Information Systems for a Connected Society

CAiSE'03 Forum

Forum for short contributions at
The 15th Conference on
Advanced Information Systems Engineering
Klagenfurt/ Velden, Austria, 16 – 20 June, 2003

Proceedings Short Papers

Edited by
Johann Eder
Tatjana Welzer

Table of Contents

definitions by replacing each construct of the parent schema by a disjunction (OR) of the corresponding constructs in the child schemas. Child schema constructs may later be made Void by contract transformations occurring further down the tree. The tree is traversed in this fashion from the root to the leaves until all the nodes are visited. The resulting view definitions are the GAV definitions for the global schema constructs over the local schemas.

For LAV views, the pathway from a local schema to the global schema is processed as above to derive the view definitions. The process is simpler than for GAV views because there is now only a single pathway being traversed, with no branching. More generally, views can be derived for any schema in terms of any set of other schemas provided that pathways linking all the schemas exist.

View definitions can be simplified after they have been generated. This saves later work for the query optimiser when these definitions are substituted into global queries for GAV query processing (which is what AutoMed supports).

The AutoMed query language, IQL, has bags (multi-sets) as its basic collection type, and supports two kinds of operator for manipulating them: bag union, ++, and also a family of operators all derived from one function `fold`. `fold` applies a given function to each element of a bag and then 'folds' a binary operator into the resulting values. For example, `sum = fold (id) (+) 0` and `count = fold (lambda x.1) (+) 0`. Selection, projection, join and group-by operations can also be defined in terms of `fold`. Optimisations for `fold` apply to all operators that can be defined in terms of it. Two particular optimisations can be applied to view definitions generated from BAV pathways. Firstly, instances of `Void` can be removed. `Void` is equivalent to the empty bag and thus.

```
fold f op e Void      = fold f op e []       = e
Void ++ e = [] ++ e = e ++ [] = e ++ Void = e
Void OR e             = e OR Void            = e
```

Secondly, due to the step-wise fashion in which view definitions are generated, **loop fusion** may also be applicable. This optimisation replaces two successive iterations over a collection by one iteration provided the iterations satisfy certain algebraic properties. We refer the reader to [4] for a discussion of IQL, and for references to relevant work on fold-based functional query languages and optimisation techniques for them.

References

1. E. Jasper, N. Tong, P. McBrien, and A. Poulovassilis. View generation and optimisation in the AutoMed data integration framework. Technical report, AutoMed Project, 2003.
2. M. Lenzerini. Data integration: A theoretical perspective. In *Proc. PODS02*, pages 247–258, 2002.
3. P.J. McBrien and A. Poulovassilis. Data integration by bi-directional schema transformation rules. In *Proc. ICDE'03*, 2003.
4. A. Poulovassilis. The AutoMed Intermediate Query Language. Technical report, AutoMed Project, 2001.

A Methodology for Multilevel Databas

Eduardo Fernández-Medina and Mario Piattini

Escuela Superior de Informática. University of Castilla-La
Paseo de la Universidad 4, 13071, Ciudad Real. (SPAIN). Tel: 3
{Eduardo.FdezMedina, Mario.Piattini}@uclm.es http://www.

Abstract. In a Connected Society, the importance of database priva rity increases considerably. Security must be considered as a fun quirement in Information Systems (IS) development, and has to b consideration at all stages of the development, instead of being an marginal requirement considered once the IS is already finished. W methodology for designing secure databases, which allows to desig tabases taking into account constraints regarding sensitive informati requirements phase.

1. Introduction

The Connected Society forces companies and enterprises to evolve, formation properly in order to achieve their objectives and survive. Organizations depend increasingly on IS, which rely upon large dat databases need increasingly more quality and security. Indeed the ve organization depends on the correct management, security and conf information (Dhillon and Backhouse, 2000). The protection of datal requirement, which must be considered carefully, not as an isolated element present in all stages of the database life cycle, from the requ to implementation and maintenance. For this purpose, Hall and Chap pose different ideas for integrating security in the system developn database security is only considered from the cryptographic point of sequence, we need to commit our efforts to design databases that are

All the solutions that have been offered up until now are very im partial and isolated, and they do not solve the problem of database pr Moreover, they do not deal with the security problem at the design le hands, in the traditional database methodologies, security is not cons come this problem, we propose a methodological approach, which all databases taking into account confidentiality from the earliest stages o

We have defined a methodology for designing secure databases tha ent UML models and the Object Constraint Language. It also ada Process and the methodology for developing secure databases that is p

---

[1] Partially supported by DOLMEN (TIC2000-1673-C06-06) and RETISSI 5023-E) of the Research Projects Subdirection of the Ministry of Science and

tano et al. (1994). The methodology allows to create conceptual and logical models of multilevel databases, and to implement them by using *Oracle9i Label Security*. There are a few proposals that try to integrate security into the software development process such as the Semantic Data Model for Security (Smith, 1991) and the Multilevel Object Modeling Technique (Marks et al., 1996), but they have only been prototypes. One more recent proposal is UMLSec (Jürgens, 2002) where UML is extended to develop secure systems. This approach is very interesting, but it only deals with IS, whilst databases are not considered.

2. Methodology Overview

This methodology allows us to classify the information according to its confidentiality properties and to which user roles will have access permissions. It is also possible to specify security constraints on that classification. The methodology ends implementing the multilevel database with Oracle*9i* Label Security.

The methodology will be iterative and incremental, driven by use cases, and centered on the architecture. The stages of the methodology are as follows: *Requirements gathering*, *System Analysis*, *Multilevel Relational Logical Design*, and *Specific Logical Design*.

