicidad, la descripción, y el valor por defecto.

2.1. Descripción

Esta extensión de UML reutiliza un conjunto de estereotipos definidos previamente en [12], y define un conjunto de valores etiquetados, estereotipos y restricciones, que nos permiten crear modelos MD seguros. Los 20 valores etiquetados que hemos definido se aplican a ciertos componentes que son especialmente particulares del modelado MD (hechos, dimensiones, etc.), permitiéndonos representarlos en el modelo MD y en los mismos diagramas que describen el resto del sistema. Estos valores etiquetados representarán la información sensible (en base a los tipos de datos definidos) de los diferentes elementos del modelado MD, y nos permitirán especificar restricciones de seguridad dependiendo de la información de seguridad y del valor de los atributos de los elementos del modelo. El estereotipo *UserProfile* nos ayudará a identificar una clase especial que definirá el perfil de los usuarios del sistema. Un conjunto de reglas bien formadas definen la semántica del modelo. El uso correcto de nuestra extensión es asegurado por la definición de restricciones tanto en lenguaje natural como en OCL. De este modo, hemos definido 7 nuevos estereotipos: uno que especializa el elemento de modelo *Class*, dos que especializan el elemento de modelo *Primitive* y cuatro que especializan el elemento de modelo *Enumeration*.

En la Fig. 1 hemos representado porciones del metamodelo de UML[1] para mostrar donde se ubican nuestros estereotipos. Sólo hemos representado las jerarquías de especialización, debido a que el hecho más importante de un estereotipo es la clase base de la cual el estereotipo se especializa. En esta figura, los nuevos estereotipos se muestran en color gris oscuro, mientras que los estereotipos que reutilizamos a partir del perfil previo [12] están en un color gris claro y las clases propias de UML permanecen en blanco.

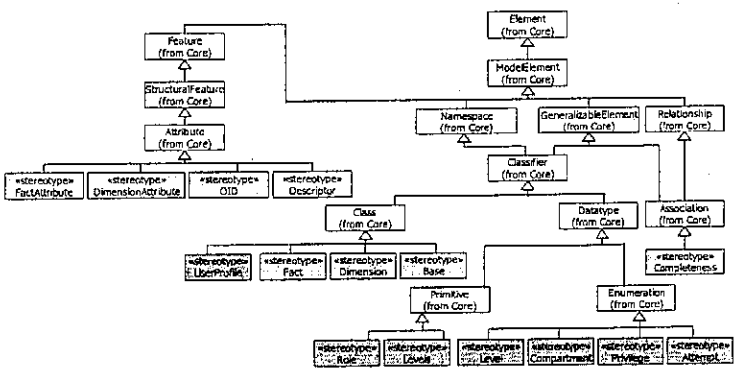


Fig. 1. Extensión de UML con estereotipos

[1] Todas las metaclases vienen del *Core Package*, un subpaquete del *Foundation Package*. Nosotros basamos nuestra extensión en UML 1.5 debido a que éste es el actual estándar aceptado. De acuerdo a nuestro conocimiento, UML 2.0 no es aún un estándar aceptado.

2.2. Prerrequisitos de la Extensión

Este perfil de UML reutiliza estereotipos que fueron previamente definidos en otro *profile* de UML en [12]. Este perfil proporciona los estereotipos, valores etiquetados y restricciones que son necesarias para el cumplimiento de las propiedades del modelado MD conceptual. Para facilitar la comprensión del *profile* de UML que presentamos y usamos en este artículo, ofrecemos un resumen de la especificación de estos estereotipos en la Tabla 1.

Tabla 1. Estereotipos del *profile* de UML para el modelado conceptual MD [12]

Nombre	Clase Base	Descripción
Fact	Class	Clases de este estereotipo representan hechos en un modelo MD
Dimension	Class	Clases de este estereotipo representan dimensiones en un modelo MD
Base	Class	Clases de este estereotipo representan niveles de jerarquía de dimensiones en un modelo MD
OID	Attribute	Atributos de este estereotipo representan atributos OID de clases Fact, Dimension o Base en un modelo MD.
Fact-Attributes	Attribute	Atributos de este estereotipo representan atributos de clases Fact en un modelo MD
Descriptor	Attribute	Atributos de este estereotipo representan atributos descriptor de clases Dimension o Base in a MD model
Dimension-Attribute	Attribute	Atributos de este estereotipo representan atributos de clases Dimension o Base en un modelo MD
Comple-teness	Association	Asociaciones de este estereotipo representan la completitud de una asociación entre una clase Dimension y una clase Base o entre dos clases Base

2.3. Tipos de Datos

Necesitamos la definición de algunos tipos de datos nuevos (ver Fig. 2) que serán usados en nuestras definiciones de valores etiquetados. En la Fig. 1 podemos ver las clases base de las cuales estos nuevos estereotipos se especializan.

