VIVAT ACADEMIA
N° 3 - NOVEMBRE 2002

Círculo de Cuore Circle de España

CUORE

SECRETARÍA DEL CUORE

C/ Gil de Ontañón, 21

Tel. 91 367 53 65

Fax 91 377 46 69

28027 Madrid (España)

www.cuore.es

e-mail: dabreu@papcongresos.es

Estimado Socio/a:

Parece que fue ayer cuando estábamos haciendo las maletas para salir de Lanzarote al finalizar el XI Congreso, y ya tenemos ante nuestros ojos el XII. Cuando se redacta este editorial los miembros de la Junta Directiva estamos ultimando los detalles de nuestra cita anual, a celebrar en Valencia los días 4, 5 y 6. Simultáneamente, estamos cerrando este número de la revista, que llegará a vuestras manos durante la celebración del Congreso. Además, proseguimos las deliberaciones para adjudicar el Primer Premio CUORE, que patrocina ORACLE, cuyos ganadores conoceremos durante la celebración del Congreso. Como veis, estamos tocando todos los resortes para hacer realidad el objetivo fundamental de nuestra asociación: fomentar el intercambio de conocimiento y experiencias entre nuestros socios.

Pero toca ahora comentar el contenido de este número de nuestra revista. Comenzamos con una entrevista con Jorge Edelmann, Presidente de ORACLE. En ella nos habla de la situación de la empresa y su futuro, en estos momentos de incertidumbre en diversos ámbitos.

Se incluye después una colaboración de Francisco José Fernández Olmedo sobre el relevante tema de la gestión de contenidos, que se aborda en este caso con Vignette y Oracle.

Hemos entrevistado a Albert Trepat de Computer Associates, empresa que colabora con nosotros desde hace tiempo, y es co-patrocinadora de nuestro Congreso. En ella nos habla de su oferta de soluciones y de su postura ante el fenómeno del software libre. Disponemos también de un artículo de Gerard Cristofol, de Tecsidel, sobre un tema tan "candente" como la firma electrónica y su papel en el desarrollo del comercio electrónico. Francisco Monteverde, representante de otra empresa que tradicionalmente colabora con nosotros y co-patrocina nuestro Congreso, Sun Microsystems, nos habla acerca de la situación y futuro de su empresa, sobre LINUX, Java y otras cuestiones de gran interés.

El año pasado comenzamos la iniciativa "Vivat Academia" que llega ahora a su tercer número, incorporando a nuestra publicación la nueva "savia" de los trabajos universitarios. Traemos por primera vez a nuestras páginas un artículo de Diego Cutrone, de la empresa argentina Euronecz, acerca de mecanismos de actualización diferida de tablas.

Los expertos de ORACLE nos han enviado sendos artículos sobre servicios web, sistemas para gestión empresarial y la iniciativa de ORACLE para extender el conocimiento de su software en la Universidad. Incluimos también nuestra habitual sección "revista de libros", en la que se examina las últimas novedades aparecidas en el sector editorial. No podía faltarnos, por último, nuestra habitual sección de citas, para terminar de configurar un número que esperamos os complacerá.

Mi despedida tiene que ser, más que nunca, un "hasta pronto", pues espero encontraros dentro de pocas fechas en nuestro Congreso de Valencia. Recibe un cordial saludo.

José Ángel Alonso
PRESIDENTE

Enter

Edita:
Cuore

Coordinador
Rafael Rojo

Consejo Editorial
José Ángel Alonso
Pedro Poveda
Rafael Rojo
Adolfo Sánchez
Manuel Pérez
David Abreu

Impresión
Moncaba

Preimpresión
Lufercomp

Dpto. Legal
M-24195-1992

Sumario

Cuore

VIVAT ACADEMIA

Nº 3 - NOVIEMBRE 2002

Una metodología para diseñar bases de datos seguras implementadas en Oracle *9i* Label Security

Eduardo Fernández-Medina y Mario Piattini

Grupo ALARCOS
Esucela Superior de Informática. Universidad de Castilla-La Mancha. Ciudad Real
{Eduardo.FdezMedina, Mario.Piattini}@uclm.es

RESUMEN. En este artículo argumentamos la importancia que tiene la seguridad en las bases de datos, y la necesidad de considerar la seguridad como uno de los requisitos fundamentales en su desarrollo. Así, proponemos la integración de la seguridad en todas las etapas del análisis y el diseño de bases de datos, en lugar de ser considerada como un requisito marginal que solamente se considera en las últimas etapas del desarrollo. Se presenta una metodología para diseñar bases de datos seguras y varios modelos y lenguajes que han sido extendidos partiendo de algunos de los estándares más aceptados de modelado. La metodología nos permite diseñar bases de datos teniendo en cuenta aspectos de seguridad de la información en todas las etapas del proceso de desarrollo, desembocando en la última etapa en el sistema de gestión de bases de datos seguras "Oracle*9i* Label Security". La metodología es ilustrada con un ejemplo de parte de un caso real de diseño de base de datos segura que ha sido llevado a cabo en un organismo oficial. Las aportaciones de la metodología son principalmente, los modelos de casos de uso y clases extendido, el modelo multinivel extendido, los lenguajes de especificación de restricciones, el modelo de procesos de la metodología, y el proceso de adaptación de las bases de datos seguras diseñadas, a Oracle*9i* Label Security. Este artículo muestra una visión general de la metodología, y se enfoca a la implementación de bases de datos seguras con Oracle*9i* Label Security.

1. INTRODUCCIÓN

La información es uno de los factores más importantes para las compañías y para todos los sectores de la sociedad en la era de las comunicaciones y por lo tanto ha de ser gestionada de manera adecuada. De hecho, a menudo la supervivencia de las organizaciones dependen de la correcta gestión, seguridad y confidencialidad de sus informaciones (Dhillon y Backhouse, 2000). Así, puesto que la mayoría de los datos que componen la información se almacenan en bases de datos y almacenes de datos, éstos son puntos clave para examinar cuando analizamos la seguridad de la información. Por este motivo, la protección de las bases de datos es un aspecto que ha de ser considerado explícitamente, como un factor presente en todas las etapas del ciclo de vida de las bases de datos, desde la captura y análisis de requisitos hasta la implementación y mantenimiento (Devanbu y Stubblebine, 2000; Ferrari, 2001; Ghosh et al., 2002). Para este propósito, Hall y Chapman (2002) proponen diferentes ideas para integrar la seguridad en el proceso de desarrollo de los sistemas de información, aunque en su propuesta, la seguridad de las bases de datos es sólo considerada en el ámbito de la criptografía.

