Review on Metadata Management and Applications

Md. Anisur Rahman, Mehedi Masud
Computer Science Department, Khulna University, Bangladesh
e-mail: anis@mcrlab.uottawa.ca, mmasud@scientist.com

Abstract—In large scale distributed and heterogeneous software systems, metadata is considered as one of the most important components for the architecture and construction of information systems. Metadata is utilized to improve communication between heterogeneous information systems – for the purposes of obtaining and providing information, for communication between user client workstations and information servers, and for electronic business between information systems. Metadata management is much challenging and is a hot topic for research at present. In this paper, we review work that deals with metadata in three different points of view. Firstly, we survey some proposals about applying model management to classical metadata problems. Then we inspect how Metadata can help managing multimedia data. Finally, we review some Web-Service specific metadata management issues. We survey approaches that are of foundational nature as well as those that are application-oriented. We believe that combining both foundational and practical aspects is important to achieve the goal of managing metadata with the same ease as, for example, database management systems have achieved in the management of classical relational data.

Keywords: Heterogeneous databases; Model management for metadata problems; Composition of metadata

1. INTRODUCTION

Metadata are considered as the future of networked information systems. The ubiquity of the World Wide Web (WWW), the increasing need for access to heterogeneous distributed information and the increased use of multilingual and multimedia sources all demand some common representation and understanding of metadata[11]. Metadata is attached to data to aid in its interpretation. Information systems process and interpret the data using the associated metadata which, in more and more cases, are getting very large. Hence the importance of dealing with an appropriate way of managing these large metadata sets.

Model management is a new approach to metadata management that offers a higher level programming interface than current techniques. The main abstractions are models (e.g., schemas, interface definitions) and mappings between models. It treats these abstractions as bulk objects and offers operators for matching metadata objects, merging them, computing the difference between given metadata objects, composing these objects, and generating them. It is a powerful approach to metadata management which is generic in the sense that it not limited to a specific language or application. Generic model management [3,4,13] aims at simplifying the development of metadata-intensive applications, such as data integration, software engineering, website management, network modeling, management, and network modeling.

Such applications manipulate a variety of models (database schemas, XML schemas, UML / ER diagrams, ontologies, etc.) and mappings between models (SQL view definitions, XSLT transformations, XML-to-relational shredding specifications, ER-to-SQL DDL mappings, etc.).

Finding correspondences between models is required in many application domains, such as data integration, e-business, data warehousing, web services, and peer-to-peer semantic query processing. This task is often referred to as matching. In generic model management, matching is embodied in the operator Match. The operator takes two models as input and returns a mapping between the models as output. This operator is special as matching typically involves information that is not contained in the input models [16].

Real applications in industry face serious problems on how to handle large amounts of a variety of digital media. Therefore, there is a growing demand for database and information systems support in the area of modeling, management, and processing digital media. There is a need to explicitly capture a fair amount of content-information as well as application-specific semantics by means of a variety of metadata, e.g. multimedia indexes, attribute-based annotations, and intentional descriptors, to allow appropriate access to, selection of, and processing of digital media (or multimedia). One of the key problems to be solved is the development of multimedia metadata. Multimedia Content Description Interface (aka MPEG-7) developed by the Moving Picture Experts Group (MPEG) addresses this content management challenge though it has some limitations.

As web services become more prevalent, tools will be needed to help users find, filter and integrate these services. WSDL (Web Services Description Language) is used as the metadata language of web services. Composing existing services to obtain new functionality will prove to be essential for both business-to-business and business-to-consumer applications. The dynamic composition of services is difficult.
using just the WSDL descriptions. This is essentially due to the hardness of dynamically dealing with metadata management.