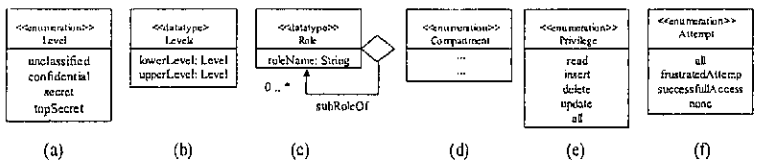


Fig. 2. Nuevos tipos de datos

El tipo *Level* será la enumeración ordenada compuesta por todos los niveles de seguridad que se han considerado (estos valores serán normalmente *unclassified, confidencial, secret y top Secret*). El tipo *Levels* será un intervalo de niveles compuesto por un nivel mínimo y uno máximo. El tipo *Role* representará la jerarquía de los roles de usuario que se pueden definir para la organización (por simplicidad, no consideramos herencia múltiple). El tipo *Compartment* es la enumeración compuesta por todas las categorías de usuario que se han considerado en la organización. El tipo *Privilege* será una enumeración de todos los privilegios que han sido considerados

(estos valores típicamente son *read, insert, delete, update, all*). El tipo *Attempt* será una enumeración de todos los tipos de acceso que han sido considerados (estos valores, son *all, frustratedAttempt, successfullAccess, none*).

2.4. Valores Etiquetados

En esta sección, mostramos la definición de diversos valores etiquetados de la extensión. La Tabla 2 muestra 14 de los 20 valores etiquetados que forman la extensión. Los 6 restantes son SecurityLevel, SecurityRoles y SecurityCompartments aplicados (con la misma semántica que a las clases) tanto a los atributos como a las instancias, y que por motivos de espacio no se han incluido en este artículo.

Todos los valores por defecto de valores etiquetados de seguridad del modelo son colecciones vacías. Por otro lado, el valor por defecto de valores etiquetados de seguridad para cada clase es la menos restrictiva (el menor nivel de seguridad, jerarquía de rol de seguridad que ha sido definida para el modelo y el conjunto vacío de categorías). El valor por defecto de los valores etiquetados de seguridad para los atributos es heredado de las clases a la cual pertenecen.

Si necesitamos especificar la situación en la que los accesos a la información de una clase tienen que ser registrados en un archivo log para una auditoría futura, deberemos usar los valores etiquetados *LogType* y *LogCond* en esa clase. Por defecto, el valor de *LogType* es *none*, de esta manera por defecto la auditoría no es necesaria. Por otro lado, si necesitamos especificar una restricción de seguridad, podemos usar OCL y el valor etiquetado *InvolvedClasses* para especificar en qué situación la restricción debe ser cumplida. Por defecto, el valor de este valor etiquetado es la clase con la cual la restricción está asociada. Finalmente, si necesitamos especificar una restricción de seguridad en la que un usuario o un conjunto de usuarios (dependiendo de una condición) puede o no acceder a la clase correspondiente, independientemente de la información de seguridad de esa clase, debemos usar excepciones unidas a los siguientes valores etiquetados: *InvolvedClasses, ExceptSign, ExceptPrivilege* and *ExceptCond*. El valor por defecto de *InvolvedClasses* es la propia clase. Para *ExceptSign* el valor por defecto es +, y para *ExceptPrivilege* es *Read*.

Tabla 2. Valores Etiquetados de la extensión

Valores Etiquetados del Modelo			
Nombre	Tipo	M[2]	Descripción
classes	Set(OclType)	1..*	Especifica todas las clases del modelo. Este nuevo valor etiquetado es útil para navegar a través de todas las clases del modelo.
securityLevels	Sequence (Levels)	1..*	Especifica todos los niveles de seguridad que pueden ser usados por los elementos del modelo (ordenados desde el menos al más restrictivo).
securityRoles	Role	0..*	Especifica la estructura de roles jerárquica que ha sido definida para la organización. Este tipo será administrado como un árbol.
security-Compartments	Set (Compartment)	0..*	Especifica el conjunto de categorías que han sido definidas para la organización.

