Information System Architecture for Data Warehousing from a Federation

Alberto Abelló [(a)], Marta Oliva [(b)], José Samos [(c)], and Fèlix Saltor [(a)]

[(a)] U. Politècnica de Catalunya (UPC), Dept. de Llenguatges i Sistemes Informàtics
[(b)] U. de Lleida (UdL), Dept. d'Informàtica i Enginyeria Industrial
[(c)] U. de Granada (UGR), Dept. de Lenguajes y Sistemas Informáticos

Abstract. This paper is devoted to 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 Warehouse, Database Security

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 integration concept in the definition of a Data Warehouse (DW) given in [Inm96] ("A Data Warehouse is a subject-oriented, integrated, time-variant, nonvolatile collection of data in support of management's decision-making process") 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.

We are using the seven levels schema architecture presented in [ROSC97]. Its different schemas (drawn in the leftmost side of figure 5, 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 is defined to aply a MultiLevel Security (MLS) mechanism, based on the model of Bell and LaPadula [BL75]. MLS is a Mandatory Access Control (MAC) mechanism to protect data where access right authorizations are not used but access decisions depend on security levels (many authors say labels instead of levels), 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". Each one of the Authorization Schemas represents a subset of the Federated Schema, which is accessible by a class of federated users (subjects) with a certain Clearance Level.

External Schema defines a schema for a class of users and/or applications.

User External Schema is the conversion of an External Schema, which are still expressed in the CDM, 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 Data Warehousing.

The paper is structured as follows: section 2 presents the example used along this paper; section 3 constitutes the core of the paper and explains the new schemas one by one; in section 4, the schemas are presented all together; and, finally, section 5 contains the conclusions and future work.

2 An example

To illustrate our architecture we use an example, that was 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.

```
class f_customers [U] {
 alte_graliz_of customers_db1,
                customers_db2
    by db_discr delete_effect propagate
 alte_graliz_of
    f_producers [C], 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_producers [C] {
 alte_graliz_of producers_db1
                producers_db2
    by db_discr delete_effect propagate
 alte_spaliz_of f_customers
    by f_customer_type
    delete_effect propagate
 aggregation_of
  db_discr: {db1, db2} [inherited]
  f_prod_code: pr_codes [inherited]
  f_customer_type: strings [inherited]
  f_telephone_number: strings [inherited]
  ...
  class_key f_prod_code, db_discr
}
```

```
class f_receivers [U] {
 alte_graliz_of
    receivers_db1, receivers_db2
    by db_discr delete_effect propagate
 alte_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
  f_risk: integers [S]
  f_precautions: strings [C]
  ...
  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_producer: f_producers [C]
  f_receiver: f_receivers
  ...
  class_key f_prod_code, db_discr
}
```

Fig. 1. Federated Schema

The five classes of the Federated Schema showed in figure 1, as well as the partial ordered set of security levels of the federation system (for instance, *Unclassified* < *Confidential*< *Secret*, so that *Secret* dominates *Confidential* and finally *Confidential* dominates *Unclassified*), 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). The classification is indicated by the initial into square brackets (for example *f_risk: integers [S]*), or nothing if it takes as default the classification of the class (for instance *class f_wastes [U]*).

3 The parts

In this section, we are going to dissect each one of the schemas helping on Data Warehousing. As it is shown in figure 2, 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.

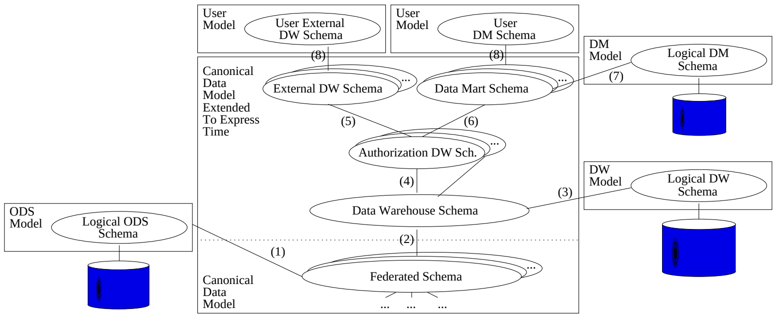

Fig. 2. Data Warehousing schemas architecture from the Federated Schema

3.1 The Operational Data Store

Once we have a Federated Schema, the first thing we could do is materializing it (tag (1) in figure 2). 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 Operational Data Store (ODS). The ODS is defined in [IIS98] as 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.

If we analyze the characteristics of an ODS enumerated above, we can see that a Federated Schema also satisfies "the collective, integrated, operational needs". 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. 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 1.

3.2 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. Star Schemas reflect the natural multidimensional point of view of the users. 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).

