CONF-9469B7--2

Metadata: a User's View

Francis P. Bretherton

University of Wisconsin - Madison
fbretherton@ssec.wisc.edu

Paul T. Singley

Oak Ridge National Laboratory
sin@ornl.gov

Abstract

An analysis is presented of the uses of metadata from four aspects of database operations:(1) search, query, retrieval; (2) ingest, quality control, processing; (3) application to application transfer; (4) storage, archive. Typical degrees of database functionality, ranging from simple file retrieval to interdisciplinary global query with metadatabase-user dialog and involving many distributed autonomous databases, are ranked in approximate order of increasing sophistication of the required knowledge representation. An architecture is outlined for implementing such functionality in many different disciplinary domains utilizing a variety of off the shelf database management subsystems and processor software, each specialized to a different abstract data model.

1: Introduction

This paper presents an analysis of issues concerning the management of metadata, from the perspective of a concerned scientific user. It is apparent that scientists and their information managers in many different disciplines are struggling with similar problems, yet with very little awareness of what has been accomplished in other areas. The goal, towards which this analysis is only a first step, is to establish a framework for dialog and partnership between users and computer scientists and database management system vendors within which each group can contribute to successful solutions. Because such a dialog has scarcely begun, the conclusions are necessarily tentative. They may, however, stimulate further discussion.

1.1: The authors' biases

The authors are atmospheric scientists, of whom only one (P.T.S.) has had formal training in information management. This study was motivated by frustration with the complexity of managing exponentially increasing streams of data from automated measurement systems and computerized analyses in a scientific environment which demands interdisciplinary collaboration across an ever widening range of natural and social science disciplines on issues that are of great public importance [1]. Documenting and understanding the changes in the global environment that are anticipated over the next 100 years, as well as the interactions of human societies with that environment, places enormous demands on an information system that is still evolving out of a myriad of disconnected and independent pieces [2]. A Workshop on Metadata for Scientific and Technical Data Management, sponsored by the IEEE Mass Storage Systems & Technology Technical Committee, was held in College Park, Maryland, May 16-18, 1994, with participants from a wide variety of backgrounds. During discussion of a white paper drafted by one of us (F.P.B.) and entitled "A Metadata Reference Model: A Strawman"[3], many different views were expressed [3]. Some of these views are reflected here, as is further reading of the published and grey literature.

1.2: What is metadata?

Metadata is generally loosely defined as "information which makes data useful". Metadata typically describes the structure of a data set or the interpretation to be placed on collections of similar items within that data set, rather than focusing on the individual instances usually regarded as primary data. It is, however, an overloaded term, meaning different things in different contexts.

*Managed by Martin Marietta Energy Systems, Inc., under contract DE-AC05-84OR21400 with the U.S. Department of Energy.

To a computer systems engineer metadata means physical level information like file names and formats, data types, and hash tables *i.e.* what is necessary to decode a sequence of bytes into basic elements recognized by a general purpose programming language. To a database manager metadata may mean the contents of a schema, *i.e.* names for all the classes of data objects in the database, a precise statement of all their attributes, and of the relationships between them, and a characterization of the questions that can be asked of the database. It may also mean a collection of rules and heuristics modeling standard operating procedures in some disciplinary domain, which can be used to frame and interpret interactions with users and other databases. To a physical scientist metadata may be a critical calibration constant, *i.e.* a number to be placed in a formula used to transform the data, or it may mean a natural language description of the measurement process of which the data was the outcome. To an intelligent novice exploring a new domain, it may simply be a guide to where to find more information.

With the increasing power and scope of computerized information management and the development of networking and distributed workgroups, all these senses of the term have to be integrated into a seamless whole. Such integration is needed to enable better communication both among groups of specialists, and, even more important, between specialists in different disciplines who need to establish a common factual base.