2. METADATA: A SHORT OVERVIEW

Metadata describe a data source, a particular collection of data (a file or a database or a table in a relational database or a class in an object-oriented database), an instance of data (tuple in a relational database table, object instance in a class within an object-oriented database) or data associated with the values of an attribute within a domain, or the particular value of an attribute in one instance. Metadata can describe data models. Metadata can also be used to describe processes and software. It can describe an overall processing system environment, a processing system, a process, a component of a process. It can describe a suite of software, a program, a subroutine or program fragment, a specification. It can describe an event system, an individual event, a constraint system and an individual constraint. It can describe a process and/or event model. Metadata can describe people and their roles in an Information Technology (IT) system. It can describe an organization, a department, individuals or individuals in a certain role.

The topic of Metadata has recently found more limelight than in the past, largely due to a sudden realization of its necessity in making the WWW usable effectively. Metadata is essential for WWW to scale up to an astronomical number of users, for finding information of relevance, and for integrating together data and information from heterogeneous sources. Metadata are essential for refining queries so the latter return what the user intends. It is also essential for understanding the structure of information, its quality and its relevance. Metadata are required for explaining answers from ever more complex information systems. It assists in distilling knowledge from information and data. It assists in multilingualism and in multimedia representations. The engineering of systems from components (data, processes, software, events, and subsystems) is assisted by metadata descriptions of those components. Metadata have been used in information systems engineering for many years, but usually in a specialist, one-off and uncoordinated way.

Commonly, the metadata have been human readable, but not specified sufficiently formally, nor accepted sufficiently widely, to be interpreted unambiguously by IT systems. In addition to information systems such as WWW (update, retrieval) and systems engineering as described above, metadata are essential for e-business from advertising through catalogue information provision through initial enquiry to contract, purchase, delivery and subsequent guarantee or maintenance.

3.2 Difficulties in Metadata Management

There are three identifiable types of difficulties in metadata management, namely metadata definition and management, technology, and standards [11]. Metadata definition and management is about defining, creating, updating, transforming, and migrating all types of metadata that are relevant and important to a user’s objectives. Metadata management technology includes metadata design tools that allow users to model the schema of metadata across all data sources, and metadata repository systems that allow the users to extract metadata from various data sources, search and query metadata, and exchange metadata with other users, etc. Metadata standards include not only those for modeling and exchanging metadata, but also the vocabulary and knowledge ontology. These difficulties have stunted universal adoption of metadata management technologies. There are efforts such as the Dublin Core Metadata Initiative’s Metadata Terms to develop standards on certain metadata vocabularies. Standard knowledge ontology is also needed to organize such types of metadata as content metadata and data usage metadata. With respect to the vocabulary and knowledge ontology, where there are suitable industry standards, the standards may be adopted in full or in part. Appropriate procedures need to be defined and followed within the enterprise in documenting the capture, update, transformation, migration, replication of metadata and relevant transformation rules and business rules, etc [11].

2.2 Future of Metadata

Metadata have moved centre-stage as one of the most important components of the architecture and construction of modern information systems. The idea of separating the primary information resources from data and processes (metadata system) to provide access to those resources is extremely important. This allows changes of access policy – such as changes in access restrictions for certain kinds of users in certain roles, changes in categorization and classification, and changes in descriptive metadata depending on viewpoints of different authorized users – without accessing the data resource itself [12].

People who are publishing valuable information to the Internet want to be able to create or at least to control the metadata describing their resources. Metadata that are generated not by a single entity such as a search engine, but by many different entities requires some recognized standard metadata formats. Without standard metadata formats and semantics, metadata would be just as unprocessable and unmanageable as the original data. Existing barriers in business, modeling, and technology will have to be addressed for metadata in order for them to play the important role of alleviating barriers between heterogeneous users and applications. Metadata collection has attained a sufficient level of maturity; however, metadata management today is at an elementary phase. In the future, there is a need for an extended ecology of metadata artifacts that will constantly evolve [7]. Before increased automatic metadata management can be
readily exploited in enterprise activities, such as e-commerce, education, and government, metadata will require extensible models, richer nuances, and underlying trust mechanisms. The future of the Internet will rely on this evolution.

