The Role of Metadata for Data Warehousing*

Martin Staudt† Anca Vaduva‡ Thomas Vetterli†

† Swiss Life, Information Systems Research (CH/IFuE),
P.O. Box, CH-8022 Zurich, Switzerland
<firstname>.<lastname>@swisslife.ch

‡ University of Zurich, Department of Computer Science,
Winterthurerstr. 190, CH-8057 Zurich, Switzerland,
vaduva@ifi.unizh.ch

Abstract

Metadata has been identified as a key success factor in data warehouse projects. It captures all kinds of information necessary to extract, transform and load data from source systems into the data warehouse, and afterwards to use and interpret the data warehouse contents. This paper gives an overview about the role metadata plays for data warehousing and reviews existing standards, commercial solutions and research actions relevant to metadata management. It turns out that an overall solution for managing all metadata in a central or federated repository is still missing regarding a global metadata schema as well as system aspects and interoperability among involved tools producing metadata. The divergence of proposed standards will probably prevent a breakthrough within the near future.

1 Introduction

The topic is as old as data exist: metadata have ever been needed to describe the meaning or properties of data with the aim to better understand, manage, and use that data. A classical example are libraries. Books (data) may be classified, managed and retrieved only by means of appropriate metadata (i.e., title, author and content keywords).

Metadata is commonly understood as any information needed in information technology in order to analyse, design, build, implement and then use computer systems [2]. In the case of information systems, metadata particularly facilitates managing, querying, consistent use and understanding of data.

Many recent efforts within both the academic and industrial community have concentrated on issues related to metadata. The generation, storage and management of metadata promise to better support the exploitation of the huge amount of data available nowadays in every conceivable electronic form. Since everything computers work with is inherently data and (a

*This work was supported in part by the Swiss Federal Office of Professional Education and Technology under grant KTI-3979.1 (SMART).

kind of) metadata accompanies any data, the notion may be found in any thinkable application domain and takes various forms depending on its use.

This paper aims to clarify the role of metadata in the particular case of data warehousing[1]. Starting with the pioneering work of Inmon [16], the popularity of data warehousing grew exponentially during the last years. Competitive organizations are just on the way to build data warehouses or to extend, reengineer, and improve already existing one(s). An abundance of software products for building and exploiting data warehouses are on the market.

Data warehousing is a collection of concepts and tools which aims at providing and managing a set of integrated data (the data warehouse) for business decision support within an organization. In this way, important business trends may be discovered and explored and better and faster decisions may be achieved regarding multiple aspects of the business like sales and customer service, marketing, and risk assessment.

In order to cope with the complexity of building, using and maintaining a data warehouse, a metadata management system is indispensable. It may be used by other components in the data warehouse system or directly by humans to effectively and efficiently achieve their particular tasks.

The rest of this paper is organized as follows: In Section 2 we give an introduction to data warehousing discussing the relevant tasks and architectural components of a data warehouse environment. Section 3 summarizes the objectives pursued with metadata in the data warehouse context. Based on that, Section 4 and 5 confront the requirements for a metadata management system and for a metadata schema dedicated to data warehousing with the solutions and concepts provided by current tools, standards and research projects. In particular, a classification of relevant metadata types is given in Section 5. In Section 6 we summarize our observations and point to several open problems.

2 Data Warehousing: Tasks and Architecture

The tasks of data warehousing comprise the processes of designing, building, using, and maintaining a data warehouse. The data warehouse system is first designed at *designtime*, then the warehouse is populated at *buildtime*, and finally it is employed at *usetime*. *Maintenance* partly reiterates the first two phases. After recalling the main differences between a data warehouse and operational systems within an enterprise, we consider each of these phases.

2.1 Data Warehouses versus Operational Systems

A data warehouse typically incorporates data collected from a variety of heterogeneous data sources (database systems, flat files, indexed files, so-called legacy systems[2], Web pages, etc.) including current operational enterprise-internal systems as well as external data sources. Data has to be extracted, cleansed, transformed and stored in an integrated form in the data warehouse.

Compared to traditional operational systems which focus on transaction processing, data warehouses are designed for ad-hoc, complex queries with optimized response time. This may not be achieved in operational systems where database schemas are normalized and

[1] A more comprehensive overview, in particular with a broader discussion on metadata in general, can be found in [28].

[2] The term *legacy systems* is used for all those 'old-fashioned' applications in enterprises which store and manage data mostly in proprietary formats and which have grown over time often in a chaotic manner with dubious effects on data quality and interoperability.

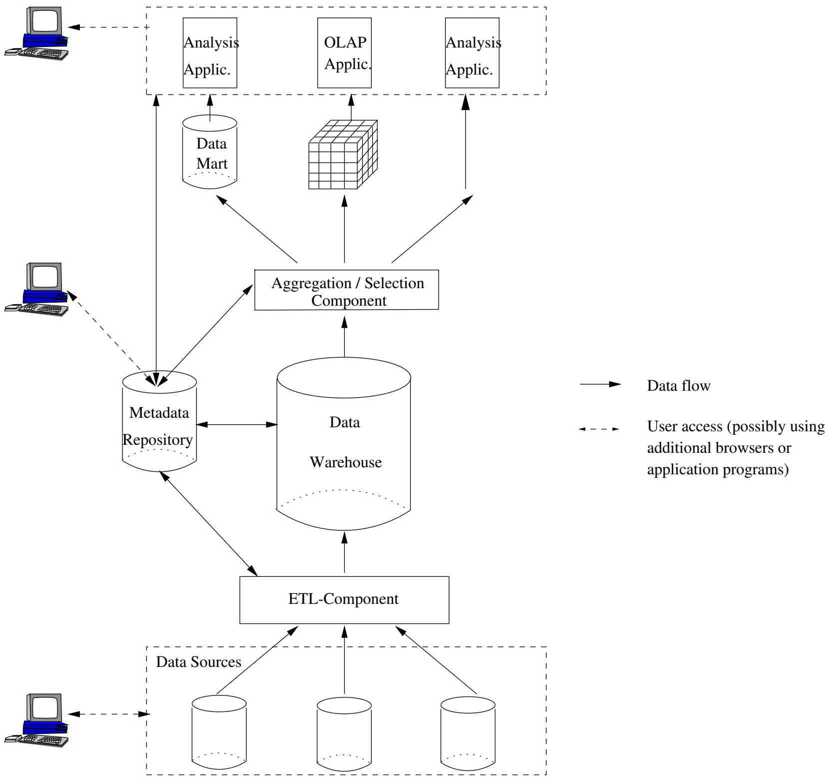

Figure 1: A data warehouse system at buildtime and usetime

a multitude of table joins would be required for complex queries inherently yielding weak performance.

An operational system normally provides current data values for the daily business, while a warehouse supplies a view of the business over a period of time by means of historical data, necessary for business trend analysis. Data in a warehouse is typically updated only at certain points in time (e.g., daily) and thus it does not necessarily contain the most recent data values. This is justified since on the one hand data warehouse systems serve for decision support and (long-term) analysis applications, which do not need absolutely current information. On the other hand there exists a tradeoff between the currentness of data and the substantial effort required to bring the data into the warehouse and to organize it in the special shapes (e.g., aggregations) suitable for such applications.

2.2 Building the Data Warehouse

Figure 1 depicts a typical architecture of data warehouse systems with the interactions that happen between the components at buildtime and runtime.

The warehouse is populated with the data extracted from the sources through two processes: the initial *loading* phase performs data extraction, cleaning, transformation, and storing into the target source (the data warehouse), and the regular *refreshment* that propagates (executing the same activities) changes into the warehouse. We consider the main components that participate in the loading and refreshment process:

Data sources contain structured, unstructured or semi-structured data and follow different data models, e.g., traditional operational database systems supporting so-called online transaction processing (OLTP) applications.

As the name suggests, the **E**xtraction, **T**ransformation and **L**oading **(ETL) component** extracts data from the sources, transforms it, and stores it in the data warehouse. Data transformation may take various forms: reconciliation of syntactic and semantic differences between operational sources, consolidation, and mapping from local data models to the global one. Examples of possible transformations include: elimination of duplicates, the calculation of derived data (like the age inferred from the date of birth), enrichment of data (like completion of addresses based on the value of the postal zip code), and so on.

The data warehouse represents the kernel of the data warehouse system. It serves as an (enterprise-wide) collection of integrated data, usually stored in a relational database system.

The **aggregation and selection component** performs further processing steps. For example, it computes the views to be stored in the data marts or prepares data to be fed into analysis applications.

Data marts are data stores that are subordinated to a data warehouse. This means they render information views of the same single integrated pool of data. Data marts are built with the aim to provide specific application requirements of a certain group of users, e.g., of a department or a geographical region.

The metadata repository plays a key role in every phase of data warehousing. During buildtime, metadata governs the extraction, transformation and loading of data into the data warehouse.

2.3 Using the Data Warehouse

Data warehouses are built for analysis purposes. The analysis involves examining data and possibly identifying relationships that may exist between different elements.

The most popular analysis means are **OLAP[3]-tools** which enable users to examine data within a multidimensional model allowing to instantaneously retrieve and summarize data. The multidimensional model provides measures (i.e., business facts to be analyzed like sales or shipments) and dimensions for these measures (i.e., the context in which the measures have values assigned, e.g, products, customers, time, region). Usually, dimensions have associated hierarchies that specify aggregation levels for viewing the data. Since the data is appropriately organized, query performance is significantly increased.

Data mining offers more powerful means for extracting information. It aims at the automatic detection of implicit, previously unknown and potentially useful patterns in data and their translation into valuable information. Successful application domains include fraud detection, loan approval, and portfolio trading. Data mining techniques comprise classification, clustering, association discovery, and deviation and change detection techniques [6]. Note that for effective data mining, data quality has to be high (i.e., as few as possible missing, incorrect and stale data).

Metadata helps the user to understand the content of the warehouse. Information about the meaning of data elements, availability of reports etc. are indispensable to successfully use the data warehouse.