Viewed from this perspective, another aspect of metadata becomes apparent. Two humans exchanging messages make sweeping assumptions about the context in which those messages are to be interpreted. If those assumptions differ between the parties, misunderstandings are likely to arise. Most frequently, failures occur because one assumes the other is aware of some fact when in reality they are not. Or a term may unknowingly be used in two different senses. As humans, we are skilled in detecting symptoms of such misunderstanding, and at the first suspicion ask for clarification. Unfortunately, when computers are involved it is necessary to make all assumptions explicit, and sufficient metadata must be exchanged to ensure that both parties share a common basis of contextual knowledge. Thus what is adequate metadata depends on with whom or with what one is communicating. As the context of permissible communication is expanded, the breadth of ancillary information, *i.e.* metadata, has to grow too.

2: A functional analysis

2.1: Priorities for scientific databases

In the past, widely available database management systems have been dominated by commercial applications. These tend to have schemas that are relatively static, though the contents may be updated frequently. Priorities are transactional integrity and high volume query and update capability, though as the tools (such as SQL queries) have become available exploratory queries from management have become more prominent. Scientific databases, on the other hand, are generally continually adding not only more data from measurements, but also new types of data such as derived products that provide value added to most scientific users. Deletions tend to occur only *en masse*, when entire datasets are discarded as obsolete or not worth maintaining. Success is measured by the discovery of new relations within the data and by the new questions they stimulate, not by transactional efficiency. Thus the flexibility to deal with rapidly evolving schemas, and the effective documentation of the database contents must be fundamental priorities. Highly repetitive applications, and the efficient handling of complex transactions are much less prominent requirements.

With the advent of inexpensive personal computers and extensive networking, many organizations are trying to exploit the contents of locally maintained databases as enterprise-wide assets. This trend is particularly evident in the domain of computer aided manufacturing [4], where the synthesis of the originally separate computerized functions of design, parts management, process planning, and shop floor control can achieve major increases in resource productivity and quality of the finished product. To achieve effective integration across many different organizational subunits, basic issues have to be addressed [4]. It must be possible to gain an enterprise-wide view the current state of an activity which is spread over many departments. This requires computer interpretable codification of which departments do what and how they relate to each other. Such information is stored in a *data repository* (or more consistently a *metadata repository*), and must be linked to the process of assembling current data from many different locations. At the same time, the work flow over the computerized network must be structured so as to eliminate centralized functions which have a small but direct role in a large fraction of individual operations *e.g.* an enterprise-wide file directory. In a large organization, such choke points can easily lead to network saturation, or even worse, enforce synchronization on

otherwise independent parallel processes, negating the advantages of distributed computing with local control. Another issue is the consistent use of names throughout the organization, or at least the tabulation of local synonyms and the identification of homonyms. This has been addressed by the Information Resources Dictionary System (IRDS) project [5].

The scientific community faces similar challenges, but there are significant differences in emphasis. We suspect that, even within a single discipline, the measurement processes, theories, and established knowledge, which provide the technical work environment will *never* be coded with sufficient completeness to eliminate human judgement in its interpretation. Even if it were technically feasible, research scientists would not permit it, because they make their living by challenging and changing the established order. There is, by design, no central authority capable of imposing standards on such matters, so codification will be tolerated only insofar as it is uncontroversial and obviously essential to achieve more interesting things. The synthesis of information from disparate sources will thus in large measure continue to require interaction with a knowledgeable user. Exchange of "facts" between disciplines inevitably oversimplifies the qualifications that scientists are trained to make, so any utilization of such information in modeling has to involve fuzzy logic, statements of probability, or other heuristics which reflect uncertainty, and humans will be required at least to monitor the reasonableness of the outcome.

2.2: Aspects of metadata usage

To fix ideas on the role of metadata, the four aspects of scientific database operations shown in Figure 1 were considered in detail in the Strawman [3]. Only some generic conclusions are reported here.

The aspect "search, browse, retrieval" in Figure 1 is driven by a human user's need to answer questions efficiently, for example "Is it likely to be of use to me?", "Is it really what I want?" and "How do I get it?". A conclusion is that, in general, an effective scientific information system needs to offer, besides primary data from measurements, a rich set of documents containing a high density of scientific guide information, together with an expanding range of derived products such as analyses and theoretically inspired interpretations which summarize, index, or integrate the primary data with information from other sources.