3. GENERIC MODEL MANAGEMENT

Many information system problems involve the design, integration, and maintenance of complex application artifacts, such as application programs, databases, web sites, workflow scripts, formatted messages, and user interfaces. Engineers who perform this work use tools to manipulate formal descriptions, or models, of these artifacts, such as object diagrams, interface definitions, database schemas, web site layouts, control flow diagrams, XML schemas, and form definitions. This manipulation usually involves designing transformations between models, which in turn requires an explicit representation of mappings, which describe how two models are related to each other. Some examples are given in [3] as follows:

- mapping between class definitions and relational schemas to generate object wrappers,
- mapping between XML schemas to drive message translation,
- mapping between data sources and a mediated schema to drive heterogeneous data integration,
- mapping between a database schema and its next release to guide data migration or view evolution, mapping between an entity-relationship (ER) model and a SQL schema to navigate between a database

As the above applications mostly involve manipulating descriptions of data, rather than the data itself, Bernstein et al. [3, 4] classify these as metadata management applications.

Today’s approach to implementing such applications is to translate the given models into an object-oriented representation and manipulate the models and mappings in that representation. Most of manipulation is programmed using object-at-a-time primitives. In [3,13], the authors have proposed to avoid this object-at-a-time programming by treating models and mappings as abstractions that can be manipulated by model-at-a-time and mapping-at-a-time operators. They believe that an implementation of these abstractions and operators, called a model management system, could offer an order-of-magnitude improvement in programmer productivity for metadata applications.

3.1 Models and Mappings

Models: Models are defined in [3] as a set of objects, each of which has properties, has-a relationships, and associations. A model is identified by the root object and includes exactly the set of objects reachable from the root by paths of has-a relationships.

Mappings: A mapping between models $M1$ and $M2$ is a model, $map12$, and two morphisms, one between $map12$ and $M1$ and another between $map12$ and $M2$. Thus, each object $m$ in mapping $map12$ can relate a set of objects in $M1$ to a set of objects in $M2$, namely the objects that are related to $m$ via the morphisms. For example, in Figure 1, $Map_{we}$ is a mapping between models $Emp$ and $Employee$, where has-a relationships are represented by solid lines and morphisms by dashed lines.

Model Management Operators: In a model management system, models and mappings are syntactic structures. They are expressed in a type system, but do not have additional semantics based on a constraint language or query language. Despite this limited expressiveness, model management operators are powerful enough to avoid most object-at-a-time programming in metadata applications. For a complete solution, metadata problems often require some semantic processing. Summary of the main model management operators are as follows [3]:

- **Match** – takes two models as input and returns a mapping between them
- **Compose** – takes a mapping between models A and B and a mapping between models B and C, and returns a mapping between A and C
- **Diff** – takes a model A and mapping between A and some model B, and returns the submodel of A that does not participate in the mapping
- **ModelGen** – takes a model A, and returns a new model B based on A (typically in a different data model than A’s) and a mapping between A and B
- **Merge** – takes two models A and B and a mapping between them, and returns the union C of A and B along with mappings between C and A, and C and B.

Application Scenarios: The operators mentioned in the previous section can be used in various applications like Schema Integration, Schema Evolution, Round-trip Engineering etc. The following example [3] illustrates how the operators might be used to generate data warehouse loading script.

**Problem:** Given a mapping $map1$ from a data source $S1$ to a data warehouse $SW$, another mapping is required to be created between a second source $S2$ to $SW$, where $S2$ is similar to $S1$. Figure 2 depicts the problem.

**Solution:** The following steps will solve the problem.

Call Match($S1$, $S2$) to obtain a mapping $map2$ between $S1$ and $S2$.
Call Compose(map1, map2) to obtain a mapping map3 between S2 and SW, which maps to SW those objects of S2 that correspond to objects of S1.

Call Diff(S2, map3) to find the sub-model S3 of S2 that is not mapped by map3 to SW, and map4 to identify corresponding objects of S2 and S3.

Call ModelGen(S3) to generate a warehouse schema for S3 and merge it into SW.

In [3,4,21,23], the authors have studied the concepts and algorithms for generic model management. A prototype of a generic model management (Rondo) has been given in [18]. Algebraic operators that are used to manipulate models and mappings are introduced in [16]. State-based semantics of the operators have been clarified and some algorithms have been given in [3,23].

