Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2010 Proceedings

Americas Conference on Information Systems (AMCIS)

8-2010

Computer-Aided Warehouse Engineering (CAWE): Leveraging MDA and ADM for the Development of Data Warehouses

Christian Kurze
Chemnitz University of Technology, christian.kurze@wirtschaft.tu-chemnitz.de

Peter Gluchowski
Chemnitz University of Technology, peter.gluchowski@wirtschaft.tu-chemnitz.de

Computer-Aided Warehouse Engineering (CAWE): Leveraging MDA and ADM for the Development of Data Warehouses

Christian Kurze, Peter Gluchowski
Chemnitz University of Technology, Thueringer Weg 7, D-09126 Chemnitz, Germany
{christian.kurze | peter.gluchowski}@wirtschaft.tu-chemnitz.de

ABSTRACT

During the last decade, data warehousing has reached a high maturity and is a well-accepted technology in decision support systems. Nevertheless, development and maintenance are still tedious tasks since the systems grow over time and complex architectures have been established. The paper at hand adopts the concepts of Model Driven Architecture (MDA) and Architecture Driven Modernization (ADM) taken from the software engineering discipline to the data warehousing discipline. We show the works already available, outline further research directions and give hints for implementation of Computer-Aided Warehouse Engineering systems.

Keywords

Data Warehouse, Business Intelligence, Model Driven Architecture, Architecture Driven Modernization.

INTRODUCTION

Data Warehouses provide crucial business information to improve strategic decisions and therefore play an important role in current decision support systems (Inmon, 2005). During the last decade, data warehouse industry has reached full maturity and acceptance in the business world. Furthermore, technical improvements opened the possibility to replace storage spaces of gigabytes into terabytes. Privacy, security, and compliance requirements require more efforts in development, documentation, and metadata management. There is also a shift that puts initiative into the hands of business users, not IT (Kimball, Ross, Thornthwaite, Mundy and Becker, 2008). This is what makes the point for model driven data warehouse development: the data warehouse has to provide data in a business user-understandable way. The purpose of a detailed system specification becomes twofold: First, it should sum up the requirements towards the system and therefore provide documentation of the system (for end users and IT staff). Second, it provides a resilient basis for implementing the targeted system. By following an MDA approach, the system specification can be transformed into the real system.

Forward-, Reverse- and Re-Engineering

Within the context of software engineering, the terms *Forward-*, *Reverse-* and *Re-Engineering* are well discussed (Chikofsky and Cross, 1990). Furthermore, their application has been transferred to data engineering problems by Aiken (1996). We are adopting them to data warehousing and thus give a short summary about their meaning and transfer this meaning to the development of data warehouses in a later section.

All three terms describe different directions of the development process: *Forward-Engineering* is a process that uses requirements artifacts to derive design artifacts and actual source code. It is the traditional development process of new systems. The level of abstraction decreases during the process. *Reverse-Engineering* deals with activities that derive information from previously designed artifacts (e.g. design descriptions or code). The process aims at deriving design descriptions from implementation artifacts and requirements artifacts form design artifacts. The level of abstraction raises during the development process. *Re-Engineering* covers examination and adoption of a system in order to use this system in a derived form. Re-Engineering results from the combination of Reverse-Engineering and Forward-Engineering; it requires a Reverse-Engineering which results are input for Forward-Engineering.

Applied to data warehousing one has to distinguish different layers: operational systems, ETL processes, multidimensional data storage, and applications (Chaudhuri and Dayal, 1997). Each step has to be taken in order to get a fully fledged data warehouse system. In order to support a triple-driven design of data warehouses as a combination of data provided by

operational systems, user requirements against the data warehouse as well as goals defined independent of available data and user requirements (Guo, Tang, Tong and Yang, 2006), we want to define forward- and reverse-engineering for data warehouses as a whole spreading over the already given layers:

- *Forward-engineering of data warehouses* requires a reverse-engineering of operational systems in order to support data-driven requirements engineering as well as forward-engineering of multidimensional data storage, applications and ETL processes.
- *Reverse-Engineering of data warehouses* is also based on a reverse-engineering of operational systems as a requirement for reverse-engineering of ETL processes. Furthermore, it consists of reverse-engineering of multidimensional data storage and applications.
- *Re-Engineering of data warehouses* as a combination of forward- and reverse-engineering is based on a reverse-engineering of operational systems, and a re-engineering of ETL processes, multidimensional data storage, and applications.

These three process directions are the basis of the following ideas presented in the paper at hand.

Model Driven Architecture (MDA) and Architecture Driven Modernization (ADM)

The Model Driven Architecture (MDA) is an approach that uses models in software engineering (Object Management Group, 2003). It starts by the idea to separate the specification of a system from the way the system uses the capabilities of its platform. Therefore, the Computing Independent Model (CIM) provides a view independent of the future implementation. It is sometimes referred to as a domain model and uses vocabulary familiar to the practitioners of the domain in question. The CIM forms the basis for the Platform Independent Model (PIM). It can be used with different platforms of a similar type. The Platform Specific Model (PSM) combines the PIM with additional details on how the system uses a particular platform. Model transformations play an important role in MDA: they convert models of the same system into each other. Figure 1 summarizes the basic principles of MDA.

