

Meta data Models

Meta data Models

Author: Prakash Kewalramani

Organization: Evaltech, Inc.

**Evaltech Research Group,
Data Warehousing Practice.**

Date: 09/11/2007

Email: erg@evaltech.com



Abstract:

This article gives the details about the Meta data, Meta data categories, Architectures for Metadata Management and Enterprise Information Portals.

Metadata is divided into two categories: technical and business metadata.

Technical metadata provides the description of data within an IT infrastructure — where the data is located, what the names and access methods to the servers are, what kinds of data types are being stored, and other attributes. Business metadata is far more valuable to information consumers than technical metadata. These descriptions are inferred and derived from existing specifications, business rules and technical metadata.

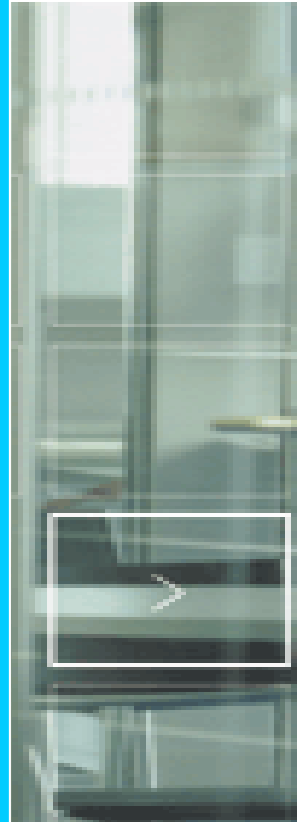
Regarding the way metadata is managed and shared between applications, we generalize three approaches for metadata management in a heterogeneous environment:

centralized, decentralized, and shared or federated. In addition, a mixed approach is also discussed in this article.

Enterprise information portals allow shared repository integrating technical and business metadata and providing the base for various data access tools and end-user interaction. Eight components are critical for a complete EIP solution: security, caching, taxonomy, multi-repository support, search, personalization, application integration, and a metadata dictionary. All these are described in this article.

Intellectual Property / Copyright Material

All text and graphics found in this article are the property of the Evaltech, Inc.



Meta data Models

INTRODUCTION TO METADATA

Metadata, which is loosely defined as “data about data,” is acquiring increased visibility and importance as organizations grapple with the complex process of delivering integrated information technology solutions to their end users. There is tremendous confusion about what metadata actually is, how it is used, and how companies should go about the process of creating, managing, and leveraging their metadata investment. We believe that enterprise metadata management has entered a new phase and is on the verge of becoming a crucial information asset that companies can leverage for competitive advantage. The traditional definition of metadata is that it is data about the structures contained within databases — especially relational databases. This classic definition views metadata as a way to describe database properties and attributes, including database, table, and column names; column attributes (size, data type, constraints); and primary key/foreign key relationships. However, given the increasingly complex, heterogeneous, and distributed computing environment that many companies are deploying, metadata is acquiring new meanings. This trend has become particularly evident in the decision support marketplace, but it is also becoming important in the enterprise resource planning (ERP) or packaged application marketplace. For example, metadata is used in decision support to describe the back-end data extraction and transformation process used to prepare data to be loaded into a data warehouse. This process involves identifying source target data locations and attributes, mapping the source data to target database(s), and defining the business logic to scrub, consolidate, transform, and otherwise prepare the data to be moved into the target. A trivial example of the business logic used in the transformation process would be to change every instance of “1” in a source field identified as gender to “M” in a target field. Complex transformations might involve consolidating records from multiple sources, such as account information, customer demographics, sales data, customer support calls, and Balanced View returns, into a single target record. Some companies are establishing application integration capabilities at either the application API level or the application database level. Many of these systems contain triggers that are used to transmit information to linked systems, either on a scheduled or event-driven basis. Using metadata for decision support, end-user query and analysis tools can browse, query, and search both relational and non-relational databases, such as multidimensional cubes. Database administrators and end users also use metadata to determine the lineage and usability of data found in the data warehouse. Ultimately metadata can also be used to describe various business processes, particularly the steps of a workflow process, and how the business processes relate to different applications or to the modules of an ERP application.

THE IMPORTANCE OF METADATA

Metadata is the fundamental basis for establishing enterprise-wide data integrity. As business needs change, an organization’s business applications will evolve. Metadata is useful for analysing how the changes will affect the application, its modules, and the other applications with which it must interact. Metadata does this by providing both an integrated data model and a description of how the physical databases relate to that model, which can be linked with the various applications that use that data. A good metadata management tool can identify and highlight the elements of the enterprise data model that will be affected by application modifications. Metadata is the glue that binds IT applications and data. It identifies the various databases and their constituent data objects (tables and columns) and establishes their relationships. This enables an organization with a metadata strategy to establish consistency in its data values. For example, many organizations suffer from data value fragmentation, with various systems containing inconsistent data value definitions for key business objects and concepts, such as revenue, customer, product, and vendor. An enterprise metadata strategy enables a company to publish and maintain a catalog that is a system of record for defining the attributes of customers, products, vendors, and such related business concepts as revenue,

Meta data Models

profit, and invoice. A carefully designed and implemented metadata strategy that supports metadata interoperability will also enable a company to support a best-of-breed architecture for both decision support and transaction processing applications. This approach ensures that the applications that interoperate with the enterprise metadata repositories will all be using the same data.