[2] M se refiere a Multiplicidad

Valores Etiquetados de la Clase			
Nombre	Tipo	M	Descripción
SecurityLevels	Levels	1..*	Especifica el intervalo de posibles valores de niveles de seguridad que una instancia de esta clase puede recibir. Si el nivel superior e inferior son iguales, todas las instancias tendrán el mismo nivel de seguridad. En caso contrario, el nivel de seguridad de la instancia en concreto será definida de acuerdo a una restricción de seguridad.
SecurityRoles	Set(Role)	0..*	Especifica un conjunto de roles de usuario. Cada rol es la raíz de un subárbol de la jerarquía de roles de usuario general definida para la organización. Todas las instancias de esta clase pueden tener los mismos roles de usuario, o pueden ser subárboles de los roles que han sido definidos para la clase. Una restricción de seguridad puede decidir los roles de usuario para cada instancia de acuerdo al valor de algunos atributos de la instancia.
Security-Compartments	Set (Compartment)	0..*	Especifica un conjunto de categorías. Todas las instancias de esta clase pueden tener las mismas categorías de usuario, o un subconjunto de ellas. Una restricción de seguridad puede decidir las categorías de usuario para cada instancia de acuerdo al valor de algunos atributos de la instancia.
LogType	Attempt	0..1	Especifica si el acceso debe ser registrado: ninguno, todos los accesos, sólo accesos frustrados, o sólo accesos satisfactorios.
LogCond	OCLExpression	0..1	Especifica la condición para que el acceso sea registrado.
Involved-Classes	Set(OclType)	1..*	Especifica las clases que deben estar involucradas en una consulta para hacer cumplir una excepción.
ExceptSign	{+,-}	0..1	Especifica si una excepción permite (+) o rechaza (-) el acceso a las instancias de esta clase a un usuario o a un grupo de usuarios.
Except-Privilege	Set(Privilege)	1..*	Especifica los privilegios que el usuario puede recibir o perder.
ExceptCond	OCLExpression	0..*	Especifica la condición que los usuarios deben cumplir para ser afectados por esta excepción.
Valores Etiquetados de la Restricción			
Nombre	Tipo	M	Descripción
Involved-Classes	Set(OCLType)	0..1	Especifica las clases que están involucradas en una consulta, las que deben ser cumplidas en la restricción.

2.5. Estereotipos

Necesitamos definir un estereotipo para especificar otros tipos de restricciones de seguridad (ver Tabla 3). El estereotipo UserProfile puede ser necesario para especificar restricciones dependiendo de una información particular de un usuario o un grupo de usuarios, por ejemplo, dependiendo de la nacionalidad del usuario, edad, etc. Así, los tipos de datos y valores etiquetados definidos previamente, se aplicarán a los estereotipos *Fact*, *Dimension* y *Base* para considerar otros aspectos de seguridad.

Tabla 3. Estereotipo *UserProfile* de nuestra extensión

Nombre	**UserProfile**
Clase base	Class
Descripción	Clases de este estereotipo contienen todas las propiedades que los sistemas administran de los usuarios.
Restricciones	- Esta clase no está asociada con otras clases

	Self.AssociationsEnd.size()=0 - No hay más de una clase de este tipo **Context** Model **Inv** self.classes->forAll(oclIsTypeOf(UserProfile))->size()<=1 - El nombre de la clase de este estereotipo será *PerfilUsuario* self.className=*PerfilUsuario*
Valores Etiquetados	Ninguno
Icono	

2.6. Reglas bien formadas

Hemos identificado y especificado tanto en lenguaje natural como en OCL un conjunto de reglas bien formadas que describen la semántica estática del modelo. En este artículo, por restricciones de espacio no se incluye un detalle de cada regla. Estas reglas están agrupadas básicamente de la siguiente forma:

- Coherencia entre los valores etiquetados de los elementos del modelo y los valores válidos definidos para el modelo.
- Coherencia entre los valores etiquetados definidos por alguna restricción para las instancias de clase, y los definidos para la clase.
- Coherencia de los valores etiquetados definidos para los atributos y los definidos para la clase a la que pertenecen.
- Coherencia de los valores etiquetados de varias clases base que intervienen en la categorización de una dimensión.
- Coherencia de los valores etiquetados de varias clases que intervienen en una asociación (dependiendo de la cardinalidad de ésta).