2.4 Designing the Data Warehouse

The design phase comprises defining the structure of the warehouse (including source schema integration) and developing software that carries out data extraction, cleaning, data integration, and loading into the warehouse. The process covers also the development of end-user

[3] Online Analytic Processing

applications for extracting information from the warehouse (e.g., browsing, OLAP, and other analysis applications).

Data marts contain selected and aggregated portions of the main warehouse. In order to serve OLAP-applications they often have multidimensional schemas and can then be implemented either by special multidimensional databases or relational databases. In the latter case, the data is organized as a star schema with separate tables for measures (including aggregations) and dimensions.

The development of ETL-components is a rather complex task. There are three alternatives to design ETL-software. The first option uses *hard-coded scripts*, which however are difficult to maintain. The second option allows the specification of transformation rules. These rules are used to generate procedural code (*code-generation* systems), improving the ability to quickly adapt to changing requirements. In contrast to the code-generation systems, the *engine-based* systems (the third alternative) directly interpret the metadata, dynamically executing the transformations.

The metadata repository plays a key role for data warehouse design and the development of software components as well. Designers and application developers consult the repository where, for example, results of CASE-tools, design experiences, and system documentation are stored.

2.5 Maintaining the Data Warehouse

Warehouse administrators perform the maintenance of the data warehouse system. This process concerns the periodic refreshment of the data warehouse, updating of software when application requirements change. It may also require to specify further transformation rules or to compute derived data or additional aggregation and summarization. The development of new applications that access the data warehouse may be seen as being part of maintenance as well.

The maintenance process of data warehouse systems also leads to updates of the metadata repository, which may happen either automatically or manually, depending on the metadata type and use. Manual updates can fall under the responsibility of different people. For example, metadata that refers to the terminology of business users is not necessarily the concern of warehouse administrators but of people responsible for knowledge management in an enterprise.

3 Metadata for Data Warehousing

Because of the complexity and extensive use of metadata, a compact, precise definition of the notion may hardly be provided. Therefore, we explain metadata by discussing its purposes and the role played during the execution of the data warehouse processes. Metadata may be used in three different ways:

- *passively*, by providing a consistent documentation about the structure, the development process and the use of a data warehouse system. The availability of documentation supports all "actors" (i.e., end-users, system administrators, and application developers) in achieving their tasks.

- *actively*, by storing certain semantic aspects (i.e., transformation rules) as metadata which is interpreted and executed at runtime. In this case, the data warehouse processes are *metadata driven*. As a consequence, code (i.e., the active metadata) and additional documentation are uniformly and consistently managed in the same repository and the currentness of the documentation is possibly improved.

- *semi-actively*, by storing static information (e.g., structure definitions, configuration specifications) to be read by other software components during their execution. For example, query parsers need metadata in order to verify the existence of an attribute. In contrast to the *active* use, metadata is only read, not executed at runtime.

The generation and management of metadata serves two purposes: (1) to minimize the efforts for development and administration of a data warehouse and (2) to improve the extraction of information from it. The first objective mainly concerns

- *supporting system integration.* Schema and data integration rely on metadata about the structure and the meaning of the individual data sources and of the target system. Transformation rules to be applied to the original data are stored as metadata as well. Furthermore, the integration of different tools is only possible if they share their "data" which actually is, in this case, the metadata of the data warehouse system.

- *supporting analysis and design of new applications.* Metadata increases control and reliability of the application development process by providing information about the meaning of the data, its structure, and origin. Furthermore, metadata regarding design decisions adopted for existing applications may be reused.

- *improving the flexibility of the system and the reuse of existing software modules.* This objective is valid only for the active and semi-active use of metadata. Semantic aspects likely to change frequently are explicitly stored as metadata outside the application programs. Maintenance is therefore substantially easier, and the system may be extended and adapted without difficulty. This approach also enables the reuse of these "code fragments".

- *automating of various administration processes.* Metadata also drives the execution of the diverse warehouse processes (like loading and refreshing). Information about their execution (access logs, number of records added to the warehouse etc.) is also stored in the repository for easy access by the administrator.

- *enforcing security mechanisms.* Metadata should provide the access rules and user rights for the whole data warehouse system. Access control in a data warehouse system may require sophisticated methods. For example, the operational sources may contain harmless information about single figures of an enterprise but the summarization of the values stored in the data warehouse may be top secret. On the other hand, individual incomes of the employees on the source site are secret, but the total volume stored in the data warehouse may not be a critical information.

The second objective refers to the effective extraction of information from data:

- *improving data quality.* Data quality includes dimensions like consistency (whether the representation of data is uniform and no duplicates, no data with overlapping and confusing definitions exist), completeness (whether data is missing), accuracy (the conformity of the stored with the actual value, including precision and confidence of the data), timeliness (whether the recorded value is up-to-date). Quality assurance rules have to be defined, stored as metadata and checked each time the data warehouse is refreshed. In addition, high data quality requires the support of data tracking. Metadata provides information about the creation time and the author of the data, the source of the data (data provenance), the meaning of data at the time it was captured (data heritage), and the path followed from source to the current site (data lineage) [8]. In this way, users may reconstruct the path followed by data during the transformation process and verify the accuracy of returned information.

- *improving interaction with the data warehouse system.* Interaction may be performed either by means of simple queries and reporting applications or by using complex analysis applications. Metadata provides information about the meaning of the data, the terminology and business concepts used within the enterprise and their relationship to the data. Thus, metadata improves query, retrieval and answer quality. It allows to pose precise, well-directed queries and reduces the costs for users accessing, evaluating and using appropriate information.

- *improving data analysis.* Methods for data analysis cover a large spectrum, starting with simple reporting applications that include summarizations, continuing with OLAP, and ending with complex data mining applications. In this context, metadata is necessary to understand the application domain and its representation in the data warehouse in order to adequately apply and interpret results.

- *enforcing a unique terminology and communication language within the enterprise.* The availability of a metadata management system as a unique documentation source for users brings other benefits as well: it ensures a consistent means for people to communicate, understand, and interpret information provided by the data warehouse system, eliminates ambiguity and guarantees consistency of information within the enterprise, and it enables sharing of knowledge and experience.

4 Data Warehouse Metadata Management

Metadata is stored and maintained in a repository, which is a structured storage and retrieval system, usually implemented on top of a database management system. Metadata needs for a specific application domain (like data warehousing) actually impose the repository structure (e.g., the metadata schema) and the semantics of metadata to be stored.

In order to fulfill functional and architectural requirements for metadata repositories, several standards have been defined which influence commercial products and research prototypes in one or the other way. Based on the selected repository tool and/or architecture, a specific metadata schema for data warehousing can be implemented (see Section 5).

4.1 Requirements for Metadata Repositories

The typical functional requirements for metadata repositories are presented in detail in [5]. We pick up the most important aspects and summarize possible architectural alternatives influencing the position of a repository within a data warehouse environment.

4.1.1 User Access

The main purpose of a metadata repository is to provide the necessary information that may help users to achieve their tasks. Thus, it has to offer suitable mechanisms for querying, navigating, filtering and browsing the metadata it manages.

The structure (schema) of the repository has to support *querying* according to specific conditions. For example, it should be possible to pose queries for selecting all activities which make up the refreshment process, all logical metadata related to a specific business element, or metadata that has a certain origin, a certain purpose, a certain production time, etc. This require metadata to be "labeled" with attributes that contain the appropriate values.

Relationships/dependencies between individual metadata elements are important for understanding the system. Ideally, the repository provides not only explicit but also implicit (hidden) relationships between aspects of managed metadata. An essential functionality requirement is the *navigation* within the metadata collection. Starting with a certain element, the user may navigate to other elements along existing relationships. Navigation is "driven" by the underlying schema which is specified on a conceptual level by the *metamodel* of the repository.

Filtering refers to the selection of relevant information when search criteria are not necessarily provided by the structure of the repository. This means, besides querying of fixed attributes, filtering presumes the search of keywords within textual descriptions. In this way, all information related to a certain topic may be provided.

Browsing requires an appropriate, user-friendly (and thus highly graphical) interface for interacting with the repository. User views play a central role for browsing, since they restrict access to information according to user interests. Each user view has a defined starting point for browsing and navigation which provides the elements the user may start with when exploring the repository.

Manually *editing and updating* a metadata repository can be a very complex task, if, e.g., long sequences of operations should be recorded and many interrelated objects are touched. This can be supported by adding to the metamodel additional models which formalize certain types of update processes and, e.g., allow the generation of suited forms and guidance mechanisms for editing tools.

4.1.2 Interoperability and Tool Access

The interaction of software components and tools with the repository requires appropriate mechanisms, in particular:

- a comprehensive application programming interface (API) for metadata read and write access by other software components,
- interfaces ensuring the interoperability with other repositories. One way to make two repositories interoperable is to adopt a common interchange representation format at both sides and to use it when repository data has to be imported or exported, and
- a flexible core data model that allows to easily define domain-specific metadata types and to extend a given set of types concerning additional tools or new data sources.

4.1.3 Change Management

Change management deals with the handling of changes inside and outside of the repository. A *notification mechanism* is necessary to propagate changes to tools that registered their interest in being notified. Also users who previously "subscribed" are informed.

