Information System Architecture for Secure Data Warehousing

Alberto Abelló [a] aabello@lsi.upc.es

Marta Oliva [b] oliva@eup.udl.es

José Samos [c] jsamos@ugr.es

Fèlix Saltor [a] saltor@lsi.upc.es

[a] Universitat Politècnica de Catalunya (UPC), Dept. de Llenguatges i Sistemes Informàtics (LSI)

[b] Universitat de Lleida (UdL), Dept. d'Informàtica i Enginyeria Industrial (IEI)

[c] Universidad de Granada, Dept. de Lenguajes y Sistemas Informáticos (LSI)

Abstract

This paper is devoted to Secure Data Warehousing architecture and its data schemas. We relate a federated databases architecture to Data Warehouse schemas, which allows us to provide better understanding to the characteristics of every schema, as well as the way they should be defined. Because of the confidentiality of data used to make decisions, and the federated architecture used, we also pay attention to data protection.

Key words: Federated Databases, Data Warehousing, Database Security

Resum

Aquest report est dedicat a una arquitectura de emmagatzematge de dades segur i els seus esquemes. Nosaltres relacionem una arquitectura de bases de dades federades amb els esquemes del magatzem de dades, cosa que ens permet comprendre millor les caracterstiques de cada esquema, al mateix temps que facilita l'estudi dels mecanismes per definir-los. Degut a la confidencialitat de les dades que s'utilitzen a la presa de decisions i l'arquitectura federada que utilitzem, tamb prestem atenci a la protecci de dades.

Paraules clau: Bases de Dades Federades, Emmagatzematge de Dades, Seguretat a Bases de Dades

Contents

1 Introduction **1**

2 Data Warehousing and Multilevel Security Background **2**

3 An example **3**

4 The parts **3**

4.1 The Operational Data Store 4

4.2 The Data Warehouse Schema 4

4.3 Authorization DW Schemas 6

4.4 External DW and Data Mart Schemas 8

5 The whole **8**

6 Conclusions and future work **9**

Bibliography **10**

1 Introduction

Data Warehousing is a relatively new area of study. The idea behind this work is to use the advances already done in other subjects to benefit it. The presence of the word "integration" in the definition of a Data Warehouse (see section 2) invites to choose federations as a first class candidate to contribute its advances.

Specifically, we propose to locate the Data Warehousing schemas in an architecture for federated databases, and study them from this point of view.

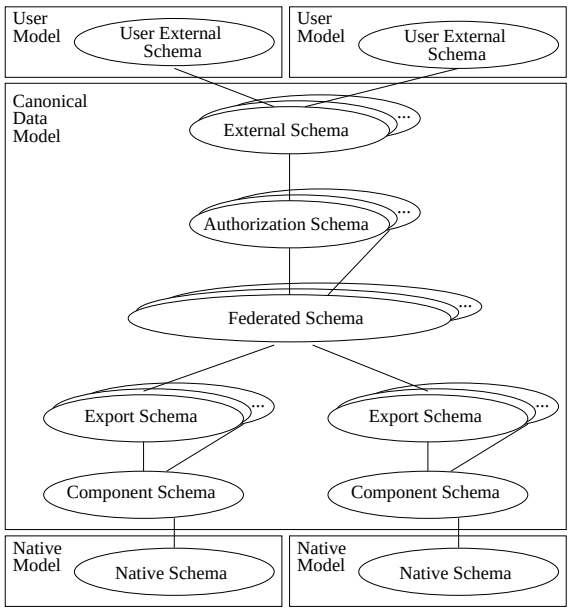


Figure 1: Seven levels schema architecture

We are using the seven levels schema architecture depicted in figure 1. This architecture was presented in [ROSC97], as an extension of that in [SL90], in order to separate different issues in the process to obtain the user schemas from the federated ones. Its different schemas, bottom-up, are:

Native Schema is the Conceptual Schema of a Component DataBase (CDB) expressed in the native model.

Component Schema is the conversion of the Native Schema into the Canonical Data Model (CDM), in our case BLOOM (defined in [CSGS94]).

Export Schema represents the part of the Component Schema that is available to a class of federated users.

Federated Schema is the integration of multiple Export Schemas. Each Federated Schema supports one semantics.

