MODELING AND SERVICES FOR ADAPTIVE COLLABORATIVE DELIVERY OF ANNOTATED MULTIMEDIA RESOURCES

Jérôme Godard

DOCTOR OF
PHILOSOPHY

Department of Informatics,
School of Multidisciplinary Sciences,
The Graduate University for Advanced Studies (SOKENDAI)

2004 (School Year)

March 2005

A dissertation submitted to
the Department of Informatics,
School of Multidisciplinary Sciences,
The Graduate University for Advanced Studies (SOKENDAI)
in partial fulfillment of the requirements for
the degree of Doctor of Philosophy

Advisory Committee:

Frédéric Andrès	National Institute of Informatics, SOKENDAI
William Grosky	University of Michigan
Hiroshi Ishikawa	Tokyo Metropolitan University
Yusheng Ji	National Institute of Informatics, SOKENDAI
Akifumi Makinouchi	Kyushu University
Katsumi Maruyama	National Institute of Informatics, SOKENDAI
Kinji Ono	National Institute of Informatics

(Alphabetical order of last name except chair)

Abstract

Adaptive systems are becoming essential for supporting the overload and the diversity of digital documents to be archived, retrieved and disseminated. They indeed represent the most promising solution for individuals to be able to handle the amount of data which they are daily flooded with during their professional activities or on their personal devices. However, providing adaptive management of heterogeneous resources remains an important research issue as it requires extensive and global environmental knowledge management to be effective. Communities, as they are sharing interests and accesses to resources, present very interesting characteristics that enable systems to deliver automated and personalized services; the scope of such processes being to take advantage of collaborative involvement in order to provide relevant knowledge management to users, to ensure the consistency of data manipulation, and to improve the distribution of resources within communities.

This dissertation presents a collaborative information management framework dedicated to the personalized delivery of multimedia documents. We first propose an Information Modeling for Adaptive Management (IMAM) which is an innovative set of algebraic structures making it possible to categorize and manipulate any piece of environmental knowledge that is useful for an enhanced delivery of data. In addition, this modeling is the generic basis for the definition of operators and services in collaborative environments; these functions rely on contextual information for personalizing the retrieval and the distribution of multimedia documents. Then, after ensuring an efficient storage management of annotations in XML with a mapping to the Extended Binary Graph data structure, we define a powerful indexing strategy that makes the retrieval of annotations faster. Finally, we design adaptive services based on IMAM's formal modeling. These services comply with complex distributive models structures such as Peer-to-Peer and perform personalized query optimization and data placement in collaborative environment.

Acknowledgements

Many people have contributed to this work and provided efforts during my years at National Institute of Informatics in Tokyo. Thus I want to thank everyone who helped me shape the ideas explored in this dissertation.

I wish to express my sincere gratitude to my principal advisors, Prof. Emeritus Kinji Ono and Prof. Katsumi Maruyama, for their guidance and patience. Their support encouraged me to pursue my research as a SOKENDAI Ph.D. student.

I am grateful to Ass. Prof. Frédéric Andrès for being the driving force behind the work produced during the last three years. Our discussions and sometimes disagreements have been a very good basis for elaborating and structuring my research horizon; he gave me the freedom and means to mature my ideas while always pointing me interesting directions. His trust has been very pleasant as he allowed me to follow the leads I was believing in.

I would like to thank Prof. William Grosky for providing valuable remarks, allowing me to improve some papers drafts and clarify my arguments. He showed me what it takes to step back and see a big picture, and yet keep the details in focus.

I would also like to thank all the other committee members, Ass. Prof. Yusheng Ji, Profs. Hiroshi Ishikawa and Akifumi Makinouchi, for their interest in my work and the honor of having them as evaluators. I appreciated receiving their comments on my work and this manuscript.

The selecting committee of Egide made my graduate work possible as they partially supported it with a grant for research activities from the French Ministry of Foreign Affairs (bourse Lavoisier).

I would like to express my sincere appreciation to the people who played a part in this memorable experience. I salute Dr. Emmanuel Planas for giving me the opportunity to conduct research activities in Japan, and Dr. Kitsana Waiyamai for suggesting me to make my motivations clear. I am indebted to Ass. Prof. Asanee Kawtrakul for her kindness, to Shiho Iwasawa for her continuous help since I arrived in Japan, and also to Akiko Uchida and Nahoko Iwanaga for cheerfully dealing with the University bureaucracy. I enjoyed a lot listening to Prof. Henri Angelino; I thank him for sharing his experience of research and wise advices. It has been a pleasure to share informal discussions and some interesting ideas with Dr. Vincent Oria, Dr. Marco Mesiti, and Christophe Lucas.

I owe my deepest thanks to Tokiko. Her love, support, and patience over the last few years made me feel very lucky and happy, especially during the hard period of dissertation writing.

Contents

List of Tables

List of Figures

Part I

A Model for Collaborative Management of Resources

Chapter 1

Introduction

> "We build too many walls and not enough bridges."
> - Isaac Newton (1642-1727)

This chapter presents the background of the dissertation and outlines its structure. In Sect. 1.1, we introduce multimedia documents collaborative management. The deficiencies of today's multimedia documents collaborative management are pointed out in Sect. 1.2. Then, Sect. 1.3 highlights the approach to multimedia documents collaborative management explored in the dissertation, called adaptive collaborative delivery of annotated resources, and formulates our main objectives. An overview of the structure and contributions of the dissertation is given in Sect. 1.4 by introducing the whole architecture we are proposing.

1.1 Multimedia Documents Collaborative Management

Multimedia documents are widely used on interconnected devices; individuals manipulate them for personal, professional, or social reasons. Furthermore, the range of document types is getting larger and size of files is increasing. This situation implies to pay much attention to the management of this amount of heterogeneous data for users not to be swamped by a mass of irrelevant data. From a processing point of view, management for multimedia documents means: acquisition, content analysis, indexing, storage, retrieval, and distribution. All these processes are required to evolve with users' needs and behaviors.

With the advent of collaborative data management through networks appear new behaviors, which create new needs that current information systems do not meet; as a matter of fact, online communication, collaboration, and communities are the basis for exciting new application areas, where promising technologies are arising. For online communication and collaboration, researchers strove to make audio and video as flexible and easy to use as text in order to help people access and collaborate around multimedia, in real-time and on-demand. New communication paradigms and techniques are increasing awareness and interaction among

geographically distributed participants. These technologies are directly applied in the consumer space, developing enhanced media-browsing interfaces and services. For online communities, sociological principles and data mining techniques provide a framework for enhanced services, including formation of peer support networks, development of reputations, and incentive structures that encourage continued contribution for the collective good.

1.2 Problem Statement: Identification of 3 Major Issues

Despite the already long history of multimedia documents management, little progress has been made in personalization of shared data distribution. Users involved in communities typically arrange their multimedia collections in file systems which provide poor annotating mechanisms and hierarchical directory structures for organization and searching. Although this approach seems to be sufficient at first sight, there are many issues which make the collaborative management of large multimedia collections quite inefficient. Therefore, systems addressing adaptive services are facing complex tasks and remain hard to build. Three major reasons contribute to this complexity:

I. **Lack of generic collaborative knowledge management**: Collaborative multimedia documents management strongly address the need to access and use information about documents, and also about users (preferences, history, hardware and network environment...). Some multimedia document formats already include annotation, which is most of the time within the file structure; MP3 music files for instance can contain metadata in order to describe their content (author, title, album...). Moreover, document-inherent metadata (e.g. ID3 tags in MP3 files) remains unused and is only available to media-specific applications (e.g. an MP3 player). Many format-centered approaches do exist, but no framework enables communities to handle the full range of multimedia document types; categorization is limited to strict classification hierarchies. Similarly, ontology-based knowledge management are nowadays relevant if they are dedicated to one specific field only. Then, the capture of metadata is also an issue. Multimedia documents present properties that enable applications to automatically extract features and directly populate metadata. However, this kind of tool is related to one type of multimedia document only (e.g. color distribution for pictures, voice recognition for audio extracts), and is not sufficient to provide comprehensive annotation for the whole document. Therefore manual text insertion is still required. Finally appears the consistency problem when annotation updates are performed.