The repository has to provide *version and configuration management*. Important changes of metadata (e.g., due to schema updates of operational sources schemas) require the creation of different versions and their storage in the repository. Problems may arise with inconsistent

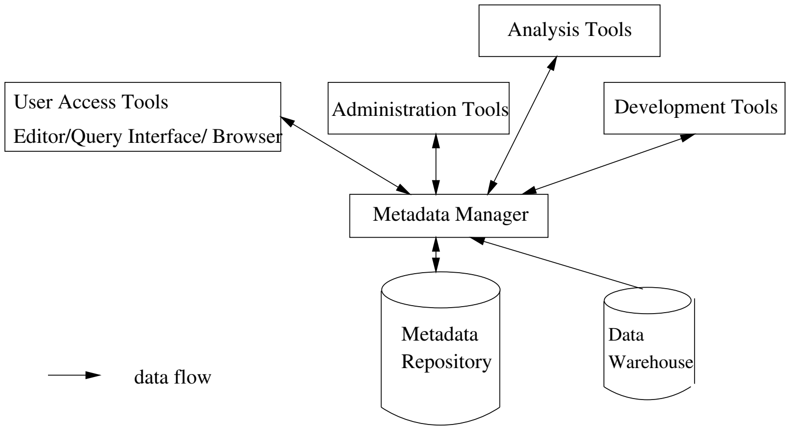

Figure 2: Metadata Repository and the Tools using it

links, e.g., descriptions on the conceptual level which are not valid anymore when changes on the implementational level (like changes of code) occur. This requires additional detection and notification mechanisms for discovering such errors in structure and actuality of the metadata.

Impact analysis is a feature which enables administrators to evaluate the impact of potential changes in the data warehouse system before they are actually executed. For example, changes in the schema of sources may have consequences for transformation rules: type mismatches, violations of referential integrity, etc.

4.1.4 Repository Architecture

In Figure 2 we show the tool classes involved in metadata management. The repository is controlled by the *metadata manager* which provides the functionality of a database management system (including persistent data storage, concurrency control, recovery), change management operations, and tool access interfaces. These interfaces have to cover also the interaction of users with the repository which is performed by means of software as well. The *user access tools* offer, besides browsing, navigation and querying also editing facilities for manually entering the metadata elements and the relationships between them.

Administration tools may use metadata for system integration, for improving data quality, and enforcing security mechanisms. *Development tools* need metadata for the design of new applications while *analysis tools*, e.g., improve their results by means of metadata or record report structures, etc.

Figure 2 depicts also an interaction between the metadata manager and the data warehouse since certain information for deriving metadata may be stored in the data warehouse itself. Considering as example data quality parameters, some values needed to derive them may belong to each record in the data warehouse (e.g., probabilities regarding the accuracy of each birthdate) and are thus stored in the data warehouse.

Regarding the enterprise-wide architecture of repositories, a single repository for managing all kinds of metadata would be the first choice. This corresponds to *centralized metadata management*: metadata is uniformly and consistently managed, and accessed by all possible consumers. However, centralization does not always work in reality. The reasons are multiple: either the historical evolution of various departments may have implied an asynchronous development of repositories, or political and organizational aspects do not allow to physically manage a single repository within the enterprise. Furthermore, different tools with divergent data models, following diverse representation formats and standards have to be inevitably used for achieving the numerous tasks of data warehousing. The consequence is a multitude of proprietary metadata storage models, redundant storage of metadata in different repositories or, worst of all, no repository at all. In this case, there are two alternatives to handle metadata management: to choose either a totally decentralized or a federated approach.

Decentralized metadata management tries to cope with this situation by simply mutually importing metadata (based on certain interchange standards) wherever necessary.

Federated metadata management strikes a trade-off between the advantages of centralization and those of local control. On the one hand, it provides a global conceptual view of metadata existing in an enterprise. On the other hand, repositories may be further on maintained individually, with different access rights enforced on them and with different tools using them.

The content and structure of a repository is directly related to the modeling of the system it belongs to. It is widely recognized that (at least) four levels are required for modeling complex information systems. Each level contains the modeling constructs (or the modeling language) used to define the information on the level below. On level 0 are the actual data items (e.g., the customer data). The levels above contain the metainformation: level 1 contains metadata (e.g., the database schema), level 2 specifies the schema used to store the metadata (the so-called metadata schema). Usually, level 2 also includes common modeling languages like UML. Level 3 contains the metametamodel that unifies the different modeling languages specified on level 2. The conceptual schema of the metadata repository is situated on level 2, often called *metamodel* or *information model*. The application context data warehousing and the three ways of organizing metadata management mentioned above strongly influence how the metadata schema(s) look like (see Section 5).

4.2 Repository and Exchange Standards

Based on the functional and architectural requirements for data warehouse repositories stated above, we now turn to relevant standards for repository systems and for metadata exchange.

4.2.1 Repository Standards

Standards for repositories try to establish reference architectures to be followed by repository vendors and make these products uniformly usable in different application contexts, e.g. by incorporating the four level paradigm mentioned above.

IRDS The IRDS (Information Resource Dictionary System) standard, developed by ISO/IEC [18] addresses the requirements and architecture of a dictionary system. An Information Resource Dictionary (IRD) is defined as a shareable repository for the definition of information resources relevant for an enterprise [1]. This may include information about:

- *data* needed by the enterprise;
- the computerized and possibly non-computerized *processes* which are available for presenting and maintaining such data;
- the available physical *hardware* environment on which such data can be represented;
- the organization of human and physical *resources* which can make use of the information;
- the *human* resources responsible for generating that information.

PCTE The PCTE (Portable Common Tool Environment) standard from ECMA (European Computer Manufacturer's Association) [9] and ISO/IEC describes the basis for a standard software engineering environment, including a repository and communication between tools.

The most important aspect of PCTE with regard to the process of constructing and integrating portable tools is the provision of an object base (the repository) and a set of functions to manipulate the various objects in it. The basis for the Object Management System is derived from the Entity Relationship (E/R) model and defines objects and links as basic items of a PCTE object base. All entities in the object base are typed. Type rules are defined for objects, links and for attributes.

The PCTE object base may be split according to a number of schema definition sets (SDSs). Each SDS provides a consistent and self-contained view of the data in the object base. A process at any time views the data in the object base through a working schema. A working schema is obtained as a composition of SDSs in an ordered list.

4.2.2 Exchange Standards

Standardization of metadata exchange is one basic requirement for interoperability. The following exchange standards partly rely on the representational standards XML, OIM, UML and MOF which are described in Section 5.

CDIF The CDIF (Case Data Interchange Format) family of standards, standardized by the EIA (Electronic Industries Association)[4] and adopted by ISO, defines the structure and content of a transfer to exchange metadata between two CASE tools. The CDIF standards cover many of the important modeling techniques used by CASE tools today (e.g. data-flow modeling, data modeling, physical relational data base design, etc.). An actual transfer of metadata between two tools is usually via files, an IDL (Interface Definition Language) binding compliant to the CORBA standard of OMG (Object Management Group) is also available.

MDIS MDIS (Meta Data Interchange Specification) [30] is an exchange standard from the Meta Data Coalition (MDC). MDIS provides a metamodel that addresses the main types of data models (relational, hierarchical, network), and a standard import/export mechanism that enables the exchange of these metadata objects between tools. MDC plans to integrate MDIS with OIM.

XMI The XMI (XML Metadata Interchange) is an OMG standard to exchange MOF based metamodels. The standard consists of:

- a set of XML Document Type Definition (DTD) production rules for transforming MOF based metamodels into XML DTDs;
- a set of XML Document production rules for encoding/decoding MOF based metadata;
- design principles for XMI based DTDs and XML streams;
- concrete DTDs for UML and MOF.

Because XMI is based on XML, it can be used for metadata interchange with and between non-CORBA based metadata repositories and tools.

[4] http://www.eigroup.org

XIF The XIF (XML Interchange Facility) is an MDC standard to exchange OIM models. Like XMI, XIF is based on XML. There are no details available from MDC about XIF.

4.3 Commercial Metadata Management Solutions

For all components in the architecture of a data warehouse system, commercial tools exist in abundance[5]. Each of these tools is a metadata consumer and/or producer. The creation, maintenance, sharing and coordination resp. integration of their metadata requires a lot of effort. Several commercial offerings for metadata management are based on a centralized strategy, dominating, however, are decentralized proposals.

4.3.1 General Purpose Repositories

The centralized approach to data warehouse metadata management can naturally be realized by a general-purpose standalone repository. According to a recent Gartner Group survey [13], there are two vendors (CA/Platinum, Viasoft) selling repository products which are classified as "leaders" in this market segment, and two vendors (Unisys, Microsoft) are classified as "visionaries". We will present the "stand-alone" repository solutions from Viasoft, Unisys and the offerings from Platinum/Microsoft, who recently joined forces in the repository area.

UREP The Universal Repository (UREP) from Unisys corporation[6] is an object-oriented, extensible repository.

The repository services provided by UREP are based on the service sets defined by the PCTE standard. These services include:

- *version control*: maintaining a history of information;
- *transactions*: enabling users to define and enact transactions (short and long transactions);
- *user and session management*: identifying a repository user, defining access rights etc.;
- *metadata service*: extending the information model.

Two types of models exist within UREP: The *Technology model* describes repository types and operations that represent generic software technologies (database paradigms, business analysis, etc.). The repository types and operations that a particular tool uses, belong to the *Tool model*. Both types of models are extensions of the UREP information model. The metadata service provides mechanisms for:

- defining object modeling constructs such as types, relationships, integrity rules etc.;
- extending the UREP information model;
- reusing object types and operations that are defined in the repository, and
- extending repository functionality by defining subtypes and by overriding inherited operations.

UREP has been licensed by different companies to be included in their product palette, e.g., by Sybase corporation and BEA systems.

[5] An excellent resource is Larry Greenfield's site at http://pwp.starnetinc.com/larryg/index.html.
[6] http://www.marketplace.unisys.com/urep/

Microsoft Repository The Microsoft repository[7] is an object-oriented, extensible repository, consisting of two major components:

- a set of APIs based on Microsoft's Component Object Model (COM) to describe information models and
- a (relational) repository engine (either Microsoft SQL Server or Microsoft Jet, the database system in Microsoft Access) which serves as storage component for these models.

The Microsoft repository is targeted towards software vendors and users wishing to support the management of metadata in a variety of scenarios including software development and data warehousing, using OIM (see 5.2.2) as the underlying model.