La seguridad en bases de datos, no sólo es importante para las compañías por el riesgo de pérdidas de confidencialidad, integridad y disponibilidad. A veces, las bases de datos almacenan también información sobre aspectos íntimos o personales de los individuos, como información de identificación, datos médicos, o incluso otros aspectos más delicados como creencias religiosas, ideologías, tendencias sexuales, etc. En estos casos, la protección de esta información es muy importante por la responsabilidad hacia los individuos a los que se refiere la información. Para intentar garantizar la protección de estos tipos de datos, existen leyes de protección de datos personales, como por ejemplo, la *Ley Orgánica de protección de Datos Personales* (Ley Orgánica, 15/1999). Por lo tanto, las bases de datos que almacenen información de carácter personal, deberían contar con los mecanismos adecuados para garantizar la seguridad de la información, y además que permitan cumplir las leyes de protección de datos existentes.

La labor de proteger la información, se convierte en una tarea nada trivial, sobre todo si consideramos la escasez de recursos que se le otorgan a ella (SIC, 2001), y la complejidad en los requisitos de seguridad que aportan los constantes cambios tecnológicos en las tecnologías de la información, como el acceso a bases de datos a través de la

web, el avance del comercio electrónico, de los almacenes de datos, e incluso en las técnicas de minerías de datos, que frecuentemente ponen en conflicto incluso la seguridad nacional con los derechos de privacidad e intimidad de los individuos (Thuraisingham et al., 1997).

Por lo tanto, es muy importante diseñar bases de datos que sean más seguras. A pesar de que ya existen soluciones muy interesante, no resuelven el problema de la protección de las bases de datos de manera global, sino que ofrecen soluciones parciales y aisladas, y no actúan en el análisis y diseño de las bases de datos. Para abordar este problema, presentamos un enfoque metodológico para diseñar bases de datos teniendo en cuenta los aspectos de seguridad desde las etapas más tempranas hasta el final de su desarrollo con Oracle*9i* Label Security (OLS). Además, esta metodología está basada en las metodologías y estándares de modelado más aceptados, para garantizar que las organizaciones que estén interesadas en utilizarla no necesiten realizar un esfuerzo considerable para adaptarse a ella. Las metodologías de diseño de bases de datos tradicionales no resuelven el problema puesto que la seguridad no es considerada en sus etapas (Batini et al., 1991; Connolly y Begg, 2002).

Uno de los estándares más aceptados en estos momentos es UML (Booch et al., 1999), y uno de los procesos de desarrollo de software más utilizados es el *Proceso Unificado de desarrollo de Software* (Jacobson et al., 1999). De acuerdo a Muller (1999), UML puede ser utilizado, mediante un proceso adecuado, en el desarrollo de bases de datos. Por lo tanto, la idea de extender los modelos de UML con características de seguridad, y utilizarlos junto con una adaptación del Proceso Unificado para diseñar bases de datos seguras, resulta atractiva (Fernández-Medina y Piattini, 2001). Así, una metodología para diseñar bases de datos, basado en el lenguaje UML y en el Proceso Unificado, con características adicionales de seguridad, nos permitiría diseñar bases de datos con la sintaxis y potencia de UML y con las nuevas propiedades de seguridad, listas para ser utilizadas cuando los requisitos de seguridad de las bases de datos las necesiten. Esta medida nos permite cumplir con las condiciones impuestas por Chung et al. (2000) para diseñar de manera sistemática sistemas de información seguros, integrando los requisitos de seguridad en el diseño, y ofreciendo al diseñador modelos para especificar aspectos de seguridad. Así, también cumpliríamos los desafíos que consideraron Devanbu y Stubblebine (2000), de unificar la seguridad en la ingeniería del software tanto en los procesos como en los modelos de los sistemas.

La metodología desemboca en un proceso que hace posible la implementación de las bases de datos seguras que han sido diseñadas, utilizando OLS, que es un componente de Oracle*9i* que nos permite crear y gestionar bases de datos multinivel, y para el cual no se ofrece ninguna metodología para soportar las etapas de análisis y diseño de las bases de datos seguras. Por lo tanto, podemos decir que la propuesta es una metodología para diseñar bases de datos, basada en el Proceso Unificado, que considera diferentes extensiones de varios modelos de UML y lenguajes de restricciones, y que concluye con la implementación de la bases de datos a través de OLS.

1.1. Caso Práctico

Para validar la metodología que ha sido presentada en detalle en Fernández-Medina (2002), y que se resume y sintetiza en este artículo, hemos utilizado el método de investigación "*Investigación en acción*" (Avison et al., 1999). El proceso cíclico de Investigación en Acción ha sido aplicado en el diseño de una base de datos de un organismo oficial, cuyo nombre no mencionamos por cuestiones de confidencialidad. Esta base de datos está gestionada por una aplicación que tiene diversos problemas de confidencialidad, que han sido resueltos a través de un nuevo diseño de la base de datos.

El objetivo general de la aplicación es el control del presupuesto y la contabilidad de un sector. La aplicación está compuesta por trece módulos, algunos de los más importantes son los siguientes:

- Presupuesto de gastos: Su objetivo es el de gestionar los gastos presupuestados.
- Presupuesto de ingresos: Su función es la de gestionar los ingresos presupuestados.
- Terceros: Gestiona los datos
ciones con personas y entida
ternas al organismo antes cit
- Administración del sistema: E
dulo de la aplicación admini
configuración del propio sist
- Recursos de otros entes: Gestió
cursos contratados por este o
mo a organizaciones externas.
- Tesorería: Se encarga de ges
los pagos y los cobros realizad
- Gastos con financiación afecta
labor es gestionar los gastos e
dentro de un proyecto.
- Operaciones no presupuest
Gestiona operaciones econó
fuera del presupuesto.

Analizando los detalles de segur
en concreto de confidencialidad, ob
vamos que los módulos que mayor
rés podrían tener son los cuatro pri
ros (presupuestos de gastos, presupu
de ingresos, terceros y la administra
del sistema). En el funcionamiento
esta aplicación, con respecto a la in
mación contenida en la base de da
solamente se consideran controles
confidencialidad mínimos, a pesar
que algunos datos, debido a su natur
za deberían ser protegidos.

En la gestión de toda esta inform
ción, existen ciertos mecanismos de
guridad, pero que no garantizan al c
por cien la confidencialidad de la inf
mación, puesto que tales mecanism
han sido incluidos una vez integrada
aplicación (la seguridad no ha sido in
grada en el proceso de desarrollo de
aplicación), y que protegen la inform
ción en el ámbito de las "operacione
y no de los "datos". Si bien estos mec
nismos evitan que un usuario pueda l
var a cabo ciertas acciones a través de
aplicación, dejan descubiertos ciert
canales, y además no evitan que pued
ser creados nuevos programas que e
traigan la información, o incluso mod
ficar los existentes creando de este mod
"Caballos de Troya".