KEY ORGANIZATIONAL METADATA ISSUES

No single metadata repository can meet the needs of all applications and users within a large organization. We believe that large, global organizations will deploy a mix of central and distributed Meta data repositories. Central repositories contain metadata that describes corporate data structures and business rules, whereas distributed; decentralized repositories contain either specialized metadata that overrides the corporate metadata or additional metadata not appropriate for the corporate repository. Distributed repositories may be regional (USA, India) or they may be deployed within functional business units (accounting, sales/marketing, product development, customer service). A regional repository contains local business rules that super cede corporate business rules, for example the local U.S. rules about employee termination that differs greatly from the corporate European or Japanese rules. A department repository contains metadata about databases and applications specific to that department, such as audit procedures in the accounting department or quality assurance rules for product development. Therefore, a key issue that organizations must address is metadata consistency within multiple repositories distributed across the enterprise. This problem can be resolved by establishing a metadata policy that defines the system of record for metadata consistency. This record is the gold standard for the enterprise. The metadata policy should also describe the processes and procedures required to add, delete, modify, and replicate metadata definitions within the system of record as well as between the system of record and the distributed repositories.

Another important consideration that developers must address is how to present metadata to users in a way that is consistent across applications but also context-specific to the application. Separating metadata into a metadata storage layer and a metadata presentation layer accomplishes this. The metadata storage layer contains the various categories of metadata, and the metadata presentation layer gives the user a context within which to review and evaluate metadata. For example, the process of loading data into a data warehouse may extract data from multiple sources, consolidate it on a staging platform, and filter it through a series of transformation functions before it is loaded into the data warehouse. The various steps in this metadata-driven process often affect different metadata elements. To understand the lineage of the data and verify that it is the most appropriate data for analysis, the user will want to understand this process in terms that make sense to the user. An experienced programmer responsible for maintaining and enhancing the transformation process can access this same metadata. These two individuals have extremely varied needs for how they will access and use the metadata stored in the enterprise metadata repositories.

Structural metadata is used for creation and maintenance of the data warehouse. It fully describes data warehouse structure and content. The basic building block of structural metadata is the data warehouse model that describes its data entities, their characteristics, and how they are related to one another. The way potential data warehouse users currently use, or intend to use, enterprise measures provides insight into how to best serve them from the data warehouse; i.e., what data entities to include and how to aggregate detailed data entities. The data warehouse model provides a means of documenting and identifying structural metadata. This includes both strategic and operational uses of enterprise measures, as well as multi-dimensional summarization. Structural metadata also includes performance metrics for programs and queries so that users and developers know how long programs and queries should run. Data warehouse performance tuning also uses these metrics.

Access metadata is the dynamic link between the data warehouse and end-user applications. It generally contains the enterprise measures supported by the data warehouse and a dictionary of standard terms including user-defined custom names and aliases. Access metadata also includes

Meta data Models

the location and description of data warehouse servers, databases, tables, detailed data, and summaries along with descriptions of original data sources and transformations. Access metadata provides rules for drill up, drill down and views across enterprise dimensions and subject hierarchies like products, markets, and customers. Access metadata also allows rules for user-defined custom calculations and queries to be included. In addition, access metadata contains individual, work group, and enterprise security for viewing, changing, and distributing custom calculations, summaries, or other analyses.

Roles

Maintaining consistent access control mechanisms is a hard problem for most organizations. The variety of different systems sometimes-containing similar information make implementing rational consistent access control hard. When each system has separate mechanisms for establishing access it takes a great deal of work to keep them in synch. In order to simplify this area at MIT, we designed a central authorization system called Roles. This system allows for the maintenance of authorizations in a single system. Then this system can drive the authorizations in both the system of record and the warehouse. In our case the financial reporting authorizations are maintained in Roles and fed to both SAP and the Warehouse always keeping them in synch. Furthermore, the interface for maintaining them can be distributed out to the departments, so that the people who know what the authorizations should be are maintaining them.

An enterprise access control infrastructure has the advantage of:

- Achieving consistent access control over multiple systems
- Distributing the maintenance of access controls to a person close enough to know and care whether the authorizations are correct.
- Having a simple way to maintain and view authorizations across many systems.
- Granting authorizations at the highest possible level, but enforcing them at a detailed level, which minimizes the changes to authorizations due to account or organizational restructuring.