3.1.1 Future Scope of Generic Model Management

Developing formal semantics for the operators that combines the state-based and a more structural approach, developing practical materialization algorithms, finding appropriate architectures and techniques for coupling model management applications, tools, and conventional programming languages, developing powerful user interfaces for building model-management solutions and supporting user feedback during script execution, finding mechanisms for deducing equivalence and entailment of scripts, etc. Furthermore, applying model management to practical problems will help validating the expected answer from MAPONTO is of the form:

T: Emp(ssn, name, dept, proj).

O: Employee(x), O: hasSsn(x, ssn), O: hasName(x, name), O: Department(y), O: works_for(x, y), O: hasNumber(y, dept), O: Worksite(z), O: controls(y, z), O: hasName(z, proj)

Here the prefixes T and O are used to distinguish predicates in the relational schema and the ontology.

To achieve the objective, the authors envision a two step process, where (a) the columns Ai of each table are linked to elements in the ontology; then (b) a formula is proposed by the tool on the basis of heuristics. The basic idea underlying MAPONTO is to represent the ontology as a graph consisting of nodes (corresponding to concepts) connected by edges (corresponding to properties). Semantic connections in the ontology, expressed in the formula Φ, are then based on paths in this graph. The policy “Fewer connections are better” lead to look for minimal–cost spanning trees connecting the concepts which have one or more properties corresponding to table columns- called Steiner trees. Such a tree is the translated to a logical formula by joining the concepts and properties encountered in it. For example, if concepts C and D are connected by the tree consisting of edges p and q, traversing intermediate node G, the formula produced is C(x) .p(x,y), G(y), q(y,z), D(z).

In [1] the authors considered ontologies expressed by first order formulas. We think there is scope for extensions of this work to cope with the ontologies which are expressed by Description Logic. There is also scope for finding mapping between ontologies and other metadata (like XML schemas, MPEG-7 documents etc).

3.2 Ontology for Metadata Management On Web

Ontology typically contains a hierarchy of concepts within a domain and describes each concept’s crucial properties through an attribute-value mechanism. Further relations between concepts might be described through additional logical sentences. Constants are assigned to one or more concepts in order to assign them their proper type. Ontologies may play a major role in supporting the metadata management over the web. E.g, schema-to-ontology matching can be used to compose mappings between schemas with high accuracy. In [1,6,13] some interesting works can be found where the authors tried to show how ontology can be used in metadata management. The next sub-section briefly describes the work of Mylopoulos et. al. (MAPONTO) [1] which tries to refine semantic mappings from relational tables to ontologies by using heuristics.

3.2.1 MAPONTO Approach

In MAPONTO a framework is assumed where an ontology and a relational schema are given. The general objective is to find a mapping relating predicates in the ontology and relational tables. For example, given the ontology in Figure 3 [1], and relational schema Emp(ssn, name, dept, proj), an expected answer from MAPONTO is of the form:

T: Emp(ssn, name, dept, proj).

O: Employee(x), O: hasSsn(x, ssn), O: hasName(x, name), O: Department(y), O: works_for(x, y), O: hasNumber(y, dept), O: Worksite(z), O: controls(y, z), O: hasName(z, proj)

Here the prefixes T and O are used to distinguish predicates in the relational schema and the ontology.

To achieve the objective, the authors envision a two step process, where (a) the columns Ai of each table are linked to elements in the ontology; then (b) a formula is proposed by the tool on the basis of heuristics. The basic idea underlying MAPONTO is to represent the ontology as a graph consisting of nodes (corresponding to concepts) connected by edges (corresponding to properties). Semantic connections in the ontology, expressed in the formula Φ, are then based on paths in this graph. The policy “Fewer connections are better” lead to look for minimal–cost spanning trees connecting the concepts which have one or more properties corresponding to table columns- called Steiner trees. Such a tree is the translated to a logical formula by joining the concepts and properties encountered in it. For example, if concepts C and D are connected by the tree consisting of edges p and q, traversing intermediate node G, the formula produced is C(x) .p(x,y), G(y), q(y,z), D(z).

