A Data Warehouse Engineering Process

Sergio Luján-Mora and Juan Trujillo

D. of Software and Computing Systems, University of Alicante
Carretera de San Vicente s/n, Alicante, Spain
{slujan,jtrujillo}@dlsi.ua.es

Abstract. Developing a data warehouse (DW) is a complex, time consuming and prone to fail task. Different DW models and methods have been presented during the last few years. However, none of them addresses the whole development process in an integrated manner. In this paper, we present a DW development method, based on the Unified Modeling Language (UML) and the Unified Process (UP), which addresses the design and development of both the DW back-stage and front-end. We extend the UML in order to accurately represent the different parts of a DW. Our proposal provides a seamless method for developing DWs.

Keywords: data warehouse, UML, Unified Process, software engineering

1 Introduction

In the early nineties, Inmon [1] coined the term "data warehouse" (DW): "*A data warehouse is a subject-oriented, integrated, time-variant, nonvolatile collection of data in support of management's decisions*". Building a DW is a challenging and complex task because a DW concerns many organizational units and can often involve many people. Although various methods and approaches have been presented for designing different parts of DWs, no general and standard method exists to date for dealing with the whole design of a DW.

In the light of this situation, the goal of our work is to develop a DW engineering process to make the developing process of DWs more efficient. Our proposal is an object oriented (OO) method, based on the Unified Modeling Language (UML) [2] and the Unified Process (UP) [3], which allows the user to tackle all DW design stages, from the operational data sources to the final implementation and including the definition of the ETL (Extraction, Transformation, and Loading) processes and the final users' requirements.

The rest of the paper is structured as follows. In Section 2, we briefly present some of the most important related work and point out the main shortcomings. In Section 3, we summarize our DW engineering process: first, we present the diagrams we propose to model a DW (the results achieved so far), and then we describe the different workflows that make up our process. Finally, we present the main contributions and the future work in Section 4.

2 Related Work

During the last few years, different approaches for the DW design have been presented. On the one hand, different data models [4,5,6,7,8], both conceptual and logical, have been proposed. These approaches are based on their own visual modeling languages or make use of a well-known graphical notation, such as the Entity-Relationship (ER) model or the UML. However, none of these approaches has been widely accepted as a standard DW model, because they present some important lacks. Due to space constraints, we refer the reader to [9] for a detailed comparison and discussion about most of these models.

On the other hand, different DW methods [10,11,12,13] have also been proposed. However, all of them present some of these problems: they do not address the whole DW process, they do not include a visual modeling language, they do not propose a clear set of steps or phases, or they are based on a specific implementation (e.g., the star schema in relational databases).

A key approach is Kimball's *Data Warehouse Bus Architecture* [14], which addresses planning, designing, developing, deploying, and growing DWs. However, this approach also lacks a modeling language that comprises the different tasks.

In conclusion, no general and standard method exists to date for dealing with the whole design of a DW.

3 Data Warehouse Development

The goal of our work is to develop a DW engineering process to make the developing process of DWs more efficient. In order to achieve this goal, we consider the following premises:

- Our method should be based on a standard visual modeling language.
- Our method should provide a clear and seamless method for developing a DW.
- Our method should tackle all DW design stages in an integrated manner, from the operational data sources to the final implementation and including the definition of the ETL processes and the final users' requirements.
- Our method should provide different levels of detail.

Therefore, we have selected the UML as the visual modeling language, our method is based on the well-accepted UP, we have extended the UML in order to accurately represent the different parts of a DW, and we extensively use the UML packages with the aim of providing different levels of detail.

The rest of the section is divided into two clear parts: in Section 3.1 we present the results achieved so far, and in Section 3.2 we outline our current and future lines of work.

3.1 Data Warehouse Diagrams

The architecture of a DW is usually depicted as various layers of data in which data from one layer is derived from data of the previous layer [15]. Following this consideration, we consider that the development of a DW can be structured into an integrated framework with five stages and three levels that define different diagrams for the DW model, as shown in Fig. 1 and summarized in Table 1.

