Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2005 Proceedings

Americas Conference on Information Systems (AMCIS)

2005

Using Reference Models for Data Warehouse Metadata Management

Florian Melchert
University of St. Gallen, florian.melchert@unisg.ch

Alexander Schwinn
University of St. Gallen, alexander.schwinn@unisg.ch

Clemens Herrmann
University of St. Gallen, clemens.herrmann@unisg.ch

Robert Winter
University of St. Gallen, robert.winter@unisg.ch

Using Reference Models for Data Warehouse Metadata Management

Florian Melchert
Institute of Information Management
University of St. Gallen
Mueller-Friedberg-Str. 8,
9000 St. Gallen, Switzerland
Florian.Melchert@unisg.ch

Alexander Schwinn
Institute of Information Management
University of St. Gallen
Mueller-Friedberg-Str. 8,
9000 St. Gallen, Switzerland
Alexander.Schwinn@unisg.ch

Clemens Herrmann
Institute of Information Management
University of St. Gallen
Mueller-Friedberg-Str. 8,
9000 St. Gallen, Switzerland
Clemens.Herrmann@unisg.ch

Robert Winter
Institute of Information Management
University of St. Gallen
Mueller-Friedberg-Str. 8,
9000 St. Gallen, Switzerland
Robert.Winter@unisg.ch

ABSTRACT

Based on experience from applying the Common Warehouse Metamodel to metadata management at a large Swiss bank, the applicability of metadata reference models in complex application domains is analyzed. In order to explain the context of the case, the state-of-the-art of the Common Warehouse Metamodel and its application for data warehouse management are summarized. Based on the project experience documented in this paper, benefits of the reference model approach are described and recommendations for future developments of the Common Warehouse Metamodel are proposed.

Keywords

Data Warehousing, Metadata, Metadata Management, Common Warehouse Metamodel, Reference Model.

INTRODUCTION

Like many other companies particularly in services industries, a large Swiss bank initiated a data warehouse program in the late 1990ies in order to create an integrated foundation for consistent decision support. In the course of the program, a large number of departmental management support systems were replaced by analytical information systems based on the centralized data warehouse (DWH) system (Meyer, 2000). While applications of the DWH were growing in scope and size across the bank, DWH program management became confident that additional value could be created by a centralized metadata repository (MDR). Although a 'classical', centralized DWH architecture was being followed, DWH metadata were distributed across several organizational units at that time.

As a prerequisite for a centralized MDR, an integrated conceptual metadata model for technical as well as application-oriented metadata had to be created. By using Object Management Group's (OMG) Common Warehouse Metamodel (CWM) as a reference model for the MDR, DWH management hoped to minimize modeling costs and maximize modeling quality as well as stakeholder acceptance at the same time.

This article reports the experience made with the application of the CWM standard within the bank's metadata management project. As there are hardly any reports on CWM application projects, it provides new insights into the process of implementing a metadata management solution based on CWM. By examining a specific part of the CWM in detail, the limitations of the standardized metamodel are depicted. It is shown how the bank's metadata team used extensions to meet their metadata requirements. Finally, the article describes the results of a CWM assessment workshop that was conducted with several companies to identify major areas to which CWM coverage should be extended.

After a brief discussion of related work in section 2, the most important features of CWM are summarized in section 3. The evaluation of the CWM application in the banking case is described in section 4. More general recommendations for future research regarding metamodel extensions are given in section 5. Section 6 concludes with a short summary.

RELATED WORK

While the standardization of metadata is discussed in numerous domains resulting in a plethora of metadata standards, the specific requirements of data warehousing solutions are usually addressed insufficiently (Staudt, Vaduva, & Vetterli, 1999). Only a few authors, like e.g. (Marco & Jennings, 2004) propose metadata models that are comprehensive, address DWH-specific requirements and can therefore serve as sound basis for developing company-specific DWH metadata solutions. However, these types of models suffer from not being established as an official standard as they have been developed by a single party. In addition, there are only few recommendations or mechanisms for the adaptation or expansion of such metamodels to project-specific requirements.

Concerning standardization, the CWM represents one of the most suitable approaches in the DWH domain. It has been constructed as a metadata standard for this very application domain and it is already supported by a large number of DWH software vendors. Although the original specification of the CWM standard is very comprehensive in terms of the underlying metadata model and descriptions of model elements, it lacks advice on how to apply the quite complex and large model in practice. In order to overcome this problem, (Poole, Chang, Tolbert, & Mellor, 2002a) present very detailed recommendations of how to use the CWM constructs for representing different types of DWH metadata and explain the different mechanisms for extending the metamodel. However, they do not give specific recommendations for the process of matching existing metamodels with the CWM in order to find differences that require metamodel adaptations to real world data warehousing environments.

THE COMMON WAREHOUSE METAMODEL

CWM was specified as a reference model for metadata that is produced or used in the domain of data warehousing (OMG, 2003a; Poole, Chang, Tolbert, & Mellor, 2002b). Its main purpose is to enable easy interchange of warehouse and business intelligence metadata between software tools, system platforms and metadata repositories in distributed heterogeneous environments. The CWM specification is a joint effort of IBM, Unisys, NCR Teradata, Hyperion Solutions, Oracle, UBS, Genesis Development und Dimension EDI. It was recognized as an official standard by OMG in 1999.