The aspect "ingest, quality assurance, and reprocessing" is driven by the need to acquire a high quality dataset with a

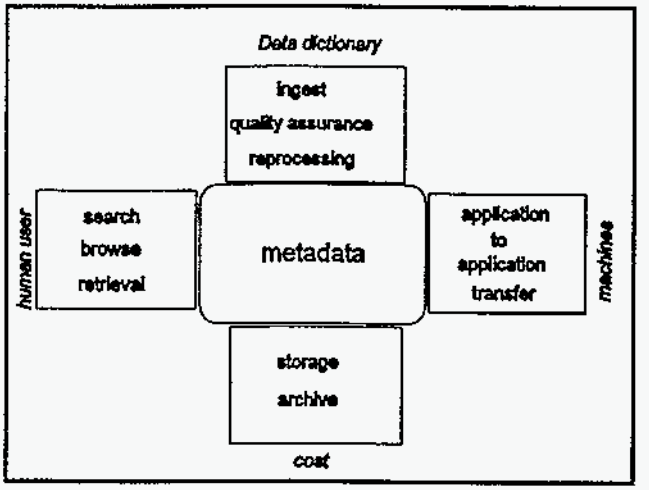


Figure 1 Aspects of metadata usage

precisely defined data dictionary, and to ensure the logical and scientific integrity of the database, together with adequate documentation of provenance, processing, and material relevant to data interpretation. It requires input both from expert scientists and from individuals skilled in knowledge engineering, and requires as much care and attention for derived products as it does for original measurements. CASE-like tools to assist in this process are urgently needed.

The aspect "application to application transfer" is driven by the need to transfer without human intervention information in the database to other databases on different platforms with different database management regimes. In a loose association of autonomous units, a good strategy for enabling graceful evolution is to provide at each level of the communication process not a single interface or transfer standard, but rather a handful of choices of such standards, together with a negotiation process whereby the parties can select that which best meets their needs. Addition of a new choice to the set, or subsetting an existing one which has fallen into disuse, then permits incremental adaptation to changing circumstances with the minimum of disruption.

The fourth aspect "storage and archive" was originally mistakenly supposed to be driven by need for efficient implementation of search and retrieval within an overall goal of total cost minimization, including the time of users of the database. After discussion, it became apparent that implementing this goal is a responsibility of overall management, whereas the provision for cost allocation purposes of data on design options and usage of system resources is just one of several similar requirements on practitioners of the discipline of database design and

operations. Meeting these requirements requires, however, organization of metadata in much the same manner as in other disciplines.

2.3: A taxonomy

At the IEEE workshop it became apparent that there are two different kinds of metadata: *guide* and *control*. Guide

A Taxonomy of Database Functionality - Part 1

ID. Overall Functionality:
Example(s);
Knowledge and @functions private to database;
Knowledge that must be shared or exchanged with user or another database;
Additional capabilities required of user or database environment.

1. Store and provide human interpretable descriptions:
File storage and retrieval;
Physical file representation, @store & @retrieve file;
Identifier as full node and file name
e.g. tigger.eds.lro.gov /usr/paul/file.name;
Map required information to node and file name, viewer for file format.

2. Locate information:
GCDIS Master Directory, Boolean library search;
@Parse query, map from keywords and index terms to data objects in database, @retrieve matching data objects;
Query expressed by keywords or index terms, identifier for retrieved data object(s);
Map required information to keywords and index terms, viewer for data objects.

3. Navigate among servers:
WWW, GILS [6];
Node names, maps from HTML buttons to objects pointed to (internal to the node and at other nodes), @retrieve and display objects;
Menus and descriptive text, viewer for data objects;
Map required information to menu terminology, HTML or equivalent language.

4. Advanced search for information:
Library search with feedback [7];
Content analysis which @generates maps of data objects to index terms, maps from index terms and synonyms to objects, @parse query, @modify default heuristics based on user profile, @rank data objects by similarity to query;
Query language conventions, principles of content analysis, history of user-system dialog;
Map retrieved data objects to required information, refine queries reflecting principles of content analysis.