State of the Warehouse

The warehouse is now enjoying some success. The usage has been steadily climbing this past year. More and more people are incorporating the warehouse in their normal work. The central offices that are the information providers, are finding many benefits as well. They don't have to write and maintain as many feeds now, since people can get information directly from the warehouse. Also, having other ways to analyse their own data helps uncover problems sooner. Some of their common reports can be run against the warehouse in far less time, because of the more efficient reporting structures.

We're feeding many systems currently using a variety of methods, such as database links, file extracts, and Perl programs. Most of the maintenance of this is done by the person getting the feed and so as they need changes they can just do it themselves, since their program define the format, sort etc.

Metadata is information about enterprise data. However, metadata is much more than this. It is descriptive information about the context, quality, condition, and characteristics of data. For example, metadata describes data elements and their attributes (name, size, data type, etc.), records or data structures (length, fields, columns, etc), and the data itself (where it is located, how it is associated, ownership, etc.).

Meta data Models

Metadata is divided into two categories: technical and business metadata:

Technical Metadata

Technical metadata provides the description of data within an IT infrastructure — where the data is located, what the names and access methods to the servers are, what kinds of data types are being stored, and other attributes. This is the type of metadata most IT tools are capable of creating including code generators. Most ETL environments are able to create technical metadata from the existing sources. Technical metadata also stores information about the transformation processes, logs and audit trails, giving a full technical view to the environment.

Business Metadata

Business metadata is far more valuable to information consumers than technical metadata. These descriptions are inferred and derived from existing specifications, business rules and technical metadata. The key issue for many information consumers is data quality — Where did the data come from? How was it transformed? How current is it? By providing direct access to all this metadata, knowledge workers are empowered to answer questions directly and promptly in a self-sufficient manner. Information consumers can readily see all the information around business rules, processes, audit trails, and transformations in a user-friendly format. For IT professionals and developers, business metadata must be scalable so that the designers can add classes and attributes to the metadata creating brand new semantic objects. These objects can be URL addresses to reports on the web, information about different roles an object has, or simply descriptive comments on how to use the data found in a metadata repository. Simply put, metadata is the key enterprise catalog for searching and accessing information within organizational data warehouses.

Architectures

As metadata develops into the fundamental underpinning of robust enterprise data integration and repository building and maintenance, architectural issues surrounding metadata become increasingly important.

Different Standards

The quest for the ideal metadata architecture has created numerous standards in the market, confusing users more than helping them. The Metadata Coalition has been one of the carrying forces but large database players have sponsored a competing model with OMG – the Object Management Group.

MDC — OIM

The Metadata Coalition (MDC) has developed, with the help of its partners, a set of modeling standards to be able to store and share metadata between different applications and heterogeneous systems. The model is simply called Object Information Model (OIM) and is currently a working documenting that will likely change over time.

OMG — MOF

The Meta Object Facility (MOF) specification provides a set of CORBA interfaces that can be used to define and manipulate a group of interoperable meta-models. The MOF is a key building block in the construction of CORBA based distributed development environments. This specification enhances metadata management and interoperability in distributed object environments in general, and in particular, distributed development environments. The MOF also defines a simple meta-model with sufficient semantics to describe meta-models in various domains starting with the domain of object analysis and design. Integration of meta-models across domains is required for integrating tools and applications across the life cycle using common semantics.

The Problem with Focusing on Technical Issues

The problem with trying to define perfect metadata models is that vendors easily forget the main focus — the information consumer. The value proposition for organizations is that end-users

Meta data Models

(knowledge workers, analysts, managers, etc.) need a means to locate and browse relevant information and reports to help them manage on a day-to-day basis. Building a metadata-centric solution must, by definition, address the needs of the information consumer.

Enterprise Repository Synchronization — Part of the Solution

Another issue is the interchange of metadata between back-end repositories and front-end tools like business-intelligence reporting solutions. To be a true enterprise solution, metadata must seamlessly move from back-end systems to front-end tools. This enables knowledge workers to gain access to business rules, making the time they spend on query and analysis activity more productive.

Back End to Front End

For the end-user benefit there is a great variety of reporting tools available for pulling out the information stored in the data warehouse. The greatest return on investment (ROI) is always achieved if the data is already cleansed and transformed into a format that is useful for the business purposes. Some of these tools also give access to the operational source systems, which is the least desirable architecture — using a reporting data warehouse or data mart is much easier to administer and gives vast performance boost over the operational sources. To improve the end-user usability, the system must provide information about the contents and the quality of the data. This information is provided through so called metadata repositories. Most providers in the market are making their own repositories with more or less the basic information about the technical and business environment. Like other DW solution vendors, Hummingbird has put a lot of effort to create an ETL metadata repository and to open it so that the reporting tool providers can connect to it as easily as possible. Some vendors have also built proprietary “standards” to access the meta-information from the business intelligence tools instead of accessing it directly from the source.