Integration with other OMG standards

The main characteristic of CWM is its coherence to OMG's Model Driven Architecture (MDA) (OMG, 2003b). MDA is intended to integrate all OMG standards into a holistic modeling framework (Kent, 2002). As a consequence, CWM provides several links to existing standards in order to guarantee reuse of existing model constructs during CWM construction, and to facilitate the instantiation of the platform independent reference model to platform specific models. Figure 1 provides an overview of the most important artifacts of CWM and its links to artifacts of other standards. The artifacts are classified by their primary purpose in the context of CWM implementation (metadata access, metadata modeling and metadata exchange) and the abstraction level of the OMG metamodel architecture (Poole, Chang, Tolbert, & Mellor, 2003) on which they are situated.

According to the OMG terminology, CWM itself represents a metamodel which is situated on level M2 of the four-level metamodel architecture. The main goal that is pursued with the CWM is to provide a semiformal model of a language for the definition of data-warehouse-related metadata. In other words, the CWM is a language-oriented model of metadata, hence a metamodel. The metamodel in turn is constructed using Unified Modeling Language (UML) and represents therefore an instance of the UML metamodel.

Both the UML metamodel and the CWM are instances of the Meta Object Facility (MOF), which can be seen as a meta-metamodel defining the language for the construction of metamodels (OMG, 2000). MOF comprises elements that are very similar to those found in the UML metamodel, but situated on the next higher abstraction level (Poole et al., 2002b). Because CWM and UML are both MOF compliant, it is possible to exchange model elements of the CWM and the UML metamodel. For this purpose, MOF provides platform-independent interface definitions called MOF Reflective Interfaces. In order to generate application programming interfaces (APIs) for the exchange of metadata that is based on a MOF compliant meta-model like the CWM, so-called MOF-to-IDL-Mappings can be used. These provide a set of rules by which the metamodel constructs can be transformed into interface definitions represented in CORBA Interface Definition Language (IDL) (Balzert, 2000). Apart from using APIs, the CWM also supports the file-based exchange of metadata. This is achieved by using XML Metadata Interchange (XMI) which is another OMG standard. XMI provides mechanisms to transform MOF compliant

metadata (i.e. metadata that is based on a MOF compliant metamodel) into XML documents of a specific format which can be interchanged through file transfer (OMG, 2002; Poole et al., 2002b). The mechanisms mainly comprise two types of production rules

1. XMI DTD production rules can be used to transform a MOF compliant metamodel into an XML document type definition (DTD). This DTD in turn defines the syntax of an XML document that contains metadata based on the metamodel. In case of CWM, a corresponding DTD generated by XMI DTD production rules is already part of the CWM standard.
2. The second type of production rule provided by the XMI standard is the so-called XMI document production rule and can be used to transform metadata into XML documents that are based on the generated DTD (Jeckle, 1999; OMG, 2002).