II. **Lack of contextual data management**: Being able to manage annotated multimedia documents through XML files makes it necessary to store and retrieve data efficiently. Unfortunately, the logical organization on devices is bound to the underlying physical storage system. Multimedia management applications usually depend on databases for indexing collections with user-defined keyword annotations and media-specific metadata import. Although it is a step in the right direction, these systems use proprietary databases without access to the stored organization structures or to metadata outside the

application. Moreover traditional databases are not flexible enough and typically too heavy for small devices. Collaborative multimedia documents management misses a data model that would fit XML and support DBMS-like services for any kind of devices being able to connect to others, to avoid document duplicates, to provide powerful information structure to express relations or dependencies between contextual information entities, and finally to ensure data consistency.

III. **Lack of personalization in services**: Users expect much from digital devices and embedded applications; they can easily stop using these tools if the results do not reach some standards. Thus, high quality of service is required for collaborative management of data. The main strategy that increases users satisfaction is to provide them automated and personalized access to very relevant data. Current systems enabling users to share and access data do not take advantage of environmental knowledge and lack relevance evaluation processes. Automated services can moreover reduce the network consumption by adapting the access to data. For instance, the personalization of query answering can be done either by selecting content, or by transforming content; the content may have to be transformed prior to export, in order to match user's device restrictions (bandwidth, size, memory...). P2P systems, which allow people to share multimedia files, are using some metadata to perform users' queries. Unfortunately, these annotation structures vary depending on the type of file and on the applications used to insert these metadata. Thus, this heterogeneity makes it impossible to offer users automated services that would be able to deliver accurately any kind of multimedia document.

We definitely cannot consider data management without tackling information management anymore, and must address the need for annotated, classified material with a rich context, which should support retrieval on a higher level than just common content based text search. There is obviously a huge lack of generic information modeling that would provide a unique and sufficient structure for the whole available knowledge about any type of multimedia documents (OMG or UML, for instance, are not easily portable to other domains), and environmental entities. It should moreover enable communities to retrieve and access more relevant information faster.

1.3 The Vision for Adaptive Delivery of Resources

Personalization of services is becoming a very important trend for many communities and software companies. Being convinced that this is indeed the breakthrough for the next generation of Information Systems (such as file systems [GGL03], digital library, decentralized distributed databases [Gra04]), we decided to improve the range and the quality of services offered to people who are manipulating heterogeneous multimedia documents by considering the whole combination of elements which services are depending on. After noticing that knowledge management is the key issue for building such kind of innovative system, we decided to study collaborative cases where environmental information is available (through annotations) and reliable. A community moreover presents the very attractive property to address and share one main topic of interest

(and possibly subtopics); this is the main requirement to fulfill for being able to define effective knowledge structures such as ontology as part of an information model. This information obviously has to be handled through metadata, which represent the most powerful solution to structure documents and related knowledge. Communities, once more, are the best environment to collect annotations; indeed, people involved in communities are motivated enough to spend some time for providing relevant and accurate annotations.

Therefore, our goal is to build a generic collaborative management framework for heterogeneous documents, that fully takes advantage of annotations (related to resources, users, communities, and devices). This solution must offer global management services adapted to categories of users having specific behaviors and expectations. The services to be proposed to users cover the usual database functions, the management of transactions to ensure the capability of the system to work in heterogeneous distributed environments, and user-personalized automated properties. This last point is a very important element of our vision. Personalized services are based on contexts (age, languages abilities, professional activities, hobbies of the user, users' communities, time...) that give systems clues about users' expectations and abilities. Then, users can get the exact view of the information they might be looking for and a better access (if they are allowed to).

Multiple points of view of multimedia documents are needed to serve many multi-disciplinary communities of interest and individual preferences [CG04]. This is also an imperative issue addressing the digital gap between the poorest communities of the developing world and the developed countries, as it has been pointed out by UNESCO[1]. Altogether, the variety of viewpoints that may be linked to a given object calls for several services and multiple ways to index such an object according to its representation at a specific time. Context-dependencies imply the manipulation of multidimensional data [KH98]; moreover the different types of services described above require considering both document-centric and data-centric perspectives. XML is surely the perfect tool to fulfill all these conditions. Furthermore, the concept of multi-dimensions tends to be used in the XML framework as a way to enforce the usage of multimedia data. Managing multimedia documents and avoiding redundancy between the different entities makes it necessary to define a data structure so as to make data retrieval more efficient, and to provide appropriate indexing methods. Obviously, the design of a convenient indexing method depends on the kind of data retrieval.

We propose a generic Information Modeling for Adaptive Management (denoted IMAM) which is the basis for the definition of operators and automated services in collaborative environments. This modeling aims at supporting and categorizing metadata dedicated to any multimedia document. The primary element of IMAM is a monolingual thesaurus-like tree structure which allows communities to accurately categorize multimedia documents and all the knowledge that is available about them; then operators can manipulate and compare the documents, produce indexes based on semantic, and ensure the consistency of the information. It is then very important to ensure that data is efficiently managed on logical and physical points of view. This requirement implies to consider the structure and language used for the representation of metadata (in our case XML). The next and last step is to design adaptive services based on IMAM's formal modeling. These services comply with complex distributive models structures such as Peer-to-Peer and mobile environments

[1] http://www.unesco.org/webworld/cmc

and perform personalized query optimization and authoritarian data placement. Thus, users can access in a faster manner more relevant documents among huge amounts of heterogeneous data and devices.

Our work clearly stands where databases, information retrieval, and distribution processing meet. Therefore, it makes it quite difficult to take into account all requirements related to each field and to merge them in a coherent way. However, it also makes it possible to provide innovative services, and to open future promising prospects.

1.3.1 Framework Components Description

As we pointed it out above, information modeling must be based on a coherent and powerful structure that can support all the layers of the architecture it is applied to (physical, logical, semantical, transactional). In our framework, it has to handle physical entities (servers, *access points* being devices used as sub-servers for communities, and fixed and mobile devices; see an illustration on Fig. 1.1), *resources* (i.e. any mono-type multimedia document that can be related to at least one topic), knowledge management entities (*resource descriptions*, community's, user's, and device's *profiles*), and semantic elements (types of documents, i.e. *categories*, attributes, i.e. *descriptors*, and characteristics, i.e. *descriptors' values*). This information structure relies on contextual information for personalizing the retrieval and the distribution of multimedia documents. Since we want to provide a generic solution, our model is deeply related to XML; as a matter of fact, the Extensible Markup Language, with its ever-increasing number of extensions (and numerous drawbacks...), has become the standard for data analysis and exchange, and perfectly fits the requirements for handling annotation.

The global idea we have about the multimedia data management is to be embedded into a pervasive computing world. The data storage systems and the transactions have to support distributed and heterogeneous constraints coming both from the users and the data. From our point of view, the best way to achieve efficiently this goal is to process database-like services on devices, with processes that behave quite autonomously. Many aspects have to be considered in order to provide such a complicated distributed document management; e.g. data model (supporting both storage and transaction processes), databases operators (indexing, querying...), transaction protocols (e.g. based on SOAP and/or JXTA to provide mixed strategy: Peer to Peer for transactions between devices, and C/S for transactions between devices and DBMS server)... We present an appropriate data model [GAO02] in Sect. 5.3.3) which is based on the Extended Binary Graph [AO98b] core data structure in association with XML.

As users are more and more located in a pervasive computing environment with many mobile devices (so called symbiotic environment) [AOT01], it is important to categorize the elements, to which processes have to be applied. Then we can identify the needs and constraints related to each category of elements. The global environment is made of:

- *Traditional* DBMS servers; which provide a safe and strong basis for archiving the *resources* and performing costly processes.

- Mobile devices; which provide the main constraints for defining adaptive services.
- Fixed devices; which can be used as sub-server (what we call *access-points*), with higher capacities than mobile devices.
- Network services; which are the distributive layer of the architecture and articulate the propagation of the *resources*.

Every element in this architecture has to be able to communicate with elements of the three other entities. Within this framework, we want individuals or automated processes to provide powerful and relevant access to data for other users or processes. People are using a wide range of devices including desktops, laptops, handled personal digital assistants (PDAs), and smartphones that are connected to the Internet using very different kinds of network, such as Wireless LAN (e.g. 802.11b), cellphone network (e.g. WAP), broadband network (e.g. cable or ADSL modem), telephone network (e.g. 28.8 kbps modem), or Local Area Network (Ethernet). Occasional to frequent disconnections and unreliable bandwidth characterize many of these networks. The availability of services is thus a significant concern to people using mobile devices and working in different kinds of wireless and wired networks.