En las siguiente secciones, mostrare
mos un resumen de la metodología d
diseño de bases de datos seguras y de lo
modelos y lenguajes que han sido defi
nidos. La metodología se ilustra a través
de algunos de los principales productos
que se obtienen en la aplicación de la
metodología en el caso de estudio con
siderado.

2. Metodología de Diseño de Bases de Datos Seguras

El enfoque de esta metodología está basado principalmente en dos ideas:

- Una nueva metodología de diseño de bases de datos debería estar basada los lenguajes, procesos y estándares de modelado más aceptados del mercado. Por lo tanto, hemos definido:
- Un modelo de procesos basado en el Proceso Unificado de desarrollo de Software (Jacobson et al., 1999).
- Varios modelos para representar la *información de sensibilidad*[1] de las bases de datos que están basados en modelos de UML (Booch et al., 1999)
- Varios lenguajes para especificar restricciones de seguridad que son extensiones del Lenguaje de Restricciones de Objetos (OCL) (Warmer y Kleppe, 1998), que es el lenguaje de restricciones de UML.
- Desde el punto de vista de la confidencialidad, la información debería ser clasificada, dependiendo sólo de las características de la propia información, y no como en los modelos multinivel clásicos, donde la clasificación de la información depende de las características del sujeto que la crea. Por ese motivo, hemos definido un modelo relacional multinivel extendido, basado en el modelo multinivel de Sandhu y Chen (1998), pero con algunas características diferenciadoras.

Al igual que en Proceso Unificado, la metodología será iterativa e incremental, dirigida por casos de uso y centrada en la arquitectura. Las etapas de la metodología son las siguientes:

- *Captura de requisitos*. En esta etapa, se realiza el análisis de los requisitos del sistema, pero teniendo en cuenta los distintos requisitos de seguridad.
- *Análisis del sistema*. El objetivo de esta etapa es representar los requisitos por medio de un modelo conceptual, incluyendo información de seguridad en las clases, atributos y asociaciones entre clases cuando sea necesario.
- *Diseño lógico relacional multinivel*. En esta etapa, se crea un modelo lógico en el modelo relacional, y se clasifica la información dependiendo de sus propiedades de sensibilidad.
- *Diseño lógico específico*. La base de datos se representa mediante un modelo relacional específico, teniendo en cuenta sus particularidades. En este caso, hemos considerado el modelo de OLS pero se podría adaptar a otros (ver Figura 1).

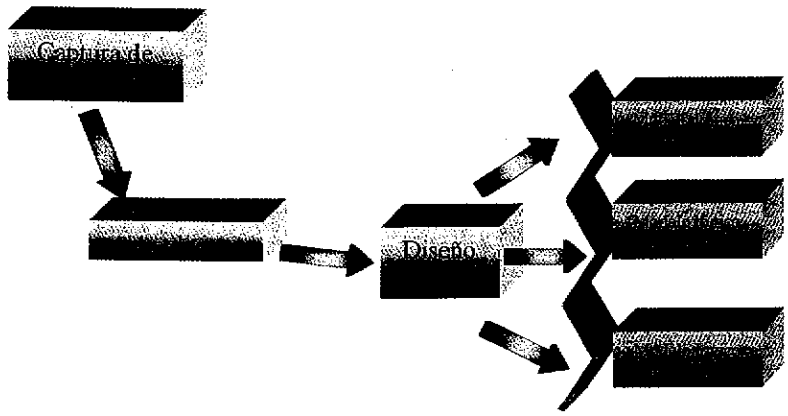

Figura 1. Esquema general de la metodología

Además, para dar soporte automatizado a la metodología, hemos desarrollado un prototipo de herramienta CASE. Este prototipo está integrada en la herramienta Rational Rose, y permite incluir y gestionar la información de sensibilidad de los elementos en diagramas de casos de uso y diagramas de clases que han sido creados por Rational Rose. También permite definir restricciones de seguridad y comprobar tanto su léxico como su sintaxis.

En las siguiente secciones mostraremos cada etapas de la metodología con más detalle, ilustrándolas con una parte del caso práctico que hemos considerado.

2.1. Captura de Requisitos

Esta etapa es una de las más importantes de la metodología puesto que, como es bien conocido, las comunicaciones entre el equipo de desarrollo y los clientes es crucial para recolectar y entender todos los requisitos. El objetivo de este paso es capturar y representar requisitos, con la particularidad de que en esta etapa también consideraremos los requisitos de seguridad.

Existen varios artefactos que pueden ser usados en esta etapa, pero el más importante es el *modelo de casos de uso extendido* (Fernández-Medina y Piattini, 2002), que permite representar actores y casos de uso, indicando características de seguridad mediante estereotipos. En particular, el modelo de casos de uso extendido introduce el concepto de *caso de uso seguro* (Secure-UC) y *actor acreditado* (Accredited-Actor). Un caso de uso seguro es un caso de uso que, desde el punto de vista de la seguridad debería ser estudiado en profundidad en posteriores etapas del desarrollo. Un actor acreditado es un actor que debe tener especial acreditación para poder ejecutar o participar en un caso de uso. Además, el estereotipo "database" se define para poder identificar actores que serán elementos de almacenamiento permanente (que podrán ser bien bases de datos o bien otros elementos de una base de datos).

Las actividades de esta etapa son la siguientes: Capturar requisitos iniciales, crear el modelo de negocio y el glosario del sistema, buscar actores, buscar casos

[1] Tradicionalmente, la información de sensibilidad que ha sido considerada por los modelos multinivel son sólo los "niveles de seguridad", pero a veces es necesario usar más información sobre sensibilidad. En este artículo, cuando usemos el término "información de sensibilidad" asociada con datos, nos referiremos tanto a "niveles de seguridad" de los datos, como a "roles de usuarios" (usuarios autorizados a acceder a los datos).

de uso, buscar elementos de almacenamiento permanentes, describir casos de uso, analizar seguridad en actores y en casos de uso, definir prioridades en casos de uso, estructurar el modelo de casos de uso, buscar relaciones entre casos de uso y, finalmente, revisión de casos de uso. Evidentemente, la actividad más importante de esta etapa, desde el punto de vista de la seguridad es el análisis de la seguridad en actores y casos de uso, que consiste en estudiar si los casos de uso tienen requisitos de confidencialidad, y si los actores necesitarán una cierta acreditación para realizar los casos de uso. La Figura 2 ilustra uno de los diagramas de casos de uso del sistema, correspondiente al módulo "Terceras Partes".