In [1] the authors considered ontologies expressed by first order formulas. We think there is scope for extensions of this work to cope with the ontologies which are expressed by Description Logic. There is also scope for finding mapping between ontologies and other metadata (like XML schemas, MPEG-7 documents etc).

4. METADATA FOR MULTIMEDIA

Metadata play a far more important role in managing multimedia data than does the management of traditional (well-) structured data or information retrieval techniques applied to text-only data. The following section highlights the necessity of metadata for digital media.

4.1 Necessity of Metadata for Multimedia

Various digital media or components of multimedia data involve very large raw data volume. This has consequences on effective management and retrieval of the digital media.
Content-based retrieval on raw data means that the query capabilities are limited to the number of available matching algorithms. Performance is lacking when queries are executed on large data sets. The use of metadata of the digital media seem to be a promising approach to enhance querying and processing and to improve response time as metadata will be of much less than the digital media themselves [5].

Semantics of multimedia data like images, video, and audio is implicit to the raw media data. By analyzing and processing, semantics can be made explicit to some extent on different abstraction level, from feature values to knowledge-based concepts. Metadata describing this semantics explicitly may not be sufficient for exact-match querying. A different querying paradigm is needed to allow a proper mapping of the user’s ideas to the explicitly available semantics [20]. Metadata that is derived or extracted from digital media have the additional advantages of being more amenable to traditional data retrieval and manipulation techniques than the raw digital media. For fuller exploitation of data and to provide information system interoperability it is necessary to support correlation of data. It is often more convenient and useful to identify and specify correlation at the metadata level rather than between raw data items [5, 25]. Keeping all these challenges in mind Moving Picture Experts Group came up with a standard for describing Multimedia Metadata and named it MPEG-7.

We can attach audio, visual, annotation, and content management description tools to the segments to describe them in detail. MPEG-7 Visual description tools include the visual basic structures (such as description tools for grid layout, time series, and spatial coordinates) and visual description tools that let us describe color, texture, shape, motion, localization, and faces. MPEG-7 Audio description tools comprise the audio description framework and high-level audio description tools that let us describe musical instrument timbre, sound recognition, spoken content, and melody. The Semantic description tools that let us describe the content with real-world semantics and conceptual notions: objects, events, abstract concepts, and relationships. We can cross-link the semantic and structure description tools with a set of links.

The MPEG-7 description tools are a library of standardized Descriptors and Description Schemes. This library is presented on the basis of the functionality they provide, but in practice, we can combine them into meaningful sets of description units making use of the Schema tools. Each application builder might want to select a subset of Descriptors and Description Schemes.

MPEG-7 definitions are expressed solely in XML Schema [12]. XML Schema has been ideal for expressing the syntax, structural, cardinality and datatyping constraints required by MPEG-7. In order to make MPEG-7 accessible, re-usable and interoperable with other domains the semantics of the MPEG-7 metadata terms need to be expressed in an ontology using a machine-understandable language. There is scope for building such an ontology represented in more expressive languages (e.g. RDF Schema).

5. METADATA FOR WEB SERVICES

Web services are application components. They communicate using open protocols and are self-contained and self-describing. They can be discovered using UDDI and be used by other applications. XML is the basis for Web services. Basic web services platform elements are SOAP (Simple Object Access Protocol), UDDI (Universal Description, Discovery and Integration), and WSDL (Web Services Description Language) [9]. Among them, WSDL is used as a metadata description language for web services.

WSDL is an XML-based language for describing Web services and how to access them. It is written in XML. WSDL is used to describe Web services. It also specifies the location of the service and the operations (or methods) the service exposes.