1.3.2 Illustrative Scenario

We have been involved in two projects (Geomedia and Digital Silk Road) focused on the management of multilingual multimedia documents; both are handling all kinds of multimedia documents (including text-based, image, audio, video formats).

Digital Silk Roads project (DSR, [Ono01, Ono03]), which has been initiated by NII and UNESCO, is focused on the collaborative management of digital multilingual cultural documents. DSR aims at creating a global repository that enables us to collect, validate, preserve, classify and disseminate cultural *resources* related to the historical silk roads. The variety of *resources* is very large; it ranges from historical restoration notes, to architectural monitoring drawings, pollution control graphs, virtual museum excerpts, or educational support documents. Maps (often represented with multidimensional layers) are the cornerstone to link geographical information to other dimensions such as language, history, economy, religion, politics, humanity and they are an excellent support and framework to visualize and to understand the analytical information and interdependency information as part of multidimensional knowledge.

The main issue we are facing with DSR is heterogeneity; as we mentioned earlier, we have to consider very different types of data, users and devices. We have the chance, as part of DSR, to work with more than 400 specialists in various fields (using 21 languages) who are *motivated* and *able* to annotate documents very accurately. DSR users are divided into two categories: contributors and end users who are supposed to manipulate any kind of device (from mobile phones to mainframe computers). Since DSR is a collaborative project, it is important to register the users as members of communities (NB each community has an *access point*, which is a device being a kind of sub-server dedicated to the community); this enables us to increase

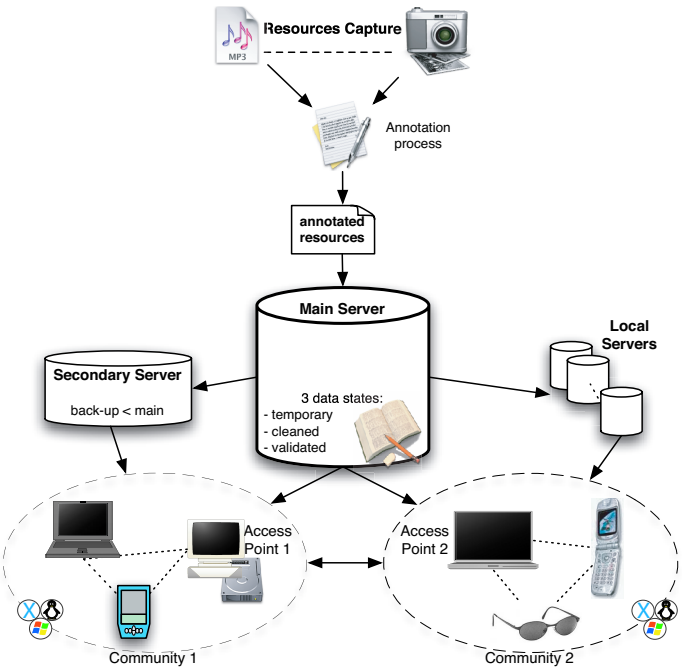

Figure 1.1: DSR framework

the environmental knowledge that is required for performing adaptive services based on users' status and abilities.

Building such a system is a great challenge, and requires to fulfill some commitments: first, it must provide an appropriate knowledge management framework that supports all the tasks it aims at covering. Then, documents and annotations have to be gathered and to be well structured (for the insertion process, we start with raw data including annotations from the author, then we apply some annotations and cleaning processes that are semi-automated, and we finally get certified data through specialists committee validation; this last step is obviously done by human beings). Thirdly, the data has to be stored safely through XML. Afterwards, it is necessary to ensure a simple and accurate access to the resources for each user. Finally, the distribution of the information has to be optimized in order to propose adaptive services. The global framework of DSR with its data distribution scheme is illustrated on Fig. 1.1. In Sect. 2.2, we present the model we use to identify, describe, and access any kind of *resources* with the aim to integrate them to DSR.

1.4 Outline and Contributions of the Dissertation

The global approach we are proposing in this dissertation confronts us to various research fields. Therefore, we have to review different issues separately in order to define and merge the best components for the most

appropriate framework. We chose to distribute the review of states of the art among the various chapters of the dissertation; this allows us to present precise and consistent overviews and to clearly motivate our choices.

The parts of the dissertation, although they focus on different topics (which are of course related), are dedicated to the enhancement of the delivery of multimedia documents in collaborative environments. In order to enable the reader to apprehend our global vision, we present on Fig. 1.2 the whole architecture we are proposing; each component in brown will be investigated and described in the corresponding parts, with the aim to contribute towards the building of a coherent framework. In the remaining of the dissertation, we motivate each element of the architecture by considering the existing solutions and by choosing the best compromises in order to make the full framework relevant and useful.

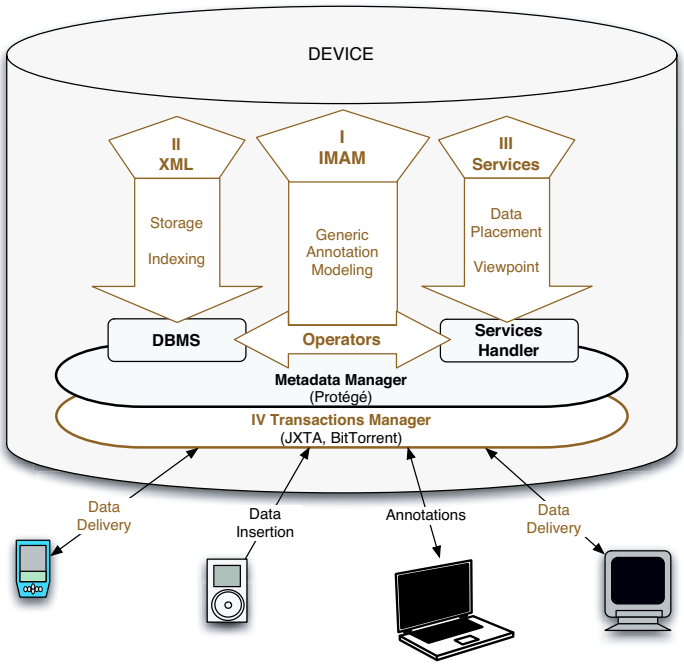

Figure 1.2: An architecture for adaptive delivery of data

The dissertation presents an initial study of adaptive services, which are based on a generic knowledge modeling and dedicated to the delivery of data within communities. It consists of four parts:

Part I. first considers the existing strategies which make it possible catching pieces of knowledge that can be useful for the delivery of multimedia documents. After pointing out their drawbacks and precisely motivating our needs with an example, it defines an Information Modeling for Adaptive Management, called IMAM[2], dedicated to communities sharing access to resources. IMAM is the primary component of the

[2]The term *imam* is used in many different contexts, and with different meanings. As we are focusing on communities, we will only refer to the common everyday use of the word which is for a person leading Muslim congregational prayers in the mosque. We called our modeling IMAM to reflect the fact that it makes it possible to give directions to groups of people in a quite authoritarian way, but

architecture described on Fig. 1.2. Indeed, this modeling aims at improving the knowledge representation in order to enhance the delivery of data between communities' members; the kernel of IMAM is a thesaurus-like extensible categorization tree that can handle any kind of multimedia document. IMAM entities have been implemented and are presented in a simple manner as XML files; they are separated into two categories (the resource description for multimedia documents, and the profile for users, communities, and devices) that allow us to capture any useful information and to categorize it through metadata.

Part II. investigates the needs and the generic issues concerning the achievement of flexible support of XML multidimensional data in a heterogeneous information management system. As it appears on Fig. 1.2, the data management is an important component of our framework and we must ensure that it does not spoil the services that are to be delivered to heterogeneous devices. We focus here on a data model and an appropriate indexing strategy in order to define a functional and effective data manipulation for our framework. The solutions proposed in this part relies on an interesting structure called EBG, which is the core of a DBMS prototype that has been under maintenance since we started working on this component. Therefore, we only provide the design of XML mapping and of the multidimensional indexing with examples using the multidimensional extension of XML called MXML.