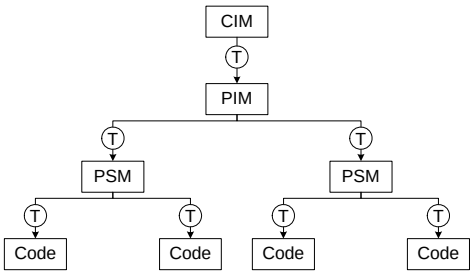

Figure 1. Basic principle of MDA (Mazón and Trujillo, 2008a)

A more recent approach covered by the MDA is Architecture Driven Modernization (ADM). It addresses system modernization and encompasses the business as well as the IT domain. The business domain contains models and correspondent diagrammatic views of the organization, business semantics, business rules, and business processes. The IT domain comprises traditional data, application and technical architectures (Khusidman and Ulrich, 2007). The aim is to provide models for reengineering of systems which forms a transformational path consisting of three steps: first, knowledge discovery from existing systems; second, target architecture specification which defines the framework the existing solution should be mapped or transformed into; and third, transformative steps that move the as-is state into the to-be state (Khusidman and Ulrich, 2007; Khusidman, 2008). The most critical issue is the so-called semantic gap between a diagrammatic representation (which is semantically rich) and its logical representation (Pardillo, Mazón and Trujillo, 2008).

MDA and ADM for the Development of Data Warehouses

Combining the ideal layers of data warehousing and the different viewpoints of MDA, Mazón and Trujillo (2008a) presented a framework for the development of data warehouses. They outline the main advantages of applying MDA to the development of data warehouses:

- Definition of a systematic, well-structured and standard way for the development of the different data warehousing layers by using the MDA approach
- Productivity increase by automatically generating (parts of) the DW system from the PIM

- Better fit of the system to the requirements, since developers focus more on conceptual models of each DW layer than on implementation details
- Portability increase, because the same PIM can be automatically transformed into different PSMs
- Integrated modeling framework for managing the metadata of the heterogeneous components of a data warehouse system
- Reuse of best practices in each implementation since they are integrated into the transformations
- Adaptability increase, since arising new technologies for one of the DW components can be integrated by only changing a few transformations
- Support for system evolution is given by changing the CIM and the PIM. Transformations can then be updated to transform the changes into a PSM and Code.

The framework of Mazón and Trujillo (2008a) focuses on data sources, ETL processes, multidimensional data modeling for the core data warehouse, modelling data cubes, and application modelling like OLAP and data mining. In difference to Mazón and Trujillo (2008a), we summarized their modelling of the relational data warehouse and data cubes into one multidimensional data layer. This is because of the difference of ROLAP and MOLAP systems; each requiring different descriptions (Chaudhuri and Dayal, 1997). However, each data warehouse layer is described from all three viewpoints (CIM, PIM, PSM) as well as an additional transformation into code. The CIM layer is the main input for multidimensional data PIMs and application PIMs as shown in Figure 2.

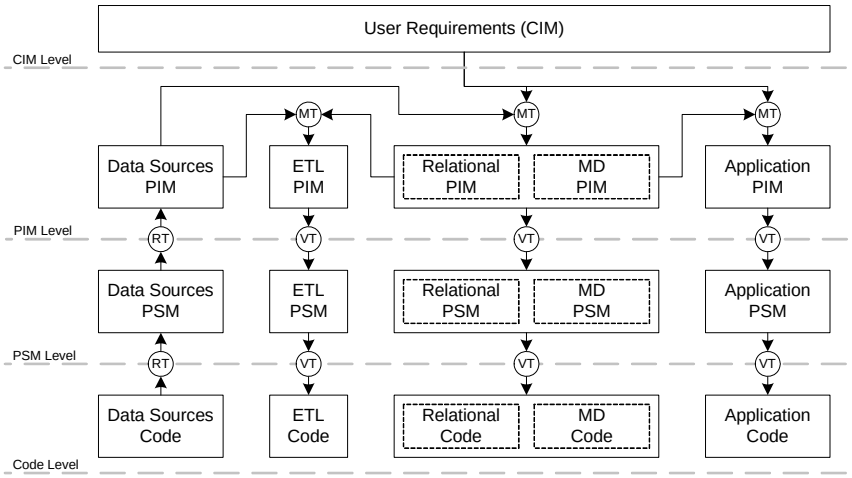

Figure 2. MDA framework for the development of data warehouses based on (Mazón and Trujillo, 2008a)

Several transformations are necessary within this framework. In order to be used in model transformations, source systems have to be modeled first. In most cases, these models do not exist and have to be developed from data sources code, i.e. the operational database system. These transformations are depicted as Reverse Transformation (RT) in Figure 2. Transformations depicted as MT are merging transformations since they have multiple input models and one single output model. Merging different models is, for example, necessary to generate ETL PIMs: source system PIMs as well as multidimensional data PIMs are required. Vertical transformations (VT) are intended to transform models from one MDA layer into a more technical one.