A WSDL document defines a web service using these major elements [10]:

<table>
<thead>
<tr>
<th>Element</th>
<th>Defines</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;portType&gt;</td>
<td>The operations performed by the web service</td>
</tr>
<tr>
<td>&lt;message&gt;</td>
<td>The messages used by the web service</td>
</tr>
<tr>
<td>&lt;types&gt;</td>
<td>The data types used by the web service</td>
</tr>
<tr>
<td>&lt;binding&gt;</td>
<td>The communication protocols used by the web service</td>
</tr>
</tbody>
</table>

The main structure of a WSDL document looks like the following:

```xml
<definitions>
  <types>
    definition of types........
  </types>
  <message>
    definition of a message....
  </message>
  <portType>
    definition of a port.......;
  </portType>
  <binding>
    definition of a binding....
  </binding>
</definitions>
```

A WSDL document can also contain other elements, like extension elements and a service element that makes it possible to group together the definitions of several web services in one single WSDL document. The <portType> element is the most important WSDL element. It defines a web service, the operations that can be performed, and the messages that are involved. The <portType> element can be compared to a function library (or a module, or a class) in a traditional programming language. The <message> element defines the data elements of an operation. Each message can consist of one or more parts. The parts can be compared to the parameters of a function call in a traditional programming language. The <types> element defines the data type that are used by the web service. The <binding> element defines the
message format and protocol details for each port.

```xml
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portTypename="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
  </operation>
</portTypename>

<binding type="glossaryTerms" name="b1">
  <soap:binding style="document">
    <transport protocol="http://schemas.xmlsoap.org/soap/http"/>
    <operation>
      <soap:operation
        soapAction="http://example.com/getTerm"/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
  </soap:binding>
</binding>
```

Figure 4: An example of WSDL document [10]

**Example:** In the example of Figure 4, the port "glossaryTerms" defines a request-response operation called "getTerm". The "getTerm" operation requires an input message called "getTermRequest" with a parameter called "term", and will return an output message called "getTermResponse" with a parameter called "value". The binding element has two attributes - the name attribute and the type attribute. The name attribute defines the name of the binding, and the type attribute points to the port for the binding, in this case the "glossaryTerms" port. The soap:binding element has two attributes - the style attribute and the transport attribute. The style attribute can be "rpc" or "document". In this case, "document" was used. The transport attribute defines the SOAP protocol to use. In this case, HTTP was used. The operation element defines each operation that the port exposes. For each operation the corresponding SOAP action has to be defined. The way of encoding input and output must be specified (e.g. "literal" etc.).

WSDL defines the interface of a Web service in terms of what are the messages that are exchanged. A WSDL document also structures the messages into pairs (that correspond to the operations provided by a service). However, WSDL does not contain any further information specifying what is the correct order of invocation of the various operations. If an operation...

6. **Summary**

The use of metadata in information systems is not new. But earlier generations of metadata management systems did not provide adequate facilities for managing metadata and there were no standards for metadata management tasks. Keeping this in mind, we reviewed some metadata related research trends and tried to summarize them in this paper.

Bernstein et al. [21,23] made a proposal to make database systems easier to use for applications like transforming data from one model to another model by making the "model" and "model mapping" first-class objects with special operations (e.g. Match, Merge, Extract etc.) that simplify their use. They presented a taxonomy that covers many of the existing approaches of Schema Matching [23, 27].

Matching of two semantically similar metadata description (Schema Matching) is the most challenging task of Metadata management. So, we gave special attention to it. Melnik [16, 17, 18], Madhavan [13,14], and Fagin [26] came up with three different algorithms to solve this problem. Rondo [18], the prototype developed by Melnik, is an attempt for generic model management. In [16] he clarified the semantics of the model management operators in a state-based fashion. Madhavan [13,14] developed a prototype named Cupid [13] for discovering mappings between schema elements based on their names, data types, constraints, and schema structures. Some of his innovations are the integrated use of linguistic and structural matching, context-dependent matching of shared types, and a bias toward leaf structure where much of the schema content resides.

Fagin et al. [8] formulate a formal semantics for the composition of mappings, and study the definability and computational complexity of this operation. To our knowledge, taken together with the work reported in [14], the work in [8] constitutes the only attempts to date to tackle the study of metadata at the foundational level.