The Growing Importance of Metadata

As organizations strive to embrace the e-business model that will drive business in the new millennium, metadata will play an increasingly important part. With the evolution of business intelligence, metadata will serve as the bonding agent that ties the various tools and technologies together on an enterprise level. A sound metadata strategy will ensure seamless access, exchange, and integration among various vendors’ business intelligence tools and repositories. It will maximize sharing and re-use of important metadata, and will eliminate redundancy. Finally, it will ensure metadata integrity, from tool to tool, and user to user. By providing the foundation components to form a metadata strategy that will enable organizations to take full advantage of their enterprise metadata resource,

Meta data is a component of data quality initiatives and data quality improvement

There are several reasons why companies initiate [data](#) quality efforts. The driving reason may be poor quality [data](#) discovered during the integration of several legacy systems into packaged solutions such as SAP, People Soft, or Oracle Financials. Another reason may include the same discovery during the development of a decision support environment. One more reason may include known and documented faults in operational data that are causing business problems such as delayed and/or rejected transactions. These are all legitimate reasons for focused efforts on improving data quality. Companies that initiate (or have on-going) data quality efforts spend a large amount of resources investigating data make-up and definition, documenting [data](#) accountabilities and responsibilities (stewardship), and mapping data across the corporation. This information can be found in Meta data. If the Meta data that is used for the corporate data quality efforts is available, the company has a tremendous competitive advantage over similar efforts at companies that do not manage and make available Meta data. If the meta data necessary for the data quality effort is not available, companies should consider taking advantage of the research and documentation created during the data quality initiatives by capturing and maintaining meta [data](#) in a centralized [data](#) asset catalog (repository) to support future IT & data quality programs.

Meta data Models

Meta data is used to control or reduce data redundancy

It is very difficult to manage something if you know little or nothing about its existence. This observation holds true about data and the prevention of duplicate or redundant data. In many companies, the data administrator or the data modeller is the first line of defence when defining new data. Often, how well these individuals define and reuse data is a result of the information on hand about existing data. As an example, the "best practices" approach to data modelling includes sharing entities and attributes from an enterprise data model across multiple project (or subject area) data models. In the absence of an enterprise model (or some form of reusable data model entities), the same data becomes defined repeatedly by different individuals in the organization. Data modelling by itself, on a project-by-project basis, does not provide the ability to share data unless there is access to the Meta data (data about data models) that already exist. If the intent is to share data across the enterprise, each application development area needs to know what data structures already exist before it can define the requirements for what does not yet exist. The data documentation that provides this ability to see what exists, and to share and re-use data, is Meta data.

Relationship between data modelling and Meta data

The information that is manually entered into the CASE (data modelling) tools is Meta data (or data about the logical or physical components of the data). Therefore, if data modelling is a part of your IT processes and data-modelling information is important to your organization, Meta data is important as well. When the data modeller creates an entity relation diagram (ERD), they define the way that data is represented logically and physically in your company. The modeller creates data entities and their attributes, the relationships between the data entities, the logical and physical names for the data, domains, and more ... that represent how data is defined for that enterprise, project, or subject area. All of this information is Meta data. The first question in this article mentioned the use of Meta data beyond the IT tools themselves. In this case, the modelling Meta data remains in the CASE tool where no one other than the modellers can view it. The business names and definitions that originate as meta data in the CASE tool should be shared with knowledge workers and application developers that are interested in how the data of the organization is defined and related.

Meta data is helpful when forcing compliance to IT standards

Forcing compliance to IT standards is typically an on-going battle. Some companies are successful and some companies are less than successful. Often the success or failure of following standards is a result of the corporation's environment (use of packages, merging companies and IT functions, centralization of IT service functions, etc.). Other times, the success or failure is based on the company's ability or willingness to force individuals to comply to the rules of IT development. Most IT standards are based on Meta data. Component naming, storage location, and component interaction, are a few examples of Meta data that can play a significant role in IT standards. Naming standards, for example, define specific ways in which components are named. Examples of standards for naming include embedded component types (identifying JCL, program, table, etc.), embedded owning applications or contexts (through prefixes/application coding), or the identification of the platform and tier on which the component is based (personal, departmental, organizational, etc.). Many common standards are created and controlled by meta data. Meta data (stored in a repository) can be used to identify components that do not follow naming standards. Meta data can be used to identify how many versions of each component exist and where they exist. In an ideal environment, standards based on Meta data could be built into the change management environment making it impossible to process components that do not follow standards. Since many IT standards are based on meta data, the ability to track and report on that meta data is very helpful when forcing compliance to IT standards.

Meta data Models

Metadata Models

Managing the different types of metadata requires repositories with a powerful metadata model. Several vendors offer general repositories supporting data warehousing environments as well as other application domains such as software development. In addition, most data warehousing tools use built-in repositories for their specific functionality thus typically covering a particular subset of the metadata. Several research efforts have also proposed metadata models suitable for data warehousing].

