The Process of Metadata Modeling in Industrial Data Warehouse Environments

Claudio Jossen, Klaus R. Dittrich

Database Technology Research Group
Department of Informatics
University of Zurich
CH-8050 Zurich
jossen@ifi.unizh.ch
dittrich@ifi.unizh.ch

Abstract: Modern application landscapes and especially huge enterprise applications, like data warehouses, used for decision support or other analyzing purposes get more and more complex. To manage, use and maintain these systems the need for metadata management has increased. In consequence of new tasks being identified by new groups of data warehouse users, the role of metadata management implies more than simply surf data schemas. It becomes necessary that metadata systems integrate different kinds of metadata and offer different views on the metadata as well. In this paper we discuss the process of identifying metadata model requirements, defining a new metadata model and finally implementing it in a metadata schema. The process is illustrated by a possible metadata model and schema, which were developed to meet the requirements of a complex data warehouse environment in Helsana Versicherungen AG, the largest Swiss insurance company. The paper describes the implementation of the metadata model based on the metadata standards Resource Description Framework (RDF) and RDF Schema (RDFS). The presented model and schema are just one possible solution and are not leading to a universal metadata model. The goal of this paper is to discuss the process of metadata modeling and to help metadata architects to develop their own metadata models and schemas.

1 Introduction

Data warehousing (DWH) has become a very important part of large companies' application landscape. The group of DWH application users is no longer restricted to the management or a few power users and data architects who know data schemata and data flows very well. Under the label of Business Intelligence (BI), there are now many types of applications using data warehouse technology to fulfill the needs of very different kinds of users. Modern data warehouses belong to the group of most important enterprise-wide technologies, like enterprise application integration (EAI) or enterprise information integration (EII). In fact, they are realizing some aspects of data integration especially in the field of expert systems and Decision Support Systems (DSS).

To keep such complex and large systems manageable for professionals and allow end-users to learn about the company's data flows and structures, the provision of a metadata management system is the best way. This obviously requires a metadata model. However, there is currently none available covering all needs of possible metadata applications. Standardized metadata models defined by groups of companies exist, but they can not contain all possible kinds of metadata another company may require. In this paper a metadata model is a set of elements, which are needed to describe a metadata schema. A metadata schema is a possible implementation of a metadata model.

This paper shows the process of creating a metadata model by following the principles of keeping a model as small and simple as possible and to include only as much complexity and functionality as needed. The goal of this paper is *not* to show a new metadata model that is applicable to *any* DWH environment. It just describes the way of metadata modeling and one possible solution depending on a company's requirements. Creating a metadata model and using it for the design of metadata schemas should be as adaptable to different metadata as it is to create a relational data schema for different kinds of data processing applications using the relational model.

In the field of metadata representation and modeling there are several metadata formats developed by different people from the research community as well as from industry [Vad01]. In the data warehouse field there are two main metadata standards: The common warehouse model (CWM) [Vet00] and the Resource Description Framework (RDF). CWM [Cw03] is part of the Object Management Group's standard family related to the Unified Modeling Language (UML) [Omg02] and is dedicated to the needs of metadata modeling in data warehouses.

A more generally used metadata standard is RDF[1] which is used as a metadata model in very different areas ([Sto05] is an example). In fact, not RDF itself is used to model metadata, but dialects based on the RDF/XML syntax [Tho06]. RDF is the basic standard for other semantic web metadata standards which are built on top of it: RDF Schema (RDFS) and the Web Ontology Language (OWL)[2].

Originally, there has been a third well known standard, Open Information Model (OIM). It was developed by the Meta Data Coalition (MDC), but in 2000 they joined the OMG and supported CWM as their standard.

[1] See http://www.w3.org/RDF/ for a description of the standard and its related technologies
[2] See http://www.w3.org/2004/OWL/ for the detailed standard

We decided to use our own RDF dialect instead of CWM because we do not cover some mandatory parts of CWM but need some additional aspects[3]. This means basically, that the MDMS will store not only data warehouse metadata but also metadata like interface descriptions (out of WSDL documents), business term definitions and their relations etc. Additionally we require a query language to access our data in different ways. The World Wide Web Consortium (W3C) has defined such a standard for querying RDF: SPARQL. SPARQL becomes more and more supported by RDF-supporting Frameworks and is really easy to understand because of its SQL-like syntax[4].