5. Exchange of files - with partially self describing abstract data structures:
NetCDF [8], HDF;
Byte transform rules (*e.g* little and big endian), @transformation utilities;
Full node and file name, file format structure, format definition metadata;
Abstract format and data definition language, file generators and viewers.

6. Exchange of files - with controlled terminology for specified metadata:
Spatial data content and transfer standards [9], mmCIF [10];
Domain specific file format structure, map of keywords to objects in file, @check data integrity (permitted attribute values, etc);
Full node and file name, byte transform rules, subject domain, keywords, permitted attribute values;
Map required information to/from file format, data definition language, naming conventions.

7. Query of structured database:
SQL, Object oriented DBMSs;
Physical structure of database, @parse query, @compute value restrictions, @retrieve matching instances;
Logical structure of database (this may be the table names, key elements in all objects in database, relationships of key elements, or it may be an inheritance hierarchy), query, retrieved data objects;
Map required information to logical structure of database, query language.

8. Usage triggered, metadata controlled, transformations:
Calibration factors, unit conversions, gridding data, transformations between grids, statistical summaries, graphical display;
Allowable transformations for each "object", rules/heuristics underlying transformations and metrics for "best", @select "best" transformation, @execute transformation;
Specific data which trigger rules/heuristics, profiles expressing user preferences, results of transformations;
Understand principles governing transformations including conceptual structure & defaults, languages for describing data relationships and rules.

9. Provision of information relevant to life cycle costs:
Required for cost effective system design and operation must be allocatable to individual users and projects;
Cost model determining rules/heuristics for unit cost of each resource, @map resources consumed to each user/project, @statistics of usage patterns;
Cost of resources consumed for each user/project, information to assist cost minimization;
Project budgets/incentives for cost minimization, project strategies for usage optimization, management strategies for total information system optimization.

A Taxonomy of Database Functionality - Part 2

10. Computer aided capture of ingest and processing metadata to document pedigree of data products:
- Context dependent metadata entry forms for use by scientists, no examples known;
- @Adapt entry form to context as determined by previous entries, @check data integrity (permitted attribute values, etc)
- Identifier for data input/product being documented, conceptual level model describing data generation or processing procedures, allowable metadata values, and logical mapping between descriptions, values, and data products;
- Community consensus on appropriate conceptualization of the process, data and rule definition languages sufficient to describe the essential relationships of the process being documented, judgements on inclusion of optional metadata.

11. Share information between databases enabling distributed queries and transformations:
- Schlumberger [11], Federated Databases Sytems [12];
- Logical structure of database (this may be the table names, key elements in all objects in database, relationship of key elements), allowable transformations for each "object", rules/heuristics underlying transformations, and metrics for selecting 'best" transformation;
- Integrate private knowledge into global schema and metaschema, mapping of each name space into common identifiers, @parse query, @serialize elements of query and synchronize where required, @retrieve and assemble elements of query, @execute transformations, status of each database, status of transfers and transformations;
- Transfer language with mappings from conceptual level to different logical representations, name & synchronization services.

12. Remote query at conceptual level with user-database dialog to resolve ambiguities - assuming common terminology:
- See Wald & Sorenson [13]
- Logical structure of database, process description, @parse query, @retrieve matching instances, @describe alternative interpretations & retrieved objects;
- Conceptual structure of database and process, allowable query vocabulary and semantics, query, alternative interpretations of query & preferred interpretation, retrieved objects;
- Language & viewer for describing conceptual structure, discipline specific query vocabulary.

13. Global query to many distributed autonomous evolving databases:
- As (11) but scalable with central metadatabase and local monitoring agents, Hsu [14];
- Local database as (11), except hosts agent which @monitors relevant local metadata, @parse & @execute local and inter-database queries;
- Central metadatabase maintains complete structural and process knowledge base at the global conceptual level, interacts with local agents, receives global query from user, @parses and @serializes it and @transmits to the appropriate node an instruction form for inter-database query and result assembly ;
- User maps required information to metadatabase knowledge, language for describing metadata structure and semantics.