- **Stages**: we distinguish five stages in the definition of a DW:
  - Source, that defines the data sources of the DW, such as OLTP systems, external data sources (syndicated data, census data), etc.
  - Integration, that defines the mapping between the data sources and the DW.
  - Data Warehouse, that defines the structure of the DW.
  - Customization, that defines the mapping between the DW and the clients' structures.
  - Client, that defines special structures that are used by the clients to access the DW, such as data marts (DM) or OLAP applications.
- **Levels**: each stage can be analyzed at three levels or perspectives:
  - Conceptual: it defines the DW from a conceptual point of view.
  - Logical: it addresses logical aspects of the DW design, such as the definition of the ETL processes.
  - Physical: it defines physical aspects of the DW, such as the storage of the logical structures in different disks, or the configuration of the database servers that support the DW.
- **Diagrams**: each stage or level require different modeling formalisms. Therefore, our approach is composed of 15 diagrams, but the DW designer does not need to define all the diagrams in each DW project: for example, if there is a straightforward mapping between the Source Conceptual Schema (SCS) and the Data Warehouse Conceptual Schema (DWCS), the designer may not need to define the corresponding Data Mapping (DM). In our approach, we use the UML [2] as the modeling language, because it provides enough expressiveness power to address all the diagrams. As the UML is a general modeling language, we can use the UML extension mechanisms (stereotypes, tag definitions, and constraints) to adapt the UML to specific domains. In Fig. 1, we provide the following information for each diagram:
  - Name (**in bold face**): the name we have coined for this diagram.
  - UML diagram: the UML diagram we use to model this DW diagram. Currently, we use class, deployment, and component diagrams.
  - Profile (*in italic face*): the dashed boxes show the diagrams where we propose a new profile; in the other boxes, we use a standard UML diagram or a profile from other authors.

Table 1. Data warehouse design framework

The different diagrams of the same DW are not independent but overlapping: they depend on each other in many ways. For example, changes in one diagram may imply changes in another, and a large portion of one diagram may be created on the basis of another diagram. For example, the DM is created by importing elements from the SCS and the DWCS.

In previous works, we have presented some of the diagrams and the corresponding profiles shown in white dashed boxes in Fig. 1: *Multidimensional Profile* [16,17] for the Client Conceptual Schema (CCS) and the *ETL Profile* [18] for the ETL Process and the Exporting Process. In light gray dashed boxes, we show our

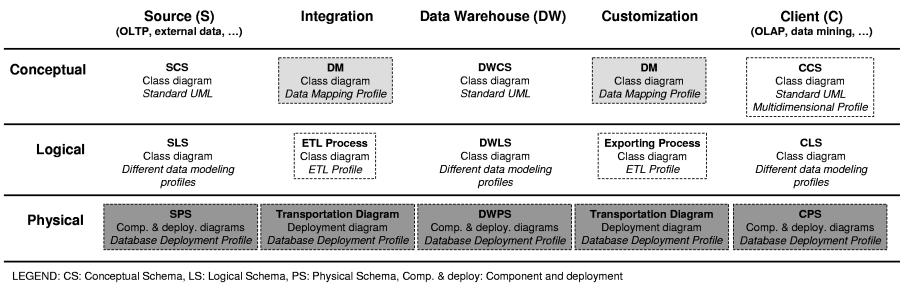

Fig. 1. Data warehouse design framework

last contribution (submitted to review process), the *Data Mapping Profile* for the Data Mapping (DM) between the Source Conceptual Schema (SCS) and the Data Warehouse Conceptual Schema (DWCS), and between the DWCS and the CCS. Finally, in dark gray dashed boxes, we show the profile we are currently working on, the *Database Deployment Profile*, for modeling a DW at a physical level.

On the other hand, the Common Warehouse Metamodel (CWM) [19] is an open industry standard of the Object Management Group (OMG) for integrating data warehousing and business analysis tools, based on the use of shared metadata. This standard is based on three key industry standard: Meta Object Facility (MOF), UML, and XML Metadata Interchange (XMI). We use the CWM when we need to interchange any DW information among different applications.