Part III. tackles the essence of our work as it provides adaptive services which offer a wide possible range of improvements for the management and especially the delivery of information within communities. The main goal of these services that are performed by automated processes is to improve the access of communities' members to shared multimedia documents. This is done by exploiting the structured and categorized knowledge handled by our modeling; IMAM entities are compared and evaluated in order to identify relevant pieces of data for the users. We define two adaptive services based on IMAM: the viewpoint, which acts as a query optimizer, and an authoritarian data placement, which dispatches the multimedia documents on the community's devices according to the attractive potential that resources have for each user. These services represent an emerging and promising area of research and applications. We describe a partial implementation of the data placement that focuses on the relevance evaluation part of the service and present some preliminary results of experiments that are being performed. This dissertation presents an initiating approach that aims at answering some fundamental challenging issues in this area by focusing on users satisfaction.

Part IV. investigates the remaining issues that need to be addressed in order to fully take advantage of IMAM and its services, and particularly focuses on the transactional aspects that are required for IMAM services to be fully efficient. This component is the bridge between the community's devices and must be the vector of merged services (viewpoint & data placement). We identify the key issues that have to be solved and the attractive directions that should be followed in order to perform effective adaptive services within the architecture presented in Fig. 1.2. Moreover, we uncovered some exciting open problems, which are described in Chap. 12, where we also summarize the work and contributions presented in this dissertation.

---

nevertheless with respect of each individual.

Chapter 2

Modelling Contextual Information

> "Knowledge is elusive and volatile; it escapes measurement."
> - Umberto Eco (1988), "Foucault's Pendulum"

Elaborated data management and adaptive services must consider all available and relevant knowledge related to data and elements involved in the processes [Fit99]. Knowledge Management is in itself a wide area where many domains are merging; thus, as distribution and adaptability are increasingly involved in, it appears to be a key issue in more and more applications. But it still lacks global approaches that consider the knowledge management from the acquisition to the dissemination, in particular for the shared information within users communities.

As text obviously is still (and indeed for many more years) the only reliable basis to build generic and portable strategies for the management of heterogeneous data, we chose to use metadata (through annotations) dedicated to multimedia documents within XML as a knowledge capture requirement. This *information about the data* has to cover four layers: users, communities, devices, and *resources* (which are homogeneous pieces of data, i.e. mono-type).

Our strategy, using well-structured knowledge management, is to precisely manipulate resources via the metadata we have about them. First, it is important to keep safely all the information we get about the resources. Then, we have to ensure the quality and the validity of the information we are storing. The third step is to properly disseminate the resources depending on the information we have about users. This approach is based on combined manual and automated processes for all the following services: annotation, storage, distributed back-up, data placement, information sharing, and relevance feedback. The main contribution of this chapter is to provide a global *resource* description, a categorization, and a manipulation framework that fits XML and enables us to enhance the collaborative distribution of information, by capturing all the knowledge that might be useful for improving the relevance of distributed semi-automated processes.

In this chapter, we address the need for a generic information modeling that would allow communities

to handle and take advantage of all the knowledge they are dealing with. After reviewing the knowledge management issues we are facing and the existing approaches, we describe a modeling made of conceptual structures and operators that is the basis for the adaptive collaborative delivery of resources we want to provide. This chapter is organized as follows:

- In Sect. 2.1, we investigate relevant knowledge management issues and give an overview of the related work.
- In Sect. 2.2, we introduce conceptual structures used for representing resources and related annotations. We present the *Resource Categorization Tree* and related environmental knowledge structures, which allow us to identify and manipulate any kind of *resource* depending on the environment.
- In sect. 2.3, we define the structural semantic of the key resource-management operators based on the conceptual structures we introduce, and ensure the consistency of the operators.

Then, Chap. 3 presents an architecture which is relevant to IMAM needs and proposes an implementation of IMAM. Sect. 3.3 concludes Part I.

2.1 Structured Knowledge Management

2.1.1 Metadata Management

Metadata is descriptive information that can be applied to data, environmental entities, or applications; it enables applications to capture and handle information embedded within the file and into a content management system. Metadata content goes from structural specifications to deep semantic descriptions. Most of current usage of metadata is related to the management of database schemas, interface definitions, or web pages layouts; it is in fact easy to notice that structural metadata is quite simple to produce and to manage, whereas semantic manipulation of information is a huge challenge.

Relevant descriptions, searchable information, and up-to-date author and environmental information can be captured in a format that is eventually understood by users as well as by software applications, and hardware devices. We claim that information systems do not enough take advantage of metadata, and so are missing great opportunities to improve their knowledge management tasks. From what has been explained in the previous chapter, two solutions appear for collaborative platforms that aim at manipulating complex sets of metadata:

- To build an application that integrates and exploits any kind of metadata structure; it implies to map all the structures to each others and to update the mapping each time a new structure appears.
- To define a generic metadata structure that supports any kind of multimedia documents.

The first solution generates many inconsistency issues and becomes very heavy as the number of mapped structures grows. Example of metadata format dedicated to one type of file is given in Sect. 1.2 and shows

the limitations of most existing formats. The second solution first seems too restrictive and complex; but it is quite possible to make it more flexible by defining a core structure that can be extended for particular uses. The MPEG-7 standard also known as *Multimedia Content Description Interface*[1] aims at providing standardized core technologies allowing description of audiovisual data content in multimedia environments. MPEG-7, with its Description Schemes and Description Definition Language, has an interesting approach. Unfortunately, as its design goes deep into details and is quite complicated, common users have been reluctant to use it. Adobe is providing an open source, W3C-compliant way of tagging files with metadata across products from Adobe and other vendors, called Extensible Metadata Platform[2] (XMP). XMP is extensible, meaning that it can accommodate existing metadata schemas; therefore systems do not need to be rebuilt from scratch. However, many companies and communities reject it because of its origins. Finally, W3C recently provided a recommendation describing CC/PP[3] (Composite Capabilities/Preference Profiles) which defines the description of device capabilities and user preferences as a profile. This structure, using RDF, is quite attractive as it contains very precise and coherent vocabularies. Unfortunately, its complex and very detailed structure makes it unusable for most of the users.

2.1.2 Related Work

Context-dependence

The significance of context-aware computing is dramatically increasing and promises strong improvements in human-machine interaction. Indeed, autonomous interactivity performed by an application or agents [BL01] can successfully be based on active and passive context-aware features [BD03]. But in all cases, it is important to balance the degree of autonomy in order not to bother users. According to Hess and Campbell [HC03], context is one of the factors that differentiates ubiquitous computing from traditional distributed computing.

Many approaches for context management are available in the literature and various theories have been proposed to formalize context [BS01]. The range of fields using context-dependence is quite wide. It goes from very abstract analysis [BBG01] to Artificial Life applications. Our understanding of context-dependence is slightly different as we consider contexts as dimensions; this approach has been deeply investigated for the definition of Multidimensional XML [SGR00] (MXML) which is an extension of XML including the management of context-dependent information. Versioning, which is vital in a collaborative project, is also a context-dependent issue.

In order to avoid the drawbacks of the two casual versioning schemes (store last version + backward deltas, and store all versions), an adaptive document version management scheme [BMN03] enables the system to continuously evaluate if it is pertinent to keep each version of a document. However, this strategy does not allow us to take fully advantage of the context-dependence. A very complete versioning management of XML documents has been proposed [CTZ01] but it does not consider distribution issues. XML versioning

[1] http://www.itscj.ipsj.or.jp/mpeg7/
[2] http://www.adobe.com/products/xmp/main.html
[3] http://www.w3.org/TR/2004/REC-CCPP-struct-vocab-20040115/

using MXML has been defined [GS03] and so represents a nice opportunity to deal with versioning through XML. Another possible interpretation of distributed knowledge versioning is adaptive point of view, i.e. personalized data access. This issue is very interesting to us since it is similar to the kind of query optimization we want to provide. Giving a formal approach of a multidimensional logic, [WLO01] defines a set of contexts with properties that seems to be very convenient for MXML.