The presented framework is the basis for our further developments. Based on related work we are going to extend the framework and explicitly show how to use it in forward-, reverse-, and re-engineering scenarios.

RELATED WORK

Within this section we outline related work that fits into the framework presented above. We order the different resources according to the layers of data warehousing as well as MDA viewpoints. Complementary material is presented subsequently. The list is not intended to be exhaustive but to present the most important works to consider in MDA based data warehouse development.

Requirements Management

Requirements management is a tedious task independent of the target system's domain. Within the data warehouse discipline, there should be an integrated triangulation of requirements (Guo et al., 2006): *Data-driven* approaches start with an analysis of transactional systems and therefore take the data given into account. *Goal-driven* approaches focus on the need to align the contents of the data warehouse with corporate strategy and business objectives. *User-driven* approaches integrate the end-user into the design process. All three approaches should be integrated, in order to gain reasonable and accepted results. A recent summary on different methodologies for requirements elicitation is given by Romero and Abelló (2009).

Multidimensional Data Modeling

Many different approaches exist to model multidimensional data, but no graphical notation has gained broad acceptance among practitioners. One problem can be seen in the absence of a commonly accepted and mathematically proven model like the relational model. There should be a clearly defined model which is capable of modeling complex requirements, e.g. Pedersen et al. (2001), as well as a multidimensional algebra that defines the operations available (Romero and Abelló, 2007). Another issue is the size of real-world systems. Therefore, there have to be mechanisms and tools which are able to cope with very large models (Kurze and Gluchowski, 2009). An approach to automatically derive multidimensional models form operational systems based on ADM is provided by Mazón and Trujillo (Mazón and Trujillo, 2008b).

ETL processes

The development of ETL processes takes most of the developer's time. Due to the complexity of possible extraction and transformation peculiarities, it is difficult to completely automate the generation of ETL processes. It might help to use an automated data-driven requirements engineering which derives multidimensional data from operational data in an algebraic way. These algebraic steps can be transformed into ETL models and, furthermore, into actual implementations.

Some graphical notations have been proposed, for example (Vassiliadis, Simitsis and Skiadopoulos, 2002; Trujillo and Luján-Mora, 2003; Muñoz, Mazón, Pardillo and Trujillo, 2009; Simitsis, Skoutas and Castellanos, 2010). There are also works which concentrate on the automated transformation into implementations, for example (Simitsis, 2005; Muñoz, Mazón and Trujillo, 2009).

Application Modeling

There is a plethora of applications in data warehousing. We would like to show that there are models for complex tasks like data mining. The modeling is currently supported by UML profiles, for example association rule mining (Zubcoff and Trujillo, 2007) and time-series analysis (Zubcoff, Pardillo and Trujillo, 2009). Furthermore, what-if-analyses are also supported by an UML profile (Golfarelli and Rizzi, 2008).

Tool Support

In order to use the MDA approach for data warehouses, there has to be sufficient tool support which allows metamodelling, domain specific languages, and model transformations. A reasonable conceptual architecture for metamodelling has been proposed by Karagiannis and Kühn (2002). This architecture is very promising since it allows an extension with ontological aspects in order to better support semantic issues in data warehousing (Živković, Kühn and Murzek, 2009).

A great deal of work has been done to create meta-CASE toolsets for developing modeling environments for domain specific visual languages. An example of such work has been presented by Zhu, Grundy, Hosking, Liu, Shuping and Mehra (2007). They summarize plenty of related work and created a toolset which simplifies the development of modeling environments. All models are persistently saved in an XML format so that they can also be stored in a database. Other important works include, for example, MetaEdit+ (Steven, Kalle and Matti, 1996), Meta-MOOSE (Ferguson, Parrington, Dunne, Hardy, Archibald and Thompson, 2000), GMF (Eclipse, 2009), and DSL Tools (Cook, Jones, Kent and Wills, 2007). Those frameworks often require considerable effort to be understood and to be used effectively.

ENRICHMENT OF MAZÓN AND TRUJILLO'S FRAMEWORK

We examine the framework in detail and extend/modify it if necessary. Concerning the requirements, three different approaches can be identified and have to be integrated: goal-driven, data-driven, and user-driven approaches (Guo et al., 2006). Therefore, we explicitly show these three approaches in the adopted framework, as depicted in Figure 3. The *arrow from data sources PIM towards user requirements* represents the data-driven approach, *end-users* and *corporate strategy* the user-driven and goal-driven approach, respectively. The requirements rectangle has been moved above the application layer

since requirements are expressed according to these applications. A user does not express requirements against ETL processes or multidimensional data storage. They express their requirements against the applications they can use to analyze data.