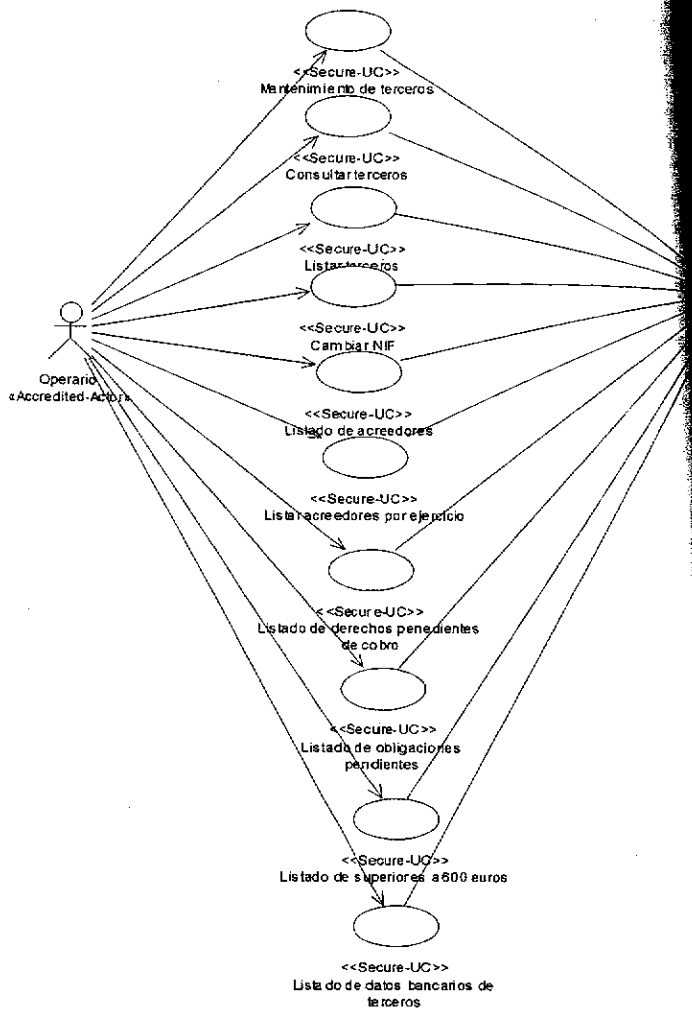

Figura 2. Diagrama de casos de uso de "Terceras Partes"

2.2. Análisis del Sistema

El objetivo de esta etapa es construir el modelo conceptual de la base de datos, considerando todos los requisitos que han sido capturados en las etapas previas. El modelo conceptual estará compuesto básicamente por el *diagrama de clases extendido* (un análisis en profundidad de este modelo puede ser encontrado en Fernández-Medina y Piattini (2002)), junto con un conjunto de restricciones de seguridad que han sido expresados a través del lenguaje OSCL (Piattini y Fernández-Medina, 2001).

Las particularidades de los diagramas de clases extendidos es que pueden ser usados para especificar información de sensibilidad en clases, atributos y asociaciones, que indican las condiciones que los sujetos han de cumplir para acceder a ellos. Además, a los usuarios se les asigna también información de acreditación que delimita la información a la que podrá tener acceso. Los tipos de información de sensibilidad que han sido considerados en la metodología son *niveles de seguridad* y *roles de usuarios autorizados*. Si asignamos un nivel de seguridad por ejemplo a una clase, significa que los sujetos tendrán que estar clasificados al menos en el mismo nivel de seguridad que la información para poder acceder a ella. Si asignamos un conjunto de roles a un elemento, significa que un sujeto tiene que pertenecer al menos a uno de esos roles para tener acceso a la información. Por ejemplo, si clasificamos la clase "Paciente" con el nivel de seguridad "Alto Secreto", y definimos el conjunto de roles autorizados para la clase con "Personal Médico", la información de esta clase solamente será accesible para los usuarios del rol "Personal Médico" y que estén clasificados en el nivel de seguridad "Alto Secreto".

El lenguaje OSCL permite especificar restricciones de seguridad que definen la información de sensibilidad para las clases, atributos o asociaciones, dependiendo de ciertas condiciones basadas en las propias características de la información. Por ejemplo, es posible definir una restricción que especifique que el nivel de seguridad de los objetos pertenecientes a una clase serán más restrictivos si el valor de uno de sus atributos tiene un valor determinado. La sintaxis es fácil de entender, puesto que este lenguaje está basado en OC lenguaje de restricciones sobre ob de UML.

Las tareas de esta etapa son l guientes: Análisis de la arquitec análisis de casos de uso, análisis d ses, análisis de la seguridad y análi paquetes. Todas esas tareas se com nen de muchas más subtareas que motivos de espacio no desarrollare No obstante, analizaremos más en fundidad la tarea de análisis de la se ridad, que es la más importante desd punto de vista de la seguridad. E compuesta de las siguientes acciones

- Definir los niveles válidos de segu dad para la base de datos.
- Asignar niveles de seguridad a clases, atributos y asociaciones, niendo en cuenta las propiedades

sensibilidad de la información y las *restricciones inherentes*[2] del modelo de clases extendido.

- Clasificar clases, atributos y asociaciones en diferentes roles de usuarios acreditados si es necesario. Esta actividad tendrá en cuenta la jerarquía de usuarios definida para la organización, el rol de usuarios de cada elemento de la jerarquía y las restricciones inherentes al modelo.
- Especificación de restricciones de seguridad, que define la información de seguridad o sensibilidad de diferentes elementos del modelo, dependiendo del cumplimiento o no de una cierta condición, que habitualmente depende del valor de un atributo determinado.
- Analizar otros tipos de restricciones de seguridad (no sólo restricciones de confidencialidad).
- Definir la información de acreditación de los usuarios, que estará compuesta de los niveles de seguridad y los roles que juegan los usuarios.

La Figura 3 ilustra un ejemplo de un diagrama de clases extendido. Los niveles de seguridad[3] que han sido definidos para esta bases de datos han sido Sin Clasificar (SC), Secreto (S) y Alto Secreto (AS). Las jerarquías de roles que han sido definidas para esta base de datos es la que podemos observar en la Figura 4. Para cada rol, hemos definido un nombre corto, que es el usado en el diagrama. Por ejemplo, la clase "Datos Económicos" tiene definido el nivel de seguridad "AS" y el rol "OAC". Esto significa que la información de los objetos que pertenezcan a esta clase serán accesibles sólo para usuarios que tengan el nivel de seguridad "Alto Secreto", y que jueguen el rol "Operario del área de contabilidad". Cuando no exista información de sensibilidad asociada con una clase, significará que todos los usuarios podrán tener acceso a la información de sus objetos. Podemos observar también cuatro restricciones OSCL, que definen los niveles de seguridad de los objetos, dependiendo del valor de diferentes atributos. Por ejemplo, la restricción de seguridad asociada con la clase "Datos Bancarios" indica que el nivel de seguridad de sus objetos será "Confidencial", si el valor del atributo "Descripción del Banco" es igual a "Banco no español", y "Sin Clasificar" en otro caso. Puesto que los objetos de esa clase pueden tener niveles de seguridad diferentes, son expresados en la clase como un rango de niveles de seguridad.