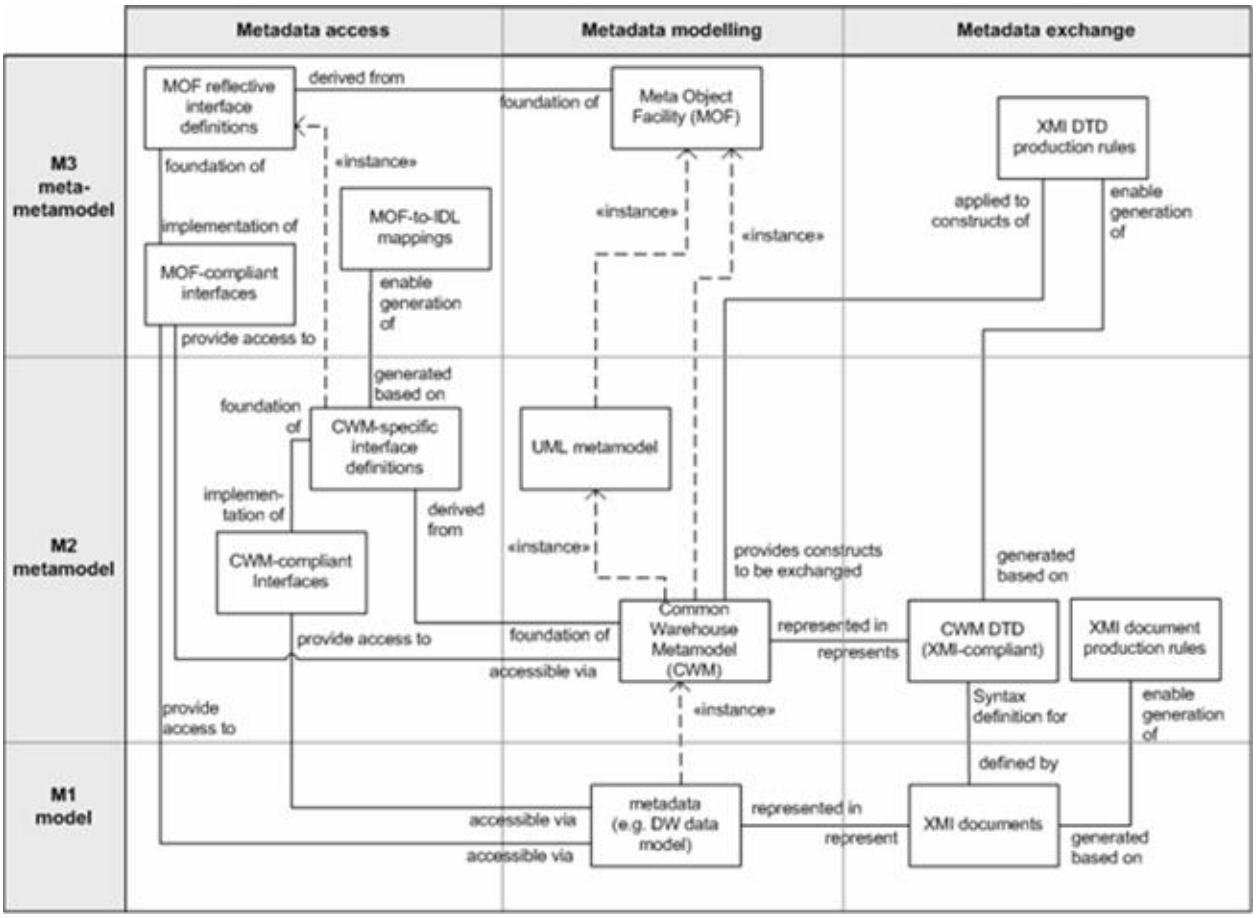

Figure 1. Linkages between CWM artifacts and other OMG standards

Modeling principles

By embedding the CWM into the OMG metamodel architecture and by integrating it with other related standards, the CWM development team has produced a standardized language for modeling, accessing and exchanging data warehouse metadata. The language is defined as an UML model comprising about 200 classes and about 150 associations. In order to make the model manageable in terms of comprehensibility and clarity, a modular structure has been created. This was achieved by decomposing the model into 22 UML packages, each containing the classes and associations needed to define a certain type of metadata.

The development team limited the number of model elements to a minimum by making extensive use of inheritance: Each class of the model is integrated into an overall class hierarchy and inherits as many attributes from its parent classes as possible. Moreover, association inheritance has been introduced by the development team as an extension to the UML

standard: An association between two classes is omitted from the model if the parents of the two classes are linked by an association of the same type.

Figure 2 shows an example of the association inheritance principle: The included classes define constructs to represent data structures. The model elements which are needed to represent relational data structures are combined into the CWM package Relational. The elements which are needed to represent record data structures are combined into the package Record. Both the classes Table and RecordDef represent some kind of container and therefore inherit from the same element Class which in turn inherits from the class Classifier. The Elements Column and Field represent data structures that can be included into these containers as container elements and therefore inherit from the same element Attribute, which in turn inherits through several steps from class Feature. The classes Class, Classifier, Attribute and Feature are combined into the package core that defines the basic model elements of the CWM. Table and RecordDef could each be linked to the class representing their respective container element (Column or Field) by a composition. But instead, the composition between the parent classes Classifier and Feature is reused, avoiding two additional model elements.

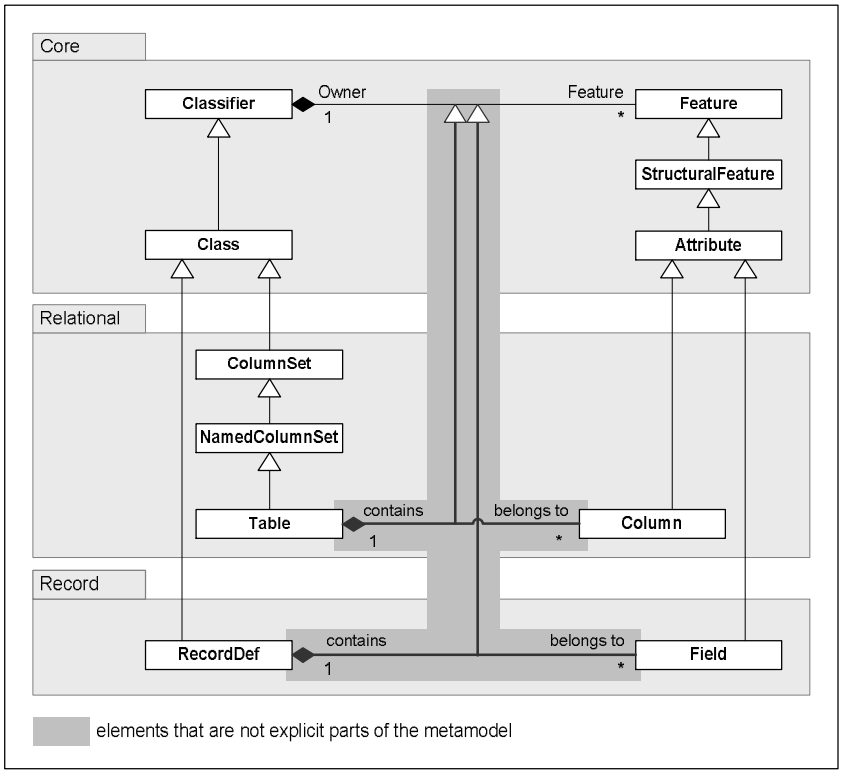

Figure 2. Example of CWM inheritance principles (Tolbert, 2001)

In addition to decomposing the metamodel into 22 packages, the modularity of the CWM is further increased by avoiding associations between classes situated in different packages. The only type of linkage that is allowed to cross packages boundaries is inheritance. This characteristic makes it possible to reduce the metamodel to a subset of packages that define only those types of metadata really needed for a specific implementation.

Model structure

Figure 3 illustrates how the 22 CWM packages are assigned to five different layers. Every package can rely on constructs within packages on a lower or on the same layer but is not dependent on any package of a higher layer (Poole et al., 2002b). Each layer combines all types of metadata associated with a specific problem domain of data warehouse development. The packages on the Object Model layer define basic modeling constructs that are used to design the CWM. The package Core contains the most important classes and every other package is dependent of this package. On the Foundation layer, several building blocks are defined: basic data types, rules for data type mapping, structures for keys and indexing, expressions, annotation and elements for adding contact information or software deployment data to other model elements. These building blocks are reused in many other packages. The Resource layer contains packages for the definition of different data structures

that can be found in operational or data warehouse systems. The Analysis layer defines metadata constructs used in the context of data transformation, data representation, data analysis, and terminology management. The Management layer comprises of two packages which mainly provide model elements for metadata concerning control and monitoring of technical data movement processes.