Authorization Schema represents a subset of the Federated Schema, which is accessible by a class of federated users (subjects) with a certain Clearance Level (see section 2).

External Schema defines a schema for a class of users and/or applications. It is still expressed in the CDM.

User External Schema is the conversion of an External Schema to the user data model.

The idea of relating Data Warehousing and Federated Databases was already presented in [SSSB98]. In this paper, the architecture is studied in more depth, paying special attention to the new schemas appearing to achieve Secure Data Warehousing.

The paper is structured as follows: section 2 and 3 give some background and an example respectively; section 4 constitutes the core of the paper and explains the new schemas one by one; in section 5, the schemas are presented all together; and, finally, section 6 contains the conclusions and future work.

2 Data Warehousing and Multilevel Security Background

There are three main data structures in Data Warehousing. As they are defined in [Inm96, IIS98]:

Data Warehouse (DW) is a subject-oriented, integrated, time-variant, nonvolatile collection of data in support of management's decision-making process.

Data Mart (DM) is a subset of a DW that has been customized to fit the needs of a department or subject area.

Operational Data Store (ODS) is a collection of data containing detailed data for the purpose of satisfying the collective, integrated, operational needs of the corporation. An ODS is subject-oriented and integrated, as a DW is. However, an ODS is volatile, current-valued, and detailed, as well.

Another frequently used term in the field is "Multidimensional Schema" (also called "Star Schema" or "Snowflake Schema"). The main purpose of having Multidimensional Schemas defined is to ease the presentation, navigability, and access to the data by distinguishing two kinds of entities: those that are to be analyzed (i.e. central facts), and those used to analyze (i.e. surrounding analysis dimensions).

With regard to security, MultiLevel Security (MLS) is a Mandatory Access Control (MAC) mechanism to protect data. A secure information system needs to protect data from unauthorized accesses. The use of a MLS helps both the access control and the information flow control.

MLS is based on the model of Bell and LaPadula ([BL75]) where access right authorizations are not used but access decisions depend on security levels[1], organized as a partial ordered set, associated to each subject and each protected object. The security level associated to a subject is named "Clearance Level", and the security level where a protected object is classified is named "Confidentiality Level".

There are two rules for MLS systems:

Simple Property a subject will be able to read a protected data if its Clearance Level dominates the Confidentiality Level of the protected data.

[1] Many authors say labels instead of levels

***-Property** a subject is allowed to write/update protected data if its Clearance Level is dominated by the Confidentiality Level of the protected data.

For instance, an ordered set could be *Unclassified* < *Confidential* < *Secret* < *Top-Secret*, so *Top-Secret* dominates *Secret*, *Secret* dominates *Confidential* and finally *Confidential* dominates *Unclassified*.

The information inference problem arises because a user can obtain unauthorized information, or knowledge about the existence of unauthorized information, through accesses to authorized data. That is, when there is an information flow from an object classified at level *l1* to another object classified at level *l2* and *l2* < *l1*, it causes a covert channel.

waste	waste_code	description	risk	TC
	0101	Nitric Acid	5	Secret
	0120	Liquid	4	Secret
	0101	Acid	1	Unclassified

Table 1: Example of polyinstantiation

In multilevel environments, it is possible that subjects with different Clearance Levels have distinct views of the same reality. For example, in relational model the same primary key value of a relation can exists in several levels of classification. This situation is called polyinstantiation and is used to avoid covert channels. Table 1 shows how the value *0101* of attribute *waste_code* appears twice because the system allows that a subject with *Unclassified* Clearance Level can insert data to the *Waste* relation so this user does not know that this *waste_code* value previously exists classified in an upper level.