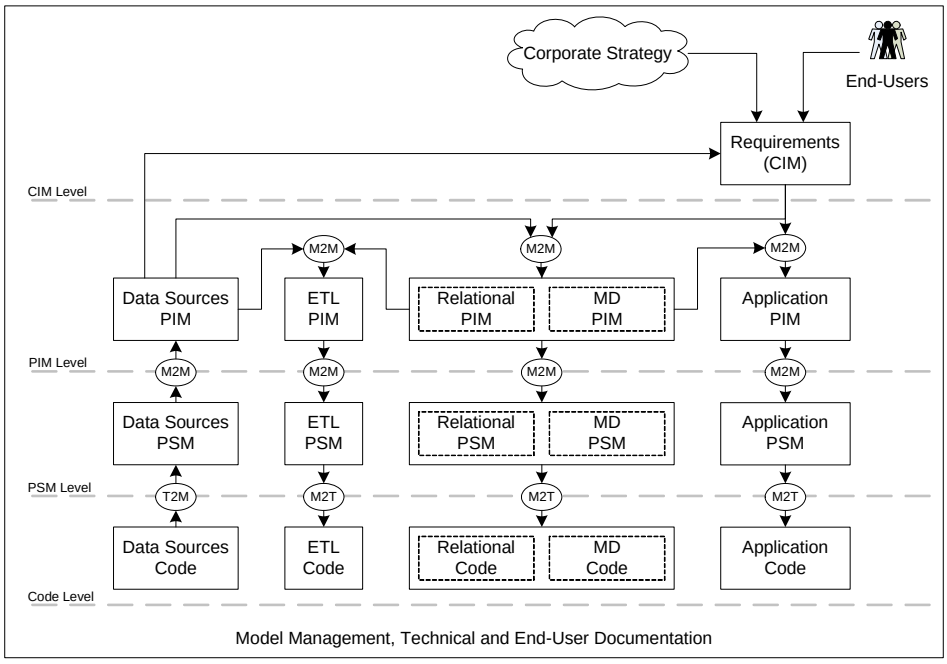

Figure 3. MDA framework for forward-engineering of data warehouses

Furthermore, the whole framework is embedded into a *model management* (Bernstein, 2003; Bernstein and Rahm, 2000) box. This box is very important since it covers the framework as a whole and offers the technical infrastructure for the MDA approach for data warehouses. It allows the interconnection of different models, e.g. via trace links (Czarnecki and Helsen, 2006). Additionally, it allows – based on models, trace links, and additional metadata annotated to model elements – an automated generation of technical and end-user documentation. If generated automatically from the models as a basis for the whole system, documentation is always up-to-date (Heinrich, Boehm-Peters and Knechtel, 2008). Documentation can be made available, for example, via web applications or wikis.

We further analyzed the necessary model transformations. According to Czarnecki and Helsen (2006), there are three types of transformations: model-to-model (m2m), model-to-text (m2t), and text-to-model (t2m) transformations. Model-to-model transformations take models as both input and output of the transformation. Model-to-text transformations take models as input and transform these models into text, whereas text-to-model transformations take given texts and reconstruct the models behind. In most cases, text-to-model transformations need sophisticated reverse-engineering technologies in order to construct appropriate models. Adopted to the MDA framework for data warehouses, it is not as important to distinguish between merging or vertical transformations as Mazón and Trujillo did than to distinguish between m2m, m2t, or t2m transformations. This characterization has impacts on the technology used to define and execute the particular transformation. The elaboration is depicted in Figure 3 which concentrates on the forward-engineering of data warehouse solutions.

Concerning the reverse-engineering of data warehouses, the same framework applies but is used in a slightly different way. The main aim is to automatically create requirement artifacts from an already existing system facilitating an easy understanding of the current system as a basis for implementing new requirements. As outlined above, most of the transformations are used in the opposite direction than in forward-engineering. One exception are the operational systems, they are still reverse-engineered. The relationship between *Data Sources PIM* and *Requirements* is not an input relationship any more, but a way to prune the reverse engineered artifacts by looking up the operational pendants for multidimensional functionality. Furthermore, it is not possible to derive a *Corporate Strategy* (goal-driven requirements) nor the *End-User* requirements (user-driven requirements) from the reverse engineered artifacts. But, same as the operational systems' models, they are used to prune the engineered artifacts and to derive potential new (currently missing) requirements. Thus, the connections shown in Figure 4 are double-arrowed to show the bi-directional usage in reverse-engineering of data warehouse systems.