2.3. Diseño Lógico Relacional Multinivel

El objetivo de esta metodología es diseñar bases de seguridad, partiendo de modelos de un alto nivel de abstracción y transformándolos en otros de un menor nivel de abstracción. Por lo tanto, una vez que tenemos disponible un modelo conceptual, es necesario transformarlo para obtener un modelo relacional multinivel. Como hemos comentado previamente, un nuevo modelo relacional multinivel ha sido definido en Fernández-Medina (2002), con algunas diferencias con respecto a los modelos tradicionales. Este modelo nos permite representar toda la información del modelo conceptual, pero en el contexto de los sistemas de bases de datos relacionales, y de los sistemas multinivel. Los tres componentes del modelo relacional multinivel son los siguientes:

- Modelo relacional de la base de datos: Este componente incluye la definición de cada relación de la base de datos. La definición de una relación está compuesta del nombre de relación y del nombre de sus atributos, incluyendo los atributos necesarios para representar la información de confidencialidad.
- Meta-información del modelo: Cada relación tendrá asociada una tupla de meta-información que incluirá los tipos de datos de los atributos, y los valores válidos de los atributos relativos a la información de sensibilidad de la tupla y los atributos.
- Restricciones de seguridad. Todas las restricciones de seguridad definidas en el modelo conceptual deberían ser especificadas también en este modelo sin sufrir pérdidas o modificaciones en su semántica. Por esa razón hemos redefinido el lenguaje OSCL, creando el lenguaje RMSCL (Lenguaje de restricciones de seguridad multinivel relacional), para expresar las restricciones de seguridad en el contexto de este modelo.

Los modelos multinivel clasifican la información en diversos niveles de seguridad. Típicamente, en esos modelos, es posible clasificar una tupla en un determinado nivel de seguridad, y este nivel dependerá del nivel de seguridad del usuario que la cree. Además, es típico también el uso de la poli-instanciación para evitar el descubrimiento de información sensible (Sandhu y Chen, 1998; Samarati y De Capitani Di Vimercati, 2001). Sin embargo, en nuestro modelo, cada elemento puede estar asociado con diferentes niveles de seguridad, sin depender éste del nivel e seguridad del sujeto que la cree, sino de las propias características de confidencialidad que tenga la información, y en este caso, la poli-instanciación no es necesaria. La clasificación de la información en niveles de seguridad se decide en tiempo de análisis-diseño, y si es necesario se define un conjunto de restricciones de seguridad. De este modo, las tuplas y atributos, cuando son creados, heredan la información de sensibilidad que ha sido previamente definida (o bien el resultado de evaluar una determinada restricción de seguridad).

Además, en nuestro modelo también se consideran roles de usuarios como información de sensibilidad, no como los modelos tradicionales, que sólo consideraban los niveles de seguridad. Las reglas de acceso a la información son las siguientes: Un sujeto puede leer un objeto sólo si el nivel de seguridad del sujeto domina el nivel de seguridad del objeto, y un sujeto tiene acceso de escritura sobre un objeto sólo si el nivel de seguridad del sujeto es igual al nivel de seguridad del objeto. Puesto que hemos considerado roles de usua-

[2] El modelo de clases extendido tiene diferentes restricciones inherentes, que todas las instancias de los objetos deben cumplir. Por ejemplo, el nivel de seguridad de los atributos deben ser iguales o más restrictivos que el nivel de seguridad de las clases a las que pertenece.

[3] Es posible en esta metodología definir un conjunto particular de niveles de seguridad, dependiendo de la complejidad de la base de datos.

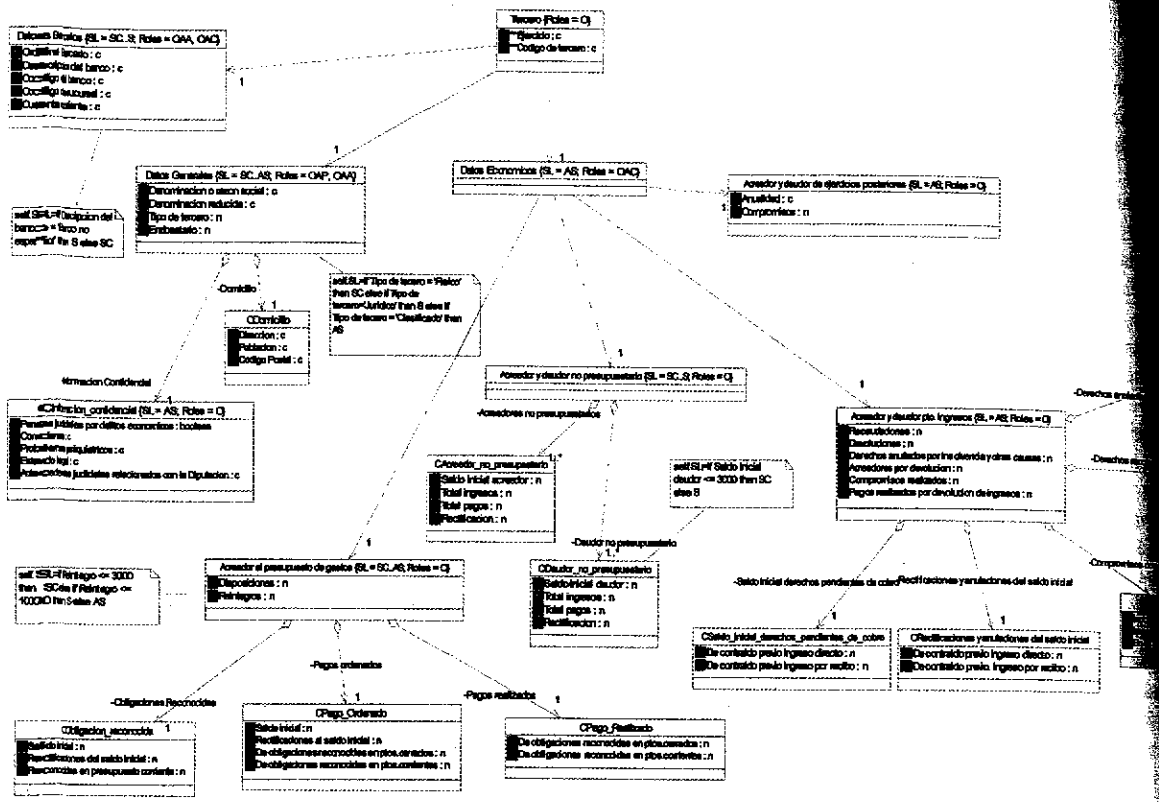

Figura 3. Diagrama de clases del módulo "Terceras Partes"