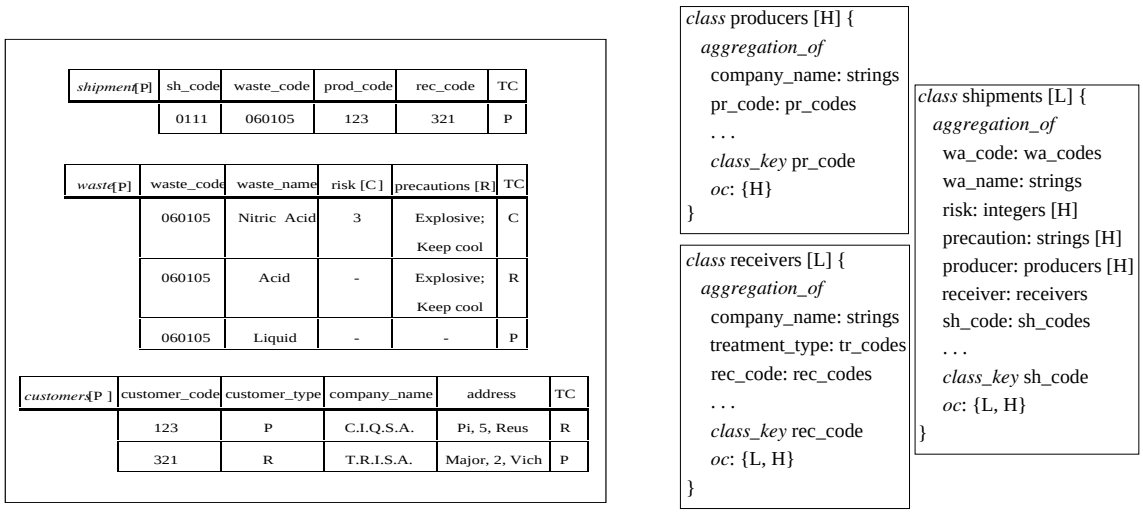
3 An example

To illustrate our architecture we use an example, previously introduced in [ROSC97], which uses a Federated Schema obtained from the integration of two CDBs that belong to an enterprise of industrial waste transports. This example will be used along the rest of the paper.

In figure 2, we can see the conceptual schema of CDB1 and some data in it (expressed in the relational data model), as well as the conceptual schema of CDB2 (expressed in BLOOM data model, which syntax can be read in [AORS99]). The five classes of the Federated Schema showed in figure 3, as well as the partial ordered set of security levels of the federation system, and the classification of all components of the Federated Schema, are obtained through data schema integration and security policies integration processes respectively. These integration processes are out of the scope of this paper, for further information you can see [GSSC95] (data schema integration) and [OS00] (security policies integration).

4 The parts

In this section, we are going to dissect each one of the schemas helping on Secure Data Warehousing. As it is shown in figure 4, all of them are obtained from the Federated Schema, which means we do not start from scratch, but assume the integration work as already done.



shipment[P]

sh_code	waste_code	prod_code	rec_code	TC
0111	060105	123	321	P

waste[P]

waste_code	waste_name	risk [C]	precautions [R]	TC
060105	Nitric Acid	3	Explosive; Keep cool	C
060105	Acid	-	Explosive; Keep cool	R
060105	Liquid	-	-	P

customers[P]

customer_code	customer_type	company_name	address	TC
123	P	C.I.Q.S.A.	Pi, 5, Reus	R
321	R	T.R.I.S.A.	Major, 2, Vich	P

```
class producers [H] {
  aggregation_of
    company_name: strings
    pr_code: pr_codes
    . . .
    class_key pr_code
    oc: {H}
}
```

```
class receivers [L] {
  aggregation_of
    company_name: strings
    treatment_type: tr_codes
    rec_code: rec_codes
    . . .
    class_key rec_code
    oc: {L, H}
}
```

```
class shipments [L] {
  aggregation_of
    wa_code: wa_codes
    wa_name: strings
    risk: integers [H]
    precaution: strings [H]
    producer: producers [H]
    receiver: receivers
    sh_code: sh_codes
    . . .
    class_key sh_code
    oc: {L, H}
}
```

Figure 2: CDB1 and CDB2 schemas

4.1 The Operational Data Store

Once we have a Federated Schema, the first thing we could do is materializing it (tag (1) in figure 4). How that materialization is performed, is completely out of the scope of this paper. What really matters is what we obtain with that materialization i.e. the ODS.

If we analyze the definition of an ODS, we can see that a Federated Schema also satisfies "the collective, integrated, operational needs". About the characteristics of an ODS enumerated in section 2, the federated data is obviously integrated, volatile, current-valued, and detailed. With regard to "subject-oriented", it does not come from the integration mechanisms but from the purpose of who integrates. Thus, it is always possible to obtain a subject-oriented Federated Schema with a small extra design effort.