14. Interdisciplinary global query to many distributed autonomous databases with metadatabase-user dialog:
- Corresponds to (12) + (13) with incomplete user understanding of global structure and semantics;
- Local database as (13), metadatabase as (13) + allowable query vocabulary and associated syntax for each discipline, together with mappings of vocabularies to "objects" being manipulated and to a common vocabulary and syntax, @describe alternative interpretations of query & retrieved objects where default descriptions are tailored to a user profile maintained in user system;
- Discipline ID for user; a query in the corresponding vocabulary, descriptions of alternative interpretations of that query, the user-preferred interpretation, descriptions of retrieved objects;
- User maps required information to metadatabase descriptions of global information model. The system requires a language & viewer for describing the information model, a consistent mapping of discipline-specific vocabulary and semantics to a common base, standards for user profile.

metadata is intended solely for use by humans and is expressed in natural language. Control metadata, on the other hand, is intended primarily for use in directly affecting database or other computer system operations, though it is desirable that it be intelligible to scientists and/or database managers. It may be expressed in a controlled vocabulary and syntax, or, for example, embedded in the structure of a relational database, as the database of a specialized knowledge representation language such as Prolog, or even in a generalized entity relationship or semantic net language from which it is translated as needed for applications. If it is not available to the computer, the user has to substitute. However, the boundary between these two kinds of metadata is not fixed. Indeed the goal of a strategy for improving our handling of metadata should be to move that boundary towards increasing the control category in an evolutionary manner consistent with the investments of time and resources required and the benefits to be expected. The Tables outline different types of database functionality in order roughly of increasing sophistication of the control metadata required. The list is illustrative rather than exhaustive and should be considered a draft rather than a final product. For each numbered item the header indicates the approximate concept in mind, and the first displayed paragraph lists some examples. The following two displayed paragraphs attempt to list the knowledge required (control metadata) divided between that which need be explicit only to the computerized system, and that which must either be exchanged across the

interface with a user or precisely understood by both parties. The second paragraph also indicates in a general way some of the functions, distinguished by the @sign, that must be implemented within the database system and utilize the control metadata. The fourth paragraph contains a very crude indication of residual functionality which must become from the user or from the information system external to the database. A possible architecture addressing how this might be accomplished is discussed in the next section.

The knowledge that must be codified as control metadata is discipline specific. Though spatially extensive variables (fields) are common to the Earth sciences and some other disciplines such as aerodynamics, imposing some uniformity of approach to measurement, analysis, and modeling across these areas, practical techniques vary widely, and diversity tends to be more apparent than commonality. Thus, in general, sharing of methodologies and tools between disciplines must come at a more abstract level than the data structures based on regular grids which are so valuable for visualization [15] and certain types of modeling studies (see item 5 in Table 1). Direct comparison in item 6 of the format required for metadata to be exchanged among Geographic Information Systems [9] with that among Macromolecule Crystallography units [10] shows that each is highly structured, but sems to hold out little hope for direct sharing of processing software between the parent groups other than a file system. In addition, the wide range of functionalities listed matches the present realities of the different areas. However, deeper analysis does suggest a methodology that holds out considerable promise of economies of scale by combining in novel ways general purpose software that is mostly already widely available, while initiative by individual groups is not merely permitted but is positively required.

3: A conceptual architecture for a metadatabase

3.1: Objectives

There are several objectives that the desired metadatabase architecture should satisfy. First it should support any desired set of functionality drawn from the Tables or similar functionality. The software tools developed under this architecture should not be specialized to discipline or limited to functionality for a particular application instead these limits should be in the contents of the database. The architecture should allow the developers to take full advantage of existing software tools from different vendors as "off-the-shelf" parts. Finally, it should support the interdisciplinary use of a collection of metadatabases.