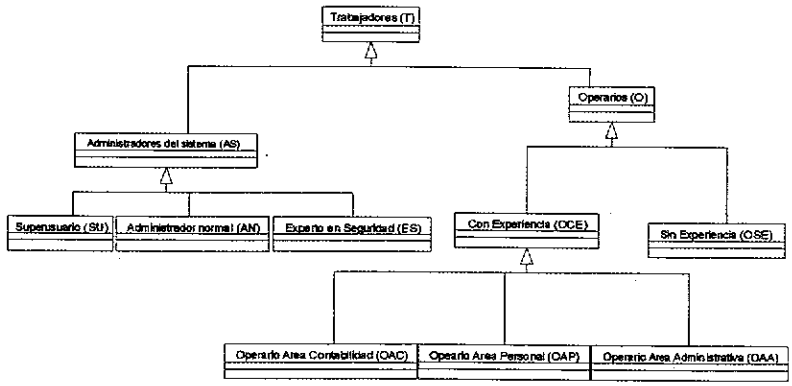

Figura 4. Jerarquía de roles

rios para la información, es necesario para ambos tipos de acceso que el usuario juegue al menos uno de los roles que han sido definidos para información a la que desee acceder.

Las principales actividades de esta etapa se refieren a la transformación de todos los elementos en el modelo de clases extendido al modelo relacional multinivel. Estas actividades son las siguientes: Adaptar los paquetes de clases, adaptar los tipos de datos, transformar clases en relaciones, transformar asociaciones binarias, transformar relaciones n-arias, transformar relaciones de generalización, transformar agregaciones, y finalmente, adaptar las restricciones de seguridad del lenguaje OSCL al lenguaje RMSCL.

Un fragmento del modelo relacional multinivel de nuestro ejemplo puede ser observado en la Figura 5. Por razones de espacio, hemos seleccionado sólo un subconjunto del diagrama de clases. Po-

demos observar que cada
tanto en el modelo relacion
sección de meta-informac
la sección de restriccione
restricciones relativas a l
hemos considerado. Adem
alizado una transformació
de datos usados en el model
a tipos de datos del model

2.4. Diseño Lógico Especí

Una vez que el modelo rela
tinivel ha sido definido, n
pensar en la forma de impl
base de datos. Básicamente
considerar dos formas:

- Considerar el modelo rel
pecífico de uno de los si
gestión de bases de datos
les, que habitualmente no
te a la gestión de la seguri
nivel.
- Considerar el modelo relac
tinivel de uno de los prototi
tentes de bases de datos mu

Si elegimos la primera opci
mos encontrar problemas imp

Modelo Relacional de la Base de Datos

DatosBancarios(Id, OrdinalBanco, DescripcionBanco, CodigoBanco, CodigoSucursal, CuentaCorriente, Tejercicio, Tcodigo, NivelSeguridad)

Tercero(Ejercicio, Codigo)

AcreedorDeudorEjerciciosPosteriores (Anualidad, Compromisos)

AcreedorPresupuestoGastos (Id, Disposiciones, Reintegros, Obl_SaldoInicial, Obl_RectificacionesSaldoInicial, Obl_ReconocidasPtoCorriente, Pag_ord_SaldoInicial, Pag_Ord_RectificcionesSaldoInicial, Pag_Ord_DeObligacionesPtosCerrados, Pag_Ord_DeObligacionesPtosCorrientes, Pag_Rea_DeObligacionesPtosCerrados, Pat_rea_DeObligacionesPtosCorrientes, NivelSeguridad)

Meta-information

<DatosBancarios, SC..S, OAA, OAC, Id: VarChar; OrdinalBanco: VarChar; DescripcionBanco: VarChar; CodigoBanco: VarChar; CodigoSucursal: VarChar; CuentaCorriente: VarChar; Tejercicio: VarChar; Tcodigo: VarChar; NivelSeguridad: NS>

< AcreedorDeudorEjerciciosPosteriores, AS, O, Anualidad: Varchar; Compromisos: Real>

< AcreedorPresupuestoGastos, SC..AS, O, Id: VarChar; Disposiciones: Real; Reintegros: Real; Obl_SaldoInicial: Real; Obl_RectificacionesSaldoInicial: Real; Obl_ReconocidasPtoCorriente: Real; Pag_Ord_SaldoInicial: Real; Pag_Ord_RectificacionesSaldoInicial: Real; Pag_Ord_DeObligacionesPtosCerrados: Real; Pag_Ord_DeObligacionesPtosCorrientes: Real; Pag_Rea_DeObligacionesPtosCerrados: Real; Pag_Rea_DeObligacionesPtosCorrientes: Real; NivelSeguridad: NS>

Restricciones de Seguridad

Context DatosBancarios **inv:**
Self.SL = (if self.DescripcionBanco = "Banco no español" then S else SC endif)
Context AcreedorPresupuestoGastos **inv:**
Self.SL = (if Reintegro <= 3000 then SC else (if Reintegro <= 10000 then S else AS endif) endif)

Figura 5. Modelo Relational Multinivel

como por ejemplo la necesidad de diseñar e implementar mecanismos para integrar la información de sensibilidad en las bases de datos, y de confirmar que todos los accesos cumplen con las políticas de control de acceso definidas en el modelo.

En cambio, si elegimos la segunda opción, tendríamos resueltos los problemas comentados anteriormente, pero aparecerían otros problemas relativos al hecho de que existen muy pocos prototipos de bases de datos multinivel, puede que no estén muy depurados, y cada uno de ellos tiene distintas particularidades para las cuales es difícil conseguir una grado adecuado de adaptación.

Afortunadamente, existe una solución intermedia, *Oracle9i Label Security*. Esta solución nos permite implementar bases de datos multinivel, y además ha sido mejorada a través de varias versiones. Además es parte de uno de los sistemas de gestión de bases de datos, Oracle. Esta alternativa es buena pero también entraña algunos problemas, ya que OLS cuenta con algunas peculiaridades que hay que tener en cuenta y que limitan en cierto modo la funcionalidad de la metodología. Por ejemplo, esta tecnología no considera clasificación de grano fino (sino que clasifica la información solamente a nivel de tupla), mientras que la metodología considera clasificación de la información incluso a nivel de atributo. Otra característica es que la información de sensibilidad de las tuplas depende de la información de acreditación de los usuarios que las crean. Afortunadamente, OLS es muy flexible, y permite al diseñador seleccionar diferentes opciones, para adaptarla a diferentes modelos. Un estudio en detalle de OLS puede encontrarse en Morán (2001).

En el contexto de OLS, una política de seguridad es el elemento más importante, e incluye diversos parámetros, como los niveles de seguridad válidos, los grupos[4] y compartimentos[5], y diferentes opciones que definen la forma de gestionar la seguridad. Teniendo esto en cuenta, las actividades de esta etapa son las siguientes: Definir el modelo de la base de datos (todas las tablas), definir la política de seguridad y sus opciones por defecto, definir la información de sensibilidad en la política de seguridad, crear los usuarios autorizados y asignarles su información de acreditación, definir información de sensibilidad para las tablas mediante funciones de etiquetado, implementar las restricciones de seguridad también mediante funciones de etiquetado y predicados de control de acceso, y finalmente, si es necesario, implementar las operaciones y controlar su seguridad.