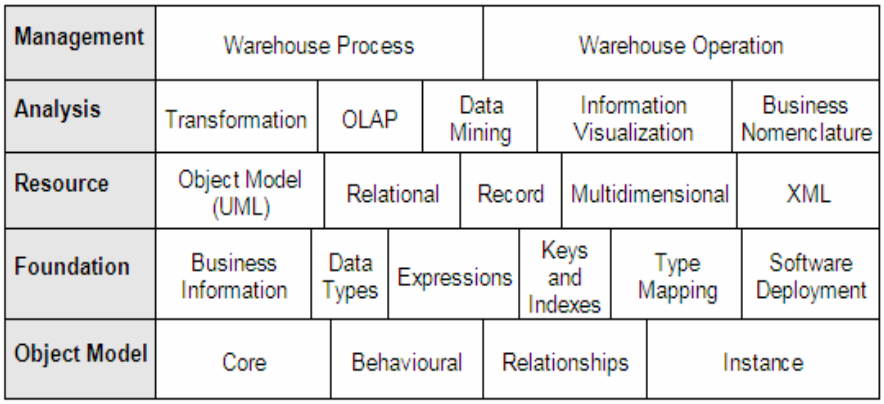

Figure 3. The CWM package structure (OMG, 2003a)

Alternatives of CWM implementation

Being platform independent, CWM can be applied to a large number of different platforms. The XML CWM DTD serves as a foundation for the exchange of metadata between platforms – only a XML interface has to be implemented for each respective platform to generate and / or process this type of XML documents. Many software vendors in the DWH domain already support this architecture. However, in most cases only a CWM subset is supported (Melchert, 2003).

An alternative to XML CWM DTD based file transfer between distributed metadata sources is to implement a centralized metadata database using CWM as its conceptual model. CWM databases can be implemented using either the object oriented or the relational data model (Muller, 1999). When compared to XML based document exchange, the database alternative allows for integrating metadata from different tools into one single metadata system. A central metadata database would support features like unified access control, unified versioning, consistency control or cross-tool reporting (Poole et al., 2003). However, the complete integration of all metadata in one single database seems to be unrealistic at the moment due to the fact that most DWH tools use on proprietary mechanisms for metadata utilization and partially rely on metadata not included in the CWM (Chisholm, 2001; H. H. Do & Rahm, 2000). A more viable approach would be to integrate only those metadata into a read-only central CWM metadata repository which are needed for certain DWH metadata management purposes (Auth, 2003).

A third alternative would be to implement distributed metadata management by using CWM compliant tool APIs. In contrast to the other two alternatives, this alternative is aimed at the propagation of metadata updates rather than at the ex-change of large amounts of metadata between different tool repositories. When CORBA IDL is used for specifying inter-faces, a wide range of implementation choices is preserved.

The CWM developers' vision for deploying these three CWM implementation approaches has been documented by proposing the CWM Metastore concept: The API alternative is seen as complementary extension of both a direct, XMI based metadata transfer between tools or a centralized metadata database (Poole et al., 2003). In addition to a data definition language (DDL) script for generating CWM compliant relational data structures, specifications for creating, updating and querying metadata instances are provided. By that means, the object oriented character of CWM can be preserved although a relational data model is used for metadata storage. In addition, the CWM Metastore proposal simplifies CWM utilization because repository structure and constraints are made transparent to developers and users.

APPLICATION EXPERIENCE

Project objective

The goal of the project described in this article was to develop a CWM-based model of data warehouse metadata that allows for the integration of metadata from different software tools within the data warehouse system.

The bank's metadata team decided to use CWM as reference model mainly due to its vendor independence and its conformance with other OMG standards. In contrast to other metadata standards, CWM is aimed specifically at defining metadata in the domain of data warehousing, which made it more suitable for the bank's DWH metadata project than models from other domains or domain independent models. In order to figure out whether the CWM is suitable for the project from a practical point of view, the metadata team decided to test the practicability of the reference model by using it in two different scenarios. As the first scenario, the data extraction and transformation process was chosen, in which legacy data files are cleansed and loaded into a relational database. As this process is widely automated and controlled by metadata from different sources, it provided an appropriate scenario for testing the capability of CWM in mapping existing metadata onto the integrated metamodel. Existing metadata particularly specified relational and record-based data structures, as well as data transformation rules.

While the first scenario is mainly focused on technical metadata, a second scenario was chosen in order to evaluate the suitability of the CWM standard to represent the bank's business metadata, especially glossaries of business terms.