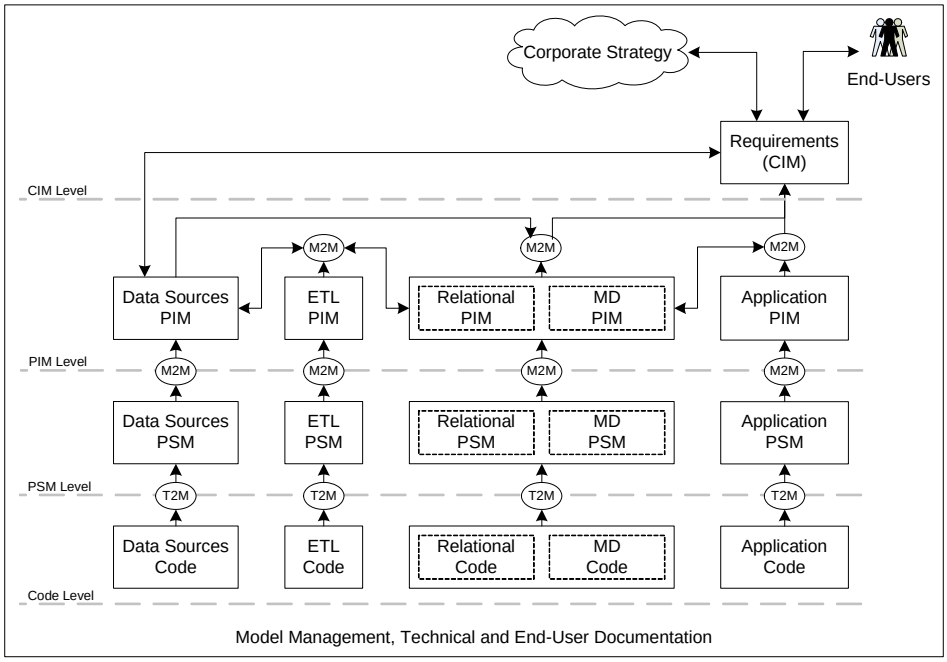

Figure 4. MDA framework for reverse-engineering of data warehouses

Re-Engineering of data warehouse systems requires a reverse-engineering according to Figure 4. The results are the input for a forward-engineering as shown in Figure 3. In order to define suitable requirements within the transformation phase between reverse- and forward-engineering a proper triple-driven requirements engineering has to be done to get consistent and solid requirements. Especially the integration with the already implemented requirements is of particular interest. A first approach is presented by Blaschka (2000) who introduced several operators for the evolution of data warehouses.

IMPLEMENTATION CLUES

Most objectives against CAWE are related to metamodelling. Therefore, we propose an architecture based on the work of Karagiannis and Kühn (2002) which is shown in Figure 5. The meta²-model provides the core concepts to create metamodels and mechanisms and is therefore aligned in the centre of the architecture and connected with all other components. Models and metamodels are stored in the model repository and the metamodel repository, respectively. Both repositories are interconnected in order to delegate changes in metamodels to the corresponding models, i.e. to keep them in sync. The functionalities applicable to models and metamodels are stored in the mechanism repository. They can be either stored directly in the mechanism repository or externally. In the latter case, the mechanism repository only provides a reference to the particular mechanism as well as an interface description how to interact with the mechanism. Each repository fulfils typical repository-specific tasks as described by Bernstein and Dayal (1994). The component *persistency services* is separated from the single repositories: it manages the persistent storage of all models, metamodels, and mechanisms and offers transparency independent of the particular physical storage method (e.g. database or file system). Additionally, it enables the distribution of parts of models and metamodels. Access to the different repositories is provided by *access services* via API or file-based interaction. The *viewer and builder components* on top of the architecture support use and maintenance of the whole platform.

The metamodelling platform architecture covers the required metamodelling aspects. Model transformations are stored in the mechanism base and applied appropriately to models and metamodels. ADM specifics are also supported, since model discovery, i.e. text-to-model transformations, can be seen as mechanisms. More complex model discovery tasks may be stored externally (e.g. external applications). A reference to call an application as well as the required input and output (meta) models are stored within the mechanism repository. Transformation from PSM to PIM is also stored in the mechanism repository. If more complex algorithms or rules are needed, external applications can be called analogous to model discovery.

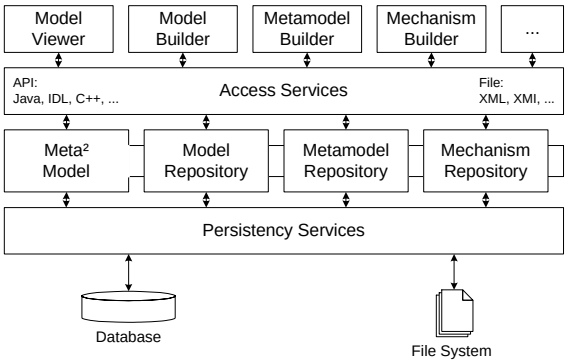

Figure 5. Metamodelling platform architecture (Karagiannis and Kühn, 2002)

We have chosen the Eclipse Modeling Project (Gronback, 2009) as a basis for implementing the architecture since it is an open source project with a large community of committers developing the whole platform. The *meta²-model* is represented by the Ecore metamodel which is part of the Eclipse Modeling Framework (EMF) (Steinberg, Budinsky, Paternostro and Merks, 2009). The *model* and *metamodel repository* are currently file based. All (meta-) models in the EMF can be exported in XMI. *Persistency services* are offered for XML or binary storage. Additionally, there are implementations for database storage, e.g. Connected Data Objects (CDO) or Teneo. (Meta-) model access is transparent independent of the actual storage mechanism. The *mechanism repository* is mainly to store model transformations implemented in the Atlas Transformation Language (ATL). More complex algorithms for model discovery are written in Java. All models and model transformations can be packaged into Eclipse plug-ins. Therefore, the *access services* via file or API are transparent, too. The *model and metamodel viewer and builder* on top of the architecture are also covered within EMF. An integrated tree based editor as well as a graphical editor for metamodels are provided. Furthermore, tree-based and graphical model editors are automatically generated based on metamodels, as shown in Figure 6.