Microsoft Repository 2.0 ships with Visual Studio 6.0 and SQL Server 7.0. Microsoft has a collaboration agreement with Platinum to port the repository to non-Microsoft platforms. According to Microsoft, there are more than 75 third parties shipping tools that use the repository for their metadata management.

Platinum Repository The CA/Platinum repository [8] currently exists in two versions:

- *Repository/MVS* is a mainframe-based repository, first released in 1988, using DB2 as its storage platform.
- *Repository/OEE* (Open Enterprise Edition) is a client/server solution which stores its data in Oracle, Sybase or Microsoft SQL Server.

Both versions provide models based on the E/R approach. The Platinum repository provides full bi-directional interfaces to many CASE tools, including ERwin, Bachman, Sterling/Cayenne and Oracle Designer. Scanners (a specialized set of tools for parsing and importing file definitions, SQL statements etc.) are available for all popular languages.

Platinum and Microsoft have a collaboration agreement, giving Platinum the right to port the Microsoft repository to non-Microsoft platforms and to leading RDBMSs (Oracle, DB2, Sybase). According to Platinum, the next release of Repository/OEE, called the PLATINUM Enterprise Repository V2.0, will contain all the functionality currently available in Repository/OEE and will be based on the Microsoft-PLATINUM Repository V2.0 engine.

Rochade Viasoft's Rochade repository[9] uses an object/network-based database structure which handles objects and links between them. The following four basic building blocks are maintained:

- *item types* describing groups into which similar items are categorized,
- *attribute types* describing aspects of an item,
- *link types* describing relationships between items, and
- *rule types* describing processes associated with an item/link/attribute or another rule.

[7] http://msdn.microsoft.com/repository/
[8] http://www.platinum.com
[9] http://www.viasoft.com

The heart of the Rochade repository is the repository information model (RIM). A RIM provides a scoping mechanism for accessing the repository and defines the details necessary for supporting Rochade subject areas, projects and external tools. The RIM of Rochade provides unlimited extensibility. Its data architecture recognizes five data levels, each level describing and defining the level beneath it. This is not congruent with repository standards like PCTE or IRDS which both stipulate a four-level architecture.

As the competing Platinum repository product, Viasoft's Rochade offers rich interface support to other tools, e.g., Bachman, System Architect, ERwin.

4.3.2 Decentralized Tool-specific Metadata Management

As strategies for decentralized management of metadata we can distinguish between metadata *exchange standards* defined or adopted by vendor coalitions and *"open" product APIs*. The exchange standards require that the coalition members agree on a common sub-model to exchange metadata between their tools. Each vendor is free to specify additional metadata for his own products usage. IBM is one example for a vendor pursuing this approach. In the second case, a vendor offers an API that enables other parties to import and/or export metadata from their products. Examples of vendor interfaces that enjoy wide support from a variety of third-party tool vendors include Ardent Software and Informatica.

Ardent Software Ardent Software[10] offers several products for data warehousing including the product line of the former Prism Corporation (Prism Warehouse Executive and Prism Warehouse Director), and also Ardent's DataStage product. Warehouse Executive is an ETL-tool which generates code (Cobol, Java, ABAP/SAP). Warehouse Directory is the metadata store. A key feature is the integrated metamodel that contains business, technical, operational and quality data. Third-party tools can also access the repository metadata in order to facilitate end-user queries and operational data management.

Ardent announced the MetaConnect initiative, allowing users to exchange metadata among different warehouse tools. The initiative is based on so-called MetaBrokers. The core of each MetaBroker is built by a patented technique which enables the decomposition and recomposition of metadata into simple units of meaning. This semantic translation facilitates the exchange of metadata between extraction, transformation and loading, business intelligence and data modeling tools. MetaBrokers use a tools standard import/export file for metadata exchange. Metadata exchange between two tools does not rely on a central repository, but only on a MetaBrokers transient storage. When an exchange has been completed, the transient storage is deleted. Currently, MetaBrokers are available for Erwin, ER/Studio, Impromptu, and Business Objects. Ardent also announced the MetaStage product as a materialized repository for their MetaBrokers. The Ardent activities can be seen as a first step towards federated metadata management.

Informatica Informatica Corporation[11] sells the ETL-tool PowerMart and its add-on PowerCenter. PowerMart has its own metadata repository which coordinates and drives a variety of core functions including data extraction, transformation, loading and management. The PowerMart Repository Manager is used to create and maintain it. PowerCenter supports the integration of different PowerMart Repositories.

The Metadata Exchange (MX2) Architecture provides a set of application program interfaces (APIs) which can be used by OLAP, query and access tool vendors to integrate

[10] http://www.ardentsoftware.com/
[11] http://www.informatica.com/

their products with Informatica's open metadata repository. MX2 is based on UML for information modeling, and COM for object interoperability. According to Informatica, the MX2-Architecture is compatible with Microsoft's OIM. The MX2 APIs allow read and write access to the underlying repository. Different vendors announced their support for the MX2 architecture, e.g. Brio Technology, Cognos, Business Objects and others.

4.4 Research Contributions

The management and exchange of metadata is also in the focus of various research projects and has led to tools which have been or might be applied in the data warehousing context. We mention three example systems and discuss related approaches from the areas of ontology management and general information integration concerning overlaps and similarities with requirements in data warehousing.

4.4.1 ConceptBase

The ConceptBase[12] deductive object base system [19] developed at Aachen University of Technology is a client-server database system for conceptual information, i.e. metadata. The foundation of its representation language Telos is a simple data structure which uniformly represents objects, classes, meta classes, attributes, and class-instance as well as subclass-superclass relationships. Telos also provides for a logical sublanguage based on many-sorted first order logic. The main predicates available allow to express the basic object-oriented structuring principles plus certain arithmetic and aggregation operators. Formulas in this language are employed for specifying deduction rules and integrity constraints and constitute the main building block of the query and view language of the system.

Since the class-instance relationship is stored explicitly in ConceptBase, an object is allowed to have multiple classes. Moreover, classes may be instances of meta classes, meta classes may be grouped into meta meta classes, and so on. There is virtually no limit in this hierarchy though most applications do not go beyond 4 levels. Integrity constraints, deductive rules and queries can be formulated at any of these levels. The view language of ConceptBase forms the basis for view-specific C++ API's which allow automatic initialization and incremental maintenance of view data derived from the object base contents and managed in client applications.

In a number of different application settings, ConceptBase was able to support various data representation frameworks like the IRDS standard, but also showed the necessary flexibility for dedicated and application-specific modeling schemas. [21] gives a comprehensive overview on applications and experiences with the system ranging from software and requirements engineering to business process modeling projects and terminology management. A concrete application of ConceptBase in the data warehouse project DWQ is described in Section 5.

4.4.2 H-PCTE

H-PCTE[13] is a PCTE implementation maintained at the University of Siegen. Versioning of objects is supported explicitly and even schema level operations, i.e. the dynamic creation and change of types, are allowed and complemented by appropriate object modifications. While the PCTE standard is restricted to an API for navigational access to objects only, the H-PCTE system offers both a set-oriented algebraic [14] as well as an SQL-like query language named P-OQL.

[12] http://www-i5.informatik.rwth-aachen.de/CBdoc/
[13] http://pi.informatik.uni-siegen.de/pi/hpcte/

H-PCTE is, like ConceptBase, a main-memory based system. It is both possible to reserve fragments of an object base for exclusive access and uploading in the address space of a single application, or to work on a shared adress space with several applications. Java clients can access H-PCTE via the Java-API which employs an additional server process for bridging application and repository and in particular for providing a notification mechanism: observer tags are attached to objects inside the repository which are of interest to an application; changes are then steadily reported back.

H-PCTE was mainly applied as a repository component for software engineering environments.

4.4.3 Lore

The Lore system[14] being developed at Stanford University is a so-called light-weight repository in the sense of a very simple basic data model and at least in the beginning of the project with read-only and single-user functionality [25]. The first data model Lore supported was OEM (Object exchange model) which basically consists of nodes and labeled links between them. The increasing importance of XML led to a replacement of OEM in Lore. The main difference between both languages in fact can be reduced to a missing analogue for DTD's in OEM which however was consistent with the original idea of schemaless data. As soon as a DTD exists, it imposes schema restriction on the respective data.

The query language Lorel is an OQL adaption and emphasizes the importance of path expressions derived from the link labels between objects. All other objects reachable from one object by traversing paths with the same label sequence are considered as solutions satisfying such expressions. An important feature of Lore is the DataGuide which provides a facility for schema reconstruction or data condensation.

4.4.4 Related Approaches

Ontology management [11] is a classic field of Artificial Intelligence where structural information (including concepts, their definitions, relationships and properties) is collected and maintained - often in domain-specific contexts - in order to support various applications like natural language processing or machine learning. With regard to data warehousing, ontologies are in particular important as a means to organize business items and the descriptions of their usage in the various applications. A variety of ontology management systems have been developed, however they are used only in very general contexts. Most of them employ frame-based languages for describing concepts and additional powerful assertion languages. Two very prominent representatives are the Cyc Knowledge Server system [23] and the Ontolingua Server[15]. The Cyc project's main goal is to establish a huge global collection of commonsense knowledge which is made available through the Knowledge Server. Among various other application areas, Ontolingua is currently employed to construct a World Fact Book, containing a substantial collection on all aspects of knowledge about the world's nations.

While *information integration* actually is an old topic, it has received growing interest with the explosion of Internet and Web technologies during the last years. Schema integration becomes even more relevant if the data sources to be integrated have heterogenous formats or are only weakly structured. Projects like TSIMMIS [12] and Garlic [32] propose a *virtual* integration approach realized by source specific wrappers solving the format mismatch, and mediator components responsible for central integration and query services. While data

[14] http://WWW-DB.Stanford.EDU/lore/

[15] http://ontolingua.stanford.edu

warehouses *materialize* the data to be integrated, the way of managing schema descriptions and related information about the sources in respective system catalogs (Lore in the case of TSIMMIS) is relevant for data warehousing, too. In a similar way, the Information Manifold [24] serves as a bridge between several hundred information sources and end-users and employs an overall ontology for accessing the available contents.