An ever increasing amount of multimedia data is being made available on the web. Metadata are expected to play a pivotal role in managing these multimedia data. MPEG-7 [15] is a standard for multimedia metadata. It copes with many challenges of multimedia management but still suffers from some limitations. As we saw, Van Ossenbruggen et al. [2,19] identify the problems and requirements regarding the semantic content description of media units. They compare Mpeg-7 and Semantic Web approaches and discuss syntactic, semantic, and ontological problems. They also analyze the ability of the W3C and ISO efforts to define syntax for describing media semantics. Their finding is that the closed approach of Mpeg-7 hinders the required modularity for description design, obstructing the needed interoperability of syntactic and semantic levels. They suggest that this can be overcome by expressing conceptual and contextual descriptions in semantic languages such as RDF, RDF schema, or OWL. They also suggests for making an attempt to move MPEG toward modularity.

Klas et al. [17] presents critical requirements for the management of MPEG-7 media descriptions and the resulting
consequences for XML database solutions. They identify the limitations of current database solutions with respect to the management of MPEG-7 media descriptions. They suggest for the development of new generation of XML database solutions which recognize the central importance of exploiting the type information contained in schema definitions for the adequate management of XML documents. Their finding is that the new solutions should include sophisticated (multidimensional) value, text, and path index structures, profound extensibility with custom functionality and index structures, and classic DBMS functionality such as transactions, fine-grained concurrency and access control, and reliable means for backup and recovery.

Hammiche et al. [5] identifies the problems of finding multimedia data that fulfill the requirements of user queries. They propose a framework for querying multimedia data based on a tree embedding approximation algorithm, combining the MPEG-7 standard and ontology. WSDL [10] is a standard for describing (metadata) Web Services. It has some good features but is not enough to deal with applications like dynamic composition of web services [22, 24]. The main limitation of WSDL is that the input and out of it are text. For combining concepts ontologies are needed.

7. Conclusion and Future Directions

This paper has reviewed metadata related research from a certain number of different points of view, namely Model Management for metadata, Multimedia metadata, and Web Service related metadata. There are many lessons that can be learnt from the main trends presented in terms of future directions about the research on metadata. Principled study of metadata: Metadata is an area that has been plagued with ad hoc solutions. The whole intent of work on metadata management has been to remove as much as possible ad hoc solutions from the handling of the main metadata management tasks in favor of a more principled way of handling these tasks.

Foundations: The idea of organizing the main operations for managing metadata into a set which constitutes an algebra amounts to bringing metadata research closer to a foundational approach. Up to date, metadata have mainly been handled by practitioners who deal with heterogeneous data applications. Only recently have trends emerged towards the study of foundations of metadata. Like in other subfields of the theory of data management, we anticipate that investigating foundations of metadata by using the usual computer Science tools for this task like logic, formal languages, and automata will yield more substantial results of lasting and far reaching consequences than the more ad hoc and application oriented approaches used to date. We anticipate seeing two directions emerging from the current state of the art in metadata research. On one hand, we foresee the further development of a core set of metadata related operations that would ultimately yield an algebra, much in the spirit of the relational algebra in relational databases; on the other, there is a need for a logical counterpart of the metadata algebra.

Applications: It is obvious that the increasing heterogeneity of today’s networked data sources will lead to an increase of the amount of metadata intensive applications in the future. We believe that the combination of foundational and practical aspects is crucial to managing the increasingly complex amount of metadata that are being associated with applications.

REFERENCES


Anisur Rahman received his PhD in Computer Science at the University of Ottawa, Canada. He completed his Masters from Asian Institute of Technology, Thailand, in 2000. He is also a faculty at Computer Science Department, Khulna University, Bangladesh. His research interests are data integration, schema mappings, model management. He has published several research papers at international journals and conference proceedings.

Mehedi Masud received his PhD in Computer Science at the University of Ottawa, Canada. He is an Associate Professor at the Department of Computer Science, Taif University, KSA. His research interests include issues related to P2P and networked data management, query processing and optimization, and information security. He has published several research papers at national and international journals, conference proceedings.