3.2: Definition of a metadatabase

The metadatabase envisioned in this paper is a repository for persistent information structured according to various abstract data models *e.g.* blobs, documents, spreadsheets, relational, object oriented, frames, rules, *etc.*, together with the capability for manipulating the contents in ways consistent with those data models. A metadatabase is partitioned into a number of logically disjoint subsystems, one for each abstract data model. Each subsystem consists of the information store and a software tool that supports a specific set of operations on the information store. Each such operation is a method which transforms an input message and a prior state of the contents of that information store into an output message and a subsequent state of the contents. The input and output messages to the subsystems are in a format required by the abstract data type *(e.g.* an SQL query). From an information systems perspective, each metadatabase subsystem can be viewed as an encapsulated data object communicating solely by message passing.

3.3: Description of the architecture

The proposed architecture for a metadatabase that meets the objectives set out above is a set of three conceptual layers with software tools joining adjacent layers (figure 2). The lowest, and most familiar, layer is the physical layer where the physical implementation of the system in the computing environment exists. Next is the logical layer where communication occurs with the metadatabase subsystems. This communication takes place in the input/output language appropriate for each subsystem software tool. Connecting the physical and logical layers are the software tools that are used today to manage and manipulate database and information stores. Examples of these tools range from relational database management systems RDMBS, to text indexing and searching systems such as a wide area information server (WAIS). Above the logical layer is the conceptual layer that consists of a representation of the contents of the metadatabase, expressed in a sufficiently expressive language or semantic net model to define concepts by description,

independent of their instantiation at the lower levels. Connecting the conceptual and logical layers is a metadatabase management system (MDBMS) which translates messages from the external interface into and out of representations of the metadatabase at the conceptual level; manages messages to, from, and between the various logical level subsystems; and translates results into user or other external views. The remainder of this section considers the elements of this architecture in greater detail.

Physical level:

The domain at this level involves things like files on various media, hash tables, *etc.*, or metadata for management of hierarchical mass storage. It is private to the software tool managing a particular subset of a metadatabase, and is not considered further here.

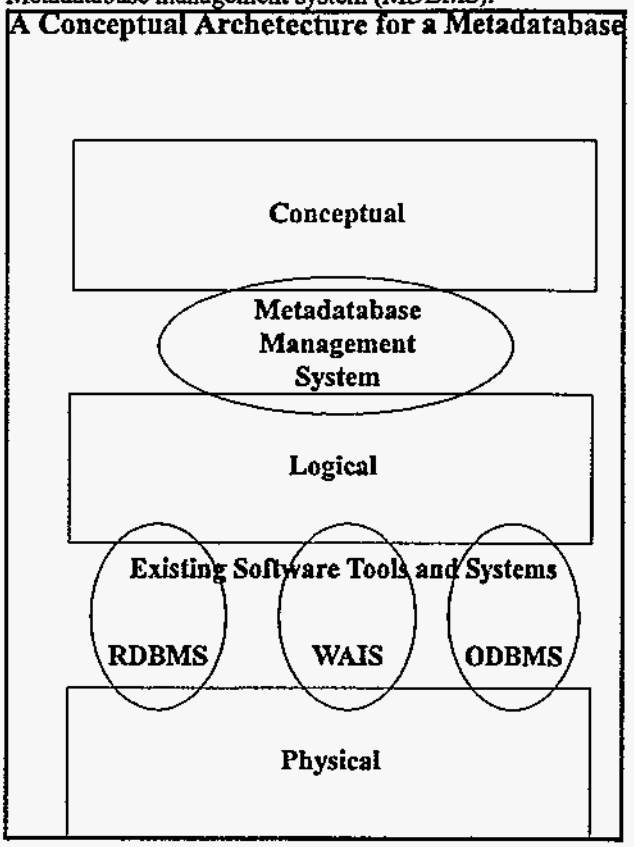
Logical Level:

The logical level is the level at which communication occurs with the metadatabase subsystems. Communication takes place in the input/output language appropriate for each subsystem software tool. A mentioned above the subsystems are different abstract data models. Entities and/or relationships relevant to a specific scientific discipline are expressed as an instance of the appropriate abstract data model. Thus the names of all explicit entities and relationships are defined together with permissible attribute types and values (e.g. "green" is a permissible value of attribute <color>, and is of type <string>) The domain of this instance (often confusingly also called a model) is the scientific knowledge within the discipline. The primary scientific data themselves describe particular instances of these concepts, processes and transformations.