Therefore, to obtain an ODS, we just have to physically store federated data (solving problems related to CDBs interdependencies, polyinstantiations, etc.), if better response times are desired. Its schema will be exactly one of the Federated Schemas. In our wastes transporting example, the schema of the ODS would be that in figure 3.

4.2 The Data Warehouse Schema

The second thing we could do with the Federated Schema is defining the historic storage schema needed to support the decision-making process. A decision about which data is going to be stored needs to be made. We need to choose a set of integrated data that could be interesting to analyze. Most of the literature seem to suggest the usage of Star Schemas at this point.

The main advantages of Star Schemas are their simplicity and proximity to the business analysis concepts. It makes them quite easy to be understood by the final users. However, even more important than that is

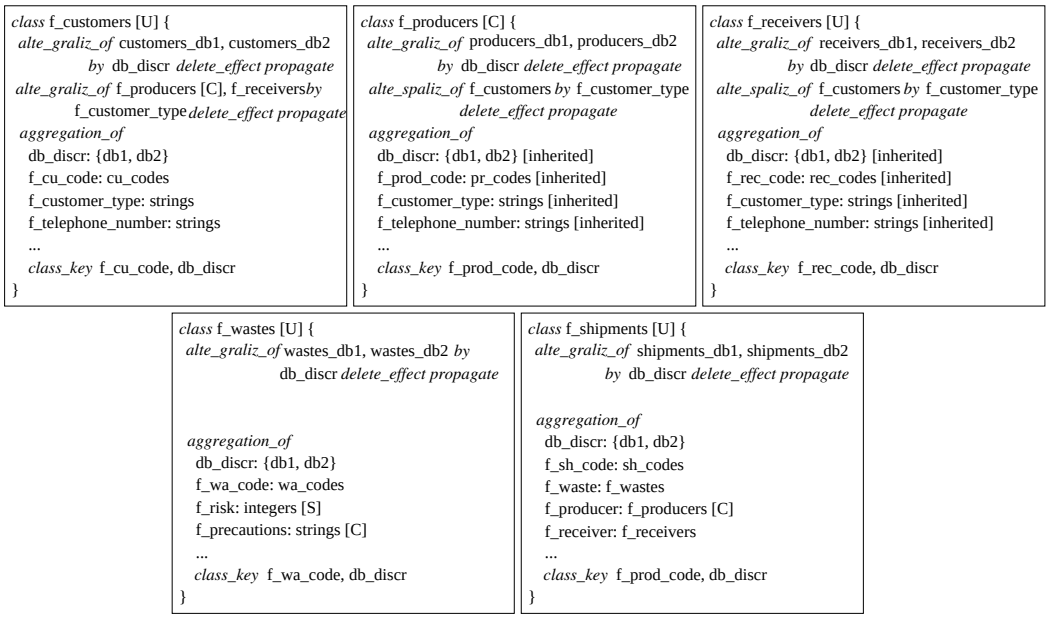


Figure 3: Federated Schema

the fact that they imply a given kind of queries. That structure is quite concrete, and allows to propose specific optimizations, access paths, and storage methods. They are probably the best way to study some small isolated facts with regard to the desired analysis dimensions. However, they are not as good at keeping the data of the whole business. In our architecture, the DW is what [KRMT98] calls the "Storage Structure", and it is only accessed to solve a small number of specific queries. As it is outlined in [IIS98], there is not an homogeneous access pattern in the DW, and that is why isolated Star Schemas do not fit well (some kind of connected "Facts Constellation" is needed).

We do not want to have little knowledge islands but a huge, fully connected continent to travel around. Star Schemas do not seem semantically rich enough to represent the business process all in once, and accomplish that goal. The DW is used to represent the data all together. Thus, its strength does not have to be in easy querying, but in good integration and data semantics representation. Precisely because of that, we propose the CDM of the federation as a basis for a good data model for the DW. Object-Oriented models were found as good CDM in [SCGS91], thereby we are arguing to have an object-oriented model as CDM for the DW. Moreover, the importance of the time dimension in analysis tasks, as well as in the DW (notice the presence of the words "time-variant" in its definition) suggests to temporally extend the model.