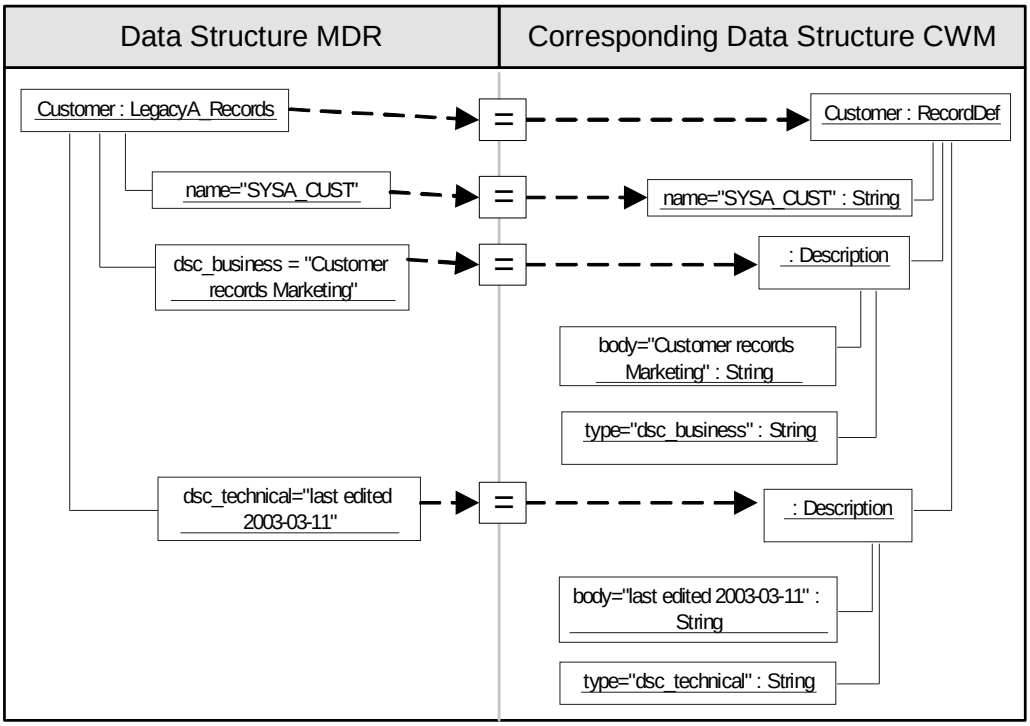

Figure 4. Mapping MDR attributes to CWM classes

Evaluation process

Although there are numerous approaches to the automatic matching of data models (H.-H. Do, Melnik, & Rahm, 2002), the bank's metadata team decided to do the mapping between the CWM and the bank's metadata repository (MDR) model manually in order to reduce the overall cost of the evaluation process.

The manual mapping process started on the highest level of granularity, as the package structure of the CWM (cf. figure 3) helped to identify the major subject areas that had to be further investigated. In order to decide in detail which existing MDR construct had to be mapped to which part of the CWM, it was necessary to compare both the existing MDR model and the reference model on the attribute level. For this purpose, the metadata team tried to identify one CWM element for each attribute of the bank's metadata model. This search process had four different types of results:

- An MDR attribute is identical with a specific CWM attribute, i.e. it has both the same designation and the same meaning as the CWM attribute. This is the most desirable test outcome because it does not require any modification to the reference model.

- An MDR attribute has the same meaning as a CWM attribute, but another designation. In this case, the CWM model element can be used without modification, and the old name of the MDR attribute will be saved as a synonym.
- An MDR attribute corresponds with a CWM class. This situation was dealt with by mapping the original MDR attribute to the most suitable attribute of the CWM class. In case of multiple MDR attributes corresponding with the same CWM class, one attribute of the class was used to distinguish these different meanings. Figure 4 illustrates an example where the MDR attributes "*dsc_business*" and "*dsc_technical*" are both mapped to the CWM class "*description*" and the attribute "*type*" is used to differentiate between the two possible meanings.
- An MDR attribute does not correspond to a CWM data structure. Here, the attribute was marked as candidate for the extension of the CWM.

At the end of the mapping process, the MDR attributes for which no corresponding CWM construct could be identified, were modeled as CWM extensions.

Test scenario 1 – technical metadata

The mapping of technical metadata to the CWM constructs revealed that 27 MDR attributes corresponded with CWM attributes and 23 MDR attributes could be represented by CWM classes. In particular, constructs from the following CWM packages were identified as relevant for the representation of the evaluated part of the MDR model:

- From the package *Record*, the elementary classes *RecordDef* and *Field* were used to represent records and record fields.
- In the package *Relational*, most elements for modeling metadata of relational databases could be matched.
- Metadata for representing technical extraction, transformation and loading (ETL) processes were provided by the package *Transformation*. The representation of simple data type conversion rules however required much more effort and involved more modeling elements than the old MDR solution.
- The package *Business Information* enabled the representation of descriptive attributes and responsibilities via the classes *Description* and *Responsible Party*.
- The package *Software Deployment* was used to represent elements describing the deployed information systems, their components and database interfaces.
- The package *Core* has to be used anyway as it is the foundation for all other CWM packages.

Apart from the packages listed above, the bank's metadata team identified aspects which are covered insufficiently or were even completely missing in CWM (see section 4). Examinations on the attribute level showed that usually only basic attributes (like object name, description) are provided by CWM. Additional attributes that are an integral part of the existing metadata pool like e.g. the timestamp of last update or technical information like data delimiter or default value did not have an appropriate CWM counterpart. In order to overcome these insufficiencies, the reference model had to be extended to cover 66 additional attributes.

CWM provides three different extension mechanisms (stereotypes, stereotypes in combination with tagged values and extension classes), the most suitable of which should be used (Poole et al., 2002b). In the bank case, modeling extension classes was considered to be the most appropriate way of extending the model, as this allowed for defining additional, non-textual attributes. Furthermore, this mechanism suits best to the modular character of CWM, so that the extension can be identified and grouped better than using stereotypes and tagged values.

After specifying the necessary extensions and selecting the appropriate extension mechanism, the bank's metadata team implemented enterprise specific extension classes to complete the test scenario. It became obvious that most of the extensions required the definition of new attributes. The attributes were integrated into new classes that were derived by building child classes to existing CWM classes. By doing this, the definition of additional associations became unnecessary in most cases, as inherited CWM associations could be reused instead. The constructed extension classes were combined into an own CWM extension package, following the recommendations of the OMG (OMG, 2001).

Test scenario 2 – terminology metadata

Apart from the technical metadata defining record structures and relational databases, the bank's metadata repository also contained business terms and abbreviations that were collected from multiple operational source systems and comprised business-related as well as IT-related terminology. Although the metadata team had already managed to partly consolidate the

different collections of terms into the MDR and to identify several links between terms of different origin, the existing metadata structure did not provide sufficient flexibility in representing different kinds of relationships like e.g. synonyms, homonyms and hierarchies.

Therefore, the main goal for the second test scenario was not only to map the existing structure to CWM constructs, but also to identify the possibilities provided by the reference model to model all desired kinds of terms and term relationships.

At the beginning of the mapping process, the package *BusinessNomenclature* was identified as primary source of CWM elements for terminology management. It allows for the definition of *terms*, i.e. domain-specific representation of abstract *concepts*. The associations provided in the package allow for representing terms and concept hierarchies as well as multiple connections between terms and concepts. Additionally, collections of interrelated concepts can be combined into *taxonomies* that represent a structured terminology of a specific domain. Terms can be combined to term directories, which are called *glossaries* and can be linked to taxonomies.

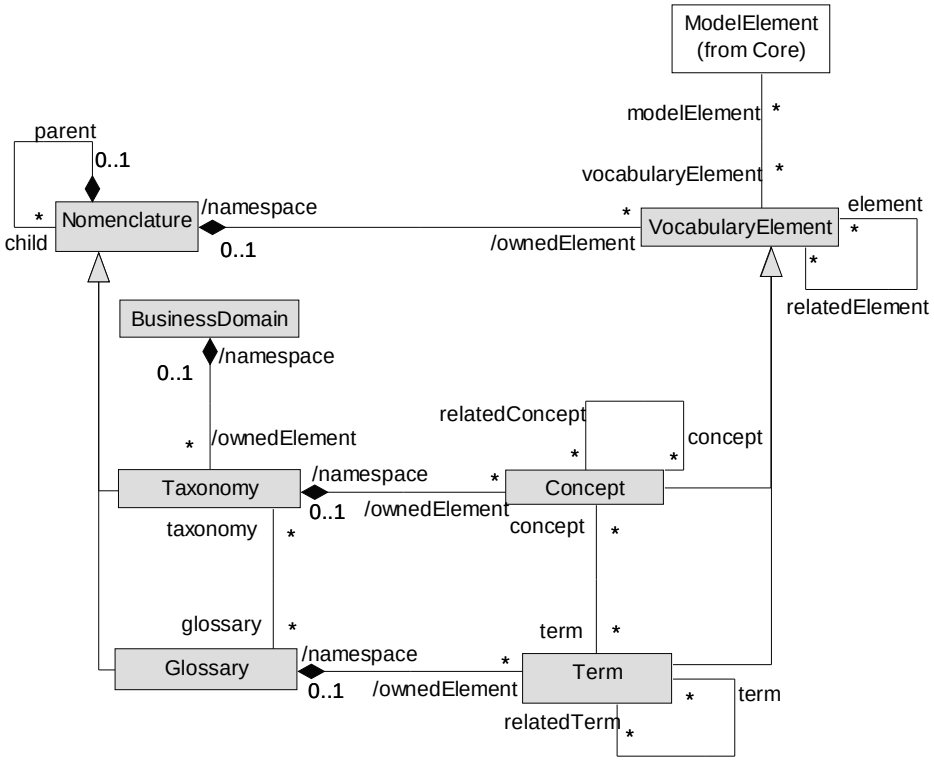

Figure 5. Major elements of BusinessNomenclature package (Poole et al., 2002b)

The comparison on the attribute level revealed that in contrast to the first scenario, the CWM provides all model elements that are required for the comprehensive terminology management that the bank was looking for. Most of the existing terminology metadata was kept in glossaries and could easily be mapped to the CWM glossary and term constructs. The major difference could be identified in the management of term abbreviations: The bank's glossaries included a separate attribute for specifying an abbreviation for each business term. While the CWM classes do not provide a specific attribute for abbreviations, each abbreviation has to be represented as a separate term that can be linked to the corresponding long text term. This also provided greater flexibility as more than one abbreviation could be maintained for a single term.

In summary the CWM BusinessNomenclature package provided very powerful constructs for representing the bank's terminologies.

Interpretation of the evaluation results

The evaluation revealed that the CWM standard is mainly intended to cover a broad range of metadata categories, thereby preferring model width over model depth. Especially in areas where technical metadata is used to automate technical processes like e.g. extraction, transformation and loading of data warehouse data, the standard does not provide all necessary

information. Although CWM deployment in those more technical areas requires numerous extensions, the standard model still serves as a solid basis. In the majority of cases that could be identified in the first test scenario, the extensions comprise only attributes that can be added to existing classes and neither additional classes nor new associations were required.

In the second scenario that dealt with metadata for terminology management, the CWM standard provided comprehensive support so that model extensions were hardly needed at all. One reason for this might be the fact that terminology metadata is not used for the automation of technical processes and does therefore not require as much detailed information as the technical scenario.

Regarding the results from both test scenarios, the CWM proved to be a suitable reference model for the bank's data warehouse metadata solution. While the coverage of technical metadata was rated only satisfactory, the overall structure of the model and the extensibility was rated good. The positive experiences supported the decision to roll out CWM-based metadata management to additional DWH application areas. Concerning the physical implementation of a CWM-based metadata management solution, the metadata team prefers a central repository, since software licenses as well as implementation skills within the bank's IT department are focused on relational database technology. Insofar the application of the CWM Metastore concept was regarded as promising starting point for further metadata integration efforts.

RECOMMENDED AREAS FOR EXTENDING THE CWM

Although the test scenarios signaled a general suitability of the CWM for the given metadata integration task, the examination has to be extended to all other metadata scenarios of interest to the bank. As mentioned in section 3, the CWM is mainly aimed at facilitating the metadata interchange between existing software tools, system platforms and metadata repositories. Being a platform-independent conceptual standard, CWM can also serve as a recommendation for the metadata that has to be managed to ensure effective and efficient usage of an enterprise data warehouse. From the experiences made in the application project described above and the results from a workshop with data warehouse practitioners of large financial service companies (Melchert, Auth, & Herrmann, 2002), the following four metadata areas were identified that are considered relevant for successful data warehousing, but not yet covered by the CWM standard.

1. In order to ensure a high quality level of the data provided by the data warehouse, the data warehousing process has to be supported by a proactive data quality management (English, 1999; Helfert & Herrmann, 2002). This process includes rule-based quality measurement, notification in case of deviation from quality standards or violations of quality rules, and quality indicators intended for the end users of the data warehouse (Helfert & Herrmann, 2002). For each of these tasks, a specific type of metadata is needed (Vassiliadis, Bouzeghoub, & Quix, 2002). By providing the class measurement (located in the Warehouse Operation package) the CWM does, in fact, enable the representation of data quality metrics. There are however no elements concerning quality conditions, quality acceptance levels, or rules for measuring and reporting data quality. Although it might be possible to model data quality rules as data transformations, it seems to be more appropriate to define explicit data-quality-related metamodel classes.

2. The CWM does not provide constructs for representing data-security-related aspects like access rights, users or roles (H. H. Do & Rahm, 2000). Such metadata however are important components of the data warehouse security concept that aims at preventing unauthorized access to the data warehouse in two directions. On the one hand, a specific authorization mechanism has to cope with very flexible data access which is common in data analysis (e.g. OLAP) (Priebe & Pernul, 2000). The single user has to be granted access only to certain aggregation levels of data warehouse data (Rosenthal & Sciore, 2000). Currently, there is no dominating mechanism to achieve this: Most data access control approaches are based on the proprietary metadata structures of specific software products (Priebe & Pernul, 2000). Integrating security-related metadata into the CWM would improve the security support and facilitate the establishment of a standardized access control mechanism for data warehouse data. On the other hand, not only the data, but also the functionality of data warehouse tools has to be protected against unauthorized access. Usually, every component of the data warehouse systems architecture manages separate security metadata in order to control access to its functionality. Hence, every user of the data warehouse has to be separately managed for every tool he or she is using. Extra work is required and inconsistencies may occur e.g. when people change their jobs. By establishing a standardized format for access control metadata, the CWM would contribute to solving these problems and allow for easier tracking of security problems and the realization of a single-sign-on-mechanism for the data warehouse system.

3. Although CWM does provide metamodel elements for the representation of reporting-related metadata, these constructs are not sufficient for the use within an enterprise context. The package Information Visualization provides elements for a high-level definition of different kinds of data representations like reports, charts, web pages, etc., but it does not allow for the assignment of reports to users or for arranging reports in an enterprise report directory. Data warehouse systems are accessed by a large number of users who may use different data analysis tools. The provision of a common format for

not only defining reports but also for managing reports and their assignments to data warehouse users as well as for tracking report usage would improve data warehouse management in two ways. Firstly, a central directory of all end users of data warehouse tools and their report usage is established. Hence it becomes possible to eliminate duplicate reports, to track inconsistencies and to improve report management. This may help to reduce the workload in query processing, e.g. by preprocessing one report that replaces the duplicate (or very similar) reports of two or more users. Secondly, data warehouse usage can be centrally tracked so that the information requirements of end users can be better understood and areas of high or low usage can be better identified. This information can be used to target the data warehouse specifically to actual user requirements. Thirdly, a central pool of usage metadata provides an excellent basis for data warehouse cost allocation and for calculating the return on investment of the data warehouse.

4. Since being designed for facilitating the exchange of metadata between software tools, CWM focuses on the definition of constructs for modeling technical metadata that describe static aspects of the data warehouse system. Dynamical elements are only included through constructs for the definition and the execution of technical data transformation process that are defined in the packages Warehouse Process and Warehouse Operation. Organizational aspects of the data ware-house are covered by the model elements for defining business terms (in the package Business Nomenclature) or contact persons (in the package Business Information). The explication and implementation of organizational processes however is necessary for establishing a permanent metadata management within an enterprise (Auth, 2003). Hence, is seems reasonable to enhance the CWM by modeling constructs for the definition, execution, and control of organizational processes. In his work on the construction of a reference process model for metadata management, Auth gives recommendations for possible CWM extension classes (Auth, 2003).

Although the above CWM extension recommendations can be justified from our analysis, there is no common agreement on theoretical concepts in all areas. Hence it is questionable whether appropriate CWM standard extensions will be possible in the near future. Nevertheless the content areas are extremely relevant in the context of data warehouse metadata management and should be considered. For the time being, it may be best-suited to design extension packages to the CWM as a first modeling recommendation.

CONCLUSION

This article introduced the Common Warehouse Metamodel as reference model for data warehouse metadata and reported results from an evaluation of the model in a practical case conducted by a large Swiss bank. The results of the evaluation revealed that CWM on the one hand serves as solid foundation for the standardization of data warehouse metadata, providing a robust model structure. On the other hand it became obvious that the model as a platform-independent standard is not capable of specifying technical metadata in such detail that it can be used to automate technical processes.

In a short discussion, four additional areas of meta-data were identified, in which the CWM should be extended in order to allow for a more comprehensive metadata support for data warehousing. In addition, more business oriented metadata applications (e.g. requirements specification, glossaries, DWH marketing) call for even more and more complex extensions. CWM provides basic mechanisms to consistently specify these extensions. Growing experience with actual projects will hopefully help the CWM approach to be adopted by more and more users and may even foster a marketplace for extension package reuse.