2.7. Comentarios

Además de las reglas bien formadas (que en realidad son restricciones inherentes al modelo) identificadas, el diseñador puede especificar restricciones de seguridad con OCL. Si la información de seguridad de una clase o de un atributo depende del valor de un atributo de una instancia, éste puede ser expresado como una expresión OCL (ver Figura 4). Normalmente, las restricciones de seguridad definidas para estereotipos de clases (hecho, dimensión y base) serán definidas usando una nota de UML asociada a la clase correspondiente. No imponemos más restricciones al contenido de estas notas que sólo aquéllas impuestas por las definiciones de los valores etiquetados, de manera que permitan al diseñador una mayor flexibilidad.

3. Caso de Estudio

En esta sección, aplicamos nuestra extensión para el diseño conceptual de un modelo MD seguro en el contexto de un sistema de salud típico. Hemos considerado un ejemplo reducido para poder enfocar nuestra atención en las especificaciones de seguridad. La Fig. 3 (a) muestra la jerarquía simplificada de roles de usuario del

sistema, y la Fig. 3 (b) muestra los niveles de seguridad que han sido definidos. En este ejemplo no hemos considerado categorías de usuario.

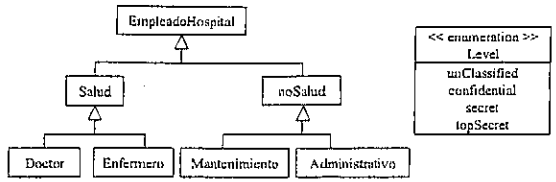


Fig. 3. (a) Jerarquía de roles de usuario (b) Niveles de seguridad

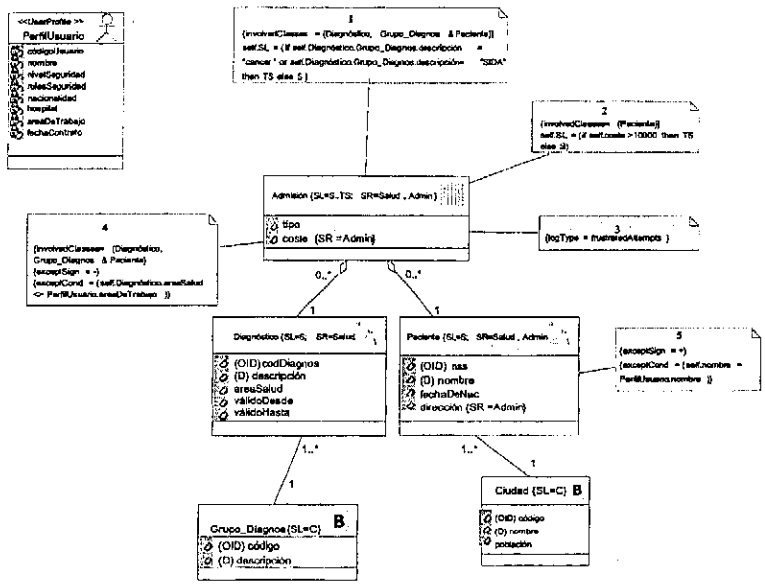


Fig. 4. Ejemplo de modelo multidimensional con información y restricciones de seguridad

La Figura 4 muestra un modelo MD que incluye una clase de hecho (*Admisión*), dos dimensiones (*Diagnóstico* y *Paciente*), dos clases base (*Grupo_Diagnos* y *Ciudad*), y una clase (*PerfilUsuario*). La clase *PerfilUsuario* (estereotipo *UserProfile*) contiene la información de todos los usuarios que tendrán acceso a este modelo multidimensional. La clase de hecho *Admisión* –estereotipo *Fact*- contiene todas las

admisiones individuales de pacientes en uno o más hospitales, y puede ser accedido por todos los usuarios que tienen niveles de seguridad *secret* o *topSecret* -valor etiquetado *SecurityLevels (SL)* de las clases-, y juegan roles de *salud* o *administrativo* -valor etiquetado *SecurityRoles (SR)* de las clases-. Note que el atributo *coste* puede ser sólo accedido por usuarios que juegan rol de *administrativo* –valor etiquetado *SR* de los atributos- La dimensión *Paciente* contiene la información de los pacientes del hospital y puede ser accedido por todos los usuarios que tienen nivel de seguridad *secreto* –valor etiquetado *SL-*, y juegan roles de *salud* o *administrativo* –valor etiquetado *SR-*.