3.2 Data Warehouse Engineering Process

Our method, called *Data Warehouse Engineering Process* (DWEP), is based on the Unified Software Development Process, also known as Unified Process or simply UP [3]. The UP is an industry standard Software Engineering Process (SEP) from the authors of the UML. Whereas the UML defines a visual modeling language, the UP specifies how to develop software using the UML.

The UP is a generic SEP that has to be instantiated for an organization, project or domain. DWEP is our instantiation of the UP for the development of DWs. Some characteristics of our DWEP inherited from UP are: use case (requirement) driven, architecture centric, iterative and incremental.

According to the UP, the project lifecycle is divided into four phases (Inception, Elaboration, Construction, and Transition) and five core workflows (Requirements, Analysis, Design, Implementation, and Test). We have added two more workflows to the UP workflows: *Maintenance* and *Post-development review*. During the developing of a project, the emphasis shifts over the iterations, from requirements and analysis towards design, implementation, testing, and finally,

maintenance and post-development review, but different workflows can coexist in the same iteration.

For each one of the workflows, we use different UML diagrams (techniques) to model and document the development process, but a model can be modified in different phases because models evolve over time. In the following sections, we comment the main details of the workflows and highlight the diagrams we use in each workflow.

3.3 Requirements

During this workflow, what the final users expect to do with the DW is captured: the final users should specify the most interesting measures and aggregations, the analysis dimensions, the queries used to generate periodical reports, the update frequency of the data, etc. As proposed in [20], we model the requirements with use cases. The rationale of use cases is that focusing "*on what the users need to do with the system is much more powerful that other traditional elicitation approaches of asking users what they want the system to do*" [20]. Once the requirements have been defined, the DW project is established and the different roles are designated.

The UML provides the use case diagram for visual modeling of uses cases. Nevertheless, there is no UML standard for a use case specification. However, we follow the common template defined in [21], which specifies for every use case a name, a unique identifier, the actor involved in the use case, the system state before the use can begin, the actual steps of the use case, and the system state when the use case is over.

3.4 Analysis

The goal of this workflow is to refine and structure the requirements output in the previous workflow. Moreover, the pre-existing operational systems that will feed the DW are also documented: the different candidate data sources are identified, the data content is revised, etc.

We use the Source Conceptual (Logical, Physical) Schema (SCS, SLC, and SPS) (Fig. 1) to model the data sources at different levels of detail. To get quality data in the DW, the different data sources must be well identified.

For example, in Fig. 2 we show the Source Logical Schema of a transactional system that manages the sales of a company. This system will feed with data the DW that will be defined in the following step of the design process.

3.5 Design

At the end of this workflow, the structure of the DW is defined. The main output of this workflow is the conceptual model of the DW. Moreover, the source to target data map is also developed at a conceptual level.

In this workflow, the main diagrams are the Data Warehouse Conceptual Schema (DWCS), the Client Conceptual Schema (CCS), and the Data Mapping

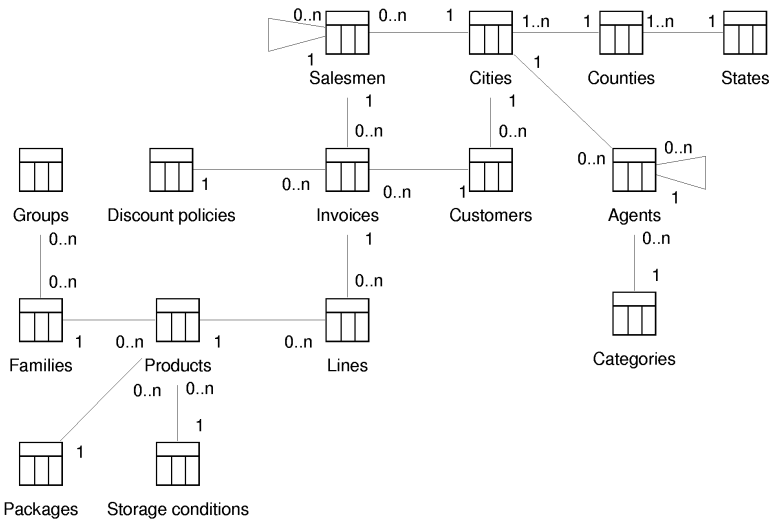

Fig. 2. Source Logical Schema

(DM). The DM shows the relationships between the SCS and the DWCS and between the DWCS and the CCS.

For the CCS, we have previously presented [16] an extension of the Unified Modeling Language (UML) by means of a UML profile. This profile is defined by a set of stereotypes and tagged values to elegantly represent main multidimensional properties at the conceptual level. We make use of the Object Constraint Language (OCL) to specify the constraints attached to the defined stereotypes, thereby avoiding an arbitrary use of these stereotypes. The main advantage of our proposal is that it is based on a well-known standard modeling language, thereby designers can avoid learning a new specific notation or language for multidimensional systems.

For example, in Fig. 3 we show level 1 of a Client Conceptual Schema, composed of three schemas (Production schema, Sales schema, and Salesmen schema). In Fig. 4 we show level 2 of the Sales schema from level 1, composed of one fact (Sales fact) and four dimensions (Stores dimension, Times dimension, Products dimension, and Customers dimension). Finally, in Fig. 5, the definition of the Customers dimension with the different hierarchy levels is showed.

3.6 Implementation

During this workflow, the DW is built: the physical DW structures are built, the DW is populated with data, the DW is tuned for an optimized running, etc. Different implementation diagrams can be created to help this workflow.

The main diagrams in this workflow are the Data Warehouse Logical (Physical) Schema, the Client Logical (Physical) Schema, the ETL Process, the Exportation

Process, and the Transportation Diagram. In the ETL Process, the cleansing and quality control activities are modeled.

For example, in Fig. 6 we show part of a Client Physical Schema. In this example, both the components and the nodes are stereotyped: the components are adorned with the DATABASE and TABLESPACE stereotypes, and the nodes with the SERVER and DISK stereotypes.

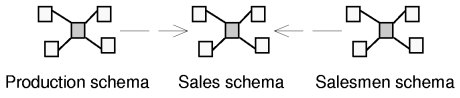

Fig. 3. Client Conceptual Schema (level 1)

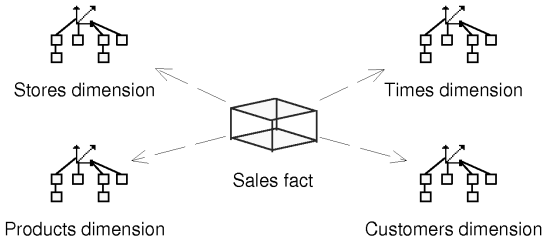

Fig. 4. Client Conceptual Schema (level 2)

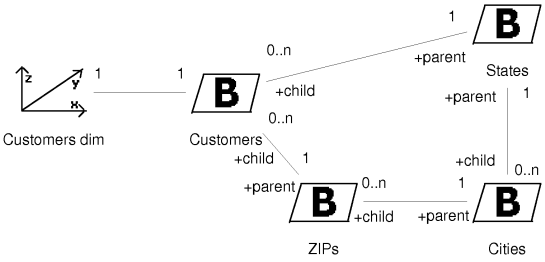

Fig. 5. Client Conceptual Schema (level 3)

3.7 Test

The goal of this workflow is to verify that the implementation works as desired. No new diagrams are created, but previous diagrams (mainly design and implementation diagrams) may be modified according the corrective actions that are taken.

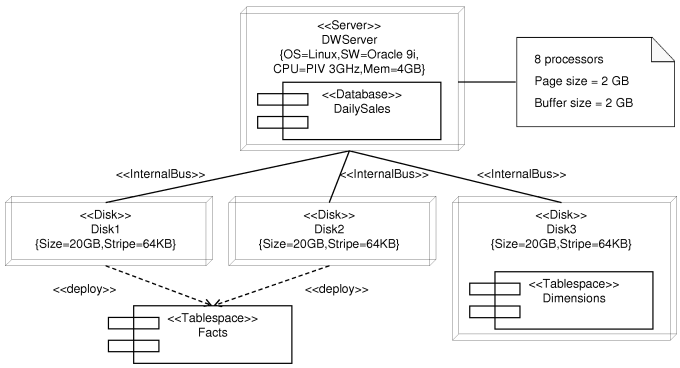

Fig. 6. Client Physical Schema

3.8 Maintenance

Unlike most systems, the DW is never done. The goal of this workflow is to define the refresh and loading processes needed for keeping the DW up to date. This workflow starts when the DW is built and delivered to the final users, but it does not have an end date (it lasts during the life of the DW).

During this workflow, the final users can state new requirements, such as new queries, which triggers the beginning of a new iteration (UP is an iterative process) with the Requirements workflow.

3.9 Post-development review

This is not a workflow of the development effort, but a review process for improving future projects. We look back at the development of the DW, revise the documentation created, and try to identify both opportunities for improvement and major successes that should be taken into account. If we keep track of the time and effort spent on each phase, this information can be useful in estimating time and staff requirements for future projects.

3.10 Top-down or Bottom-up?

Nowadays, there are two basic strategies in the building of a DW: the top-down and bottom-up approaches. The top-down approach recommends the construction of a DW first and then the construcion of DMs from the parent DW. The bottom-up approach uses a series of incremental DMs that are finally integrated to build the goal of the DW. However, in almost all projects, the DMs are built rather independently without the construction of an integrated DW, which is indeed viewed no more as a monolithic repository but rather as a collection of DMs.

Our method also allows both approaches. In the top-down approach, the DW is built first and the data sources are the transactional systems; then, each DM is built independently by using our method, and the DW becomes the only data source for all of them. In the bottom-up approach, the DMs are built first from the transactional systems; then, the DW is built and the data sources are the DMs.

4 Conclusions

In this paper, we have presented the Data Warehouse Engineering Process (DWEP), a data warehouse (DW) development process based on the Unified Modeling Language (UML) and the Unified Process (UP). UP is a generic and stable process that we have instantiated to cover the development of data warehouses. Our main contribution is the definition of several diagrams (techniques) and UML profiles [16,17,18] in order to model DWs more properly. Whereas the different diagrams provide different views or perspectives of a DW, the engineering process specifies how to develop a DW and ties up all the diagrams together. The main advantages of our approach are:

- The use of a development process, the UP, which is the outcome of more than 20 years of experience.
- The use of the UML, a widely accepted visual modeling language, for designing the different DW diagrams and the corresponding transformations.
- The use of the UML as the modeling language provides much better tool support than using an own modeling language.
- The proposal of a DW development process that addresses both the back-end and the front-end of DWs in an integrated manner.

Currently, we are working on the *Database Deployment Profile*, for modeling a DW (and databases in general) at a physical level, we are concluding the definition of the different workflows that the process comprises, and we also plan to include new UML diagrams (sequence, collaboration, statechart, and activity diagrams) to model dynamic properties of DWs. Moreover, we plan to carry out an empirical evaluation of our proposal, in order to validate the correctness and usefulness of our approach.

References

1. Inmon, W.: Building the Data Warehouse. QED Press/John Wiley (1992) (Last edition: 3rd edition, John Wiley & Sons, 2002).
2. Object Management Group (OMG): Unified Modeling Language Specification 1.5. Internet: http://www.omg.org/cgi-bin/doc?formal/03-03-01 (2003)
3. Jacobson, I., Booch, G., Rumbaugh, J.: The Unified Software Development Process. Object Technology Series. Addison-Wesley (1999)
4. Golfarelli, M., Rizzi, S.: A Methodological Framework for Data Warehouse Design. In: Proc. of the ACM 1st Intl. Workshop on Data Warehousing and OLAP (DOLAP'98), Bethesda, USA (1998) 3–9

5. Cabibbo, L., Torlone, R.: A Logical Approach to Multidimensional Databases. In: Proc. of the 6th Intl. Conf. on Extending Database Technology (EDBT'98). Volume 1377 of LNCS., Valencia, Spain (1998) 183–197
6. Tryfona, N., Busborg, F., Christiansen, J.: starER: A Conceptual Model for Data Warehouse Design. In: Proc. of the ACM 2nd Intl. Workshop on Data Warehousing and OLAP (DOLAP'99), Kansas City, USA (1999)
7. Husemann, B., Lechtenborger, J., Vossen, G.: Conceptual Data Warehouse Design. In: Proc. of the 2nd Intl. Workshop on Design and Management of Data Warehouses (DMDW'00), Stockholm, Sweden (2000) 3–9
8. Trujillo, J., Palomar, M., Gómez, J., Song, I.: Designing Data Warehouses with OO Conceptual Models. IEEE Computer, special issue on Data Warehouses **34** (2001) 66–75
9. Abelló, A., Samos, J., Saltor, F.: A Framework for the Classification and Description of Multidimensional Data Models. In: Proc. of the 12th Intl. Conf. on Database and Expert Systems Applications (DEXA'01), Munich, Germany (2001) 668–677
10. Kimball, R.: The Data Warehouse Toolkit. John Wiley & Sons (1996) (Last edition: 2nd edition, John Wiley & Sons, 2002).
11. Giovinazzo, W.: Object-Oriented Data Warehouse Design. Building a star schema. Prentice-Hall, New Jersey, USA (2000)
12. Cavero, J., Piattini, M., Marcos, E.: MIDEA: A Multidimensional Data Warehouse Methodology. In: Proc. of the 3rd Intl. Conf. on Enterprise Information Systems (ICEIS'01), Setubal, Portugal (2001) 138–144
13. Moody, D., Kortink, M.: From Enterprise Models to Dimensional Models: A Methodology for Data Warehouse and Data Mart Design. In: Proc. of the 3rd Intl. Workshop on Design and Management of Data Warehouses (DMDW'01), Interlaken, Switzerland (2001) 1–10
14. Kimball, R., Reeves, L., Ross, M., Thornthwaite, W.: The data warehouse lifecycle toolkit. John Wiley & Sons (1998)
15. Jarke, M., Lenzerini, M., Vassiliou, Y., Vassiliadis, P.: Fundamentals of Data Warehouses. 2 edn. Springer-Verlag (2003)
16. Luján-Mora, S., Trujillo, J., Song, I.: Extending UML for Multidimensional Modeling. In: Proc. of the 5th Intl. Conf. on the Unified Modeling Language (UML'02). Volume 2460 of LNCS., Dresden, Germany (2002) 290–304
17. Luján-Mora, S., Trujillo, J., Song, I.: Multidimensional Modeling with UML Package Diagrams. In: Proc. of the 21st Intl. Conf. on Conceptual Modeling (ER'02). Volume 2503 of LNCS., Tampere, Finland (2002) 199–213
18. Trujillo, J., Luján-Mora, S.: A UML Based Approach for Modeling ETL Processes in Data Warehouses. In: Proc. of the 22nd Intl. Conf. on Conceptual Modeling (ER'03). Volume 2813 of LNCS., Chicago, USA (2003) 307–320
19. Object Management Group (OMG): Common Warehouse Metamodel (CWM) Specification 1.0. Internet: http://www.omg.org/cgi-bin/doc?ad/2001-02-01 (2001)
20. Bruckner, R., List, B., Schiefer, J.: Developing Requirements for Data Warehouse Systems with Use Cases. In: Proc. of the 7th Americas Conf. on Information Systems (AMCIS'01), Boston, USA (2001) 329–335
21. Arlow, J., Neustadt, I.: UML and the Unified Process. Object Technology Series. Addison-Wesley (2002)