Our current implementation focuses on forward- and reverse-engineering of multidimensional data storage. Our system is capable of transforming requirements expressed in ADAPT (Bulos and Forsman, 2006) into relational snowflake and star schemata as well as DDL scripts for Microsoft SQL Server. Additionally, we provide multidimensional metadata for Microsoft Analysis Services. Figure 6 shows a screenshot of a simple cube modeled in ADAPT as well as three dimensions and two measures. They are transformed into a relational snowflake schema and a model according to the Analysis Services Scripting Language. Furthermore, the reverse-engineering of ideal snowflake schemata and Analysis Services descriptions is implemented. We also export the multidimensional information enriched by further metadata as an initial MediaWiki.

CONCLUSION AND FUTUTRE WORK

As the paper at hand shows, a suitable framework for MDA-based data warehouse development already exists, but it is missing stable toolsets that support complex tasks of data warehouse development for real-world data warehouses. The related work outlines that most parts of the framework are already supported by research prototypes. Nevertheless, there is no tool that supports all aspects of model-driven data warehouse development. The current paper takes steps to close this gap by extending the framework with extended requirements engineering approaches, explicit underpinning with technical architectures, and a clear definition of forward-, reverse-, and re-engineering.

Our experiences gained in two industry projects (within the telecommunications as well as the tourism industry) clearly show that there is a strong need for further research on a technical but also on a business level. Technically, there are the following requirements: There is a strong need for a model repository which is capable of collaborative modeling scenarios, i.e. it has to offer typical database features like locking on model elements, transactions, and version management of models. Furthermore, model traces are very important. They contain information about the interconnection of models. It is important in the forward-engineering scenario since they show which requirement is implemented in which physical part of the system, but also for reverse-engineering to prune the results. Metamodels are needed for a broad range of systems. Standards like XMLA or MDX have to be further examined and applied. The example of the CWM shows that finding a common standard for all aspects of data warehousing is a very difficult task. From a business point of view there has to be a well defined procedure model for model-driven data warehouse development as well as further research on organizational change management that discusses the adoption of a model-driven development approach.

The main benefits we have identified are a consistent modeling of the system from requirements to implementation, the automated derivation of business and technical documentation, and a graphical representation of multidimensional data models eases the understanding and acceptance by business users. Especially the documentation aspects are of strong interest for our industry partners. Concerning large modeling scenarios, there is still a strong research need. The same applies to reverse- and re-engineering, since smallest mistakes in modeling results in errors in data warehouse systems which are critical systems in decision support.

Our next steps are a full support of forward-engineering covering the aspects of operational systems, ETL processes, and application models. This step also includes an extended model-based triple-driven requirements management. Furthermore, a common model supporting all multidimensional peculiarities that is mathematically proven is on our agenda. From our point of view, a comprehensive model-driven development is possible with a core model similar to the relational model only. In parallel we are preparing an empirical study that tests the impact on data warehouse development by our research prototype. On the one hand we will use a case study approach with our industry partners, on the other hand laboratory experiments with students of our department.