REFERENCES

1. Auth, G. (2003). *Prozessorientierte Organisation des Metadatenmanagents für Data-Warehouse-Systeme*. Unpublished Dissertation, Universität St. Gallen, Bamberg.
2. Balzert, H. (2000). *Lehrbuch der Software-Technik* (2. Auflage ed.). Heidelberg Berlin: Spektrum Akademischer Verlag.
3. Chisholm, M. (2001). Is the Metadata Repository Dead? *DMReview*(May).
4. Do, H.-H., Melnik, S., & Rahm, E. (2002). *Comparison of Schema Matching Evaluations.* Paper presented at the Proceedings of the GI-Workshop "Web and Databases", Erfurt, October 2002.
5. Do, H. H., & Rahm, E. (2000). *On Metadata Interoperability in Data Warehouses* (No. Technical Report Nr. 01(2000)). Leipzig: Institut für Informatik, Universität Leipzig.
6. English, L. P. (1999). *Improving Data Warehouse and Business Information Quality: Methods for Reducing Costs and Increasing Profits.* New York, et al.: Wiley computer Publishing.
7. Helfert, M., & Herrmann, C. (2002, 2002-05-27). *Proactive Data Quality Management for Data Warehouse Systems - A Metadata based Data Quality System.* Paper presented at the Proceedings of 4th International Workshop on

Design and Management of Data Warehouses (DMDW 2002) in conjunction with CAiSE 2002, Toronto, Ontario, Canada.
8. Jeckle, M. (1999). Modellaustausch mit dem XML Metadata Interchange Format. *ObjektSpektrum*(5).
9. Kent, S. (2002). Model Driven Engineering. In M. Butler, L. Petre & K. Sere (Eds.), *Integrated Formal Methods. Third International Conference (IFM 2002), Turku, May 15-18, 2002* (pp. 286-299). Berlin et al.: Springer.
10. Marco, D., & Jennings, M. (2004). *Universal Meta Data Models*. New York et al.: Wiley Publishing.
11. Melchert, F. (2003). Das Common Warehouse Metamodel als Standard für Metadaten im Data Warehousing. In E. von Maur & R. Winter (Eds.), *Data Warehouse Managament. Der St. Galler Ansatz zur fachlichen Gestaltung des Data Warehousing*. Berlin et al.: Springer.
12. Melchert, F., Auth, G., & Herrmann, C. (2002). *Integriertes Metadatenmanagement für das Data Warehousing. Grundlagen, Nutzenpotenziale, Architektur* (No. Arbeitsbericht BE HSG/CC DW2/03). St. Gallen: Institut für Wirtschaftsinformatik, Universität St. Gallen.
13. Meyer, M. (2000). *Organisatorische Gestaltung des unternehmensweiten Data Warehousing - Konzeption der Rollen, Verantwortlichkeiten und Prozesse am Beispiel einer Schweizer Universalbank*. Bamberg: Difo-Druck OHG.
14. Muller, R. J. (1999). *Database Design for Smarties - Using UML for Data Modelling*. London: Academic Press.
15. OMG. (2000). *Meta Object Facility (MOF) Specification. Version 1.3*. o. O.: OMG.
16. OMG. (2001). *Common Warehouse Metamodel (CWM) Specification. Volume 2 - Extensions. Version 1.0*. o. O.: OMG.
17. OMG. (2002). *XML Metadata Interchange (XMI) Specification. Version 1.2*. o. O.: OMG.
18. OMG. (2003a). *Common Warehouse Metamodel (CWM) Specification. Version 1.1, Volume 1* (No. formal/03-03-02). o. O.: OMG.
19. OMG. (2003b). *Model Driven Architecture (MDA) FAQ*. Retrieved 2003-09-11, from http://www.omg.org/mda/faq_mda.htm
20. Poole, J., Chang, D., Tolbert, D., & Mellor, D. (2002a). *Common Warehouse Metamodel Developer's Guide*. New York, et al.: John Wiley & Son, Inc.
21. Poole, J., Chang, D., Tolbert, D., & Mellor, D. (2002b). *Common warehouse metamodel: An Introduction to the Standard for Data Warehouse Integration*. New York, et al.: John Wiley & Son, Inc.
22. Poole, J., Chang, D., Tolbert, D., & Mellor, D. (2003). *Common Warehouse Metamodel - Developer's Guide*. Indianapolis, Indiana, US: Wiley Publications.
23. Priebe, T., & Pernul, G. (2000, 2000-11-10). *Towards OLAP Security Design - Survey and Research Issues*. Paper presented at the Proceedings of the third ACM international workshop on Data warehousing and OLAP (DOLAP 2000), New York.
24. Rosenthal, A., & Sciore, E. (2000). *View Security as the Basis for Data Warehouse Security*. Paper presented at the Proceedings of the International Workshop on Design
25. and Management of Data Warehouses (DMDW'2000), Stockholm, Sweden, June 5-6, 2000.
26. Staudt, M., Vaduva, A., & Vetterli, T. (1999). *Metadata Management and Data Warehousing* (No. Technischer Report 99.04 Institut für Informatik). Zürich: Universität Zürich.
27. Tolbert, D. (2001). *The CWM Experience - Implementing a UML-Based Data Warehouse Metamodel*. Retrieved 2004-06-21, 2003, from http://www.cwmforum.org/CWM_UML_Experience.ppt
28. Vassiliadis, P., Bouzeghoub, M., & Quix, C. (2002). Towards Quality-Oriented Data Warehouse Usage and Evolution. *Information Systems, 25*(2), 89-115.