Knowledge Distribution

The number of applications using metadata to improve information retrieval processes and to deal with semantic heterogeneity is growing very fast; web services in particular already have several standards (e.g. DAML/OIL, ebXML) to describe service related information. Web-based Information systems have a typical structure which consists of three layers: semantic, application, and presentation. The Hera design methodology [VBH03] considers integration and user support as aspects to be included within these three layers, which is a very relevant strategy according to us. Nevertheless, Hera uses a RDF-based ontology model which is very convenient but lacks context management support. Metadata is also part of many semantic management frameworks for multimedia documents (e.g. audiovisual resources [TFC03]); but most of the time, these frameworks are dedicated to a precise type of data and/or to a specific domain.

Knowledge sharing is a wide area made up of many fields; moreover it covers different kinds of application, going from common memory space access to collaborative project management. It has been deeply investigated for many years and a lot of work has been produced (e.g. for software development teams [CMM03]). As a matter of fact, most of the ontology-based applications are influenced by initiatives for the definition of interoperable metadata standards (such as Dublin Core). We also would like to point out that XML, with its large set of tools and extensions is commonly recognized as the best framework to contain metadata. The distribution of data within communities can be partially automated in order to reduce the query workload [GHI01]; indeed, it is possible to evaluate what kind of data might be interesting or useful for a class of user (community), and for a specific user. It is very important to choose an appropriate heuristic [KK04] depending on the requirements related to users and environments in order to perform data placement. Then it becomes realistic to create automated data placement processes; an example of scheduled data placement [KL03] indicates that much benefit can be obtained without any human interaction.

2.1.3 Motivating Example

Ontology-based Metadata Management

DSR (see Sect. 1.3.2) has to deal with large repository of multimedia of historical and cultural resources which come along with the web. For instance, it provides access to databases containing cultural heritage photography. Meanwhile semantic understanding, access and usages of these materials are not fully possible due to the semantic gap for their annotation and retrieval [SGA02]. There are still shortcomings of appropriate

methods and tools for multimedia annotation, browsing and retrieval to help the users to find what they are really looking for. On the other hand, historical and cultural content of these databases make the process more complicated as there might be different semantic interpretations toward the subject of the visual information. Development and application of multi-lingual multimedia ontologies is the approach used for the conceptual categorization of the content on silk roads. Using domain knowledge enables DSR to improve multimedia semantic annotation and retrieval.

Ontology is defined as a specification of a conceptualization or as a set of concept-definition, a representational vocabulary[4]. Another definition of ontology which emphasizes the component-base recognition of a subject is a declarative model of the terms and relationships in a domain or, the theory of objects[5]. Based on these definitions ontology provides a hierarchical structured terminology of a domain and is completed by defining different relationships between term-sets.

Ontology is explicitly declared to be helpful for knowledge representation, knowledge sharing and integration, portability issues... Ontology has application in artificial intelligence, natural language processing, multimedia database... Examples of ontology can widely be found in Biomedicine[6]. Meanwhile recently in the field of multimedia enhanced annotation and retrieval, like photo annotation and ontology-based image retrieval and in some cases with relation to cultural heritage [Doe03] and art objects ontologies[7] are designed and applied. In the field of cultural heritage, through application of domain ontology, the domain experts can develop semantic annotation for the multimedia data like images, and users can have access to a model of the vocabularies of the subject which will guide them for a more standard and intelligent search through database and will lead to a better retrieval.

DSR is directly involved in servicing ontology management on a case study of architectural cultural heritage named caravanserais[8]. This ontology tries to provide a visual lexical model of terms or components in architectural relics sets and relationships between components based on the physical and spatial characteristics of the components. It also tries to design a multilingual ontology with the help of UNESCO expert team in order to exchange the content with experts and cover the needs of multilingual users [AAO03]. This ontology will be accessed and used by domain experts in order to reach a consensus for its content to be extended to other languages and typologies of architectural heritage. Developing ontology on this case study as part of the portal is considered as a proper example for involvement of domain experts over internet in knowledge management and application of it can help enhanced access to large visual data which DSR is dealing with.

[4]Gruber-Tom, "What is an ontology?": http://www-ksl.stanford.edu/kst/what-is-an-ontology.html
[5]Roberto Poli, "Framing Ontology - Second Part": http://www.formalontology.it/Framing_second.htm
[6]e.g. Open Biological Ontologies: http://obo.sourceforge.net/
[7]e.g. Digital Art Ontology Project: http://dao.cim3.net
[8]This multi-lingual ontology on architecture is constructed as part of a colaboration between National Institute of Informatics in Japan and the Architecture school of Paris Val de Seine in France under the *Digital Silk Roads Initiative Framework* in cooperation with UNESCO.

An Open Archives Initiative based Collaborative Portal

Several research projects such as the arXiv e-print archive[9], the Networked Computer Science Technical Reference Library (NCSTRL)[10] or the Kepler project [MZL01] in the field of digital libraries or digital research archives, tried to solve issues of sharing research information. They generally provide a common interface to the technical report collections based on the Open Archives Initiative (OAI) infrastructure[11]. This mechanism enables interoperability among large scale distributed digital archives. In many cases, network environment services include automated registration service, tracking of connected clients, and harvesting service of clients' metadata. Query service enables accesses to resources and to its related metadata. OAI has created a protocol (Open Archives Initiative Protocol for Metadata Harvesting, OAI-PMH) based on standard technologies: HTTP and XML as well as the Dublin Core metadata scheme[12]. OAI presently supports the multipurpose resource description standard Dublin Core which is simple to use and versatile.

Shortcomings of such research projects generally include a too general metadata attributes schema for fine-grained information (e.g. cultural domains) and the non-support of community building. However, OAI-PMH itself has been created to provide an XML-wrapper for metadata exchange. It has been extended in the Digital Silk Roads project to support multi-disciplinary metadata schemas such as Object ID[13] for historical buildings, Categories for the Description of Works of Art (CDWA) for historical artifacts, or VRA[14] for visual resources. In order to avoid various shortcomings and to provide a community framework for the research and education on Digital Silk Roads, the Advanced Scientific Portal for International COoperations (ASPICO) on Digital Silk Roads platform has been proposed [AGO04].

ASPICO

ASPICO is OAI-PMH 2.0 compliant as part of the distributed collaborative architecture as it is shown in Fig. 2.1. The platform provides services for data handling, registration for identification, and metadata handling based on cross-disciplinary metadata schemas to create OAI-compliant metadata and resource management. Researchers can annotate resources according to their point of views and can share their comments according to cross-disciplinary and multi cultural backgrounds. Furthermore, the cultural resource server includes an ontology management service to support multi-lingual ontologies of cross-disciplinary metadata standards and multi-lingual ontologies in Digital Silk Roads related fields (e.g. architecture, history, geography, art...). We currently use Protégé 2000[15] as the ontology server. The starting point of the ASPICO storage management has been the Dspace[16]; we extended Dspace core system to produce a multi-lingual platform and to support DSR metadata.

[9] arXiv.org e-Print archive: http://arxiv.org/
[10] Networked Computer Science Technical Reference Library (NCSTRL): http://www.ncstrl.org/
[11] Open Archives Initiative http://www.openarchives.org/
[12] Dublin Core: http://dublincore.org/
[13] Object ID http://www.object-id.com/
[14] Visual Resources Association: http://www.vraweb.org/
[15] http://protege.stanford.edu/
[16] DSpace Federation http://www.dspace.org/

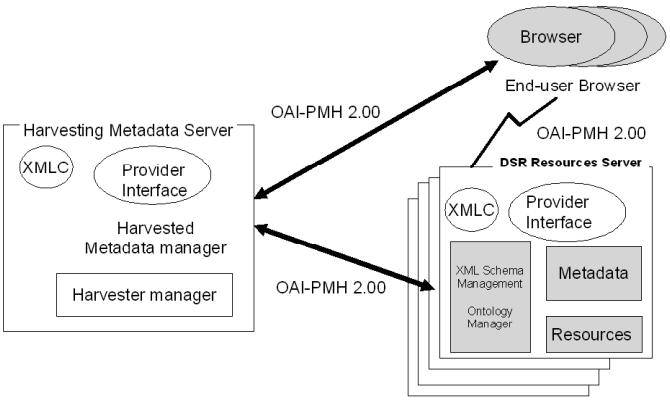

Figure 2.1: ASPICO architecture

ASPICO's scope is to offer collaborative projects such as DSR an innovative distributed information and data semantic management. It focuses on the critical lack of multi-domain semantic structure related to cultural *resources*, and supports collaboration and distribution over Internet within and between communities of interests. Distributed *resource* and semantic management systems address the need to access *resources* and related semantics wherever they are stored, to extract semantics while archiving digital *resources*, and to collaborate efficiently by sharing *resources*, annotation and forum exchange within a distributed network of communities. The ASPICO system consists of a set of autonomous virtual *resource* repositories. Each single ASPICO repository provides different points of view according to the needs of the end-users and according to the community they belong to. The *resource* semantic is collected and semi-automatically extracted from various sources (e.g. context, end-user, *resource* itself) then integrated and stored in each ASPICO node. Researchers and experts using the ASPICO system can share their knowledge related to each *resource* they collaborate on.