En la Figura 6 mostramos la forma de definir tanto la política de seguridad, como los niveles de seguridad y los grupos de usuarios en OLS. "SecurityLabel" es el nombre de la columna que almacena la información de sensibilidad en cada tabla que se asocia con una política de seguridad. La opción "HIDE" indica que la columna "SecurityLabel" será oculta, por lo tanto, los usuarios no

[4] Jerarquías de grupos, en el contexto de OLS tienen el mismo significado que jerarquía de roles en el contexto de la metodología.
[5] Este elemento representa una clasificación no jerárquica horizontal. Este concepto o ha sido considerado en la metodología.

```
CREATE_POLICY("SecurityPolicy","SecurityLabel", "HIDE, CHECK_CONTROL,
READ_CONTROL, WRITE_CONTROL")

CREATE_LEVEL("SecurityPolicy", 1000, "SC", "Sin Clasificar")
CREATE_LEVEL("SecurityPolicy", 2000,"S","Secreto")
CREATE_LEVEL("SecurityPolicy", 3000, "AS", "Alto Secreto")

CREATE_GROUP("SecurityPolicy", 1, "T" ,"Trabajador")
CREATE_GROUP("SecurityPolicy", 2, "O", "Operario", "T")
CREATE_GROUP("SecurityPolicy", 3, "AS", "Administrador del Sistema", "T")
CREATE_GROUP("SecurityPolicy", 4, "SU", "Super Usuario", "AS")
CREATE_GROUP("SecurityPolicy", 5, "AN", "Administrador Normal", "AS")
CREATE_GROUP("SecurityPolicy", 6, "RS", "Responsable de Seguridad", "AS")
CREATE_GROUP("SecurityPolicy", 7, "OCE", "Operario con Experiencia", "O")
CREATE_GROUP("SecurityPolicy", 8, "OSE", "Operario sin Experiencia", "O")
CREATE_GROUP("SecurityPolicy", 9, "OAC", "Operario Area Contabilidad", "OCE")
CREATE_GROUP("SecurityPolicy", 10, "OAP", "Operario Area Personal", "OCE")
CREATE_GROUP("SecurityPolicy", 11, "OAA", "Operario Area Administrativa", "OCE")
```

Figure 6. Definición de políticas, niveles y grupos

```
SET_LEVELS("SecurityPolicy", "User1", "AS", "S", "S")
SET_GROUPS("SecurityPolicy", "User1", "O", "O", "O")

SET_USER_PRIVS("SecurityPolicy", "User1", "FULL, WRITEUP,
WRITEDOWN, WRITEACROSS")
```

Figure 7. Definición de usuarios y asignación de privilegios

la podrán ver en las tablas. La opción "CHECK_CONTROL" obliga al sistema a comprobar que cuando un sujeto introduce o modifica una tupla, el usuario tiene control de lectura. La opción "READ_CONTROL" hace cumplir el control de acceso de lectura en la política para las operaciones de lectura, modificación y borrado. Finalmente, la opción "WRITE_CONTROL" hace cumplir el control de acceso de escritura en la política para las operaciones de inserción, borrado y modificación.

Cuando se define un nuevo nivel de seguridad, hemos de especificar el nombre de la política, un número, que indicará el orden del nivel de seguridad y que es usado internamente, un nombre corto y el nombre del nivel de seguridad. Además, cuando se crea un nuevo grupo, hemos de especificar el nombre de la política, el número del grupo, que no indicará orden, un nombre corto, el nombre del grupo, y el nombre corto del padre en la jerarquía de grupos.

Una vez que la política de seguridad, los niveles y los grupos han sido definidos, podemos identificar los usuarios en el sistema, y asignarles los privilegios necesarios. En la Figura 7, el usuario "User 1" es definido con la siguiente información: El máximo nivel de seguridad "AS", el nivel de seguridad por defecto "S", en nivel mínimo de seguridad "S", un solo grupo con acceso de lectura "Operarios", un solo grupo con acceso de escritura "Operarios", y por defecto, el grupo "Operarios". Aunque no es habitual asignar privilegios especiales a los usuarios (puesto que permiten vulnerar ciertos aspectos de seguridad), a modo de ejemplo mostramos cómo podríamos hacerlo. La descripción detallada de todas las posibles opciones puede ser vista en Morán (2001).

La forma de asignar información de sensibilidad a las tuplas, una vez que han sido insertadas es a través de funciones de etiquetado. Cuando no hay restricciones de seguridad asociadas con una tabla, las funciones de etiquetado asignan siempre la misma información de sensibilidad a todas las tuplas. En cambio, si existen restricciones de seguridad, éstas se implementarán mediante funciones de etiquetado, que decidirán, en función de los valores de ciertos atributos el valor de la etiqueta con la información de sensibilidad. La Figura 8 muestra tres funciones de etiquetado, y los comando para asignarlas a tablas. La función "Function1" crea la información de sensibilidad dependiendo del valor del atributo "Descripción Bancaria". La función "Function2" crea siempre la misma información de sensibilidad, y finalmente, la función "Function3" crea la información de sensibilidad dependiendo del valor del atributo "Refunds". Esas funciones son asociadas con tablas, y por lo tanto es posible reutilizar la misma función con varias tablas que tengan las mismas propiedades de sensibilidad.