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 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 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 at 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 use a temporal model.

The process to obtain the DW Schema, in figure 2 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 with an element of time.

In the example, we should take as source another Federated Schema containing only 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, it has to be modified to reflect the different times of interest, in order to arrive to the Data Warehouse Schema.

3.3 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. Authorization DW Schema helps Data Warehousing just as Authorization Schema helps federated databases, by defining the subset of information that a class of users/applications can access.

The process to obtain Authorization Schemas/Authorization DW Schemas from Federated Schema/DW Schema (in figure 2 tag (4)) takes into account the security policy of the federation itself. As it was previously said, the security policy is based on a MLS system. The existence of an Authorization Schema for each security level of the federation is necessary to reflect the set of elements at every Clearance Level. The set of data included in an Authorization Schema is classified at the same level, or at a smaller level, than the level corresponding to the Authorization Schema.

Another characteristic of Authorization Schemas is that they are also used to assist information inference control. 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*. 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.

```
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
}
```

Fig. 3. Authorization Schema of level U

Our example in figure 1 has class *f_receivers* classified at level U, and class *f_producers* classified at level C. Since *alte_graliz_of* at class *f_customers* is an alternative generalization of the classes *f_receivers* and *f_producers* (which indicates that an instances of *f_customers* must be classified at exactly one of its specializations), 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 (inferred information). In figure 3, it is showed the Authorization Schema corresponding to level U. We can see, in bold, how some properties have been modified (in classes *f_customers*, and *f_receivers*) to solve an inference problem.

Moreover, other classes in the Federates Schema now hide the attributes explicitly classified in levels C and S.

3.4 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 2). The external schemas are expressed in the CDM. However, they can be translated (tag (8) in figure 2) 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 users have. If the users have a multidimensional vision of the data, we will obtain Star External Schemas (by (6) in figure 2). Furthermore, 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]).

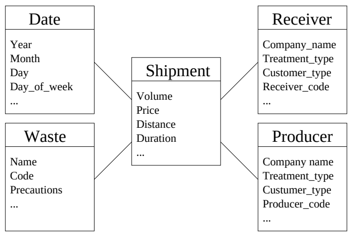

Fig. 4. Data Mart Star Schema example

In our industrial wastes example, we could define the Star External Schema depicted in figure 4 from the Authorization Schema in figure 3. In this case, the analyzers are interested in the analysis of the shipments depending on the date, kind of waste, producer, and 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, conceived to perform OLAP (On-Line Analytical Processing), are materialized (represented by (7) in figure 2), giving rise to Data Marts (DMs). A DM is defined in [IIS98] as a subset of a DW that has been customized to fit the needs of a department or subject area. Other External Schemas used for Data Mining or solving some sporadic queries would not need to be materialized.

4 The whole

In figure 5, we can see the result of merging the seven levels architecture for federated databases, and the schema levels for Data Warehousing in figure 2. It is important to notice the location of the DW Schema. If we assume that the

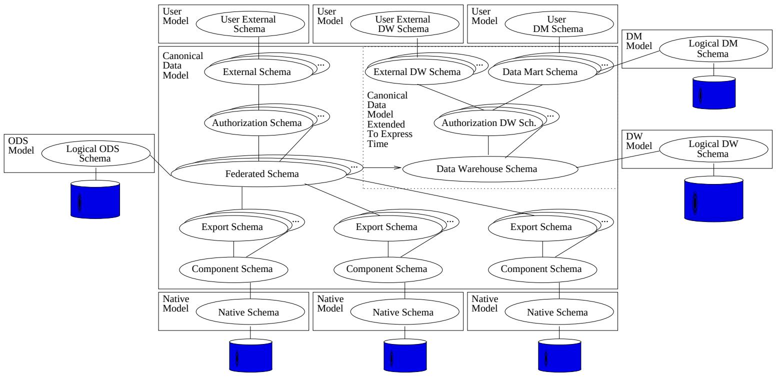

Fig. 5. Integrated architecture

presence of a processor (performing changes either in data model, in semantics, or in Universe of Discurs) 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 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), while the DM is query-driven designed, oriented to optimize response times. Thus, what we will, likely, have to store DW data 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. The former is absolutely out of the scope of this paper, and the architecture does not impose it.

5 Conclusions and future work

In this paper we tried to pay special attention to Data Warehousing schemas architecture, by using the knowledge in the Federated Databases field. This allowed us to consider the integration work as already done, besides, inviting to see 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 this 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, which should take into account some security problems as the integration process does. Moreover, the materialization of Federated Schemas, by the ODS or through the DW Schema, forces the study of new security aspects not taken under consideration for federations. 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 1998 FI-00228 from the Generalitat de Catalunya.

References

[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.

[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.

[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.