Requirement for a Uniform Representation of Warehouse Metadata

As discussed in the introduction, metadata from multiple sources needs to be integrated and thus mapped between different repositories. If every participating repository uses a different Metadata model the mapping process becomes very complex and hard to maintain. Further-more, it may not be possible to propagate all required metadata due to differences in the expressive power of the metadata models. These problems are analogous to the well-known schema translation and integration problems of federated database systems. Schema translation is performed to transform different database schemata into a common data model (e.g., object-oriented model). Schema integration then combines the homogeneously represented but independent database schemata into a combined global schema. The main problem in this step is to resolve semantic schema conflicts. Apparently, the proposed research approaches have had little impact in the commercial world so far. The metadata integration problem of data warehouses is partially more complex since not only schemata but also a multitude of additional metadata needs to be dealt with. Uniformly representing this metadata (and thus solving the translation problem) can substantially be simplified by a powerful metadata model standard if many repositories support it. Currently, there are two competing industry efforts for such a standard metadata model for data warehouses, namely OIM and CWM. The OIM (Open Information Model) effort is led by the Metadata Coalition 2 (MDC) and supported by Microsoft and other companies. CWM (Common Warehouse Metamodel) has been introduced more recently; it is an approach of the Object Management Group 3 (OMG) and supported by IBM, Oracle and others. We briefly discuss and compare these approaches in the following.

Technical and Business Metadata in OIM and CWM

The aim of OIM is to generally support tool interoperability across technologies and companies via a shared information model. OIM contains metadata from a broad range of subject areas, i.e. software development, object-oriented analysis, business engineering, data warehousing, and knowledge management. CWM, on the other hand, addresses metadata inter-change specifically in the data warehouse environment, i.e. between warehouse tools. Both metadata models make use of OMG's standard modeling language UML (Unified Modeling Language) for the specification, visualization and documentation of their metadata types. The object-orientation supports extensibility of the models by defining additional (specialized) sub models. CWM also conforms to the Meta Object Facility (MOF), an OMG metadata interface standard used for defining and representing metadata as CORBA objects.

Technical Metadata OIM Sub model CWM Sub model

OIM and CWM have similar sub models for representing technical metadata such as data schemata, (i.e. relational, record-oriented, multidimensional, XML), and data transformations. CWM includes three additional sub models describing other technical aspects of the data warehouse environment:

1. ?Warehouse Deployment: records how the software systems in a data warehouse are used and where they are installed (which computer, geographical location).
2. ?Warehouse Process: documents process flow used to execute the transformations. A warehouse process associates a transformation to a series of events (scheduled, internal or external) used to trigger the execution of the transformation.
3. ?Warehouse Operation: covers runtime metadata, e.g., about time and state of recent executions of transformations, and a historical record of metadata changes Regarding business metadata, both OIM and CWM provide basic metadata types to describe general aspects of business information such as contact information (data stewardship),

Meta data Models

- textual descriptions, etc., which can be attached to other model elements. Furthermore, both models contain some similar sub models devoted to business metadata from several areas of data warehousing, such as data analysis and knowledge management.
4. ?OIM Report Definitions / CWM Information Reporting: both provide metadata types to represent formatted reports (report formatting definitions, relationships between report fields and corresponding function / query expressions).
 5. ?OIM Knowledge Descriptions / CWM Business Nomenclature: both provide metadata types to describe and categorize information. A semantic network of taxonomies, concepts, terms, glossaries, etc. can be built allowing the sharing and the collaboration of knowledge in an organization. Besides such overlap, both OIM and CWM contain some unique sub models:
 6. ?OIM Semantic Definitions: accommodates conceptual models of user information which are basically built on three semantic concepts: entities, relationships, and dictionary entries. Schema-to-semantic mappings linking physical data with business concepts and relationships defined by means of linguistic phrases allow end-users to interact with a database without learning a data retrieval and manipulation language such as SQL.
 7. ?CWM Information Visualization: provides the foundation for the CWM Information Reporting sub model. It defines generic metadata types describing the mechanism for visualizing, i.e. rendering, an arbitrary model element in two dimensions (e.g. displaying a query result set in different formats, such as printed reports, Web page, charts, etc.).
 8. ?CWM Data Mining: contains metadata types and associations related to the data mining process, such as a generic data mining model, mining settings driving the construction of the mining model, input attributes and their usage, mining result set, etc.

Currently, both OIM and CWM do not address the management of users, access rights, user profiles and the association between users and information objects. Therefore, personalized views on warehouse data are not yet supported.

Business Metadata OIM Sub model CWM Sub model

The possibility to uniformly represent metadata from multiple sources simplifies but does not solve the metadata integration problem. In particular, semantic heterogeneity has to be dealt with when designing the warehouse schemata and defining the required transformations.