A step beyond storing schema and source information is made by advanced web-site management systems which offer facilities for model-based generating of pages or even whole applications. This can be compared, e.g., to managing reports and their basic building blocks in data warehousing. Existing proposals, e.g., as developed in the ARANEUS [3] and STRUDEL [10] projects, usually differ with regard to the formal data model (and its degree of structuring). The descriptions to be stored in a repository comprise the skeletons of the final web pages, its relationship to the data sources, and transformations between them. Advanced web applications like hyperbooks [26] can give additional hints how to represent possible access paths to metadata and data as metadata, too. In fact, for metadata based access to a data warehouse, web technology is indispensable.

4.5 Résumé

The presented approaches to tackle metadata management in general and for data warehousing as one special application show that standardization efforts obviously neither led to a unique proposal concerning architecture and exchange mechanisms nor influenced decisively the various tool implementations.

On the repository-side, the Microsoft repository is of increasing importance due to the dominating role of Microsoft in software development and standard application tools. Whether the cooperation with Platinum will make their concepts available for the Unix and mainframe world, will turn out in the future.

The research tools for metadata management feature specific benefits, which are even relevant for their commercial counterparts. For example, ConceptBase with its powerful logic-based language used for rules, integrity constraints and queries yields excellent analysis and inference facilities for metadata. Its data model has a clean formal semantics and is still the most flexible one concerning modeling layers and the definition of modeling constructs. The Java-based notification mechanism implemented in H-PCTE allows comfortable support for collaborative metadata editing and browsing tools, and even is important for handling federated metadata. Lore is one of the first XML-oriented repositories with its main intention to solve integration problems on semistructured data. In this context, also classical database problems like query optimization and view maintenance based on a declarative query language were explored.

Two important problems with metadata management concern their maintenance and integration. Metadata (like data) lose their worth if they do not reflect the actual state of the system and enterprise world. The maintenance of metadata is only in part a question of interoperability between software systems, in particular for generated metadata from, e.g., systems or process design. On the other hand the maintenance problem requires organizational concepts (responsibility, authorization, etc). This holds for manual documentation and also non-IT metadata. The idea of one global metadata base managed by a central repository becomes unrealistic at the latest when one crosses enterprise or organization borders, often even earlier. Proprietary metadata management solutions provided by tools involved in the various data warehousing tasks will continue to exist. Therefore, federated and decentralized metadata management naturally yields *meta*data integration and coordination tasks. These tasks are not supported in a satisfactory way by the current data warehouse tools, and even the general repository systems do not offer access mechanisms general enough for platform independent employment. Bi-directional toolspecific interfaces dominate. Open interfaces allow at least a replication of tool-specific metadata in a global repository. The ideas emerging from the various research projects on information integration are highly relevant for developing concepts for federated metadata management.

The main barrier for integration, however, is a missing general metadata schema for data warehousing as discussed in the next section.

5 A Data Warehouse Metadata Schema

While most of the repository requirements discussed above are more general by nature, the repository metamodel or metadata schema is truly application specific, i.e., dedicated to data warehousing in our case. It captures all metadata types and relationships between them required by the involved tools and human users. How general the metamodel is within the domain depends on the given enterprise-wide repository architecture. For the decentralized approach we have a variety of tool-specific schemas, usually overlapping with regard to several general aspects, like source schema descriptions, but without syntactic and semantic adaption. Since the main goal in a company should be directed to the central or at least federated case of metadata management where an *overall global schema* is necessary, we address in the following general categories, formalisms, standardization efforts and proposals for metadata schemas which cover all aspects relevant for data warehousing.

5.1 Metadata Classification

We present six dimensions for classifying metadata that give hints for the required main elements of the metadata schema. The dimensions themselves can be understood as direct part of this metamodel, orthogonal to or even refined by the domain-specific aspects. Their general scope, however, makes them also suited as metametamodel.

5.1.1 Criterion "Type"

One first distinction regards metadata about *primary data* in the data warehouse system and about *data processing*, that means, metadata for processes running within the data warehouse.

- **Metadata for primary data.** Primary data consists of all data managed by the data sources, the data warehouse, the data marts, and the applications. Thus, the corresponding metadata includes information related to the structure of data sources, data warehouses, and data marts. In this context, we distinguish between metadata concerning the entire schema (e.g., schema description, statistical values as, e.g., the number of entries in the database) and metadata associated with parts of the schema. Examples include quality attributes that specify the credibility of single attributes (e.g., birthdate). Furthermore, the database schema may be extended with additional attributes that associate various values to individual entries in the database, as e.g., the updating date of records, or information regarding the data collection (what, where, who, when and why). Both the attributes and the values provided for each entry can be understood as metadata. Code tables are other classical examples where no clear distinction exists between metadata and data. They relate codes used in everyday acquisition of data with their textual descriptions (e.g., 0 stands for male and 1 for female).

- **Metadata on data processing.** This is information associated with data processing: information regarding the loading and refreshment process, the analysis process, and inherently the administration. Examples are rules specified for data extraction, transformation, and aggregation which are defined by means of executable specification languages. They have to be accompanied by a textual explanatory description, usually in natural language. According to the operation type performed, rules may be further classified as filter, joiner, aggregator, etc. Process metadata also includes logfiles and schedules for establishing the time and order of the execution of (parts of) processes.

A special category constitutes metadata concerning the organisation of the enterprise. This category may be both considered as belonging to the first class (when organizational information is simply handled as primary data) or as a stand-alone class. Organizational metadata includes administrative data, information related to the staff of the enterprise, as e.g., user rights to access the data warehouse, data sources, and data marts.

5.1.2 Criterion "Level of Abstraction"

In analogy to database design steps, knowledge may be provided on three levels of abstraction: conceptual, logical and physical. The conceptual perspective includes the entire description of the business using natural language. For example, main business entities like "customer", "partner" are defined and their features and relationships with other entities explained. Rules that govern extraction/transformation are described by means of natural language in order to be understandable for any system user. Also information related to the use of the system, about predefined queries, views and existing analysis applications are provided at this level.

Logical metadata maps the conceptual perspective to a lower level that includes for example the relational schema of the databases, the description of extraction/transformation rules in pseudocode (or using a mathematical language), etc.

The physical perspective provides the implementation level. It contains the corresponding SQL code realizing certain business rules, index files of relations, and code of the analysis applications. When the distinction between logical and physical metadata is not justified either for end-users or technical users, the two levels may be merged into a single one.

5.1.3 Criterion "User View"

An important classification criterion is related to the informational objective of the metadata. The same information and structure may be seen from different perspectives, depending on the users that need it. As a consequence, metadata may be divided into subclasses that satisfy the interest of certain user groups. For example, some metadata may be relevant for certain departments, for knowledge managers (needing additional information beyond standard reports and OLAP results), for business users, or for more technical users (e.g., data warehouse administrators, analysis application programmers). In this context, a common distinction is *business* (or application-driven) versus *technical* metadata.

Business metadata is needed by end-users to better understand the application and thus better use the information system; it comprises application-specific documentation (user profiles, access maps, usage tips, navigational aids), business concepts and terminology, details about predefined queries and reports. Also, context information like measurement units specification (currency, length), date format (American or European convention) or dictionaries, thesauri, and domain-specific ontological knowledge are considered as metadata.

In contrast, technical metadata is used by database administrators who develop and maintain the system and by analysis application developers. Classical examples are the data dictionaries of the sources, data warehouse, data marts, and the code of data transformation rules.

Metadata classes defined on the basis of user views (in particular business and technical metadata) are not disjoint; they provide different extracts of the same collection of metadata, and may overlap as well.

5.1.4 Criterion "Origin"

Another dimension takes into consideration the origin of the metadata: the tool that produced it (the ETL-component, a CASE-tool used during warehouse design), the source that provided it (e.g, the data dictionary of the operational systems, of the data warehouse or data marts), or the system the metadata has been imported from, etc. Metadata may also be produced by certain users (business user, database administrator) which should be individually identifiable.

5.1.5 Criterion "Purpose"

As a counterpart to the previous criterion, a "purpose" or "use" criterion may be used to classify metadata according to activities such as extraction or transformation, building a multidimensional view, data mining, reporting, etc. However, the distinction between the different classes is vague since some metadata (e.g., schema description) may be used for most purposes: administration and maintenance, refreshment and analysis.

We may coarsely identify (in an orthogonal dimension) metadata for informational and for controlling purposes. Since all metadata may serve the informational purpose, the latter category may possibly be a subset of the former. The controlling purpose class may be divided into subclasses with regard to the operations applied to them: the metadata is either only read (schedules for processes) or read and directly exploited (transformation rules used by metadata-driven engines), or both read and also updated (logfiles).

5.1.6 Criterion "Producing/Consuming Time"

With respect to the time metadata was captured or generated, we can distinguish between three categories:

- designtime collected metadata (e.g., schema definition of sources and of the data warehouse, access rights, transformation rules, etc.),
- buildtime generated metadata (e.g., logfiles, data quality attributes, particularly data tracking),
- usetime generated metadata (e.g., usage statistics).

In analogy, the classification according to consumption time specifies when the metadata may be needed: at designtime, when the system is developed (dictionaries, CASE tools metadata, reverse engineering tools metadata, data mining metadata), at buildtime (schedules, transformation rules, data quality rules), or at usetime (configuration files, OLAP metadata).

5.2 Standards and Reference Models

Representing the above criteria within a metadata schema requires a basic representation language and, ideally, a given framework or schema kernel. We look at the most important general standards and present the existing concrete data warehouse specific reference models which are centered around the former. Two groups are influencing the scene: OMG (Object Management Group) and MDC (Meta Data Coalition). The existing commercial approaches are still undecided between adaption and proprietary solutions.