El atributo *Dirección* puede ser accedido sólo por usuarios que juegan el rol de *administrativo* –valor etiquetado *SR* de los atributos-. La clase base *Ciudad* contiene la información de las ciudades, y nos permite agrupar a grupos de pacientes por ciudades. Las ciudades pueden ser accedidas por todos los usuarios que tienen nivel de seguridad *confidential* –valor etiquetado *SL-*. La dimensión *Diagnóstico* contiene la información de cada diagnóstico, y puede ser accedida por los usuarios que juegan un rol de *salud* –valor etiquetado *SR-*, y tienen nivel de seguridad *secreto* –valor etiquetado *SL-*. Finalmente, *Grupo_diagnóstico* contiene un conjunto de grupos generales de diagnóstico. Cada grupo puede estar relacionado con varios diagnósticos, pero un diagnóstico siempre estará relacionado a un grupo. Los grupos de diagnóstico pueden ser accedidos por todos los usuarios que tienen nivel de seguridad *confidential* –valor etiquetado SL-.

Se han especificado varias restricciones de seguridad utilizando las restricciones, estereotipos y valores etiquetados definidos previamente:

1. El nivel de seguridad de cada instancia de *Admisión* es definido por una restricción de seguridad especificada en el modelo. Si el valor del atributo *descripción* de *Grupo_Diagnos* a la cual pertenece el diagnóstico que está relacionado a la Admisión es cáncer o SIDA, el nivel de seguridad –valor etiquetado *SL-* de esta admisión será *topSecret*, en caso contrario será *secret*. Esta restricción es sólo aplicada si el usuario hace una consulta cuya información viene de la dimensión *Diagnóstico* o de la clase base *Grupo_Diagnos*, unida con la dimensión *Paciente* –valor etiquetado *involvedClasses-*. De esta manera, un usuario que tiene un nivel de seguridad secreto podría obtener el número de pacientes con cáncer por cada ciudad, pero nunca si la información de la dimensión *Paciente* aparece en la consulta.
2. El nivel de seguridad –valor etiquetado *SL-* de cada instancia de *Admisión* puede también depender del valor del atributo coste, que indica el precio del servicio de admisión. En este caso, la restricción es sólo aplicable para consultas que contienen información de la dimensión *Paciente* –valor etiquetado *involvedClasses-*.
3. El valor etiquetado *logType* ha sido definido para la clase *Admisión*, especificando el valor *frustratedAttempts*. Este valor etiquetado especifica que el sistema tiene que registrar, para una auditoría futura, la situación en la cual un usuario trata de accesar información de esta clase de hecho, y el sistema lo rechaza debido a la carencia de los permisos necesarios.
4. Por razones de confidencialidad, podemos denegar el acceso a información de admisión a usuarios cuya área de trabajo es diferente del área de una instancia de admisión particular. Esto se especifica con una excepción en la clase de hecho *Admisión*, y con los valores etiquetados *involvedClasses, exceptSign* y *exceptCond*.

5. Los pacientes podrían ser usuarios especiales del sistema. En este caso, debería ser posible que los pacientes acceden su propia información como pacientes (por ejemplo, consultando sus datos personales). Esta restricción es especificada usando los valores etiquetados *exceptSign* and *exceptCond* en la clase *Paciente*.

El privilegio considerado en estas excepciones es *read*, pero no la hemos especificado en el modelo ya que es el valor por defecto del valor etiquetado *exceptPrivilege*.

Note que, usando esta extensión, es posible especificar un amplio rango de restricciones de confidencialidad en el modelado conceptual MD.