The non-redundant operations that can consistently be performed on the database at the logical level, *e.g.* "create an instance of entity <name> with attributes <instanceid>, <attribute1>,<attribute2>", "retrieve all instances which match <query>", "create entity <name> with attributes (<attributename>, <attribute type>, <value restrictions>), are limited by the abstract data model itself, and for simplicity are assumed all to be supported by the software tool for that subsystem. They may affect only actual data held by the subsystem (generally metadata as seen by a scientist), or may alter the structure of that data. They are not normally regarded as part of the logical level data model, though they are crucial to understanding the overall functionality of the system. Messages between a subsystem and the MDBMS are in a language that describes operations of the software tool as well as the entities and relations in the subsystem data store.

One of the subsystems, the processor subsystem, has no internal state but executes an extensible set of methods each corresponding to an independent procedure (a function) which takes and returns a prescribed set of arguments. These procedures are themselves stored in the database and executed as required, using late binding to pass the arguments. This subsystem is the catchall for disciplinary specific logic or arithmetic that cannot be accomplished by the methods provided by the other subsystems, and should be used as sparingly as possible.

Metadatabase management system (MDBMS):



The MDBMS consists of two components, a Message Unit and a Translator. The Message Unit dispatches, receives and, manages messages from the various logical level subsystems and the external interface. The Translator translates the messages into and out of logical representation and instanceof the conceptual level model, as well as into and out

of various user or other external views.

To support the external interface the MDBMS provides an applications program interface (API). The user, or other applications, interact with the API to request and receive information from the MDBMS and the Translator casts the requests into controlled forms (frames) for entering or retrieving data from the various subsystems. Once the request is cast into the appropriate set of frames for the proper logical subsystems the Message Unit passes these frames to the logical subsystem tools that are needed to fulfill the request. As responses return form the subsystems the Message Unit retrieves the resulting frames and returns them to the Translator to be passed to the external interface. In addition, because it must understand all logical models to perform the task of receiving and responding to user requests, the Translator and Message Unit can be used to allow information to flow between the logical level subsystems by translating messages from logical model into conceptual model and back to logical model.

Conceptual level:

The conceptual level is a representation of the contents of the metadatabase, cast in a sufficiently expressive language or semantic net model to define concepts by description, independent of their instantiation at the lower levels. This level describes all entity and relationship names and their common aliases, permissible attribute values and defaults; provision for input and output; the roles of ephemeral entities created by events such as a user initiated query on the primary data, as well as changes in the metadata itself and; the modeling of operations (*i.e.* events) in addition to static entities and relations.

A key requirement at this level is that every concept (entity or operation) which is atomic in the various subsystem models must be describable in the conceptual model language. If so, by mapping the subsystem atoms onto constructs from the atomic concepts, everything known to the subsystems can be translated to the conceptual level. Conversely, if an external input to the conceptual model is expanded into its atomic concepts but cannot be complete mapped into those of the subsystems, then there are aspects of the input that are unknowable to the system. This information should then be passed by the Central Control Unit back to the API for corrective action. Another requirement is that the details of the conceptual model be subsumable in various ways into more general concepts which resemble as closely as possible those that are natural to a human user. For each different class of user there should be a default view that can be constructed by the translator from concepts in the database. Thus the contents of the metadatabase must be self-describing.

3.4: Discussion

This conceptual architecture closely resembles that has been prototyped by Hsu and collaborators [4], [14] for the domain of computer-aided manufacturing, and is consistent with implementations that are distributed over largely autonomous units and can evolve gracefully with time. They may also be scaleable to large numbers of units, though practical experience is still limited. The architecture has not yet been tried within the domain of scientific research, and it remains to be determined how well it supports an incremental path through the levels of functionality exemplified by the Tables.