5.2.1 General Standards

UML The UML (Unified Modeling Language) standard, adopted by OMG, is a language for specifying, visualizing, constructing, and documenting artifacts of software systems, as well as for business modeling and other non-software systems. UML fits well into the four-level architecture introduced in Section 4, constituting *Level 2* (modeling languages). The corresponding *Level 3* of this architecture is occupied by the Meta Object Facility (MOF). The MOF standard from OMG defines a metametamodel with sufficient semantics to describe metamodels, UML being one of those. On the same level as UML is OMG's XMI-standard to exchange any MOF-based model between tools.

The vocabulary of UML encompasses three kinds of building blocks [7]:

1. *Things* - comprising structural things (class, interface etc.), behavioral things (messages, states), groupings (packages), and annotations (notes).
2. *Relationships* - comprising Dependency, Association, Generalization and Realization.
3. *Diagrams* - enabling the graphical presentations of things and their relationships. UML includes nine types of diagrams, namely Class diagram, Object diagram, Use Case diagram, Sequence diagram, Collaboration diagram, Statechart diagram, Activity diagram, Component diagram and Deployment diagram.

XML The XML (Extensible Markup Language), accepted by the World Wide Web Consortium (W3C) as a recommendation [33], is a standard for semistructured data and a subset of SGML (Standard Generalized Markup Language (ISO 8879)). XML is about to become the successor of HTML (Hypertext Markup Language), the current language for defining web pages. Unlike HTML, an XML document does not include presentation information. Visual presentation of XML documents is achieved with technologies such as XSL (Extensible Style Language) or CSS (Cascading Style Sheet). XSL documents are themselves well-formed XML documents.

XML is a metamarkup language allowing to define the tags actually needed in a given context, in contrast to HTML and similar markup languages which define a fixed set of tags describing a fixed number of elements. XML allows metadata markers to be embedded within a document by matching start and end tags, such as <name> and </name>, for marking up the information in between. Therefore, document designers can provide metadata that will help people find information and help information producers and consumers communicate with each other based on a common vocabulary. XML is a low-level syntax for representing structured data. An XML document structure is defined with a DTD (Document Type Definition), which can be directly included into the document, or can be stored externally through a reference to another one.

Others A number of other standards were proposed by prominent standardization bodies but lack a broad practical acceptance. NIST (National Institute of Standards and Technology), e.g., adopted the *IDEF (Integration Definition)* pair of standards. *IDEF0* is used to model functions together with data and objects interrelating them. The purpose of *IDEF1X*

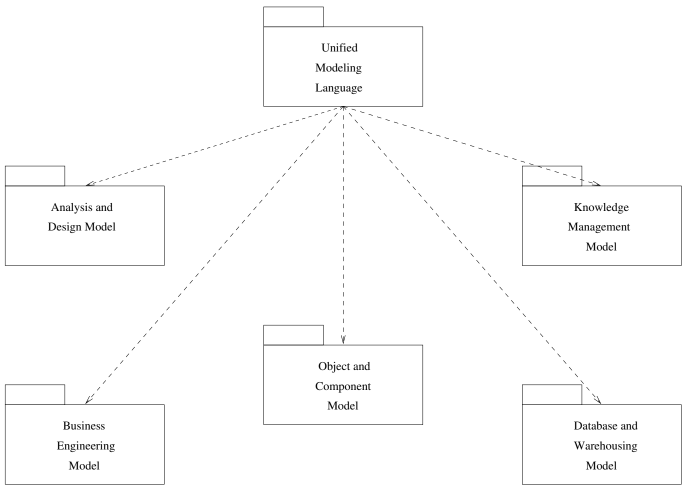

Figure 3: Open Information Model

is to model data, using as basic constructs entities, relationships and characteristics (of entities). Both standards have also been adopted by IEEE. Similarly, the ISO/IEC 11179 family of standards gives concrete guidance on the formulation and maintenance of data element descriptions and semantic content (metadata) in order to define data elements in a consistent, standardized manner.

5.2.2 Warehouse Related Standardization of Metadata Schemas

OIM for Data Warehousing The Open Information Model (OIM) is a Software Engineering standard, initially developed by Microsoft and now managed by MDC[16]. MDC is "a not-for-profit consortium of vendors and end-users whose goal is to provide a vendor-neutral and technology-independent specification of enterprise meta data". OIM is a specialization of the abstract concepts of UML into domain specific sub-models that describe metadata. Figure 3 shows the dependencies between the sub-models of OIM in UML notation[17]:

- The *Analysis and Design Model* covers the domain of object-oriented modeling and design of software centric systems. The core of the model is the *UML package*, describing version 1.3 of the UML. Other packages are *UML extensions* (describing the presentational aspects of UML elements like fonts, coordinates etc., and other general-purpose additions to the UML package), *Common Data Types* (Date, Time etc.) and *Generic Elements*, describing a set of general-purpose classes that are relevant across diverse information models.

- The *Object and Component Model* comprises the *Component Description Model*, covering the different component development life-cycle deliverables. The model is divided into three distinct layers: specification, implementation (currently not defined), and executable. The package covers the various aspects of the implementation of a component (defined as "a software package that offers services through interfaces"), but does not respect the specifics of any particular programming language.

[16] http://www.mdcinfo.com

[17] The reader should not be confused by the fact that OIM uses the UML in three distinct roles: as modeling language to design and visualize OIM itself, as main part of the Analysis and Design model subject-area to express object-oriented models, and as core-model of OIM from which sub-models inherit concepts.

- The *Business Engineering Model* provides all the necessary metadata types to describe the goals, organization, and infrastructure of a business as well as the processes and the rules that govern the business.
- The *Knowledge Management Model* comprises the *Information Directory Model*, currently not defined, and the *Semantic Definitions*. This package accommodates conceptual models of user information. Specifically, the package holds descriptions of semantic models and their relationships to the underlying database schema. These connections enable a user to query a database using English sentences. The Information Directory Model will provide metadata types to define a controlled vocabulary to classify business information.
- The *Database and Warehousing Model* (DBDWM) comprises the following packages:
  1. Database schema elements describe information maintained in relational database systems. This package contains information about schema elements (tables, views, queries etc.) and database specific data types. The concepts of this package are modeled after the ANSI SQL-92 standard, with selected extensions supported by popular relational database vendors.
  2. OLAP schema elements describe multidimensional databases. The package allows the description of *cubes* (the basic component in multidimensional data analysis), *dimension hierarchies* (for roll-up and drill-down operations), and *Aggregations* (precalculated roll-ups of data stored in a cube).
  3. Data transformations elements cover relational-to-relational transformations. A *transformation* maps from a set of source objects into a set of target objects, both represented by a *transformable object set*. Transformations can be packaged into groups. There are three levels of grouping: The first uses a *transformation task*, which describes a set of transformations that must be executed together. The second level is a *transformation step*, executing a single transformation task. Steps are used to coordinate the flow of control between tasks. The third level is a *transformation package*, consisting of a set of transformation steps.
  4. Record oriented legacy databases describe information about data maintained in files or non-relational (legacy) databases. The package does not cover detailed logical-to-physical mapping or information about any of the concrete file systems or databases that may use record structures. These will be added later in subsequent packages.
  5. Report definitions will represent information necessary for data reporting tools and their relationships to the systems they report on. This package is not specified yet.

A Metadata Schema based on the Zachman Framework A very global view on the architecture of information systems was proposed by John A. Zachman [34]. The units of his framework can also be understood as organization scheme for all kinds of metadata involved in building and using an information system and have therefore become widely recognized during the last years.

The Zachman framework is organized as a matrix which describes the various ways the stakeholders of an organization view the business and its systems. The rows of the matrix represent the five different basic roles that people play in the creation of a product:

1. *The planner* is concerned with positioning the product in the context of its environment, including specifying its scope.

2. *The owner* is interested in the business deliverable and how it will be used.

3. *The designer* works with the specifications for the product to ensure that it will, in fact, fulfill the owner's expectations.

4. *The builder* manages the process of assembling and fabricating the components in production of the product.

5. *The subcontractor* fabricates out-of-context (and hence reusable) components which meet the builder's specification.

The columns of the matrix represent the six important dimensions of the architecture of an information system, namely data, function, network, people, time and motivation. The Zachman framework supports commonly adopted techniques like entity relationship diagrams, data flow diagrams and the like - each of these techniques is placed in context with the others to provide a complete picture of the information system to be modeled.

Based on the general Zachman Framework, [17] develops a framework for managing enterprise knowledge. It consists of four layers dealing with modeling of product aspects, modeling of enterprise aspects, modeling the engineering of an enterprise, and finally a repository layer which reflects the lower layers and their relationships to each other. Data warehousing is understood as one central, first step from information resource management towards a comprehensive knowledge management environment. The Zachman view is demonstrated to be useful for defining

- a data warehouse strategy in the form of principles to guide behavior (consisting of a statement, a rationale for it, and its major implications), and

- a methodology for building and extending a data warehouse.

The methodology is complemented by a metadata schema which is organized along the dimensions (columns) of the Zachman Framework. For each dimension it contains modeling constructs across all relevant perspectives: the data or entity dimension submodel, e.g., ranges from business entities (owner perspective) down to data warehouse tables (builder and designer perspective); the motivation dimension constructs covers business goals as well as data warehouse service level objectives.

The authors claim their model not to be complete concerning all aspects of data warehousing, but to serve as a launching point for designing a metaschema tailored to a specific company's requirements and priorities.

OMG In September 1998, OMG has issued an RFP (Request for Proposal) for "Common Warehouse Metadata Interchange" [27] with deadline September 1999. Its objectives are:

1. Establishing an industry standard specification for common warehouse metadata interchange;

2. Providing a generic mechanism that can be used to transfer a wide variety of warehouse metadata;

3. Leveraging existing vendor-neutral interchange mechanisms as much as possible.

Proposals are required to cover a complete specification of the syntax and semantics needed to export/import warehouse metadata (including APIs) and the common warehouse metamodel. Proposals must be compatible with MOF/UML/XMI.