2 Metadata model requirements in a data warehouse environment

The use of data warehouses can be regarded as a kind of enterprise data integration, although some differences to other approaches of integration exist. Currently, most data warehouses are neither filled with just-in-time enterprise data nor offer application interfaces or services for the use with operational systems. At the same time most of the current reporting applications or front end tools of data warehouses are not full portal applications and not even deployable to a portal system. It is foreseeable that portal system manufactures will change this in the near future. However, this will not change the fact that data warehouses are not a full replacement for an enterprise information integration portal, although data warehouses are becoming more and more important parts of such systems.

In the field of BI and data warehousing, there are three main areas of interest if we follow the data flow from the data sources to the reporting tools data warehouse users work with. The first and oldest area of interest is the part of a data warehouse which normally covers the processes of **data extraction, transformation and load** into the data warehouse database (ETL). This part of a data warehouses is still the most complex one and gets the focus of most data warehouse projects at least during the development phase.

[3] There are case studies from the University of St. Gallen together with some huge Swiss companies like UBS, that described this problem [Mau03]. They also mentioned the lack of CWM suppporting metadata tools in the data warehousing area.

[4] For a list of frameworks and applications using SPARQL see: http://esw.w3.org/topic/SparqlImplementations There are some documents showing how widely accepted SPARQL is by now. For the description of the standard see http://www.w3.org/2001/sw/DataAccess/

The second area of interest in a data warehouse is now state-of-the-art and became necessary with the growing number of data warehouse users, especially with people joining this group, who are not expected to have deeper knowledge of data warehouses. They normally use standard reports or few ad hoc reports which are established by power users. This part of a data warehouse is sometimes even called BI but basically it covers all kinds of reporting tools querying a data warehouse, analyzing the results and delivering them to the requesting user. In the literature this group of applications/tools is called **reporting systems** [Mar04] or end user access (EUA). From a more technical point of view, this part of a data warehouse is also defined as a data mart. A data mart is basically a component of a DWH which can be used for querying a defined subset of all DWH data and for data mining [Riz06]. These subsets can be physically extracted and do not need to be disjoint.

The third area of interest is the newest and currently rarely implemented part of a data warehouse: **metadata management**. It became necessary with data warehouse environments getting more complex and the growing number of data, which is available through data warehouses to users, which do not know, where the data comes from. To work with this data the users need appropriate descriptions and additional knowledge about the data. Normally this "data about data" (basic definition of metadata) answer different kinds of questions like:

- Where does data come from? Where does data belong to? (Q1)
- How is data transformed, calculated, aggregated, etc.? (Q2)
- What meaning do data have? (Q3)

There are many more questions [Tan01] which show the need of metadata and the need of processes to collect them, to organize them and to provision them to data warehouse users. This is basically what metadata management is about, but this paper is limited to answer the three questions mentioned above (Q1, Q2 and Q3) because they are the most important, fundamental and in some cases needed ones to answer further questions. Their answers allow the development of a full metadata management as it is shown in the following chapters.

The approach discussed in this paper has been implemented as the so-called Metadata Management System (MDMS) in the largest Swiss insurance company, Helsana Versicherungen AG. It is now a productive information system running in the company for about two years with a still growing number of users.

3 Metadata Modelling

This chapter discusses the different kinds of metadata which have to be modelled in MDMS and how metadata standards are used.

The attempt to answer the questions Q1, Q2 and Q3 leads to three possible dimensions of metadata. The first one is the level of detail of metadata. In the data warehouse field this basically covers the position in a data model, for instance, metadata can be about a whole table or just about a single column. This dimension is where the answers to the first question are located.

The second dimension is the position of metadata in the data flow. This is not only meant as a physical position but also as the level of change of the corresponding data, due to calculation or aggregation, for instance. Obviously this dimension contains the answers to the second question.

The third dimension is less technical than the other two and introduces a semantic layer into our approach. The level of abstraction shows which metadata belong together by content but not by physical location or use in an information system. This dimension does not fully answer the third question (Q3). The fact of connecting metadata is the first step towards a broad and perhaps unstructured description of metadata. Adding unstructured metadata to the MDMS metadata model is not possible currently.

These three dimensions show that there are many possible solutions in metadata modeling. As mentioned in the first chapter, the decision was taken to use RDF as basic metadata model. RDF, together with its extension RDFS, offer the opportunity to create a possible metadata model covering the three dimensions.