The process to obtain the DW Schema, in figure 4 tag (2), should not be query-driven, but data-driven. We need to choose a Federated Schema containing the data of interest for the analysis, and represent time in it. It means a semantic enrichment of the Federated Schema along temporal dimensions, reflecting new, temporal integrity constraints and security restrictions. It is not as simple as extending the keys (OIDs in our case) with an element of time.

We propose the usage of two different kinds of time: "Transaction Time", and "Valid Time" (in the sense of [J+94]). The storage of the time data enters the system (Transaction Time) is mandatory and always possible. At least, two different times could be considered in this temporal dimension. The firsts one

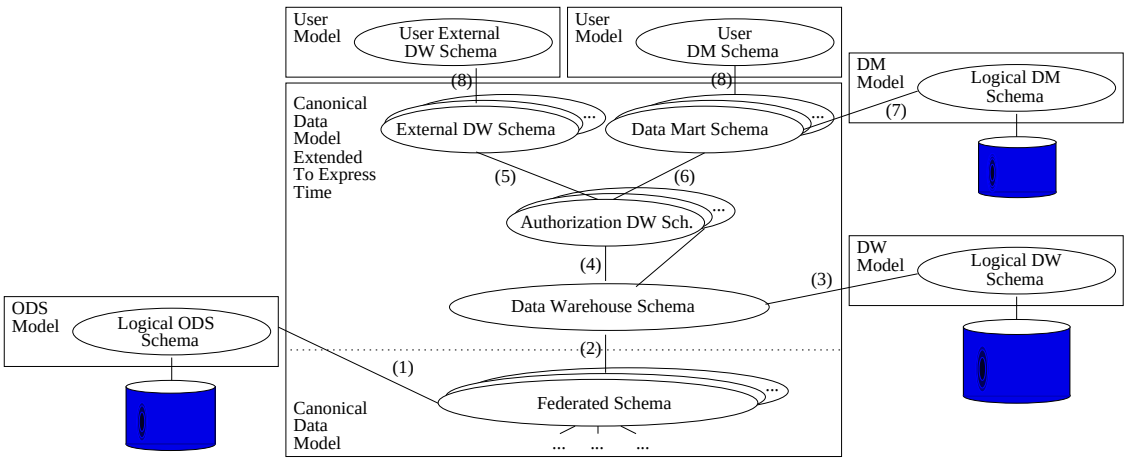


Figure 4: Data Warehousing schemas architecture from the Federated Schema

would be the time when the data was introduced in the operational applications, and the other one would be the time of entrance in the DW itself. When talking about Valid Time dimension, we could consider two different times, as well. The first one would be applied to the objects, and represented by exactly one continuous time interval. About the other Valid Time (represented by a set of non-contiguous, disjoint intervals), it will be used to tag the nexus between objects, indicating when they are valid.

Transaction Time will always be present in the DW (because we will always be able to register, at least, the Transaction Time in the DW, if the component databases do not support it), while the Valid Time will completely depend on its availability in the sources. A good data model for the DW should take both into account, and ease their representation. Transaction Time could be implicit and the Valid Time dimension explicit.

In the example, we should take another Federated Schema containing only that data interesting for the analysis (i.e. *f_telephone_number* attribute in *f_customers* should not be in it, because it does not seem interesting to be analyzed). Once we have that schema, we must modify it to reflect the different times of interest. Classes *f_producers* and *f_receivers* could have an associated Valid Time depending on the dates of their permissions to handle wastes. Each shipment should have a timestamp indicating the day it takes place. Moreover, every object would have a Transaction Time saying when it was introduced/modified in the DW, and maybe another one with the entrance date to the CDB.

All this does not mean that the users need to learn any new data model, later translation from the conceptual model to any other user model is always possible (in figure 4 tag (8)), if desired. Notice, we are not forcing to use an Object-Oriented Database to perform the historic storage, either. Any kind of system could be used, defining its schema in figure 4 tag (3).

4.3 Authorization DW Schemas

Authorized access in a DW is scantily studied, but we think the set of data, stored in a DW, that is needed to support the decision-making process has to be protected from unauthorized accesses, just as any other information system, because data helping to make decisions probably is very confidential.