A second important aspect of this approach is that increasing functionality requires representing in the database more domain specific knowledge, in particular the structure and semantics of standard concepts and processes for measurement, inference, and control within each scientific discipline. Such representation allows a user to interact with the information system at a more general level, responses being presented in the first instance based on default assumptions or probabilistic reasoning rather than detailed specifications. However, it is unlikely ever to be complete, and guide information, prepared by humans for humans but accessed from the database, will remain an integral part of the strategy. The sophistication and extent of this knowledge representation is likely to increase with time, depending on the needs, initiative, and resources of individual disciplines within the opportunity presented by a clear conceptualization of the specific tasks that have to be accomplished and the availability of appropriate general tools.

4: Conclusions

This paper has discussed the need to develop tools for building complex scientific metadatabase, and to a lesser extent the need to develop or find a modeling system for this tool development to take place. Less emphasized in this presentation but, critically important is the development or identification of languages and modeling methods required to develop the MDBMS. Without a language and modeling

methodology the constructions of generalizable software tools to develop scientific metadatabases can not occur.

This presentation heavily stresses the ability to integrate different types of available database management systems or processor software into a powerful tool for applications programmers or scientists to build complex metadatabases which they can understand. The key component of the MDBMS software is the translator which, to the authors' knowledge, has not yet been prototyped in a generalizable form. Yet some capability like this is sorely needed. As a simple example, note that relational abstract data model does not support restrictions on attribute values like $x < 1$. Thus, recording and enforcement of such items will require then access to a different subsystem, such as a rule-based model or an evaluator of algebraic expressions such as MathCAD.

References

1. Our Changing Planet: The FY 1995 U.S.Global Change Research Program, National Science and Technology Council, Washington D.C., 1994
2. Solving the Global Change Puzzle: A U.S. Strategy for Managing Data and Information National Academy Press, Washington D.C., 1991
3. Informal reports from Working Groups at the workshop, together with individual position papers including the Strawman, are available by anonymous ftp from ftp.clearlake.ibm.com
4. Hsu, C., M. Bouziane, L. Rattner, and L. Yee, 1991, Information Resources Management in Heterogeneous, Distributed Environments: A Metadatabase Approach. *IEEE Transactions on Software Engineering*, 17, 604-624
5. Judith J. Newton (ed). The Naming Forum: Proceedings of the Workshop on Data Entity Naming Conventions, NISTIR 4374, National Computer Systems Laboratory, Gaithersburg, MD 20899.
6. Christian, E., 1993, Government Indicator Locator Service (GILS): Draft Recommendations Available from echristi@usgs.gov
7. Larson, Ray R., 1992, Evaluation of Advanced Retrieval Techniques in an Experimental Online Catalog, *Journal American Socieyfor Information Science*, 43, 34 -53
8. Rew,R.K., & G.P.Davis., 1990. NetCDF: An Interface for Scientifc Data Access, *IEEE Computer Graphics and Applications*, 10, 76-82
9. Content Standards for Digital Spatial Metadata, Federal Geographic Data Commitee, 1994. Available from gdc@usgs.gov.
10. Bourne P.E. (ed), Proceedings of the First Macromolecular Chrystolographic Information File (CIF) Tools Workshop, Oct 15-18, 1993. Available from system@cuhchca.hhmi.columbia.edu
11. Barnett, J., 1993, The Schlumberger Data Model: A Meta Based Data Modeling Methodology , Available from barnett@austin.wireline.slb.com15
12. Sheth, A. P. and J. A. Larson. 1990. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys*, 22, 183-236
13. Wald, J.A. & Sorenson, P. , 1990, Explaining Ambiguity in a Formal Query Language, *ACM Transactions on Database Systems*, 15, 125-161
14. Babin, G. and C. Hsu, 1993, An Active Herogeneous Distributed Database Using a Rule-Oriented Concurrent Architecture. To be published
15. Treinish, L. A., 1993, Unifying Principles of Data Management for Scietific Visualization. lloydt@watson.ibm.com