The ASPICO system is using a multi-lingual ontology-based metadata server (the target number of languages to be supported is at least 8 including English, French, Japanese, Arabic, Farsi...). The key innovation regarding languages management is the multi-lingual ontology integration solved by classifying and reorganizing ontologies in a logical and semantic sense according to metadata sets. We use Protégé 2000 as ontology management engine. Each *resource* is mapped on a culture-dependent thesaurus such as the AAT from Getty (for English); a sample of the *resource* semantic mapping from AAT and used as part of the metadata server is given in Fig. 2.2. In this context, the design of ASPICO has directly motivated the need of a modeling that supports the whole information management of the portal.

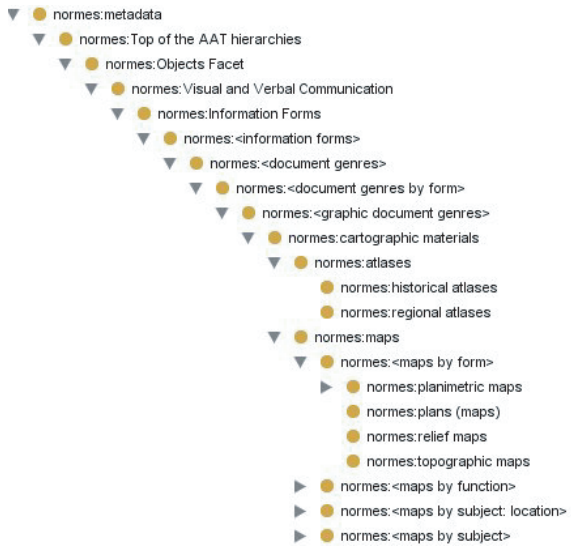

Figure 2.2: Extract of *resource* semantic mapping.

2.2 IMAM

The ability to manage *resources* for collaborative groups of users definitely relies on the available environmental information that can be gathered and processed. In fact, individuals sharing resources for business, education, or entertainment need some knowledge management support from the data repositories and devices that are used by the communities. Then it is possible for automated processes to perform tasks that improve flows of data within communities.

It is clear to us that the best strategy in this situation is to provide communities an appropriate and generic information modeling. This model has to handle any metadata related to the four layers described in Sect. 1.3.1. Moreover, we are convinced that this model must be made of a basic semantic categorization structure (such as the one that has been defined for ASPICO) on which are attached specific extensions dedicated to communities. Indeed, as we explained in Sect. 2.1.1, this extensibility is an imperative condition for the model to scale and to be reliable; in addition, it makes it easier for communities to define their own specific categories without dealing with consistency issues that would be generated by the design of the full model.

Therefore, our goal is to define a generic model for the management of distributed knowledge related to any kind of multimedia document. The first requirement that appears to us when aiming at defining a generic model is to fit standards as much as possible and to adopt efficient technologies (here comes XML, for many reasons that are described in Chap. 4). As this approach is strongly relying on XML-based annotations, it implies for users to spend a certain amount of time and to be quite precise about the information they are adding. We have the great opportunity with DSR to have access to high-quality annotations given by users

involved in communities. These annotations become very valuable once they are related to the knowledge structure presented above. Then, we need a unified model that enables us to capture this useful information about the documents and also the available knowledge about communities, users and devices; it is imperative to be very rigorous and to describe very precisely the whole information structure.

An information model must be based on a coherent formal model that can support all the layers of the management. The knowledge management entities (*resource's description*, community's, user's, and device's *profiles*) and semantic entities (types of documents, i.e. *categories*, attributes, i.e. *descriptors*, and characteristics, i.e. *descriptors' values*) are the required core elements of a collaborative information modeling [GAO03] and make up our Information Modeling for Adaptive Management (IMAM); they are defined and described in the two following sections.

Preliminary parts of IMAM have been previously introduced to XML management experts [GAG04a, GAG04b] and to the digital library community [GAA04]; we present here the full modeling including update and consistency policies, enhanced by improved knowledge management.

2.2.1 Information Modeling for Resources

Cultural information is very difficult to handle. Since it includes aspects such as politics, art, or history, it is impossible to consider it as a fully representative information, even when the sources are the most reliable ones. This is the kind of contextual issue we want to address. The considerable amount and diversity of information we are dealing with entails building a strong and powerful knowledge tree with contextual features, which fits XML.

The first postulate we declare gives a necessary condition for IMAM to exist: it requires accurate and relevant annotations. These annotations become very valuable once they are related to the framework presented in the previous section. We then need a unified model that enables us to capture this useful information about *resources*.

The *resource* is the basic element of our model; it can be any kind of unmixed multimedia document, i.e. a monotype document (pure text, picture, video...) that can be related to at least one topic. Then we add knowledge through metadata to the *resources* and obtain the atomic element of our knowledge management: *resource & annotation*.

In order to categorize and describe *resources* , we use a corpus-like knowledge tree structure as a *resource* classification which is called *Resource Categorization Tree* (RCT). It is directly defined after the semantic mapping described on Fig. 2.2 in the previous section (an illustration of RCT is given on Fig. 2.3). The knowledge management through the RCT is based on a contextual structure. Our model aims at describing as clearly as possible the information contained in the annotations provided on the *resources*. A *resource* r is represented by a final node of the RCT. The primary element in this approach is the node (or category) which contains a label of *descriptors* , which are contextual attributes (dimension) related to the node they belong to.

We define here the structure and elements of our modeling. Let us first give a few notations which will

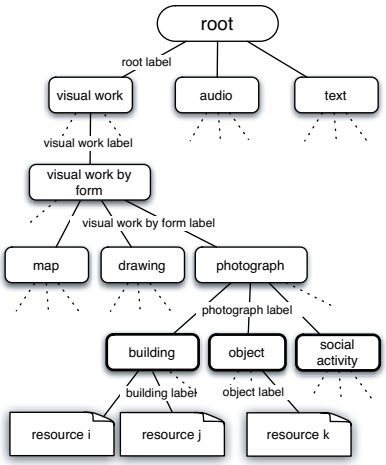

Figure 2.3: Core *Resource Categorization Tree* extract.

be used throughout the dissertation:

- α is a node of the tree; Ω is the set of nodes of the RCT.
- A *resource* is denoted r and is a leaf of the RCT; $\mathcal{R}$ is the set of *resources*.
- If we consider a branch of the RCT, the nodes α_{i-1} and α_{i+1} are respectively called immediate predecessor and any of the immediate successors of the node α_i. The root node of the RCT is denoted α_0.
- For any of the nodes α_i of the tree (except the leaf nodes), $\tau(\alpha_i)$ is the label of the node α_i (i.e. the label on the branch between α_i and α_{i+1}).

We first provide a basic structure entitled *core RCT* which structure is fixed and shall not be modified; *core RCT* is usable by any community:

Definition 1 (RCT) *The Resource Categorization Tree is an unbalanced tree denoted RCT; each node of the tree (denoted α_i) has a label being a list of descriptors (denoted δ). RCT is a tuple of nodes and descriptors lists: $RCT = \left(<\alpha_i>, <\delta_{i,j}> \right)_{\substack{i=0,\ldots,n-1 \\ j=1,\ldots,p_i}}$; where n is the number of nodes and p_i the number of descriptors in the i^{th} label. The core RCT is denoted $RCT_{\mathcal{C}}$ and has $n_{\mathcal{C}}$ nodes.*

This core structure shared by all IMAM users makes it possible for a community to partially access other communities *resources*[17]. Figure 2.3 shows an extract of $RCT_{\mathcal{C}}$; on this tree structure, bold nodes are

[17]RCT is monolingual; note that it is possible to link RCTs using different languages to each other in order to exploit some relationships and to relate *resource descriptions*; but in any case will there be a equivalence relationship between two RCTs (because of cultural and semantic differences between different languages).