Authorization Schema in figure 1 helps federated databases on data protection, because each Authorization Schema defines the subset of information that a class of users/applications can access. Authorization DW Schema helps Data Warehousing just as Authorization Schema helps federated databases.

The process to obtain Authorization Schemas/Authorization DW Schemas from Federated Schema/DW Schema (in figure 4 tag (4)) takes into account the security policy of the federation itself. In our case the security policy is based on a MLS system, so Authorization Schemas assist the fulfillment of Simple Property and *-Property rules. It is necessary the existence of an Authorization Schema for each security level of the partial ordered set of the federation itself. The set of data included in an Authorization Schema is classified at the same level, or at a level smaller, than the level corresponding to the Authorization Schema.

Another characteristic of Authorization Schemas is that they are also used to assist information inference control. In our case inference problem gets worse because the use of the semantic abstractions of BLOOM data model. Some abstractions of BLOOM are so rich semantically that the description of some characteristics of a class points out other information.

Our example in figure 3 has *f_receivers* class classified at level U, and *f_producers* class classified at level C. Since *alte_graliz_of* at class *f_customers* is an alternative generalization of the classes *f_receivers* and *f_producers*, a user with Clearance Level U cannot see *f_producers* class, but the property *alte_graliz_of* suggests the existence of at least one unauthorized class (it is a covert channel).

In figure 5, it is showed the Authorization Schema corresponding to level U. We can see, in bold, how some properties have been modified to solve inference problem.

```
class f_customers [U] {
  alte_graliz_of customers_db1, customers_db2
                 by db_discr delete_effect propagate
  gral_graliz_of f_receivers by
                 f_customer_type delete_effect propagate
  aggregation_of
    db_discr: {db1, db2}
    f_cu_code: cu_codes
    f_customer_type: strings
    f_telephone_number: strings
    ...
    class_key f_cu_code, db_discr
}
```

```
class f_receivers [U] {
  alte_graliz_of receivers_db1, receivers_db2
                 by db_discr delete_effect propagate
  gral_spaliz_of f_customers by f_customer_type
                 delete_effect propagate
  aggregation_of
    db_discr: {db1, db2} [inherited]
    f_rec_code: rec_codes [inherited]
    f_customer_type: strings [inherited]
    f_telephone_number: strings [inherited]
    ...
    class_key f_rec_code, db_discr
}
```

```
class f_wastes [U] {
  alte_graliz_of wastes_db1, wastes_db2 by
                 db_discr delete_effect propagate

  aggregation_of
    db_discr: {db1, db2}
    f_wa_code: wa_codes
    ...
    class_key f_wa_code, db_discr
}
```

```
class f_shipments [U] {
  alte_graliz_of shipments_db1, shipments_db2
                 by db_discr delete_effect propagate
  aggregation_of
    db_discr: {db1, db2}
    f_sh_code: sh_codes
    f_waste: f_wastes
    f_receiver: f_receivers
    ...
    class_key f_prod_code, db_discr
}
```

Figure 5: Authorization Schema of level U

4.4 External DW and Data Mart Schemas

Besides reflecting the security aspects of the DW, we can define the subsets of data of interest depending on the classes of users and/or applications (tags (5) and (6) in figure 4). The external schemas are expressed in the CDM. However, they can be translated (tag (8) in figure 4) to any other model.

At this point, the strength is not in the data itself, but in the needs of the users. Here, we will have a query-driven design, where what really matters is the vision the user has. If the users have a multidimensional vision of the data, we will obtain Star External Schemas (by (6) in figure 4). If most of the users have that vision, what we, likely, get is a set of stars sharing some of their dimensions. Sometimes, this is called a "Star Constellation" or "Data Warehouse Bus" (in [KRMT98]).