3. CONCLUSIONES Y TRABAJO FUTURO

Los problemas de seguridad de la información tratan con diversos aspectos de investigación, como por ejemplo las técnicas de control de acceso, los métodos criptográficos, técnicas de modelado, especificación y reutilización de requisitos, legislación en materia de protección de datos personales, estándares de seguridad, y un largo etc. Todos esos temas son muy importantes, pero además, creemos que la construcción de sistemas de información seguros debería realizarse a través de un enfoque metodológico, donde la seguridad sea considerada en todas las etapas. En este artículo hemos resumido una metodología para diseñar bases de datos seguras, que implica diferentes modelos y lenguajes que han sido diseñados pensando en la confidencialidad. Evidentemente, esta metodología y estos modelos y lenguajes no son la solución mágica al problema de la seguridad, pero es un punto de comienzo, que puede ser útil para construir bases de datos más seguras.

```
CREATE FUNCTION Function1 (DescripcionBanco: VarChar)
  Return LBACSYS.LBAC_LABEL
As MiEtiqueta varchar2(80);
Begin
   If DescripcionBanco="Banco no español" then MiEtiqueta :=
"S::OAA,OAC";
       else MiEtiqueta := "SC::OAA,OAC";
   end if;
  Return TO_LBAC_DATA_LABEL("SecurityPolicy", MiEtiqueta);
End;

CREATE FUNCTION Function2( )
  Return LBACSYS.LBAC_LABEL
As MiEtiqueta varchar2(80);
Begin
   MiEtiqueta := "AS::O" ;
   Return TO_LBAC_DATA_LABEL("SecurePolicy", MiEtiqueta);
End;

CREATE FUNCTION Function3(Reintegro: Real)
  Return LBACSYS.LBAC_LABEL
As MiEtiqueta varchar2(80);
Begin
   If Reintegro <=3000 the MiEtiqueta := "SC::O";
       else if Reintegro <= 10000 then MiEtiqueta := "S::O";
               else MiEtiqueta := "AS::O";
               end if;
   end if;
  Return TO_LBAC_DATA_LABEL("SecurityPolicy", MiEtiqueta);
End;

APPLY_TABLE_POLICY ("SecurityPolicy", "DatosBancarios",
"Scheme", , "Function1")
APPLY_TABLE_POLICY ("SecurityPolicy",
"AcreedorDeudorEjerciciosPosteriores", "Scheme", , "Function2")
APPLY_TABLE_POLICY ("SecurityPolicy",
"AcreedorPresupuestoGastos", "Scheme", ,"Function3")
```

Figura 8. Funciones de etiquetado

Existen diversas direcciones interesantes en las que extender las ideas presentadas en este artículo. Algunas de las más destacables son la mejora de los lenguajes y técnicas implicadas en esta metodología, considerando nuevos tipos de restricciones de seguridad, investigar en nuevas formas de clasificar la información, incluyendo métricas de seguridad en la metodología, mejorar el prototipo de herramienta CASE que ha sido construido para obtener automáticamente una representación de las bases de datos y de las políticas de seguridad en OLS partiendo solamente del modelo conceptual, y extender la metodología para considerar otros paradigmas de bases de datos, como bases de datos orientadas a web, orientadas a objetos, multimedia, etc.

4. AGRADECIMIENTOS

Esta investigación es parte del proyecto CALDEA (TIC 2000-0024-P4-02) financiado por el Ministerio de Ciencia y Tecnología, y de la red temática RETISSI (TIC2001-5023-E), que es una acción especial dentro del Programa Nacional de Tecnologías de la Información y las Comunicaciones.

5. BIBLIOGRAFÍA

Avison, D., Lan, F., Myers, M. y Nielsen, A. (1999). Action Research. *Communications of the ACM*, 42(1), pp. 94-97.

Batini, C., Ceri, S. y Navathe, S. (1991). *Conceptual Database Design. And entity-relationship approach 1/e.* Addison-Wesley.

Booch, G., Rumbaugh, J. y Jacobson, I. (1999). The Unified Modeling Language, User Guide. Addison-Wesley, Reading, Mass.

Brinkley, D. y Schell, R. (1995). What Is There to Worry About? An Introduction to the Computer Security Problem. *Information Security, An Integrated Collection of Essays*. Chapter 1. Eds.: Abrams, M., Jajodia, S. y Podell, H. IEEE Computer Society. California.

Chung, L., Nixon, B., Yu, E. y Mylopoulos, J. (2000). *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers. Boston/Dordrecht/London.

Connolly, T. y Begg, C. (2002). *Database Systems. A practical Approach to Design, Implementation, and Management*. Addison Wesley.

Devanbu, P. y Stubblebine, S. (2000). Software Engineering for Security: a Roadmap. *The Future of Software Engineering*. Proceedings of the 22nd International Conference on Software Engineering. Ed: Finkelstein, A. pp. 227-239.

Dhillon, G. y Backhouse, J. (2000). Information System Security Management in the new Millennium. *Communications of the ACM*. 43, 7. pp.: 125-128.

Fernández-Medina, E. (2002). Metodología para el Diseño de Bases de Datos Seguras. Tesis Doctoral. Universidad de Castilla-La Mancha. Julio.

Fernández-Medina, E. y Piattini, M. (2001). Metodología para el desarrollo de bases de datos seguras. *Seguridad en Bases de Datos*. Capítulo 8. Eds.: Fernández-Medina, E., Piattini, M.

y Serrano, M. A. Fundación Dintel. Noviembre, Madrid.

Fernández-Medina, E. y Piattini, M. (2002). UML for the Design of Secure Databases. Proceedings to the The 1st International Workshop on Security in Information Systems (SIS 2002) (Into the 4th International Conference on Enterprise Information Systems), pp. 25-38. April, 2002. Ciudad Real (Spain).

Ferrari, E. (2001). Secure DataBase Systems. Second RETISBD meeting. Murcia (Spain), June 2001.

Ghosh, A., Howell, C. y Whittaker, J. (2002). Building Software Securely from the Ground Up. *IEEE Software*. January-February. Vol 19. Nº 1.

Hall, A. y Chapman, R. (2002). Correctness by Construction Developing a Commercial Secure System. *IEEE Software*. January-February. Vol 19. Nº 1.

Jacobson, I., Booch, G. y Rumbaugh, J. (1999). *The Unified Software Development Process*. Addison Wesley.

Morán, R. (2001). Oracle Label Security. Administrator"s Guide. Release 9.0.1. http://download-west.oracle.com/otndoc/oracle9i/901_doc/network.901/a90149.pdf

Muller, R. (1999). *Database Design for Smarties. Using UML for Data Modeling*. Morgan Kaufmann Publishers, inc. San Francisco, California.

Ley Orgánica 15/1999. *Ley Orgánica de Protección de Datos Personales*. BOE núm. 298, December 14, 1999.

Piattini, M. y Fernández-Medina, E. (2001). Specification of Security Constraints in UML. Proceedings of the 35th Annual 2001 IEEE International Carnahan Conference on Security Technology (ICCST 2001), pp. 163-171. October, 2001. London (UK).

Samarati, P. y De Capitani di Vimercati (2001). Access Cont Policies, Models, and Mechanis *Foundations of Security Analys and Design*. Chapter 3. Eds.: Focardi, R. y Gorrieri, R. Sprin Verlag. Alemania.

Sandhu, R. y Chen, F. (1998). The Multilevel Relational Data Mod *ACM Transactions on Informati and Systems Security*. Vol. 1. N June.

SIC (2001). *Seguridad en Informátic Comunicaciones*. April. Nº 44. P

Thuraisingham, B., Schlipper, L., Samarati, P., Lin, Jajodia, S. y Clifton, C. (1997). Security issu in data warehousing and data mining: panel discussion, *in Database Security XI: Status an Prospects*. Eds.: T. and. Lin y S Qian. Chapman and Hall, Londo pp. 3-16.

Warmer, J. y Kleppe, A. (1998). *Th object constraint language*. Massachusetts. Addison-Wesley.