4. Conclusiones y Trabajo Futuro

En este artículo hemos presentado una extensión de UML que nos permite representar los principales aspectos de seguridad en el modelado conceptual de almacenes de datos. Esta extensión contiene los estereotipos, valores etiquetados y restricciones necesarios para un modelado MD completo y potente. Estos nuevos elementos nos permiten especificar aspectos de seguridad tales como niveles de seguridad en los datos, categorías y roles de usuario sobre los principales elementos de un modelado multidimensional tales como hechos, dimensiones y jerarquías de clasificación. Hemos usado OCL para especificar las restricciones asociadas a estos nuevos elementos definidos, evitando así un uso arbitrario. La principal ventaja de este enfoque es que utilizamos UML, un lenguaje de modelado ampliamente aceptado, que ahorra a los diseñadores el esfuerzo de aprendizaje de un nuevo modelo y sus notaciones correspondientes para un modelado MD específico. Además, UML nos permite representar algunas propiedades MD que son difícilmente representadas en otras propuestas de modelado conceptual MD.

Nuestro trabajo futuro inmediato es generar código que nos permita definir las estructuras que albergarán los datos del almacén en una plataforma destino como por ejemplo Oracle, incluyendo los aspectos de seguridad definidos en este artículo. Ello nos permitirá obligar a que las reglas definidas a nivel conceptual se cumplan por los todos los usuarios que accedan a los datos del almacén. Un trabajo futuro más lejano es extender esta propuesta para poder considerar los procesos ETL (Extraction-Transformation-Loading) tan cruciales en el campo de los almacenes de datos.

Agradecimientos

Esta investigación es parte de los proyectos CALIPO (TIC2003-07804-C05-03) y RETISTIC (TIC2002-12487-E), soportados por la Dirección General de Investigación del Ministerio de Ciencia y Tecnología, y la red VII-J.RITOS2 financiada por CYTED. Agradecemos a Sergio Luján-Mora por su participación en la especificación del *profile* de UML requerido para los prerrequisitos necesitados en este *profile*.

Referencias

1. Chung, L., Nixon, B., Yu, E., y Mylopoulos, J., Non-functional requirements in software engineering, Boston/Dordrecht/London: Kluwer Academic Publishers (2000)
2. Conallen, J., Building Web Applications with UML. Object Technology Series. Addison-Wesley (2000)
3. Devanbu, P., y Stubblebine, S., Software engineering for security: a roadmap, in The Future of Software Engineering, A. Finkelstein Editor, ACM Press (2000) 227-239
4. Fernández-Medina, E., y Piattini, M., Designing Secure Database for OLS. in Database and Expert Systems Applications: 14th International Conference (DEXA 2003).Prague, Czech Republic: Springer (2003)
5. Ferrari, E., y Thuraisingham, B., Secure Database Systems, in Advanced Databases: Technology Design, M. Piattini y O. Díaz Editores, Artech House: London (2000)
6. Gogolla, M., y Henderson-Sellers, B., Analysis of UML Stereotypes within the UML Metamodel. in 5th International Conference on the Unified Modeling Language - The Language and its Applications. Dresden, Germany: Springer, LNCS (2002)
7. Golfarelli, M., y Rizzi, S., A Methodological Framework for Data Warehouse Design. in 1st Intl. Workshop on Data Warehousing and OLAP (DOLAP'98). Maryland, USA (1998)
8. Hall, A., y Chapman, R., Correctness by Construction: Developing a Commercial Secure System. IEEE Software, 19(1) (2002) 18-25
9. Jürjens, J., UMLsec: Extending UML for secure systems development, in UML 2002 - The Unified Modeling Language, Model engineering, concepts and tools, J. Jézéquel, H. Hussmann, y S. Cook Editores. Springer: Dresden, Germany (2002) 412-425
10. Katic, N., Quirchmayr, G., Schiefer, J., Stolba, M., y Min Tjoa, A., A Prototype Model for Data Warehouse Security Based on Metadata. in 9th Intl Workshop on Database and Expert Systems Applications (DEXA'98), Vienna, Austria IEEE Computer Society (1998)
11. Kirkgöze, R., Katic, N., Stolda, M., y Min Tjoa, A., A Security Concept for OLAP. in 8th International Workshop on Database and Expert System Applications (DEXA'97). Toulouse, France: IEEE Computer Society (1997)
12. Luján-Mora, S., Trujillo, J., y Song, I. Y., Extending the UML for Multidimensional Modeling. in 5th International Conference on the Unified Modeling Language (UML 2002). Dresden, Germani: Springer-Verlag (2002)
13. Marks, D., Sell, P., y Thuraisingham, B., *MOMT: A multi-level object modeling technique for designing secure database applications.* Journal of Object-Oriented Programming, 9(4) (1996) 22-29
14. Piattini, M., y Fernández-Medina, E., Specification of Security Constraint in UML. in 35th Annual 2001 IEEE International Carnahan Conference on Security Technology (ICCST 2001).London, Great Britain (2001)
15. Priebe, T., y Pernul, G., Towards OLAP Security Design - Survey and Research Issues. in 3rd ACM International Workshop on Data Warehousing and OLAP (DOLAP'00). Washington DC, USA (2000)
16. Rosenthal, A., y Sciore, E., View Security as the Basic for Data Warehouse Security. in 2nd International Workshop on Design and Management of Data Warehouse (DMDW'00) Sweden (2000)
17. Sapia, C., Blaschka, M., Höfling, G., y Dinter, B., Extending the E/R Model for the Multidimensional Paradigm. in 1st International Workshop on Data Warehouse and Data Mining (DWDM'98). Singapore: Springer-Verlag (1998)
18. Trujillo, J., Palomar, M., Gómez, J., y Song, I. Y., Designing Data Warehouses with OO Conceptual Models. IEEE Computer (34) (2001) 66-75
19. Tryfona, N., Busborg, F., y Christiansen, J., starER: A Conceptual Model for Data Warehouse Design. in ACM 2nd International Workshop on Data Warehousing and OLAP (DOLAP'99). Missouri, USA: ACM (1999)