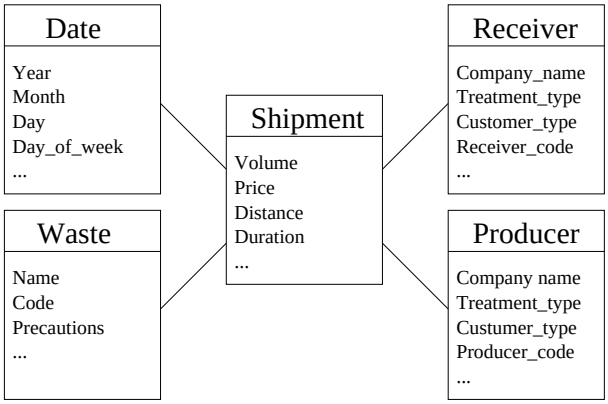


Figure 6: Data Mart Star Schema example

In our industrial wastes example, we could define the Star External Schema depicted in figure 6 from the Authorization Schema in figure 5. In this case, the analyzers are interested in the analysis of the shipments depending on the date, the kind of waste, the producer, and the receiver. That is, for them, the facts are the shipments; and the date, waste, producer, and receiver are the analysis dimensions.

Due to performance reasons, most of these Star Schemas are materialized (represented by (7) in figure 4), giving rise to DM (built with either ROLAP[2] or MOLAP[3] techniques). However, other External Schemas used for Data Mining or solving some sporadic queries would not need to be materialized.

5 The whole

In figure 7, we can see the result of merging the seven levels architecture for federated databases in figure 1, and the schema levels for Secure Data Warehousing in figure 4.

It is important to notice the location of the DW Schema. If we assume that the presence of a processor (performing changes either in data model, in semantics, or in UoD) forces the appearance of a new level, the DW should be placed in between the Federated Schema and the Authorization Schemas. However, the DW Schema is at the same level than the Federated Schema, because they are equally important. If

[2] Relational On-Line Analitical Processing
[3] Multidimensional On-Line Analitical Processing

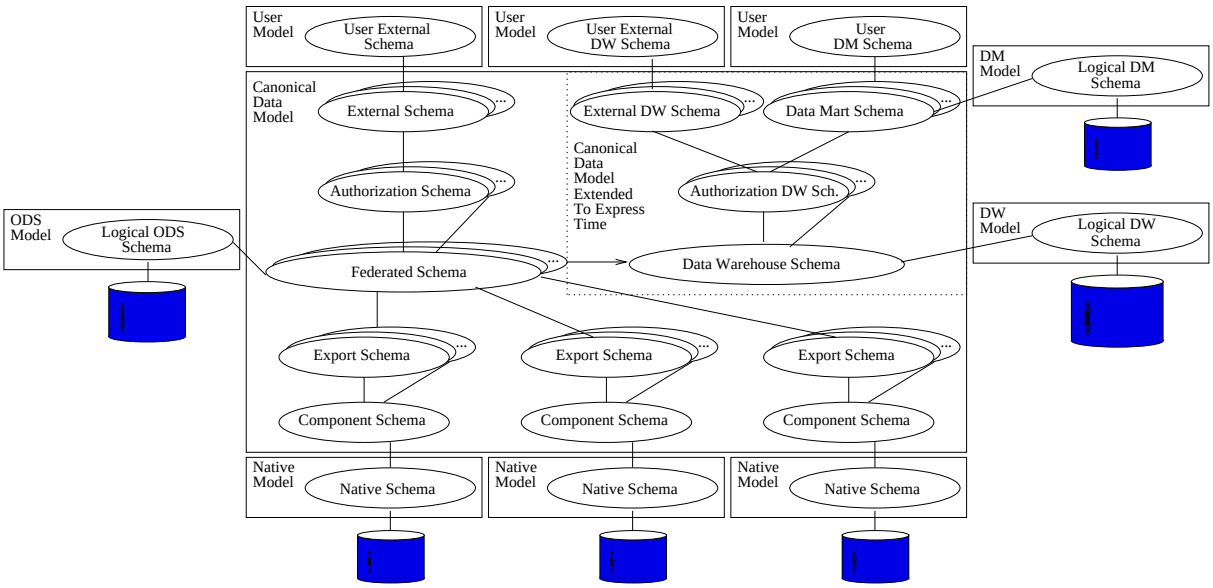


Figure 7: Integrated architecture

the Export Schemas were expressed in a temporal CDM, and we had an integration processor for it, then Federated Schema and Data Warehouse Schema would collapse into a single schema.