MDAPI The Multi-Dimensional API (MDAPI) version 2.0 [31] is a standard defined by the OLAP Council in 1998.

The OLAP Council was established in January 1995 to serve as an industry guide and customer advocacy group. Members of the council are IBM, Oracle, Sun, Platinum, Hyperion Solutions, NCR, Cognos, Business Objects, and others.

The defined API, among other things, provides metadata functions for OLAP multidimensional databases. MDAPI is principally a specification only. The OLAP Council publishes the specification, the members of the OLAP Council provide platform dependent implementations of MDAPI. Version 2 is a read-only API and does not offer a mechanism to modify data in a connected schema or the structure of a metadata schema. The API is available for COM and Java.

The API access to metadata requires an initialization by creating so-called *connection* objects. The basic information units are *members* which can be understood as abstractions of cells plus related additional features described by *properties*. A special subtype of members are *measures* with predefined attributes like scale, precision etc. At runtime, members and their properties receive concrete value assignments corresponding to their specified data types. Members are organized in *dimensions* consisting of one or more *hierarchies* with different *levels* of detail and aggregation. Special types of dimensions are available for time and measures.

MDAPI offers two basic kinds of queries, namely *Cube* for data access and *MemberQuery* for metadata retrieval.

5.2.3 Vendor Specific Metadata Schemas

Vendors of repository products are in a difficult position selecting the "right" metadata schema to support; basically, they have the option to join OMG or MDC, they can push their own "standard", or try a mix of the former strategies.

Platinum The strategy of CA/Platinum is very unclear: on the one hand, they have a collaboration agreement with Microsoft, clearly indicating their support for OIM; on the other hand, they stated their intent to submit a proposal to the OMG's warehouse standard. Finally, Platinum's own repository model has been extended to allow definitions of source, target and transformation rules used in building data warehouses and data marts. The model has no constructs for metadata about business aspects of a data warehouse, e.g. dimensions, business concepts, business key figures.

Rochade The strategy of Viasoft is at little bit less vague. Rochade offers an elaborate metadata schema for data warehousing, DW-RIM (Data Warehouse Repository Information Model). On the other side, they will provide an implementation of OIM (Viasoft is member of MDC).

DW-RIM covers both technical and business aspects of data warehousing. Specifically, DW-RIM supports the definition of transformation rules as well as business aspects of data warehousing. Transformation rules link source and target data elements and specify both the implementation as well as the business aspect of a rule. Business elements comprise aspects like available queries, reports, business terminology, dimensions and key figures.

Oracle The strategy of Oracle is quite clear, they push their own model and try to promote it as the OMG standard. They announced the *Oracle Warehouse Builder*[18] (OWB), delivering

[18] http://www.oracle.com/datawarehouse

a complete data warehousing solution. The OWB logical warehouse model consists of four sub-models:

1. The *enterprise data model* consists of information about tables, columns, foreign key joins etc.
2. The *warehouse data model* comprises contains the definition of data tables, summary tables, dimensions and hierarchies, extraction definitions and transformations.
3. The *software library* contains the list of sources that can be accessed by OWB. Additional sources for extraction can be defined using Oracle's Software Development Kit.
4. The *transformation library* stores the reusable formulas and expressions that perform transformations between objects; the transformation library contains a set of pre-defined PL/SQL functions and transformations.

Oracle announced the "common warehouse metadata standard" (CWM), which allows third-party software to integrate with the metadata repository and will be submitted to OMG.

IBM IBM is in a similar position as Oracle, offering their *Visual Warehouse*[19] solution and having stated their intent to submit a proposal to OMG. Metadata for both administrative and business users is integrated in the Visual Warehouse Information Catalog and is made available through an interface tailored to end-users, including the ability to navigate and search using business terms. Metadata exchange is supported via MDIS and IBM's own Tag Language format.

5.3 Research Contributions

Among the many research projects on Data Warehousing only a few actually concentrate on metadata aspects. Implicitly, all proposed data warehouse architectures and advanced system components for data warehouse tasks assume a repository component for storing necessary metadata. Examples are projects like Whips [22], Squirrel [15] and System 42[20].

5.3.1 DWQ

The ESPRIT Long Term Research Project DWQ[21] explores foundations of quality aspects in Data Warehousing. The project in particular proposes to link all data warehousing tasks to be executed with explicit quality information and store them in a repository.

While many results are primarily technical contributions of new techniques for the various data warehouse tasks, the most interesting efforts of DWQ in the metadata context stem from the data warehouse framework and quality-related activities [20]. One main decision consists in the adoption of two orthogonal views on data warehousing:

1. With respect to the level of abstraction, conceptual, logical and physical aspects can be distinguished: The *conceptual* perspective is directed towards the business model of information systems, while *logical* and *physical* view specify two levels relevant for realizing data warehouses.

[19] http://www.software.ibm.com/data/vw/
[20] http://www.forwiss.tu-muenchen.de/~system42/
[21] http://www.dbnet.ece.ntua.gr/~dwq/

2. With respect to the organizational focus, we have *operational* sources supporting certain departments of an enterprise, the data warehouse based on the overall *enterprise* model, and the analysis-oriented client side, mainly *dispositive* applications using data marts derived from the data warehouse.

Based on the combination of both perspectives, a metamodel for data warehouses was developed and refined to illustrate the relationship between quality requirements and data warehousing tasks. Due to the nature of the project, components for the usage side remain on the abstract level. In a detailed way the basic model was extended, e.g., to capture source and data integration. Specific quality dimensions were assigned to each data warehousing task and combined with an explicit quality model based on the Goal-Question-Metric approach.

5.3.2 Meta-FIS

The Information Systems Institute of the University of Münster works on a metadata management repository with main emphasis on management information support. In fact, the proposed ideas cover the usage aspects of data warehousing, in particular business reporting based on generated documents and business key figures [4].

As a general framework this project uses three views on management information systems. The *actor view* specifies the management personnel that has to execute certain controlling and management tasks (*task view*) and need to be provided with suitable *information objects*.

One result of the current activities is a metamodel for the information object aspects using a terminology that is rather application oriented than including the system or data source perspective. The main components of this metamodel are reports and business key figures, queries which are used to fill report elements and to compute key figures, and information objects represented as multidimensional structures.

5.3.3 MMDWE

Starting from the observation, that currently a comprehensive repository solution for all kinds of metadata relevant in Data Warehousing is not existing, the MMDWE project ('Metadata Management in Data Warehouse Environments') of the database research group at the University of Leipzig develops an UML-based schema both for technical and semantic metadata. The latter comprises constructs for representing conceptual enterprise models and multidimensional structures. Both main model components are glued together by a number of shared classes like, e.g. *Entity*, *Association*, *Attribute* and *Mapping*. Mappings are used to describe the relationships between sources and targets of transformations on the technical layer through attached transformations which filter data, aggregate data or apply general functions (e.g. join operations), but also for linking business concepts with technical entities like tables or derived relations.

A first version of the model including an insurance-domain specific sample instantiation is described in [29]. Special emphasis is put on metadata driven query generation actually exploiting the links between semantic and technical metadata. For this purpose an OQL-like intermediate query language accessing business terms is assumed. A stepwise 'unfolding' of query expressions by inspecting the involved mappings leads to the final (SQL) query executable on the data warehouse.

Future plans of the project are directed towards metadata support for data mining applications running on top of data warehouses. Metadata in this context can be used to constrain hypothesis sets and provide hints for filtering of mining results.

5.4 Résumé

Two aspects make the global schema design problem for a data warehouse repository a distinguished one, namely the active role metadata may have with regard to process execution when building or using a data warehouse, and the diversity of tools for the various data warehousing tasks.

As soon as metadata directly controls the behavior of tools, it is by nature more tool specific than a general representation of the same aspects serving, e.g., for documentation only. The emerging data warehouse business has many key players: not only the classic DB vendors providing the target platform and consulting firms giving advice for executing the projects, but also many specialized (and at least in the beginning) comparatively small data warehouse tool vendors, selling all kinds of ETL- and OLAP- tools. One system ought to be built, maintained and used, but necessarily many tools are involved and have to work together. Metadata should provide the overall view on how the system works, what it offers and how it is used. The all-from-one-hand argumentation of certain vendors in this context was only a dummy solution since missing competence in certain special areas yield only alliances and takeovers, but up to now no integration results.

The standardization side is not yet stabilized. This is not surprising since standards for specific applications require a much higher degree of detail and more precise regulations than general purpose standards. MDAPI concentrates on OLAP aspects only. CWMI is still in the call-for-proposal phase, and even which level of detail and coverage (only interfaces ?) can be expected, has not been fixed yet. Only Microsoft seems to be ahead by offering its data warehouse metamodel embedded in OIM and implemented by corresponding extensions in SQL server. The Zachman Framework based proposal in [17] up to now does not seem to have any direct commercial implications.

Data warehouse metadata management is also an attractive market for central repositories like Rochade and Platinum. They offer generic metamodels tailored to data warehousing needs from their own perspective. In most cases these models and/or the corresponding documentation are not publicly available which complicates a detailed comparison and in particular the adoption of their proposals for other tools.

As long as tool vendors continue to manage their metadata in specialized local data stores or even in a standard database with a proprietary schema which is made only in part accessible to the outside world like the MX solution of Informatica, it remains unclear whether the standalone repository vendors will reach their general goal, namely handling data warehousing metadata like any other type of metadata in a central place, accessible for all applications and users. In fact, the user side of data warehousing cannot be seen decoupled from the general representation of business metadata of a company.

A virtual integration based on an integration metametamodel providing the mapping of tool metadata to a reference metadata schema, is an ambitious goal pursued, e.g., by Ardent with its metabroker technology. Unfortunately, the integration model which is used for developing metabrokers remains implicit only and is not managed by an independent control component. The announced MetaStage product may fill this gap and in addition provides a central materialization of metadata.

In contrast to a data warehouse specific standardization, a convergence can be observed towards the general standards UML and XML, which are widely accepted as framework for representing and exchanging domain specific metadata.