We have extended Rational Rose, to include and manage automatically the security information and constraints in the multilevel database design process.

2.1. Requirements Gathering

The goal of this stage is to collect and represent requirements considering confidentiality aspects. The most important artifact of this stage is the *extended use case model*, which allows us to represent actors and use cases, indicating confidentiality properties of them through stereotypes. The extended use case model introduces the concept of *secure use case* and *authorized actor*. A secure use case is a use case that should be deeply studied from the point of view of security. An authorized actor is an actor that must have special authorizations in order to carry out a particular use case.

The activities of this stage are as follows: *Gathering initial requirements*, *creating the business model and the system glossary*, *looking for actors*, *looking for use cases*, *looking for persistent elements*, *describing use cases*, *analyzing security in actors and in use cases*, *defining priorities in use cases*, *structuring the use case model*, *looking for relationships between use cases*, and *reviewing use cases*. The most important activity is the *analysis of security in actors and use cases*, which consist of studying whether the use cases have confidentiality requirements, and whether the actors need special authorization in order to carry out the related use case.

2.2. System Analysis

The aim of this stage is to build the database conceptual model, considering all the requirements that have been collected in the previous activities. The conceptual model

is composed of the *extended class diagram* and a set of *security constraints* that are expressed through OSCL language (Piattini and Fernández-Medina, 2001).

The extended class diagram allows to specify confidentiality information in classes, attributes and associations, which indicates the conditions that the subjects have to fulfill to access them. On the other hand, the users are also provided with authorization information. The kinds of security information that have been considered in the methodology are *security levels* and *roles of authorized users*. If one security level is assigned to a class, it means that subjects have to be classified in at least the same level to access the information. If a set of roles is assigned to an element, it means that the subjects have to play at least one of those roles to access the element.

The OSCL language allows the specification of security constraints that define the information about security of classes, attributes or associations, depending on a particular condition. For instance, it is possible to define constraints that specifies that the security level of the objects belonging to a class will be more restrictive if the value of one of its attributes have one particular value. The syntax is easy to understand, because this language is based on the well-known Object Constraint Language.

The activities of this stage are as follows: *Architecture analysis*, *use case analysis*, *classes analysis*, *security analysis*, and *package analysis*.

2.3. Multilevel Relational Logical Design

Once a conceptual model has been developed, we can decide which logical database paradigm is interesting in each case. We have considered relational databases because they are the most used and widespread at present (Leavitt, 2000), but it could be possible to develop secure object-relational or object oriented databases. This stage is the bridge between the conceptual model and the implementation with a specific logical model. The components of the multilevel relational model are as follows: *Database relational model* (includes the definition of each relation or the database, considering the necessary attributes for representing the confidentiality information), *meta-information of the model* (each relation has associated a meta-information tuple, which includes the data type of the attributes, and the valid values of the attributes related to security information), and *security constraints*. The main activities of this stage deal with the transformation of all elements in the extended class diagram into the multilevel relational model.

2.4. Specific Logical Design

At the end of the previous stage, we have built a logical model of the secure database that is implementation independent. In this stage we specify the secure database in a particular logical model: Oracle*9i* Label Security. We have chosen this model because it is part of one of the most important DBMS that allows the implementation of label-based databases.

The activities of this stage are as follows: *Defining the database model* (all the tables), *defining the security policy and their default options*, *defining the security information in the security policy*, *creating the authorized users and assigning their au-*

thorizations, defining security information for tables through la *plementing security constraints through labeling functions and c* *cates*, and finally, *implementing o erations and controlling their*

3. Conclusions

Confluence of databa es, IS and their application in business, to quirements of laws and governments, make necessary more soph to ensure data security.

Traditionally, information security deals with different researcl control techniques, cryptographic methods, etc. Although all these portant, we think that we should use a methodological approach, different levels, is taken into account at all stages of the database ess. In this paper we have summarized a methodology that has be about security and usability, extending the modeling languages, pr straint languages and security models that are most accepted in th search community. We have used the methodology to design a se Spanish local government, and we have solved its confidentiality pr

4. References

Castano, S., Fugini, M., Martella, G. and Samarati, P. (1994). I Addison-Wesley.

Dhillon, G. & Backhouse, J. (2000). Information system security m new millennium. *Communications of the ACM,. 43, 7,* 125-

Hall, A. & Chapman, R. (2002). Correctness by construction devel cial secure system. *IEEE Software, 19, 1,* 18-25.

Jürjens, J. (2002). UMLsec: Extending UML for secure systems de zéquel, J., Hussmann, H. & Cook, S. (Eds.), *UML 2002 –* *c g Language, Model engineering, concepts and tools* (p many, Springer.

Leavitt, N. (2000). Whatever happened to Object-Oriented Data Trends, IEEE Computer Society, August, 16-19.

Marks, D., Sell, P. and Thuraisingham, B. (1996). MOMT: A Multile eling Technique for Designing Secure Database Application ject-Oriented Programming. Vol. 9. Nº 4, pp. 22-29.

Piattini, M. & Fernández-Medina, E. (2001). Specification of securi UML. In proceedings of the 35th Annual 2001 IEEE Interna Conference on Security Technology (ICCST 2001), pp. 16 2001. London (UK).

Smith, G.W. (1991). Modeling Security-Relevant Data Semantics. Pro IEEE Transactions on Software Engineering, Vol. 17. Nº 11, 1195-1203.