RDF is based on triple-statements with the structure {Subject, Predicate, Object}. A *subject* is a resource like a webpage or a column in a data model. The *object* is an attribute of the subject it belongs to, a webpage's paragraph or a column's definition, for instance. The kind of relationship that connects an object to a subject is defined by the *predicate*. The most used kind of relationship is just parent vs. child node which could be modelled as a tree as known from XML. In our approach, this is basically used for representing data models which are hierarchically organized. As we mentioned before, the representation and the navigation through data model is addressed by the first dimension of metadata concerning the level of detail.

If more than one tree is based on several data model representations, a way to connect them to each other or to say it more general to link resources to each other is needed. RDF offers the opportunity to link two resources together by using another predicate which defines the object of the statement to be a resource as well (rdf:resource). This can also be used to link elements of the same graph together, for instance, a child with one of its ancestors. With this feature of RDF we can model the data flow between the different data models and so realize our second dimension.

To add an object-oriented point of view to a set of RDF statements RDFS is needed. RDFS offers predicates to define resources as classes and also as class instances (rdf:class and rdf:type). This mechanism realizes the third dimension, the level of abstraction.

4 Implementation of the MDMS metadata model

This chapter describes the metadata schema of MDMS built using its metadata model and gives a short overview of its implementation.

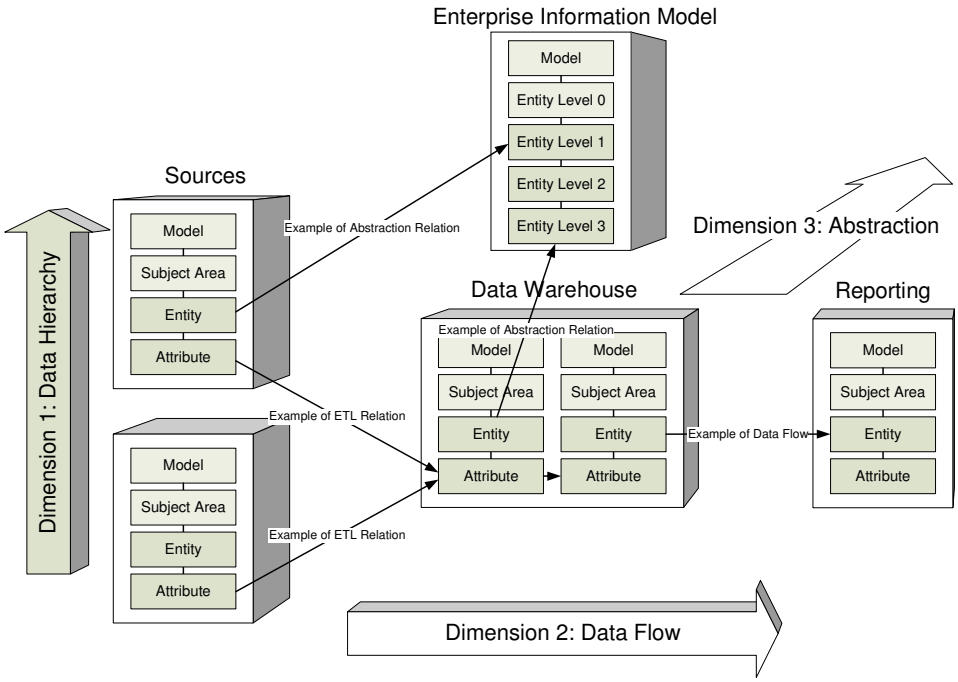

Figure 1: Data Dimensions in MDMS

The MDMS metadata schema basically contains data schemas from productive databases of Helsana. It makes no difference between data warehouse databases and databases from outside the data warehouse (source databases). There are basically two types of data schemas stored in the MDMS: data schemas and a high-level information entity schema, covering the whole company (Enterprise Information Model). The navigation through metadata inside of the data schemas and the Enterprise Information Model schema is basically a solution of the first dimension described in the last chapter. The links between all data schemas span the second dimension and the connection of entities from data schemas and entities of the Enterprise Information Model schema realizes the third dimension.