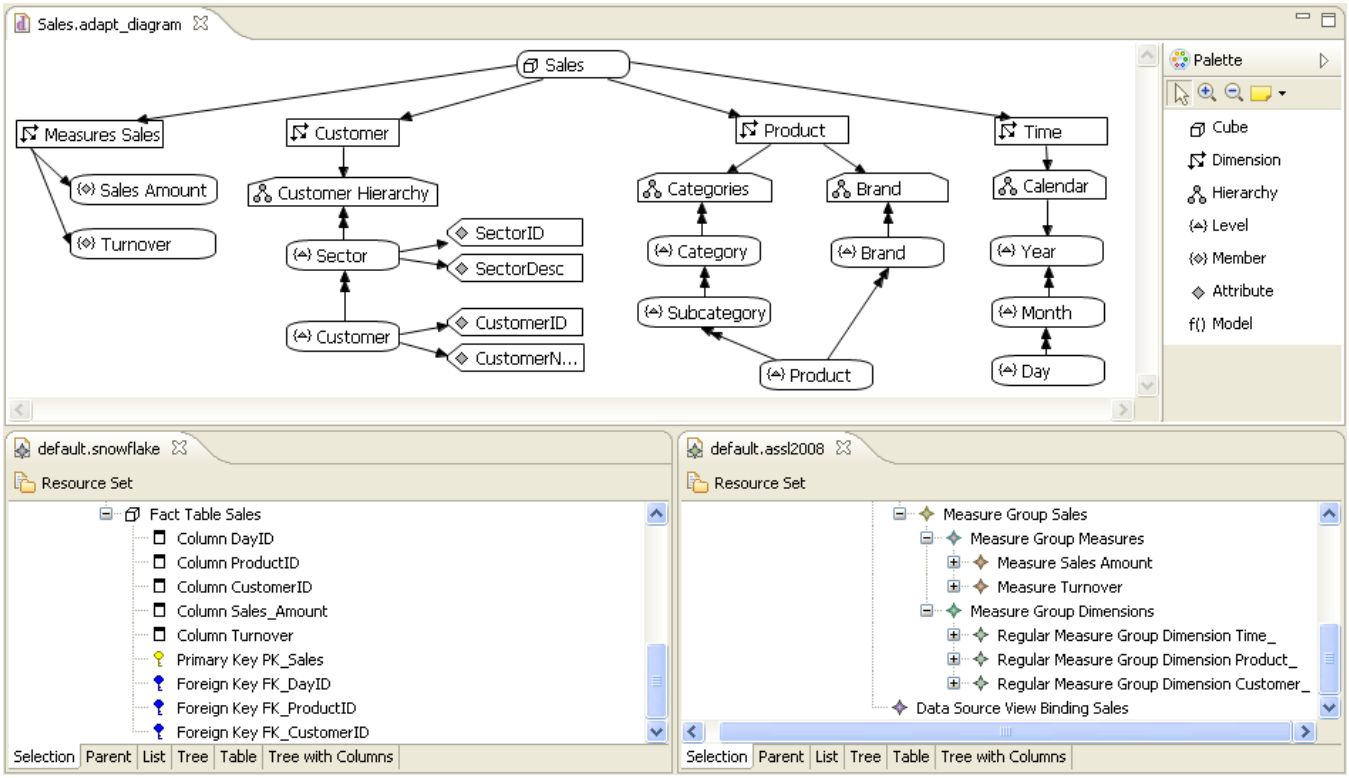

Figure 6. Screenshot of CAWE prototype

REFERENCES

1. Aiken, P. (1996) Data Reverse Engineering: Slaying the Legacy Dragon, McGraw-Hill, New York.
2. Bernstein, P. (2003) Applying model management to classical meta data problems, in *Proceedings of the First Biennial Conference on Innovative Database Research (CIDR 2003)*, January 5-8, Asilomar, CA, USA, ACM, 209-220.
3. Bernstein, P. and Rahm, E. (2000) Data Warehouse Scenarios for Model Management. In *Conceptual Modeling – ER 2000, LNCS Vol. 1920*, Springer, Berlin, Heidelberg, 57-109.
4. Bernstein, P. and Dayal, U. (1994) An Overview of Repository Technology, in *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB 1994)*, September 12-15, San Francisco, CA, USA, Morgan Kaufmann, 705-713.
5. Blaschka, M. (2000) FIESTA: A Framework for Schema Evolution in Multidimensional Databases, Doctoral Thesis, Technische Universität München (TUM), http://deposit.ddb.de/cgi-bin/dokserv?idn=962067342 [02/29/2010].