On the other hand, the double storage system (DW-DM) should not be avoided. The DW is data-driven designed, and will contain data that may not be sure that will some day be useful (which worsen performance). The DM is query-driven designed, oriented to optimize response times. Thus, what we will, likely, have is a temporal, or relational database supporting time, incrementally designed and populated, as data is generated. From this huge, central DW, we will define and feed smaller DMs, in as needed basis. Notice that we are not suggesting a methodology, but an architecture. Defining a methodology is absolutely out of the scope of this paper, and the architecture does not impose it.

6 Conclusions and future work

In this paper we tried to pay special attention to Data Warehousing schemas architecture, and their conceptual design. We made use of the knowledge in the Federated Databases field. Doing this, it allowed us to consider the integration work as already done. Besides, it invited to consider some problems, regarding data schemas and data protection, from a different point of view.

By locating the different Data Warehousing schemas in an architecture for federated databases, we obtained an integrated architecture that comprises both areas. The data warehousing terminology used by other authors was placed in that architecture. We also emphasized the characteristics of the different schemas, as well as the functions they realize. The DW Schema has been presented as the result of a data-driven design, while the DM Schemas result from a query-driven design.

As future work, we plan to study the mechanisms to define the DW Schema from the Federated Schema, by extracting the knowledge about time dimensions. This process should take into account some security problems as the integration process does. Moreover, because of the materialization of Federated

Schemas, by the ODS or through the DW Schema, we need to study in more depth aspects related to polyinstantiation. Obtaining Star Schemas from the DW Schema will be studied, too.

Acknowledgements

This work has been partially supported by the Spanish Research Program PRONTIC under projects TIC99-1078-C02-01 and TIC99-1078-C02-02, as well as the grant 1998FI-00228 from the Generalitat de Catalunya.

References

[AORS99] A. Abelló, M. Oliva, M.E. Rodríguez, and F. Saltor. The syntax of BLOOM99 schemas. Technical Report LSI-99-34-R, Dept. LSI. UPC, Jul 1999.

[BL75] D.E. Bell and L.J. LaPadula. Secure computer systems: Unified exposition and multics interpretation. Technical Report MTR-2997, (AY/W 020 445), The MITRE Corporation, Bedford, MA, Jul 1975.

[CSGS94] M. Castellanos, F. Saltor, and M. García-Solaco. A canonical model for the interoperability among object-oriented and relational databases. In Ozsu, Dayal, and Valduriez, editors, *Distributed Object Management*, pages 309–314. Morgan Kaufmann, 1994.

[GSSC95] M. García-Solaco, F. Saltor, and M. Castellanos. A Structure Based Schema Integration Methodology. In *Proc. 11th Int. Conference on Data Engineering, Taipei*. IEEE-CS Press, 1995.

[IIS98] W. H. Inmon, C. Imhoff, and R. Sousa. *Corporate Information Factory*. John Wiley & Sons, 1998.

[Inm96] W. H. Inmon. *Building the Data Warehouse*. John Wiley & Sons, 1996.

[J+94] C.S. Jensen et al. A consensus glossary of temporal database concepts. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 23(1):52–64, March 1994.

[KRMT98] R. Kimball, L. Reeves, M.Ross, and W. Thornthwaite. *The Data Warehouse lifecycle toolkit*. John Willey & Sons, 1998.

[OS00] M. Oliva and F. Saltor. Integrating security policies in federated information systems. Submitted for publication, 2000.

[ROSC97] E. Rodríguez, M. Oliva, F. Saltor, and B. Campderrich. On schema and functional architectures for multilevel secure and multiuser model federated db systems. In *Proceedings of the Int. CAiSE'97 Workshop on Engineering Federated Database Systems (EFDBS'97)*, pages 93–104, 1997.

[SCGS91] F. Saltor, M. Castellanos, and M. García-Solaco. Suitability of Data Models as Canonical Models for Federated DBs. *ACM SIGMOD Record*, 20(4):44–48, 1991.

[SL90] A.P. Sheth and J.A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22(3):183–236, September 1990.

[SSSB98] J. Samos, F. Saltor, J. Sistac, and A. Bardés. Database architecture for data warehousing: An evolutionary approach. In *proceedings of 9th Int. Conf. on Database and Expert Systems Applications (DEXA '98)*, pages 746–756. Springer LNCS1460, 1998.