Figure 1 shows how the three metadata dimensions are implemented in the MDMS[5]. The core of a data schema is built by four basic types of resources: Model, Subject Area, Entity or Table and Attribute or Column. Basically each data schema has a logical view (Model, Subject Area, Entity, Attribute) and a physical view (model, Subject Area, Table, Column). Figure 1 uses the logical terms to represent a data schema. There are some more elements, which are missing in the figure, to make it better readable. Real schemas contain many more resources like datatype domains.

```
<rdf:Description rdf:about=http://www.helsana.ch/mdm/models/BDWH>
  <rdf:li>
    <rdf:Description rdf:about=http://www.helsana.ch/mdm/entities/adr>
      <rdf:li>
        <rdf:Description rdf:about=http://www.helsana.ch/mdm/attributes/adr/sprache_c>
          <mdmErwin:ParentAttribute rdf:resource=http://www.helsana.ch/mdm/attributes/adr_typ/sprache_c/>
          <mdmERwin:Code rdf:resource=http://www.helsana.ch/mdm/code_templates/bdwh/sprache_c/>
        </rdf:Description>
      </rdf:li>
      …
    </rdf:Description>
  </rdf:li>
  …
</rdf:Description>
```

Listing 1

Listing 1 is a short sample of the RDF representation of an entity in a data schema. The data hierarchy is basically done with RDF lists. They could easily be replaced by named links like the ParentAttribute tag, which represents a foreign key relationship. Another special relationship is implemented by the Code tag. In Helsana Code values are maintained directly in the corresponding database systems and they change faster then the data schemas they belong to. This caused the MDMS project to integrated metadata from outside the MDMS, which never become stored in the MDMS repository. The only way to get them is a SQL-query on the related database system. This was done by storing groups of SQL-queries (code templates) in the MDMS repository itself and implementing a post-processor of each SeRQL[6] / SPARQL[7] query, which runs against the MDMS. This post-processor replaces the RDF data about SQL and the corresponding database (Listing 2) by the current values from the database system and transform them into RDF, too. This is a very simple way of integrating data from a relational database system into an RDF/XML-based repository.

[5] To separate the three dimensions, MDMS uses three different XML namespaces which are named based on the tools the metadata come from or where they were collected, for instance by re-engineering: http://www.helsana.ch/mdm/mdmERwin#, http://www.helsana.ch/mdm/mdmETL# and http://www.helsana.ch/mdm/mdmInfo#.

[6] The Sesame RDF Query Language (SeRQL) together with SPARQL are the supported query languages of Sesame, the RDFS framework the MDMS uses to store RDF data.

[7] See http://www.w3.org/2001/sw/DataAccess/ for more details on the standard

As long as the MDMS does not require more advanced metadata integration, there is no reason to change this, but otherwise there exists software-bridges and mapping languages to cross the gap between the relational and the RDF/XML world like R2O [Bar04], for instance, that are much more powerful.

```
<rdf:Description rdf:about=http://www.helsana.ch/mdm/code_templates/bdwh/sprache_c>
  <mdmERwin:Database>teradata</mdmERwin:Database>
  <mdmERwin:Query>Select sprache.qs_id, qs.qs_kbez_x, …</mdmERwin:Query>
</rdf:Description>
```

Listing 2

The data flow is represented by relations between attributes or columns in the various data schemas and basically models what happens in a data warehouse. Because a data warehouse environment normally contains most of the analytical databases and most of the transactional or real-time productive databases as well, the MDMS metadata schema covers most or all databases used in the company.

The data transformations in the data flow are modeled like in Listing 3. Each transformation contains at least one step and each step might have several sources and targets. Steps have an order an obviously the target of a non-final step has to be a source of its following step. The sum of all transformations between two data schemas is their mapping. There are data schemas in the MDMS with several mappings from different source data schemas and it is really easy to add some more even incomplete mappings.

The MDMS defined five types of data transformation, which are used in the Helsana DWH environment by now:

- L: Lookup
- C: Case
- B: Calculation
- S: Standard
- G: Generated

Of course there is some more information stored in the MDMS on how the transformations are done in detail, for instance the different cases of a case transformation. Some of the information is some generated automatically from the ETL tool some become entered manually. This is necessary because of the heavy use of SQL-Queries instead of standard mechanisms in ETL-jobs.

```
<rdf:Description rdf:about=http://www.helsana.ch/mdm/transformations/SourceDB1001>
  <mdmETL:Type>S</mdmETL:Type>
  <mdmETL:Step rdf:resource=http://www.helsana.ch/mdm/steps/SourceDB1001_1>
  …
</rdf:Description>
<rdf:Description rdf:about=http://www.helsana.ch/mdm/steps/SourceDB1001_1>
  <mdmETL:Source rdf:resource=http://www.helsana.ch/mdm/attributes/T_D H_RECH/sender_ean/>
  <mdmETL:Source rdf:resource=http://www.helsana.ch/mdm/attributes/T_DH_RECHPOS/ean_responsible/>
  <mdmETL:Target rdf:resource=http://www.helsana.ch/mdm/attributes/ean/ean/>
</rdf:Description>
```

Listing 3

Entities from data schemas can be connected to a corresponding entity in the enterprise information model. The entities in the information model schema itself are on different levels of aggregation (Level 0 to Level 3, with Level 0 being the highest level of abstraction), so here again appears the dimension of data hierarchy as in all other data schemas.

```
<rdf:Description rdf:about=http://www.helsana.ch/mdm/entities/InfoModel/Adresse>
  <mdmInfo:Definition>Die Adresse eines Partners…</mdmInfo:Definition>
  <rdfs:SubClassOf rdf:resource= http://www.helsana.ch/mdm/entities/InfoModel/Partner>
</rdf:Description>
<rdf:Description rdf:about=http://www.helsana.ch/mdm/entities/adr>
  <rdfs:type rdf:resource= http://www.helsana.ch/mdm/entities/InfoModel/Adresse>
</rdf:Description>
```

Listing 4

Listing 4 shows the way entities from data schemas are linked to the Enterprise Information Model of Helsana. This is done by the standard tags of RDFS. The use of RDFS allows the MDMS to use inferencing for physical aggregation of data schema elements. Any other aggregation can be easily done by writing appropriate queries in SeRQL or SPARQL.

Because there is only one global metadata schema in the MDMS there is no need of using ontology matching and OWL. The metadata integration mapping is simple and very limited because of importing database schemas from one single tool.

The data modeling application which is mainly used in Helsana is CA's All Fusion ERwin. This tool has an interface for exporting XML files in the OMG XMI standard. Although the data schemas are then written in a standard metadata format, the implementation of XMI (the schema) is still related to Erwin. The next step is to translate the exported XMI files with XML Schema Transformation (XSLT) into RDF, which has to be done by a Rich Client Application called Local Model Manager. The Local Model Manager checks and solves some dependencies with other schemas and then stores the transformed schema into a standard relational database (Microsoft SQL Server). This is done by a Java framework called Sesame[8], which supports RDF and RDFS. Sesame has been chosen because of its better performance running in the Helsana database environment compared to the better known Jena framework from HP Labs[9]. Possible reasons might be the heavy use of varchar fields in the database Schema of Jena and the huge amount of joints on the very same table (Jena stores each RDF schema in a new table) that some of the MDMS queries caused on the underlying relational databases.

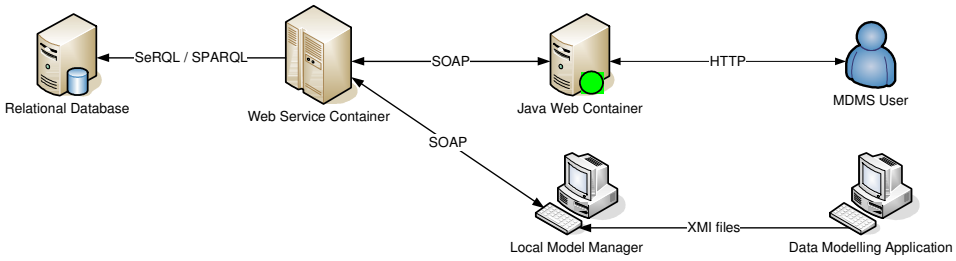

Figure 2: MDMS Architecture

The Sesame RDFS interfaces are used by Web Services running in a Web Service Container (Apache Axis2) on a Java Web Application Server (Apache Tomcat 5.5). Beside this, a web application offers a web frontend to the users of the MDMS by calling the same web services as the Local Model Manager.

The metadata MDMS supports so far are more or less well-structured data models and their relation to each other. The MDMS also offers the opportunity to store additional comments provided by users. This is still structured content by the way it can be entered (pre-defined text areas with descriptions about possible content) and by the way it is stored (pre-defined tags and relations).

[8] See http://www.openrdf.org for more details
[9] See http://jena.sourceforge.net for more details

To evolve MDMS from a system that just offers content created and managed by other applications, towards a knowledge management system with its own content created and maintained by its users, the system needs the possibility to store unstructured content[10] as well. This could be done for instance by adding support for WebDAV [Scha06] or another technology supporting the management of documents combined with the storage of metadata. Our vision is basically to expand the existing MDMS with some kind of Wiki application, which uses Semantic Web technologies to store its content [Buf06].