Table 2.1: *Descriptors* examples

root	
1	rdID
2	rID
3	title
4	authorID
5	subject
6	relations
7	date
8	rights
9	description
10	type
11	locations

building	
1	name
2	address
3	date_of_construction
4	architect

visual work by form	
1	format
2	resolution

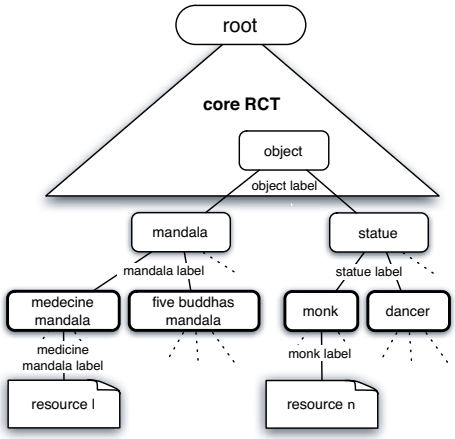

Figure 2.4: DSR *Resource Categorization Tree* extension examples.

terminal categories (thus most detailed ones of the categorization) where *resources* get connected. Then, each collaborative group can extend *core RCT* in order to define and categorize more precisely any kind of *resources* the community is especially interested in; this is done by adding new nodes and labels on *core RCT* terminal leafs. An example on Fig. 2.4 shows possible extensions of DSR RCT for communities interested in spirituality and sculpture. The extension structure of the RCT is presented in the following statement:

Proposition 1 *RCT_{c_i} denotes the Resource Categorization Tree dedicated to the i^{th} community; it is defined as follows: $RCT_{c_i} = RCT_{\mathcal{C}} \cup E_{c_i}$ where $E_{c_i} = (< \alpha_j >, < \delta_{j,k} >)_{\substack{j=n_{\mathcal{C}},\ldots,mc_i-1 \\ k=1,\ldots,p_j}}$ is the extension provided by the community c_i and $(m_{c_i} - n_{\mathcal{C}})$ is the number of nodes and labels added to $RCT_{\mathcal{C}}$ by the community c_i. Thus, we can write: $RCT_{c_i} = (< \alpha_j >, < \delta_{j,k} >)_{\substack{j=0,\ldots,mc_i \\ k=1,\ldots,p_j}}$*

It has to be clear that the RCT and its extensions must be defined and validated before a community starts to use it. We do not provide any RCT updating process once a community started using it as it would endanger the semantic coherence of the model.

This structure allows us to organize and describe all the information related to one main topic that is available about any multimedia document. Each path (from the root to a leaf node) of the RCT's skull (i.e. is nodes and parent-child relationships) provides a hierarchical semantic categorization.

The secondary (contextual) element in our approach is the *descriptor*, which brings structured and precise information about the resources:

Definition 2 (Descriptor) *A descriptor is a contextual attribute (dimension), which gives information about resources. It is denoted δ and is related to a specific node of the RCT. The set of descriptors is denoted Δ. The ordered set of descriptors of α_i is a label: $\tau(\alpha_i) = (\delta_{i,1}, \ldots, \delta_{i,p})$ where $\delta_{i,j}$ is the j^{th} descriptor of the i^{th} node and p the number of descriptors contained in label $\tau(\alpha_i)$.*

An important property of the RCT is that for any of its immediate successors α_{i+1}, the node α_i has the same label $\tau(\alpha_i)$ and so has the same set of descriptors: $\forall i \in [0, m-1], \tau(\alpha_i) = <\delta_{i,j}>_{j=1,\ldots,p}$, where m is the number of nodes contained in the full branch (path) from the root to the leaf representing the *resource* r.

The description of the resources through the *descriptors* is integrated in the RCT in order to perform effective operations on the information stored in a community repository. This is done with the *resource categorization*:

Definition 3 (Resource Categorization) *A Resource Categorization is a branch of the RCT, which is the path extending from the root to the considered resource (i.e. leaf node). A Resource Categorization R_r of a resource r is a tuple (N_r, T_r) where N_r is the non-void ordered set of nodes of r and T_r is the ordered family of labels on N_r. The set of ordered families of labels is denoted $\mathcal{T}$.*

The deeper a label is (from the root), the more precise the information about the resource is. Since $T_r = (\tau(\alpha_0), \ldots, \tau(\alpha_{m-1}))$, where $(m-1)$ is the number of arcs the *Resource Categorization* R_r contains, we write R_r as $(N_r, \tau(\alpha_0), \ldots, \tau(\alpha_{m-1}))$ or $(N_r, <\tau(\alpha_i)>_{i=0,\ldots,m-1})$. A *descriptor* can appear in several labels of the RCT, except the *descriptors* contained in the root label $\tau(\alpha_0)$; indeed, a property of *descriptors* is that they can be used only once in a *Resource Categorization*.

The value assigned to the jth *descriptor* of the ith node for a specific *resource* is denoted $\sigma_{i,j}$. Then all the knowledge about a *resource* is contained in the *resource description*; it is basically designed to structure the annotations, but it also aims at supporting the versioning of annotations. The whole annotation about a *resource* is contained in a full branch of the RCT called *Resource Description* and is defined as follows:

Definition 4 (Resource Description) *A Resource Description is the complete instance of a Resource Categorization for a resource r. A Resource Description D_r of a resource r is a tuple (R_r, S_r) where R_r is a Resource Categorization and S_r is the ordered set descriptors values $<\sigma_{i,j}>$ of the resource r. It is obvious that R_r has to be equal to $(N_r, <\tau(\alpha_i)>_{i=0,\ldots,m-1})$ so we have:*

$$D_r = \left(<\alpha_i>, <\delta_{i,j}>, <\sigma_{i,j}> \right)_{\substack{i=0,\ldots,m-1 \\ j=1,\ldots,p}}$$

The set of resource descriptions is denoted Λ.

***Example* 1** *As an illustration of the model described above, we show in Fig. 2.3 an extract of core RCT, and in Fig. 2.4 examples of possible extensions used for the DSR repository. Table 2.1 gives examples of labels and descriptors related to this RCT; e.g. the locations descriptor in the object label contains identifiers of devices where the resource description and the related resource are stored. It is important to point out the difference between the descriptors drID and rID; indeed, several resource descriptions can be related to the same resource (this will be motivated in Sect. 2.3.3).*
Note that the DSR resource descriptor list (defined by UNESCO & NII) has been influenced by the production based attributes of Dublin Core and by Getty's Art & Architecture Thesaurus.

2.2.2 Environmental Knowledge

The most common way to provide information that can enable systems to personalize data access is to manage user profiles. A profile is traditionally built through active involvement of the user, typically through fill-in forms. Users can often control the type of content provided, as well as the look and feel of the interface, by indicating their choices through their profile. The picture of the user built through the profile may consist of generic information (such as age or area code). It may also include explicitly stated choice of specific content, such as a general area of interest (e.g. music, mathematics, or gastronomy). Users could also specify general preferences for low-graphics versions of resources. This style of customisation requires the users to exert most effort and make the initial investment (such as for W3C CC/PP); it depends on the motivation and the ability of the user to set up complex customization features. If users are reluctant to spend time setting up complex personalization features [MPR00] the service may remain underutilised.

Then the profile may remain static and does not change with the user's changing needs (unless the user puts in the effort to update it). Thus dynamic profiles are needed, as automated behavior analysis enables systems to update user profiles. The main source of information for these updates is the user activity history [SHY04] but it also relies on contextual annotations that can be gathered automatically (such as localization, hardware & software characteristics...).