Separación Dinámica del Aspecto de Persistencia mediante Reflectividad Computacional

Benjamín López, Francisco Ortín, Juan Manuel Cueva

Object-Oriented Technologies Laboratory research group (OOTLab),
Departamento de Informática, Universidad de Oviedo, C/ Calvo Sotelo s/n,
33007, Oviedo, España
{benja, ortin, cueva}@lsi.uniovi.es

Abstract. El principio de la separación de incumbencias se centra en la capacidad de modularizar aquellas partes diferentes de una aplicación relevantes a un concepto, objetivo, tarea o propósito específico. Una separación apropiada de los distintos aspectos de un sistema reduce la complejidad del software, mejora su comprensión y facilita la reutilización de código. Considerando la persistencia como una incumbencia típica en la mayoría de las aplicaciones, la separación de ésta del código principal del sistema conlleva el desarrollo de programas sin tener en cuenta sus requisitos persistentes, añadiendo éstos en fases posteriores. Esta separación permite al desarrollador manejar la persistencia de los programas de forma independiente a su funcionalidad. Tras analizar distintas alternativas existentes para conseguir una separación total del aspecto de persistencia, nos hemos percatado de que, si bien unas ofrecen mayor transparencia que otras, ninguna permite desarrollar una aplicación cuyo código no posea manejo alguno de características persistentes. La reflectividad computacional es una técnica que permite adaptar la estructura y comportamiento de un sistema en tiempo de ejecución. Sobre una plataforma reflectiva no restrictiva implementada previamente, hemos desarrollado un sistema de persistencia en el que se comprueba cómo este mecanismo puede ser empleado para ofrecer una separación total de la incumbencia de la persistencia. Adicionalmente, ofrece un elevado nivel de adaptabilidad que permite cambiar dinámicamente las características persistentes de un programa en ejecución.

Palabras Clave. Persistencia, Reflectividad Computacional, Separación de Incumbencias, Persistencia Ortogonal, Separación de Aspectos.

1 Introducción

Actualmente la mayoría de las aplicaciones que manejan objetos persistentes son diseñadas con uso explícito de Sistemas Gestores de Bases de Datos (SGBD). Éstos pueden ser orientados a objetos o relacionales, empleando en el segundo caso algún mecanismo de traducción. Por tanto, el modo más común de desarrollar aplicaciones persistentes es entremezclando el código de la aplicación con sentencias OQL o SQL de control de persistencia. El no separar el código principal de la aplicación de las sentencias de gestión de la persistencia provoca una serie de inconvenientes: