ProtOLAP: Rapid OLAP Prototyping with On-Demand Data Supply

Sandro Bimonte, Elodie Edoh-alove
IRSTEA, TSCF
24 Av. Des Landais
Aubière, France
name.surname@irstea.fr

Hassan Nazih
LIMOS-UMR CNRS
Blaise Pascal University, Campus des Cezeaux
Aubière, France
hassanazih@gmail.com

Myoung-Ah Kang
LIMOS-UMR CNRS
Blaise Pascal University, Campus des Cezeaux
Aubière, France
kang@isima.fr

Stefano Rizzi
DISI-University of Bologna
Viale Risorgimento, 2
Bologna, Italy
stefano.rizzi@unibo.it

ABSTRACT
The approaches to data warehouse design are based on the assumption that source data are known in advance and available. While this assumption is true in common project situations, in some peculiar contexts it is not. This is the case of the French national project for analysis of energetic agricultural farms, that is the case study of this paper. Here, the above-mentioned methods can hardly be applied because source data can only be identified and collected once user requirements indicate a need. Besides, the users involved in this project found it very hard to express their analysis needs in abstract terms, i.e., without visualizing sample results of queries, which in turn would require availability of source data. To solve this deadlock we propose ProtOLAP, a tool-assisted fast prototyping methodology that enables quick and reliable test and validation of data warehouse schemata in situations where data supply is collected on users' demand and users' ICT skills are minimal. To this end, users manually feed sample realistic data into a prototype created by designers, then they access and explore these sample data using pivot tables to validate the prototype.

Categories and Subject Descriptors
D.2.10 [**Software Engineering**]: Design – *methodologies*; H.4.2 [**Information Systems Applications**]: Types of Systems – *decision support*

Keywords
Data warehouse design; Fast prototyping; UML profiles.

1. INTRODUCTION
Data Warehouses (DWs) are large repositories of data aimed at supporting the decision-making process by enabling flexible and interactive analyses. Data in DWs are normally represented according to the multidimensional model, which organizes them in *facts* described by *measures* and accessed via *dimensions*. OLAP clients allow decision makers to visualize and explore facts during querying sessions, whose results are displayed using interactive pivot tables and graphical displays. A basic Relational OLAP (ROLAP) architecture is composed of (i) a relational DBMS, that stores DW data; (ii) an *OLAP server*, that sits between the DBMS and the client to bridge the gap between the relational model used by the former and the multidimensional model used by the latter; (iii) an *OLAP client*, that combines and synchronizes tabular and graphical displays and allows query formulation; (iv) an ETL tool that extracts data from data sources, transforms them, and loads them into the DW.

A basic distinction is made on DW design methodologies depending on the role given to user requirements [5]: in *requirement-driven approaches*, a conceptual schema of the DW is drawn starting from the requirements expressed by users; in *source-driven approaches*, a conceptual schema is derived starting from the schemata of the data sources that will feed the DW; in *mixed* approaches, the two activities are conducted in parallel. Some attempts have also been made to apply agile practices to DW design. For instance, *4WD* aims to make the development process more efficient and predictable [6]; the main methodological principles adopted to this end are incrementality and iteration, prototyping, user involvement, and automated schema transformation.

All the approaches mentioned above are based on the assumption that the source data for feeding the DW are known in advance and available. Source data play a key role in source-driven and mixed approaches because they are used to give a strong imprint to the conceptual schema; even in requirement-driven and agile approaches, knowledge and availability of source data during the initial project stages is necessary for testing the developed prototypes and for validating the requirements through preliminary loading tests [7].

The assumption of early source data availability is true in common project situations, in which case the existing approach have proven to work well. However, in some peculiar contexts, this assumption may not be true. This is the case for instance of the French national project for analysis of energetic agricultural farms, presented in [3], that is the case study of this paper. Here, the above-mentioned methods can hardly be applied because source data will be identified and collected a posteriori, according to the information needs represented in the conceptual schema. Besides, the decision makers involved in this project found it very

DOLAP'13, October 28, 2013, San Francisco, CA, USA.

http://dx.doi.org/10.1145/2513190.2513199

hard to express their analysis needs in abstract terms, i.e., without visualizing sample results of queries, which in turn would require availability of source data. To solve this deadlock, there is a need for a fast prototyping methodology that enables users to quickly and reliably test and validate the DW schemata created by designers in situations where (i) source data is not available initially but will only be collected once user requirements indicate a need, and (ii) users' ICT skills are minimal.

In this paper we address this issue by presenting *ProtOLAP*, a tool-assisted methodology whose steps can be summarized as follows:

1. Designers create a DW conceptual schema in a requirement-driven fashion, i.e., starting from the users' analysis needs.
2. The conceptual schema is automatically translated into a logical schema and prototyped.
3. Users manually feed sample realistic data into the prototype.
4. Users access and explore these sample data using simple pivot tables, so as to validate the prototype. If the prototype is not validated, go back to step 1 and start a new iteration.
5. Once the prototype is validated and declared conformed to the analysis requirements, source data are collected, ETL is designed, and the prototype is engineered.

2. RELATED WORK

Several conceptual models (either based on the Entity-Relationship formalism, or on UML, or on ad hoc graphical constructs —see [14] for a survey) and requirement models (such as [18]) for DWs have been proposed in literature to aid designers during development and give decision makers a reliable support in discussing and validating requirements. In particular, several works address the automatic implementation of requirements and conceptual schemata onto ROLAP architectures to encourage error-free design (e.g., [2][9]). However, these methods require a significant training about OLAP and multidimensional modeling to be delivered to users.

Also in the field of ETL design, some works propose the use of conceptual models [17] and suggest the adoption of a model-driven approach [1]. However, these models reflect the complexity of ETL procedures, and consume an important part of time and resources of the development process even when their implementation is automatic. This means that they are not well suited for the early stages of agile DW development, during which close and user-assisted iteration is crucial to guarantee that users and designers can agree on requirements quickly and reliably.

In [8], the authors present a tool to feed a DW with sample data obtained using statistical methods applied to data sources. However, this tool is not integrated within a larger system for fast prototyping, thus it is not usable for our purposes. Finally, the DW testing framework proposed in [7] includes an early-loading test made by loading sample source data during the first design phases. This is aimed at letting users validate requirements when the design is not settled yet, so as to reduce the cost for correcting errors and misunderstandings; unfortunately, it requires some source data to be available, which is not the case in our context.

3. CASE STUDY

The EDEN project aims at providing ICT-based solutions to assess the energetic performance of farms. The technologies employed include for instance low-cost sensors and RFIDs, installed on agricultural equipment to provide reliable and continuous data to enable energetic performance indicators to be calculated at the finest detail (field, technical operation, etc.).

In this context, the decision makers are farmers and *life-cycle assessment* (LCA) experts who want to analyze energetic indicators of their farms, such as liters of fuels by plot, days for a plowing operation, or number of animals used for milk production per year and farm according to LCA diagnostics. Noticeably, ICT technologies are installed on the farm equipment in function of the indicators to be analyzed, e.g., computing an indicator based on worked plots requires GPS sensors to be installed on tractors.

In [3] we presented the spatial DW developed in the context of the EDEN project. The complexity of the queries supported is in line with those of common DW projects. However, the combined use of different aggregation operators along different dimensions to compute indicators, together with the relevant role played by the spatial dimensions, makes it very hard for non-skilled users to verify that the requirements elicited and modeled by designers are actually consistent with their analysis needs, which in turn makes the requirement validation process long and uncertain. The most effective approach to validation here would be to let users "play" with real data using an OLAP front-end —but this cannot apparently be done unless the relevant source data are identified and some basic ETL is provided.

4. THE PROTOLAP METHODOLOGY

In this section, before outlining ProtOLAP, we list a set of methodological requirements as emerged from the EDEN case study:

1. The querying power of a DW directly depends on its underlying conceptual schema. Thus, it is widely recognized that the success of a DW project largely depends on a comprehensive specification of the users' analysis requirements and on the accuracy of their translation into a conceptual schema. When decision makers have little or no experience with ICT and in particular with information systems and OLAP tools, and when they do not know in advance which indicators could be useful for analyses, *using an iterative design methodology* is necessary to enable a progressive refinement of the conceptual schema.
2. While conceptual and logical schemata represent a key tool for designers, they may be quite hard to understand by users. When users have little or no skills in ICT (and in OLAP in particular), *the validation of requirements with users cannot hinge on an analysis of design schemata*.
3. OLAP queries can be quite complex (for instance, due to the combined use of different aggregation operators); *delivering pivot tables with application domain data* to let users interactively play with them, could definitely make the validation of the proposed conceptual schema faster and more effective. Indeed, pivot tables showing application domain data can be considered as mockups of DW prototypes.
4. When relevant data sources cannot be identified in advance, so that the source data must be collected "on users demand" depending on the required analyses and on the indicators of interest, *a requirement-driven approach must necessarily be followed*. Besides, keeping the prototyping cycle fast and effective calls for *postponing ETL design to the phase of late engineering of the prototype*.

The ProtOLAP methodology takes into account all the previous requirements. It is sketched in Figure 1 and can be described as follows.

Initially, decision makers discuss their analysis needs with designers through natural language, mainly in terms of the indicators they need. At this time, designers can quickly draw a

draft conceptual schema; the Spatial Datacube UML profile is used to this end [4]. The draft is then progressively refined thanks to a close interaction with decision makers.

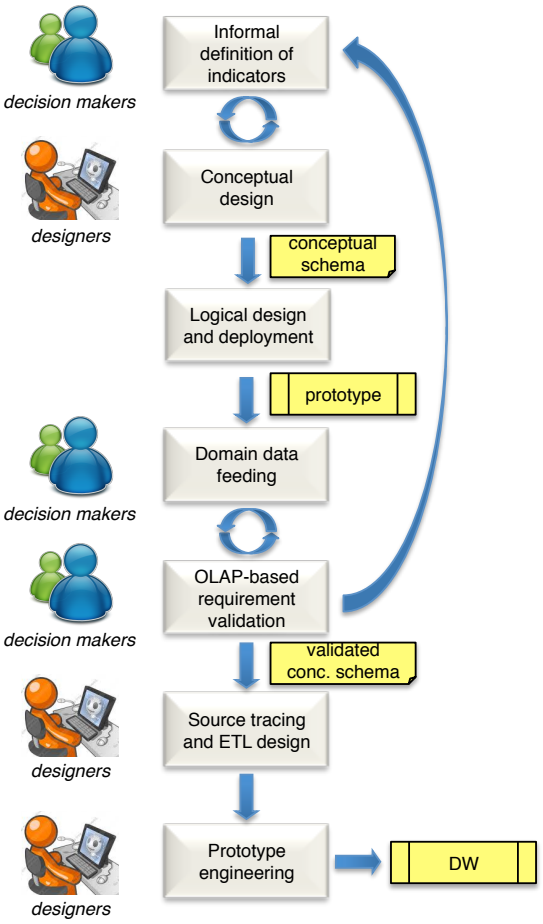


Figure 1. The ProtOLAP methodology

When a 1.0 conceptual schema is produced, i.e., one that the designers judge to be consistent and possibly complete and accurate, the next phase is triggered to produce a prototype. Here, a logical (relational) schema and the related metadata are automatically generated and deployed starting from the conceptual schema. The target RDBMS is Oracle [11], while the target OLAP server is Mondrian [12].

Now decision makers can manually insert some realistic data into the prototype by means of a user-friendly interface. In particular, they directly insert sample values for members of hierarchies, while facts and measure values are automatically and randomly generated respecting user-defined constraints and ranges.

Finally, decision makers can explore the data they inserted in an OLAP fashion using the JRubik front-end [16]. By doing so, they can understand whether the prototype (and, consequently, the underlying conceptual schema) correctly models their analysis needs. If not, a new iteration is triggered. Otherwise, the requirements are considered validated; the required data sources are traced, proper ETL procedures are set, and the prototype can be engineered. Of course, there is a possibility that the source data required are not actually available; in our EDEN project, this never happened because most data come from on-the-field sensors and decision makers have a precise knowledge of which types of sensors were available to acquire data.

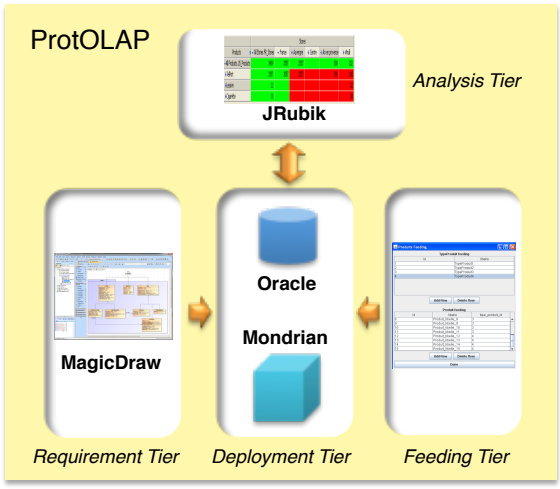


Figure 2. The ProtOLAP Architecture

5. THE PROTOLAP SYSTEM

In this section, we describe the ProtOLAP system that implements our methodology. The architecture, shown in Figure 2, is based on a ROLAP platform and is composed of four tiers: the *requirement tier*, used by designers to draw UML-based conceptual schemata by means of the MagicDraw CASE tool [10]; the *deployment tier*, that includes the Oracle RDBMS, the Mondrian OLAP server, and the tool that creates relational schemata for Oracle and metadata for Mondrian starting from the conceptual schema; the *analysis tier*, that allows decision makers to query data stored in the deployment tier using the JRubik OLAP client; and the *feeding tier*, that automatically generates a visual interface through which decision makers can feed the deployment tier with application domain data. These tiers will be described in some detail in the following subsections.

5.1 The Requirement Tier

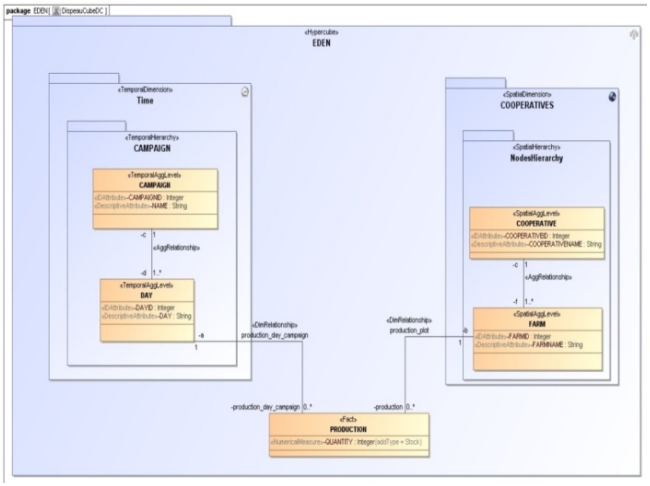
The ProtOLAP methodology is iterative, i.e., it operates by progressively refining a conceptual schema and the related prototype. So, the requirement tier must effectively support designers in easily and rapidly define multidimensional conceptual schemata that can be automatically deployed. For these reasons, we have chosen to adopt the Spatial Datacube UML profile and its implementation on the MagicDraw CASE tool.

The Spatial Datacube profile [4] extends UML with stereotypes for complex OLAP applications, and it guarantees that well-formed multidimensional schemata are created by means of a set of OCL constraints. It is organized in two models representing the static and dynamic elements of OLAP applications: the *SDW model* (Figure 3.a) and the *Aggregation model* (Figures 3.b-c).

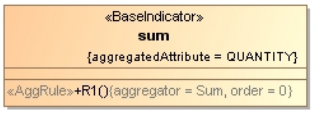
- The SDW model defines dimensions using the *package* UML element. Dimensions are composed of levels, and facts (e.g., class `PRODUCTION` with the «`Fact`» stereotype) are described by measures that are represented as attributes. Each element is typed as either spatial, thematic, or temporal (e.g., «`TemporalAggLevel`» for a temporal level like class `DAY`).
- The aggregation model represents how measures are aggregated along dimensions («`BaseIndicator`» stereotype). Aggregation rules can be defined using dimensions, hierarchies, and levels in order to define complex aggregations. In particular, measures are represented with the `aggregatedAttribute` tagged value; an aggregation

function, `Aggregator`, is defined as a parameter of a method of the fact class.

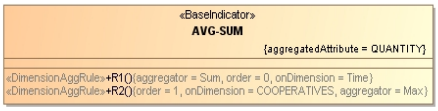
Figure 3.a shows a simplified version of the SDW model of our case study [3] where the `QUANTITY` measure, representing the milk produced, is analyzed according to two dimensions: a spatial dimension representing farms and cooperatives, and a temporal dimension that organizes days by cultivation campaigns. In the example of Figure 3.b, a base indicator summing the `QUANTITY` measure along all dimensions is represented. This indicator supports OLAP queries like: "*What is the total quantity of produced milk per cooperative and campaign?*". As previously stated, decision makers also need more complex aggregations. For example, the `QUANTITY` measure must be aggregated using the sum on the temporal dimension and the max on the spatial dimension in order to identify what is most productive farm by cooperative. This kind of queries can be addressed using the base indicator shown in Figure 3.c, that aggregates `QUANTITY` using the `Sum` operator along the `Time` dimension and the `Max` operator along the `Cooperatives` dimension.



(a)



(b)



(c)

Figure 3. Conceptual schema for our case study; (a) SDW model, (b) `Sum` base indicator, (c) `Max-Sum` base indicator

MagicDraw [10] is a commercial CASE tool for drawing UML models and architecture environments, which supports the UML profiling extension mechanism as well as declaration and automatic checking of OCL constraints. The implementation of the Spatial Datacube profile in MagicDraw allows designers to rapidly define well-formed conceptual schemata in a few clicks through a well-structured organization of the profile elements. For example, when a designer wants to change the aggregation function of a base indicator according to the decision makers needs, she can simply click on the aggregator tagged value and choose another aggregation function among the UML profile elements, as shown in Figure 4.

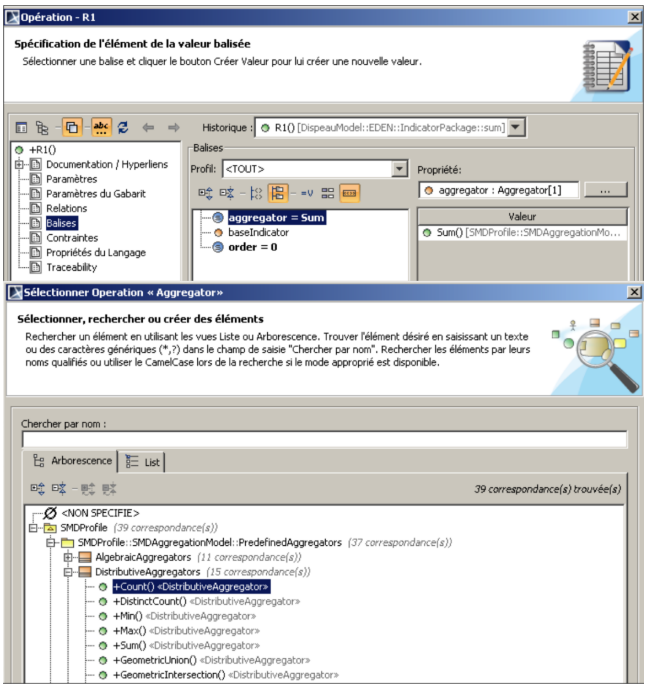


Figure 4. Conceptual tier: design with MagicDraw

5.2 The Deployment Tier

In our implementation data storage is achieved using Oracle Spatial [11], a DBMS that provides native support for spatial data and allows tables with spatial columns and indexes on these data to be created, while Mondrian is used as an OLAP server. Mondrian [12] is an open-source OLAP server belonging to the Pentaho Business Intelligent Suite, and it uses XML metadata to induce a multidimensional schema on top of relational schemata hosted on any RDBMS. For querying, Mondrian uses the standard OLAP query language MDX, that allows complex aggregations and operations to be defined.

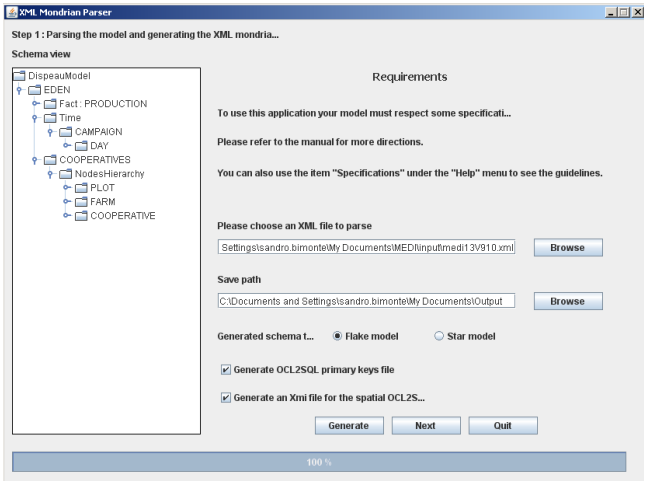


Figure 5. Parsing of the XMI file representing the UML profile

ProtOLAP takes as input the XMI (XML Metadata Interchange) file representing a conceptual schema in the form of a Spatial Datacube UML class diagram. XMI is a standard used to represent and exchange UML models and it is automatically generated by MagicDraw. A visual interface allows users to load

the XMI file and to choose the type of schema to use (star or snowflake) (Figure 5); then the tool automatically generates :

- A visual representation of DW elements.
- The star or snowflake relational schema for Oracle that enables the persistent storage of multidimensional data. Relational schemata are generated using an extended version of OCL2SQL [13], a Java open source tool capable of automatically translating OCL constraints into SQL code for Oracle. Its output is SQL code to create the DW physical schema in Oracle, together with a set of SQL queries and triggers to implement the data integrity checks modeled by OCL constraints.
- The Mondrian XML metadata representing spatio-multidimensional elements (facts, levels, etc.) and the MDX-based calculated members for complex indicators according to previously generated SQL schemata.

Remarkably, this phase is completely automated without any intervention of designers; according to [6], this accelerates software development and promotes standard processes.

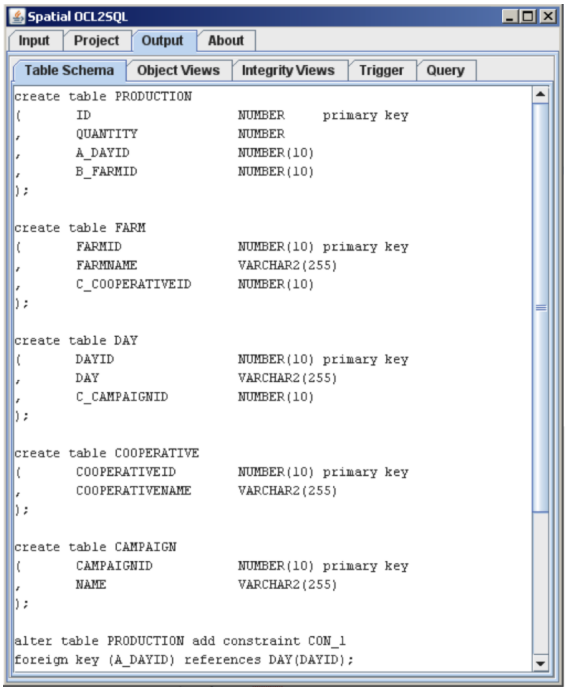


Figure 6. Deployment tier: Oracle snowflake physical schema generated from the conceptual schema of Figure 3.a

```
<!-- The Mondrian definition of the EDEN indicators -->
<Measure name="sum" column="QUANTITY" aggregator="sum" visible="false" formatString="Standard" />
- <CalculatedMember name="AVG-SUM" dimension="Measures" visible="true" formatString="#,###.##">
    <formula>Max(Descendants([COOPERATIVES].CurrentMember, [COOPERATIVES.NodesHierarchy].[FARM]),[Measures].
      [sum])</formula>
  </CalculatedMember>
</Cube>
```

Figure 7. Deployment tier: excerpt of the Mondrian XML metadata generated from UML profile of Figure 3.c

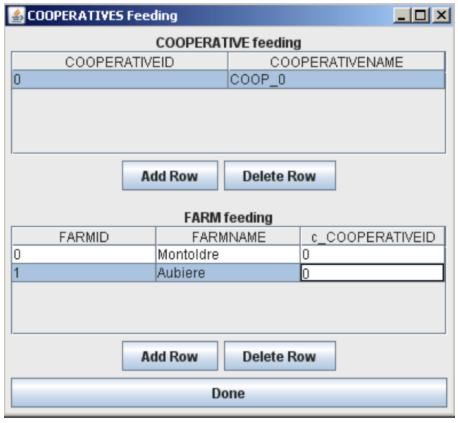
For example, starting from the conceptual model of Figure 3, ProtOLAP generates the Oracle SQL physical schema and the corresponding Mondrian-compliant XML code shown in Figures 6 and 7, respectively. Note that a snowflake schema has been chosen for implementation, as shown in Figures 5 and 6.

5.3 The Feeding Tier

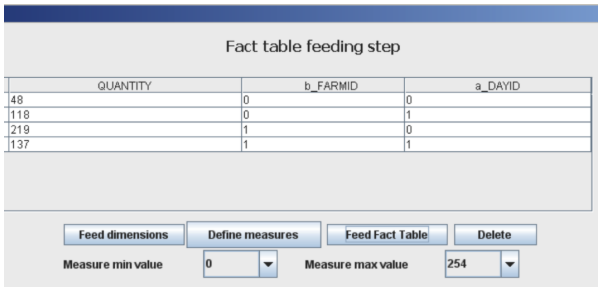
This tier offers a visual interface to feed both dimensions members and facts. Dimensions can be fed in two ways: manually, by letting decision makers insert the name of members; and automatically, by letting them automatically create a number of dummy members. Since dimensions' members are organized into hierarchies, the feeding tier in manual mode checks that decision makers correctly built the relationships between members. In automatic mode, hierarchical relations are automatically generated. Of course, while the automatic mode saves time, the manual mode is more effective because the data inserted are supposed to be part of the application domain and, as such, better understood by decision makers.

An example of the feeding process of the spatial dimension `Cooperatives` is shown in Figure 8.a. Remarkably, decision makers can manually input names of real farm and cooperatives without any knowledge of the multidimensional model. Aggregation levels and hierarchical relationships are intuitively represented by the user interface.

Once all dimensions are fed, users can set a range of possible values for measures; for example, for the `QUANTITY` measure she can choose value range [0..224]. The feeding tier automatically populates the fact tables with all possible combinations of dimension members and gives each fact a randomly-generated measure value included in the range (Figure 8.b). Then the feeding process is completed by automatically creating SQL insertion scripts to populate the physical schema.



(a)



(b)

Figure 8. Feeding tier: (a) manually fed dimensions; (b) automatically generated measures values

ProtOLAP also handles a simple form of versioning that enables decision makers to retrieve data previously inserted during another session. In this way they can incrementally check the

requirements for their DW focusing on different dimensions at different times.

5.4 The Analysis Tier

This tier is implemented using JRubik [16], a Java-based client developed on top of Mondrian and compatible with any RDBMS, such as Oracle. Using JRubik, with a few clicks decision makers can access and explore the data stored in the deployment layer.

Figure 9 presents an example of aggregated query using the base indicator of Figure 3.c. The pivot table allows aggregation rules to be more easily understood and validated. Indeed, by selecting data by farm and aggregating data on the temporal dimension using the OLAP client, decision makers can easily verify their sum aggregation value on the temporal dimension. In the same way, when they roll-up on the spatial dimension, the max aggregation function is applied by the OLAP server and they can verify/understand the modeled aggregation defined during conceptual design. As to aggregation rules, the OLAP client offers a tree visualization of dimensions members (Figure 9) which makes it easier for decision makers to validate the schemata of dimensions hierarchies.