6. Bulos, D. and Forsman, S. (2006) Getting Started with ADAPT, Whitepaper, http://symcorp.com/downloads/ADAPT_white_paper.pdf [02/29/2010].
7. Chaudhuri, S. and Dayal, U. (1997) An overview of data warehousing and OLAP technology, *ACM SIGMOD Record*, 26, 1, 65-74.
8. Chikofsky, E. and Cross, J. (1990) Reverse engineering and design recovery: a taxonomy, *Software, IEEE*, 7, 1, 13-17.
9. Czarnecki, K. and Helsen, S. (2006) Feature-based survey of model transformation approaches, *IBM Systems Journal*, 45, 3, 621-645.
10. Golfarelli, M. and Rizzi, S. (2008) UML-Based Modeling for What-If Analysis, in *Data Warehousing and Knowledge Discovery, LNCS Vol. 5182*, Springer, Berlin, Heidelberg, 1-12.
11. Gronback, R. C. (2009) Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit, Addison-Wesley Longman, Amsterdam.
12. Guo, Y., Tang, S., Tong, Y. and Yang, D. (2006) Triple-driven data modeling methodology in data warehousing: a case study, in *Proceedings of the 9th ACM international workshop on Data warehousing and OLAP (DOLAP 2006)*, November 10, Arlington, VA, USA, ACM, 59-66.
13. Heinrich, M., Boehm-Peters, A. and Knechtel, M. (2008) MoDDo - a tailored documentation system for model-driven software development, in *Proceedings of the IADIS International Conference WWW/Internet (ICWI 08)*, October 13-15, Freiburg, Germany, IADIS, 321-324.
14. Inmon, W. H. (2005) Building the Data Warehouse, Wiley, Indianapolis.
15. Karagiannis, D. and Kühn, H. (2002) Metamodelling Platforms. In *E-Commerce and Web Technologies, LNCS Vol. 2455*, Springer, Berlin, Heidelberg, 451-464.
16. Khusidman, V. (2008) ADM Transformation, Whitepaper, http://www.omg.org/cgi-bin/doc?admtf/2008-06-10 [02/29/2010].
17. Khusidman, V. and Ulrich, W. (2007) Architecture-Driven Modernization: Transforming the Enterprise, Whitepaper, http://www.omg.org/cgi-bin/doc?admtf/2007-12-01 [02/29/2010].
18. Kimball, R., Ross, M., Thornthwaite, W., Mundy, J. and Becker, B. (2008) The Data Warehouse Lifecycle Toolkit, Wiley, Indianapolis.
19. Kurze, C. and Gluchowski, P. (2009) Towards Principles for Managing and Structuring Very Large Semantic Multidimensional Data Models, in *Proceedings of the 15th Americas Conference on Information Systems (AMCIS 2009)*, August 6-9, San Francisco, CA, USA, AIS, paper 315.
20. Mazón, J. and Trujillo, J. (2008a) An MDA approach for the development of data warehouses, *Decision Support Systems*, 45, 1, 41-58.
21. Mazón, J. and Trujillo, J. (2008b) A Model Driven Modernization Approach for Automatically Deriving Multidimensional Models in Data Warehouses, in *Conceptual Modeling – ER 2007, LNCS Vol. 4801*, Springer, Berlin, Heidelberg, 56-71.
22. Muñoz, L., Mazón, J., Pardillo, J. and Trujillo, J. (2009) Modelling ETL Processes of Data Warehouses with UML Activity Diagrams, in *On the Move to Meaningful Internet Systems: OTM 2008 Workshops, LNCS Vol. 5333*, Springer, Berlin, Heidelberg, 44-53.
23. Muñoz, L., Mazón, J. and Trujillo, J. (2009) Automatic generation of ETL processes from conceptual models, in *Proceeding of the ACM twelfth international workshop on Data warehousing and OLAP (DOLAP 2009)*, November 6, Hong Kong, China, ACM, 33-40.
24. Object Management Group. (2003) MDA Guide V1.0.1, Whitepaper, http://www.omg.org/cgi-bin/doc?omg/03-06-01 [02/29/2010].
25. Pardillo, J., Mazón, J. and Trujillo, J. (2008) Bridging the semantic gap in OLAP models: platform-independent queries, in *Proceedings of the ACM 11th international workshop on Data warehousing and OLAP (DOLAP 2008)*, October 30, Napa Valley, CA, USA, ACM, 89-96.
26. Pedersen, T. B., Jensen, C. S. and Dyreson, C. E. (2001) A foundation for capturing and querying complex multidimensional data, *Information Systems*, 26, 5, 383-423.
27. Romero, O. and Abelló, A. (2009) A Survey of Multidimensional Modeling Methodologies, *International Journal of Data Warehousing and Mining*, 5, 2, 1-23.

28. Romero, O. and Abelló, A. (2007) On the Need of a Reference Algebra for OLAP, in *Data Warehousing and Knowledge Discovery, LNCS Vol. 4654*, Springer, Berlin, Heidelberg, 99-110.
29. Simitsis, A. (2005) Mapping conceptual to logical models for ETL processes, in *Proceedings of the 8th ACM international workshop on Data warehousing and OLAP (DOLAP 2005)*, November 4-5, Bremen, Germany, ACM, 67-76.
30. Simitsis, A., Skoutas, D. and Castellanos, M. (2010) Representation of conceptual ETL designs in natural language using Semantic Web technology, *Data & Knowledge Engineering*, 69, 1, 96-115.
31. Steinberg, D., Budinsky, F., Paternostro, M. and Merks, E. (2009) EMF: Eclipse Modeling Framework, Addison-Wesley Longman, Amsterdam.
32. Trujillo, J. and Luján-Mora, S. (2003) A UML Based Approach for Modeling ETL Processes in Data Warehouses, in *Conceptual Modeling – ER 2003, LNCS Vol. 2813*, Springer, Berlin, Heidelberg, 307-320.
33. Vassiliadis, P., Simitsis, A. and Skiadopoulos, S. (2002) Conceptual modeling for ETL processes, in *Proceedings of the 5th ACM international workshop on Data Warehousing and OLAP (DOLAP 2002)*, November 8, McLean, VA, USA, ACM, 14-21.
34. Živković, S., Kühn, H. and Murzek, M. (2009) An Architecture of Ontology-aware Metamodelling Platforms for Advanced Enterprise Repositories, in *Proceedings of the Joint Workshop on Advanced Technologies and Techniques for Enterprise Information Systems*, May 6-7, Milan, Italy, INSTICC Press, 95-104.
35. Zubcoff, J., Pardillo, J. and Trujillo, J. (2009). A UML profile for the conceptual modelling of data-mining with time-series in data warehouses, *Information and Software Technology*, 51, 6, 977-992.
36. Zubcoff, J. and Trujillo, J. (2007) A UML 2.0 profile to design Association Rule mining models in the multidimensional conceptual modeling of data warehouses, *Data & Knowledge Engineering*, 63, 1, 44-62.