Architectural Alternatives

We first discuss general architectures for shared metadata that may be represented according to one of the standard metadata models. Thereafter, we compare major alternatives for meta-data interoperability, in particular exchange files and metadata APIs as well as the use of wrappers for metadata mapping.

General Architectures for Metadata Management

Regarding the way metadata is managed and shared between applications, we generalize three approaches for metadata management in a heterogeneous environment: centralized, decentralized, and shared or federated. In addition, a mixed approach is discussed.

Centralized Approach

In this case, all tools and the warehouse and data mart DBMSs directly access a central repository. They do not store and maintain metadata locally. The central repository is used to manage shared as well as tool/DBMS-specific metadata. The biggest advantage of this approach is that a non-replicated and consistent management of all metadata can be achieved. Every component has access to all existing and current metadata. However, this approach would require that all tools and DBMSs depend on the central repository even for specific metadata not relevant for other components. Such a loss of autonomy is typically unacceptable, in particular if components from multiple vendors need to cooperate. In addition, there is a performance problem because all metadata accesses would require interaction with the central repository.

Decentralized Approach

Meta data Models

This approach represents the other extreme that is typical for current data warehouse environments. All tools and DBMSs possess their own (local) metadata repository and communicate with each other to exchange metadata. This supports maximal autonomy and performance for tool/DBMS-specific metadata. On the other hand, numerous bi-directional connections are needed to exchange shared metadata. Each connection may involve a complex mapping if metadata is differently represented in the local repositories. Furthermore, metadata is replicated in several components and difficult to keep current and consistent.

Shared Approach

This approach tries to combine the advantages of two previous approaches. Each tool/DBMS possesses its own repository for its local metadata thus supporting autonomy and fast access for this metadata. In addition, each component supports a metadata exchange interface to a common repository managing all shared metadata. While the metadata may be heterogeneously represented within the local repositories, the shared repository, e.g. based on a standard metadata model such as OIM or CWM, supports a uniform representation. In contrast to the decentralized approach, the number of tool-to-tool connections and the mapping overhead can be significantly reduced and metadata replication can be tracked and controlled centrally.

The shared approach represents a federated metadata architecture preserving the autonomy and supporting heterogeneity of the participating repositories. Each repository decides which part of its metadata is to be exported, e.g., by defining a corresponding export schema. The shared repository unifies and combines these export schemata within a common metadata model thus improving its usability for other tools and end-users.

Mixed Approach

The aforementioned approaches can be combined within mixed or hybrid metadata architecture. For instance, while the central approach seems to be of little relevance it can be utilized at the data access layer, e.g. when multiple client invocations of an access tool share the tool's local repository. On the other hand, the shared/federated approach seems to be the best choice but may not be achievable in its pure form in multi-vendor environments, e.g. if the operational systems or the data warehouse DBMS do not directly support a shared repository. we achieve a controlled flow of metadata from the ETL tool and the modeling tool to the data access tools, so that dependencies between metadata in those tools can be recorded. The backflow of metadata from data access tools to the shared repository enables users to inform themselves about objects created during data analysis, e.g. queries, reports, or data cubes. Data marts, data warehouse and operational systems are not directly connected to the shared repository. However, for them, the modeling and ETL repositories represent shared repositories so that the architecture can be seen as a multi-level repository federation.

Interoperability Mechanisms

Most of the discussed metadata architectures require the ability to communicate and exchange metadata between repositories. The exchange of metadata corresponds to a mapping process between two metadata models. Metadata exchange can more or less use the same methods than for data exchange between different data management systems. In particular, file exchange and API approaches can be used. In addition, a wrapper-based middleware approach can be employed to perform mappings between different metadata models. In order to provide more flexibility, some repositories implement several exchange methods at the same time.

File Exchange

The simplest method for metadata interoperability is to write metadata to a file of a specified (standardized) format that can be imported by other tools or repositories. Such exchange files are easily portable across multiple platforms. Furthermore, they can be produced and consumed even by tools and applications without a local repository. The file exchange method implies an *asynchronous* access to external metadata, actually to copies of it, which leads to two disadvantages. First, metadata is only current at the time of import. Second, the mechanism for translating between a tool's proprietary metadata format and the file format is hard-coded in the

Meta data Models

tool and needs to be adapted when the exchange format evolves. Early specifications for metadata exchange formats include MDIS (Metadata Interchange Specification) and CDIF (Case Data Interchange Format). The Metadata Coalition introduced MDIS in 1996. It allows the representation of different kinds of database schemata (relational, object-oriented, record-oriented, multidimensional) as well as inter-database relationships (textual descriptions of transformations and mappings between databases). MDIS has not been enhanced or extended since 1997. CDIF was specifically developed for transferring data models between CASE tools and is based on an integrated and extensible metadata model. CASE tools such as CA/ support CDIF Platinum ERwin and Oracle Designer.