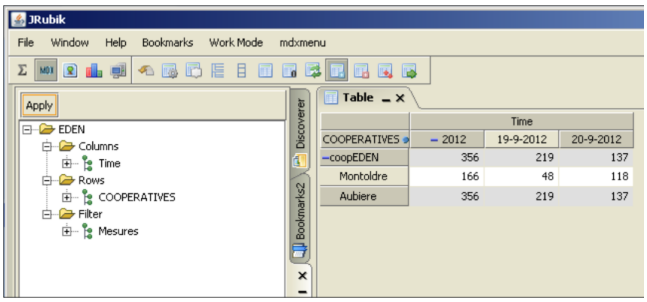


Figure 9. Analysis tier: exploring domain data with JRubik

6. CONCLUSION

In this paper we have proposed ProtOLAP, a fast prototyping methodology for DW projects that addresses project situations where source data are to be collected on-demand and users have little or no ICT skills. The methodology is requirement-based and does not require users to read and understand conceptual schemata; it is iterative and coupled with a set of tools that support, and in some cases automate, each single phase. In the context of the EDEN project, it has successfully been used to develop prototypes for 16 data marts.

To complete and improve our approach we are currently working in different directions: (i) provide project versioning features, aimed at letting designers effectively trace and manage the different iterations; (ii) pefine and carry out usability tests for the methodology and in particular for the feeding tier in the context of the EDEN project; and (iii) investigate how conceptual design can be achieved in a *by-query-example* fashion, like in [15], so as to decrease the probability of misunderstood requirements and make the overall process faster.

7. ACKNOWLEDGEMENTS

This work has been funded by the EDEN project of CASDAR and the Auvergne Region.

8. REFERENCES

[1] Akkaoui, Z., Zimányi, E., Mazón, J., Trujillo, J. 2011. A model-driven framework for ETL process development. In *Proc. DOLAP*, 45-52.

[2] Benker, T., Jürck, C. 2012. A Case Study on Model-Driven Data Warehouse Development. In *Proc. DaWaK*, 54-64.

[3] Bimonte, S., Boulil, K., Chanet, J-P., Pradel, M. 2012. Definition and Analysis of New Agricultural Farm Energetic Indicators Using Spatial OLAP. In *Proc. ICCSA* (2), 373-385.

[4] Boulil, K., Bimonte, S., Pinet, J.P. Spatial OLAP integrity constraints: from UML-based specification to automatic implementation: Application to energetic data in agriculture. *Journal of Decision Systems* (to appear).

[5] Giorgini, P., Rizzi, S., Garzetti, M. 2008. GRAnD: A goal-oriented approach to requirement analysis in data warehouses. *Decision Support Systems* 45(1), 4-21.

[6] Golfarelli, M., Rizzi, S., Turricchia, E. 2011. Modern Software Engineering Methodologies Meet Data Warehouse Design: 4WD. In *Proc. DaWaK*, 66-79.

[7] Golfarelli, M., Rizzi, S. 2011. Data warehouse testing: A prototype-based methodology. *Information and Software Technology* 53, 1183-1198.

[8] Huynh, N., Schiefer, J. 2001. Prototyping Data Warehouse Systems. In *Proc. DaWaK*, 195-207

[9] Mazón, J., Trujillo, J. 2008. An MDA approach for the development of data warehouses. *Decision Support Systems* 45(1), 41-58.

[10] No Magic 2013. MagicDraw. http://www.nomagic.com/products/magicdraw.html

[11] Oracle 2013. Oracle Spatial and Oracle Locator. http://www.oracle.com.

[12] Pentaho 2013. Pentaho Mondrian Project. http://mondrian.pentaho.com/.

[13] Pinet, F., Duboisset, M., Soulignac, V. 2007. Using UML and OCL to maintain the consistency of spatial data in environmental information systems. *Environmental Modelling and Software* 22(8), 1217-1220.

[14] Romero, O., Abelló, A. 2009. A Survey of Multidimensional Modeling Methodologies. *IJDWM* 5(2), 1-23.

[15] Romero, O., Abelló, A. 2010. Automatic validation of requirements to support multidimensional design. *Data Knowl. Eng.* 69(9), 917-942.

[16] Rubik 2013. Introduction to JRubik. http://rubik.sourceforge.net/jrubik/intro.html.

[17] Simitsis, A., Vassiliadis, P. 2008. A method for the mapping of conceptual designs to logical blueprints for ETL processes. *Decision Support Systems* 45(1), 22-40.

[18] Souza, V., Mazón, J., Garrigós, I., Trujillo, J., Mylopoulos, J. 2012. Monitoring strategic goals in data warehouses with awareness requirements. In *Proc. SAC 2012*, 1075-1082.