As metadata models are getting more complex over time and the amount of relations between them grow, too, the need of appropriate model operators [Mel03] is getting more and more urgent. These operators have been defined in another interesting research area called Generic Model Management and will help to separate sets of metadata for instance, or allowing us to implement operations like insert, delete and update that are very well established in relation databases but not yet in RDF repositories.

5 Conclusion and Further Work

This paper showed how to define a new metadata model using RDF and RDFS for a data warehouse environment that fulfills the needs of an industrial setting and how it has been implemented in the application landscape of Helsana. Although this is now a fully functional and widely accepted application solution in the company itself, there are still some points left where MDMS could be expanded and enriched to support more metadata needs.

The final goal to be reached through future development of MDMS is the combination of application metadata with database metadata. This will help us to offer information about which data were accessed or changed by which process. We hope to realize this with the extraction of application metadata from web service descriptions and additional documentation provided by software developers.

References

[Bar04] Barrasa J.; Corcho O.; Gómez-Pérez A.: R2O, an Extensible and Semantically Based Database-to-Ontology Mapping Language. Second Workshop on Semantic Web and Databases (SWDB2004). Toronto, Canada. August 2004

[Buf06] Buffa, M.; Gandon, F.: SweetWiki: Semantic Web Enabled Technologies in Wiki. Proceedings of the 2006 international symposium on Wikis WikiSym '06; ACM Press; Aug. 2006

[Cw03] Object Management Group (OMG): Common Warehouse Metamodel (CWM) Specification, Version 1.1, Volume 1; http://www.omg.org, Mar. 2003

[Mar04] Marco, D.; Jennings, M.: Universal Meta Data Models. John Wiley and Sons; Wiley Publishing, Inc., 2004

[10] In this context unstructured content means plain text from technical documents, business term definitions, user comments, etc.

[Mau03] Von Maur E.; Winter R.: Data Warehouse Management – Das St. Galler Konzept zur ganzheitlichen Gestaltung der Informationslogistik; Springer 2003

[Mel03] Melnik, S.; Rahm, E.; Bernstein, P.: Rondo: A Programming Platform for Generic Model Management. Proceedings of the 2003 ACM SIGMOD international conference on Management of data; ACM Press, Jun. 2003

[Omg02] Object Management Group (OMG): Meta Object Facility (MOF) Specification. Version 1.4; http://www.omg.org, Apr. 2002

[Riz06] Rizzi, S.; Lechtenbörger, J.; Abelló, A.; Trujillo, J.: Research in Data Warehouse Modeling and Design: Dead or Alive?. Proceedings of the 9th ACM international workshop on Data warehousing and OLAP DOLAP '06, ACM Press, Nov. 2006

[Scha06] Schandl, B.; King, R.: The SemDAV Project: Metadata Management for Unstructured Content. Proceedings of the 1st international workshop on Contextualized attention metadata: collecting, managing and exploiting of rich usage information; ACM Press, Nov. 2006

[Sto05] Stock, I.; Weber, M.; Steinmeier, E.: Document authoring, production and management: Metadata based authoring for technical documentation. Proceedings of the 23rd annual international conference on Design of communication: documenting & designing for pervasive information SIGDOC '05; ACM Press; Sept. 2005

[Tan01] Tannenbaum, A.: Metadata Solutions – Using Metamodels, Repositories, XML and Enterprise Portals to Generate Information on Demand. Addison-Wesley, Pearson Education; 2001

[Tho06] Thomsen, C; Pedersen, T.: Data Warehouse Construction: Building a Web Warehouse for Accessibility Data. Proceedings of the 9th ACM international workshop on Data warehousing and OLAP DOLAP '06, ACM Press; Nov. 2006

[Vad01] Vaduva, A.; Vetterli, T.: Metadata Management for Data Warehousing: An Overview. Intl. Journal of Cooperative Information Systems (IJCIS), Vol. 10, No. 3, 2001, p. 273-298

[Vet00] Vetterli, T; Vaduva, A.; Staudt, M.: Metadata Standards for Data Warehousing: Open Information Model vs. Common Warehouse Metamodel. ACM Sigmod Record, 29(3); Sept. 2000