Currently, most promising appears the use of XML (Extensible Markup Language), a standard language to represent data on the Web. XML supports so-called Document Type Definitions (DTD) to specify the structure of documents and other files. It is thus well suited as the basis for standardizing the formats for exchanging data or metadata. With regard to data warehousing, XML has gained substantial importance as it is supported by the standard metadata models OIM and CWM and commercial repositories (e.g., Microsoft Repository, Viasoft Rochade). CWM supports XMI (XML Metadata Interchange) for defining mappings of its metadata into XML; OIM provides an XML-based exchange format based on the former Microsoft specification XIF (XML Interchange Format). Currently, XML is not yet widely supported by modeling, ETL and data access tools but this is expected to change soon.

Repository API

A repository usually comes with a proprietary application-programming interface (API) to its contents: metadata can be retrieved, inserted and manipulated. Repositories document their API and typically provide an application development environment. The API allows other repositories and tools to synchronously access and exchange metadata. This enables access to the current metadata of a repository. The API can also provide the means to extend and customize the metadata model of the repository. The disadvantage of this method is however the time and effort to be spent on the development of such applications. Only some other vendors typically support proprietary APIs. For instance, the Meta-data Exchange (MX and its successor MX2) architecture of Informatica provides a set of COM-based APIs to the repository underlying its ETL tool PowerCenter/PowerMart. Tools of several vendors (e.g. Cognos, Business Objects) use these APIs to retrieve and push metadata from/into the repository. According to Informatica, MX2 is compatible with OIM. An MX2 software development environment is also included for developing custom applications. Relational and object-oriented data APIs such as ODBC and OLE/DB also provide access to metadata and can thus be used for metadata exchange. For instance, the multidimensional API standard OLE/DB for OLAP includes commands to query and manipulate metadata describing the schemata of OLAP data sources.

Metadata Wrapper

In federated data architectures, wrappers are used to hide API differences of heterogeneous data sources. A wrapper performs the mapping between the API of a data source and a common API that may be used by other middleware, such as a mediator providing uniform access to the underlying sources. Such an approach can also be used in the data warehouse context to provide uniform metadata access to heterogeneous repositories). In particular, it facilitates the integration of heterogeneous metadata within a shared repository. The metadata exchange between the source repositories and the wrapper and between wrapper and shared repository may be performed either API-based (synchronously) or asynchronously with exchange files. Typically, bi directional wrappers are needed to both write metadata into and read metadata from the shared repository. Several commercial repositories follow the wrapper approach.

Metadata Synchronization

While metadata changes less frequently than data, metadata updates are more difficult to deal with. This is because metadata updates not only affect the data that is described but also other metadata objects due to metadata interrelationships. These problems already occur in a local

Meta data Models

(database) environment; specific research aspects have been addressed in the area of schema evolution. In our context, we have to additionally synchronize repositories sharing metadata with each other. In particular, updates of replicated metadata need to be detected and propagated automatically in order to keep this metadata consistent. Further-more, updated metadata needs to be applied (integrated) within a repository, e.g. to keep interrelationships with metadata from other repositories consistent. For example, changes in the data warehouse schema may cause errors in the execution of some ETL jobs, queries or reports when they are not adapted appropriately. In the following we first discuss general approaches to these synchronization problems. We then turn to publish/subscribe approaches that may be used in combination with a shared repository.

Replication Control and Update Propagation

Numerous methods for replication control have been proposed in the area of distributed databases. Most of them such as ROWA (read one, write any) or voting schemes strive for one copy serializability where all replicas are mutually consistent and each read is guaranteed to get access to the most recent transaction-consistent copy of a replicated object. Furthermore, each copy can typically both be read and updated. Unfortunately, such strict approaches are too restrictive and expensive to be widely applicable. In particular, they introduce substantial delays for synchronously updating multiple database copies (e.g., during commit processing) and require global concurrency control for strictly serializing updates. Furthermore, availability problems arise when some of the copies to be updated are not accessible due to system failures etc. Such approaches are thus not appropriate for controlling replication of shared metadata. This is also because repositories are heterogeneous and largely autonomous and not ready to participate a global concurrency control scheme for serializing updates. We therefore focus on „lazy“ synchronization approaches with weaker consistency goals, which are also found in commercial DBMSs. Such strategies can be characterized by a publish/subscribe approach. A tool or repository changing (a copy of) shared metadata is called a „publisher“, a recipient of (a copy of) shared metadata is called a „subscriber“. Of course, a component may have roles, publisher and subscriber, at the same time.