Research contributions with respect to metadata are directed towards specific aspects, like introducing constructs for multidimensional representation, quality aspects or business reports, instead of pursuing the establishment of a common global metadata schema for data

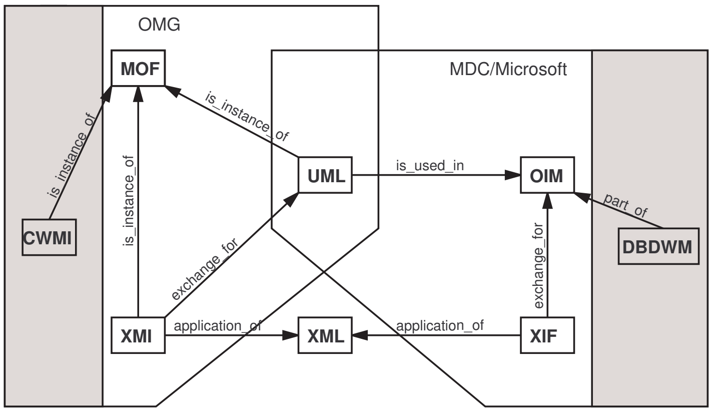

Figure 4: Standardization results

warehousing projects. The most comprehensive steps in this direction are made by the DWQ framework which is used for embedding the quality-specific metadata aspects, and by the MMDWE project.

6 Conclusions

Metadata have in general a more heterogenous structure than pure data of certain applications: They describe data and system aspects on different levels of abstraction and formalization and for different types of users. This yields specific requirements for the flexibility of both, representation and exchange languages.

We summarized the requirements on management aspects for metadata in data warehouses which do not differ significantly from those arising with general metadata and can therefore be fullfilled even by adopting the same tools. Interoperability and advanced user interaction facilities could be identified as crucial issues for successfully integrating metadata management tools into data warehouse environments. This is caused by the high demand for metadata integration and exchange due to the tendency of local and proprietary storage of metadata and the broad application bandwidth involving very heterogenous groups of human users needing different kind of metadata access support. The existing research (repository) prototypes point to various advanced techniques for handling, analyzing and reasoning with very complex and heterogenously structured metadata, and also propose improved API's and declarative query languages.

The effect of standardization efforts for metadata management in general is still unclear. Taking together the exchange standards mentioned in Section 4 and the general representation standards in Section 5 we can observe two different competing streams of standardization (cf. Figure 4) with regard to metadata representation and exchange, namely Microsoft strongly influencing MDC by OIM, and its rivals Oracle, IBM etc. who contribute to the OMG activities around CORBA.

One common intersection is UML as a general purpose modeling language suited for metadata and XML as a basis of exchange standards. Furthermore, XML might be suited as a general (low-level) metadata representation language, using, e.g. specific DTD's for handling UML models.

The main categories of metadata relevant for data warehousing were described in Section 5. These categories should be the basis of a universal metadata schema implemented in a data warehouse repository. The design of such a schema is the most important step towards integrated metadata management. It allows centralized as well as federated solutions. The integration and mapping to the standard schema can be supported by metametamodels.

The main repository vendors made their own proposals for metadata schemas suitable for data warehousing, most tool vendors follow proprietary decentralized solutions; the idea of a standardized schema supported by all involved parties currently has not yet gained full acceptance. The forthcoming CWMI standardization round of OMG will at least provide a weighty alternative to Microsoft's OIM extensions (DBDWM). A number of research projects try to fill the gap by presenting own approaches, mostly emphasizing specific aspects.

With regard to the coverage of metadata types we can state that various aspects are nowhere considered, e.g.,

- a security model attached to the representation of source systems, the target warehouse, the client applications and their respective contents, but also to processes running at build and execution time,
- metadata supporting online search, explanations, and interactive support for query formulation,
- metadata-based tracing of observations in the data warehouse on the data level through involved transformations back to the data sources.

Furthermore, data warehousing does not only consist of specifying complicated processes to be executed on data, but is itself a complex process whose creation, maintenance and evolution can be described by metadata. The handling of these aspects and their uniform embedding in a global data warehouse metadata schema requires further research.

References

[1] Purpose of an IRDS. http://www.irds.org/purpose.html.

[2] *Meta Data Europe 99: Implementing, Managing and Integration Meta Data*, London UK, March 1999. Technology Transfer Institute. http://www.ttiuk.co.uk/.

[3] P. Atzeni, G. Mecca, and P. Merialdo. To weave the Web. In *Proceedings of 23rd International Conference on Very Large Data Bases*, pages 206–215, Athens, Greece, August 1997. Morgan Kaufmann.

[4] J. Becker and R. Holten. Fachkonzeptuelle Spezifikation von Führungsinformationssystemen. *Wirtschaftsinformatik*, 40(6):483–492, December 1998.

[5] P.A. Bernstein. Repositories and object oriented databases. *SIGMOD Record*, 27(1):88 –96, March 1998.

[6] A. Berson and S.J. Smith. *Data Warehousing, Data Mining & OLAP*. McGraw-Hill, 1997.

[7] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, 1999.

[8] M.H. Brackett. *The Data Warehouse Challenge*. Wiley, 1996.

[9] European Computer Manufacturer's Association (ECMA). *Portable Common Tool Environment (PCTE) - Abstract Specification*, fourth edition, 1997. Standard ECMA-149: http://www.ecma.ch/stand/Ecma-149.htm.

[10] M.F. Fernandez, D. Florescu, A.Y. Levy, and D. Suciu. Web-site management: The Strudel approach. *Data Engineering Bulletin*, 21(2):14–20, 1998.

[11] N. Fridman Noy and C.D. Hafner. The state of the art in ontology design. *AI Magazine*, 18(3):53 – 74, Fall 1997.

[12] H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J.D. Ullman, V. Vassalos, and J. Widom. The TSIMMIS approach to mediation: Data models and languages. *Journal of Intelligent Information Systems*, 8(2):117–132, 1997.

[13] Gartner Group. *Repository Market Update 1999*, 1999. Research Note 01 June 1999, M-08-3721.

[14] O. Haase and A. Henrich. Query processing techniques for partly inaccessible distributed databases. In *Proceedings of the 15th British National Conference on Databases*, pages 123–125, London, UK, July 1997. Springer, LNCS 1271.

[15] R. Hull and G. Zhou. A framework for supporting data integration using the materialized and virtual approaches. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 481–492, Montreal, Canada, June 1996. ACM Press.

[16] W.H. Inmon. *Building the Data Warehouse*. John Wiley & Sons, 1993.

[17] W.H. Inmon, J.A. Zachman, and J.G. Geiger. *Data Stores, Data Warehousing and the Zachman Framework*. McGraw-Hill, 1997.

[18] ISO/IEC. *ISO/IEC 10027:1990 IRDS Framework*, 1990.

[19] M. Jarke, R. Gallersdörfer, M. Jeusfeld, M. Staudt, and S. Eherer. Conceptbase: A deductive object base for meta data management. *Journal of Intelligent Information Systems*, 4(2):167 –192, March 1995.

[20] M. Jarke, M.A. Jeusfeld, C. Quix, and P. Vassiliadis. Architecture and quality in data warehouses. In *Proc. of the 10th Conference on Advanced Information Systems Engineering (CAiSE '98)*, Pisa, Italy, June 1998.

[21] M. Jeusfeld, M. Jarke, M. Staudt, C. Quix, and T. List. Application experience with a repository system for information systems development. In *Proc. GI-Symposium EMISA, "Development Methods for Information Systems and their Application"*, Fischbachau, Germany, September 1999. Teubner.

[22] W. Labio, Y. Zhuge, J.L. Wiener, H. Gupta, H.Garcia-Molina, and J. Widom. The WHIPS prototype for data warehouse creation and maintenance. In *Proceedings ACM SIGMOD International Conference on Management of Data*, pages 557–559, Tucson, Arizona, May 1997. ACM Press.

[23] D. B. Lenat. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11), November 1995.

[24] A. Levy, A. Rajaraman, and J.J. Ordille. Querying heterogeneous information sources using source descriptions. In *Proceedings of 22th International Conference on Very Large Data Bases*, pages 251–262, Mumbai (Bombay), India, September 1996. Morgan Kaufmann.

[25] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom. Lore: A database management system for semistructured data. *SIGMOD Record*, 26(3):54–66, September 1997.

[26] W. Nejdl and M. Wolpers. KBS Hyperbook - a data-driven information system on the web. Technical report, University of Hannover, KBS Institute, November 1998.

[27] Object Management Group (OMG). *Common Warehouse Metadata Interchange - Request For Proposal*, 1998. OMG Document ad/98-09-02.

[28] M. Staudt, A. Vaduva, and T. Vetterli. Metadata management and data warehousing. Technical Report 21, Swiss Life, Information Systems Research, July 1999.

[29] T. Stöhr, R. Müller, and E. Rahm. An integrative and uniform model for metadata management in data warehousing environments. In *Proceedings of the Intl. Workshop on Design and Management of Data Warehouses (DMDW 99)*, pages 12.1–12.16, Heidelberg, Germany, June 1999.

[30] The Meta Data Coalition. *Meta Data Interchange Specification (MDIS Version 1.1)*, 1997. http://www.MDCinfo.com/MDIS/MDIS11.html.

[31] The OLAP Council. *The MDAPI specification*, 1998. http://www.olapcouncil.org/research/apily.htm.

[32] M. Tork Roth, M. Arya, L.M. Haas, M.J. Carey, W.F. Cody, R. Fagin, P.M. Schwarz, J. Thomas, and E.L. Wimmers. The Garlic project. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4-6, 1996*, page 557. ACM Press, 1996.

[33] W3C. *Extensible Markup Language (XML)*, 1.0 edition, 1998. Recommendation 10-February-1998, http://www.w3.org/TR/1998/REC-xml-19980210.

[34] John A. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3), 1987.