Replication Control with a Shared Repository

The shared repository architecture can ease the task of metadata synchronization. It provides an almost up-to-date and possibly versioned set of all shared metadata and allows a centralized replication control for this shared metadata. A particular publish-and-subscribe approach (also called „hub-and-spoke“ can be applied where the shared repository has both the role of a subscriber and publisher. Other publishers and subscribers access the shared repository to transfer/receive updates of shared metadata. The shared repository can register all publishers and subscribers together with the set of metadata they export or import. For update propagation each of the methods discussed can be chosen. The main choice we left open was the distinction between a notification (pull) and probing (push) approach. These alternatives exist for two propagating steps: update propagation from a publishing repository to the shared repository and update propagation from the shared repository to subscribers. This results in four possible combinations where each of the steps is either implemented by the shared repository or by the external publishers/subscribers.

1. The publishers push (propagate) their changes to the shared repository. The subscribers, on their own, detect the updates made in the shared repository and pull (import) them into their repository. This approach actually does not make active use of the shared repository for replication control: it only plays the role of a data management component.
2. The publishers push their changes to the shared repository, which further pushes those changes to the subscribers. The shared repository registers the set of subscribed metadata.

Meta data Models

3. The shared repository detects and pulls changes made by the publishers. The subscribers detect and pull updates made in the shared repository. The shared repository registers the set of each provider's published metadata.
4. The shared repository detects and pulls changes made in the publishers, and pushes them further to the subscribers. The sets of both published and sub-scribed metadata are maintained and tracked by the shared repository.

Depending on the combination, functionality for update propagation is to be implemented either in both publisher/subscriber repositories and the shared repository, only in publisher/subscriber repositories, or only in the shared repository. First is obviously the most inflexible approach because each tool and repository to be used must provide replication support. By contrast, fourth seems particularly promising for automating update propagation since it can be implemented entirely in the shared repository. However, both third and fourth require the implementation of comprehensive change detection, which can be complex due to the heterogeneity of the repositories and tools.

Enterprise Information Portals

Current data warehouses and repositories do not sufficiently support business users. The situation can be improved by the proposed shared repository integrating technical and business metadata and providing the base for various data access tools and end-user interaction. However, business users also demand access to a variety of other information not maintained in the company's data warehouse. In particular, unstructured or semis structured data such as documents, news, Web pages, or multimedia objects need to be accessed in a simple and uniform way. The recently introduced concept of enterprise information portals aims at meeting these requirements.

Eight components are critical for a complete EIP solution: security, caching, taxonomy, multi-repository support, search, personalization, application integration, and a metadata dictionary

- **Security**

Users need only sign on once for their session. From that point on, secure access to all content and applications is managed transparently. The EIP model respects the individual security models for each different application and data type, reducing security risks while improving ease of use. Users no longer have to remember multiple user ids and passwords for different applications, and the task of user administration is significantly reduced.

- **Caching**

Caching is an extremely effective strategy that yields precious network efficiencies, especially when a large number of users are accessing the same content. The EIP uses caching technology to improve response and performance.

- **Unified Search**

Completely transparent to the end user and can execute unified searches that access all structured and unstructured data sources, both internally and externally.

- **Taxonomy**

It is usually easier to navigate to relevant and important documents if they're organized by subject, rather than by the location where they're stored. Taxonomy is a hierarchy of subjects, allowing such navigation. The documents and information in an organization should be categorized according to that taxonomy to allow easy navigation. Once taxonomy has been created, new documents are automatically categorized into the hierarchy. This engine provides users with a powerful, easy-to-use navigation tool that will ultimately save an organization hundreds - even thousands - of hours that would otherwise be spent manually categorizing the documents and building a suitable, accurate taxonomy. These tools not only enhance the searching experience, but also give users the opportunity to truly understand the information that already exists in their organization.

Meta data Models

- **Multi-repository support**

It should support secure access to the broadest set of information and data repositories including groupware and email systems; document management and records management systems; line-of-business systems, including ERP applications; data warehouses; network file systems; and the Web.
- **Personalization**

It should have capability to support multiple levels of user personalization. These provide organizations with the ability to customize the appearance of the user interface for anyone important to your business - customers, employees, and trading partners. Individual users can personalize and customize their own environment to increase usability and user-friendliness.
- **Application Integration**

Application integration services enable the EIP to provide users with a centralized, unified, and consistent environment for interactions with all applications. The integration services greatly simplify both EIP deployment and interactions for the end user, thereby reducing costs and implementation time. The application integration services provide the EIP user the ability to act on the information, once business decisions have been made. The application integration services are central to the power of the EIP. Without these services, the EIP is just a one-way view into static information. These services provide the ability to act - to interact with the data and information that have been found, to edit a document, to trigger a change in the accounting system, to participate in an online discussion or email thread, to publish an updated document - all within the single, unified UI of the EIP.
- **Metadata Dictionary**

The metadata dictionary provides a single, seamless source for all data about the portal users and their preferences, and about the information sources used by the portal, regardless of where the data is stored, and what application generated it. This metadata repository aggregates data and information from all sources, providing quick and complete searching, and a full view of all of an organization's information.