However, handling the available knowledge about users is not sufficient anymore. Indeed, advanced services for collaborative distribution of information must not only rely on knowledge related to the data (in our case through the RCT) and users; in addition, they need to consider all elements that are involved in the delivery process and might influence the access to the information. We clearly need a representation of all the useful information about the contextual entities (i.e. users, communities, and devices); this is the motivation of the profile:

Definition 5 (Profile) *A profile is a set of descriptors and values that are related to one environmental entity; it is a tuple denoted:*

$$\pi = (<\delta_i>, <\sigma_i>)_{i=1,\ldots,k}$$

where k is the number of descriptors and corresponding values. The set of profiles is denoted Π.

The values can be constants (birthdate, CPU...) or variables (localization, job...); it is important to specify types in order to manage efficiently history records (a profile is time-stamped).

We initially defined in [GAG04a, GAA04] the set of information related to one *descriptor* (instance) as a list of values; the possible number of values for each *descriptor* was bounded and the list of values for one *descriptor* was ordered (from the most relevant to the less one. e.g. in the case of languages, the first one must be the user's mother-tongue and then decreasingly regarding his skills). We finally chose to have a unique value for each *descriptor* in the profile. This approach is more relevant to us as we are applying term matching analysis on the values, and as lists of values would generate higher processing costs. However, we keep the hierarchical ordering of the terms included in the value field (we use the character ";" as a separator); in addition we prohibit and prevent the insertion of sentences in profiles' *descriptor values*.

Note that a user can be involved in several communities and a device can be shared by different users being involved in different communities. We give some examples of *descriptors* with possible values for the user *profile*: (ID - *124*), (rights - *read*;*write*)...

***Example* 2** *Regarding the DSR project, there are two main different cases we have to consider about the way we want to deal with the data management:*

- *the researchers involved in the DSR project.*
- *the common users.*

This aspect is handled through the communities' profiles.

Table 2.2: Extract of IMAM profiles descriptors

	profiles		
descriptor	user	device	community
1	ID	ID	ID
2	name	type	name
3	birth_date	allocated_memory_space	main_topic_of_interest
4	main_location	available_space	other_topics_of_interest
5	usual_locations	CPU_frequency	users_involved
6	current_location	RAM	devices_involved
7	languages	screen_resolution	
8	fields_of_expertise	common_bandwidth	
9	fields_of_interest	current_bandwidth	
10	environments	access_point	
11	communities_involvement		
12	devices_used		

Then, appear characteristics that are related to the fields of interest and abilities of the user. Table 2.2 gives a short overview of the IMAM profile for users, devices, and communities; it can in fact have many more descriptors. For instance, the computer environment consists in: fixed/mobile device, CPU, RAM & video-card resources, screen resolution... Moreover, the descriptors can be chosen according to the kind of community IMAM is used for: in the case of B-to-C or B2B applications for instance, some descriptors

can be dedicated to lists of customers or providers, to specific key words or topics related to collaborative projects... We provide implementations of profiles that are described in Table 2.2 for user, community, and device in Sect. A.

2.3 Operators

For convenient reference, the signatures and informal descriptions of the operators that are used throughout this chapter are summarized in Table 2.3. In this section, we define the precise semantics of these operators applied to the structures defined in Sect. 2.2.

Table 2.3: Summary of IMAM key operators

Operators	
signature	description
CREATEDR(D_r)	creates an empty *resource description*.
GETVAL(X, δ)	[$X = D_r \vee \pi$] returns the value assigned to δ in X.
PUTVAL(X, δ, σ)	[$X = D_r \vee \pi$] assigns σ to δ in X.
r.CREATE(N_r, T_r)	initiates the use of resource r with N_r and T_r contents within the community. Once the operator has been validated, services are applied to r and the members of the community can access it.
r_1.DIFF(r_2)	returns the set of descriptors that are not in r_1 and r_2.
r.EDIT	returns the whole set (or part of it) of *descriptor values* of D_r.
r.INSERT(P, V)	populates descriptors in P with values from V.
r_1.INTER(r_2)	returns the of descriptors that are in r_1 and r_2.
r_1.SIM(r_2)	returns a tuple (ρ_N, ρ_T) that evaluates the similarities between r_1 and r_2.
UPDATEDR($D_r, \Delta_{i,j}, \pi_u$)	returns two resource descriptions; D_r might be updated whereas D'_r is non void if the operator generates a new resource description.

2.3.1 Resource Descriptions Manipulation Operators

The previous definitions allow us to define the operators (unary and binary) that we need in order to manipulate instances of *resource categorization* and *resource description* such as casual database-like operators (e.g. create, edit, insert).

Each operator has been carefully designed in order to ensure the completeness of the model. For instance, *creation* operator makes sure that interactions between *profiles* (user involved in community(ies), device used by user(s), device being the access point of a community...) do not create inconsistency from other operators.

Operator 1 (Create resource) r.CREATE(N_r, T_r)*: this first operator creates an instance of Resource Categorization with a unique identifier for the resource r and its sets of nodes and related labels. Tthis instance is empty, i.e. does not contain descriptors values:* $(<\alpha_i>, <\tau(\alpha_i)>)_{i=0,\ldots,m-1}$.
NB This instance is not a complete Resource Description since it has no descriptors value; we use another operator to insert the descriptors values (see below).

Data insertion might imply some validation steps in the case of cultural content (such as the one used for the Digital Silk Roads project, see Sect. 1.3.2). However, we describe here a basic approach which is sufficient to ensure the completeness of the operator:

Operator 2 (Insert value) $r.\text{INSERT}(P, V)$*; $P \subset \wp(\mathcal{T})$ and $V \subset \wp(\Sigma)$, with $\mathcal{T}$ being the set of ordered families of labels and Σ being the set of descriptors' values. This operator associates each value $\sigma_{i,j}$ contained in V to its corresponding descriptor $\delta_{i,j}$ from P and inserts it in the Resource Description D_r. This operator can be used to delete values by inserting a void value to descriptors.*

Operator 3 (Edit resource) $r.\text{EDIT}$*: By default, the* EDIT *operator takes no other argument than r and returns all the descriptors values contained in D_r. We define this operator as the result of the following function:*

$$\begin{aligned} \epsilon : \mathcal{R} &\longrightarrow \mathcal{T} \\ r &\longmapsto <\tau(\alpha_i)>_{i=0,\ldots,m-1} \end{aligned}$$

Thus:

$$\epsilon(r) = \left\{ <\sigma_{i,j}>_{\substack{i=0,\ldots,m-1 \\ j=1,\ldots,p}} \mid \delta_{i,j} \in \tau(\alpha_i), \alpha_i \in N \right\}$$

It is possible to use this operator with an argument being a list of descriptors: $\epsilon^\star(r, P)$. Then it returns the corresponding values if they exist:

$$\epsilon^\star(r, P) = \left\{ <\sigma_{i,j}>_{\substack{i=0,\ldots,m-1 \\ j=1,\ldots,p}} \mid \delta_{i,j} \in P \right\}$$

2.3.2 Resources Comparison

The two following operators (used for the comparison of two *resources* r_1 and r_2) have in common to be made of two levels; they are first applied to sets of nodes N_{r_1} and N_{r_2}, and then, depending on this first result, to the sets of labels T_{r_1} and T_{r_2} (we use the following notation: $\tau(\alpha_i)_{r_j}$ is the label of i^{th} node of the *resource categorization* R_{r_j}):

Operator 4 (resources Difference) $r_1.\text{DIFF}(r_2)$*: the* DIFF *operator returns a tuple (N_{diff}, T_{diff}), which is the result of a two-steps analysis. Indeed, in order to optimize the operative costs, we first check the nodes lists and notify the nodes contained in one of the Resource Categorizations R_1 and R_2 only. Then, we apply the same kind of operation to the descriptors (notation: $N^* \equiv N \setminus \alpha_0$):*

- DIFF *on nodes:*
  - *if $\tau(\alpha_1)_{r_1} = \tau(\alpha_1)_{r_2}$, then the operator returns N_{diff} being the list of nodes appearing only once in $\{N_1, N_2\}$: $N_{diff} = N_1 \cup N_2 \setminus N_1 \cap N_2$*

– *if* $\tau(\alpha_1)_{r_1} \neq \tau(\alpha_1)_{r_2}$*, then the operator returns* N_{diff} *being the list of all nodes in* N_1 *and* N_2*:* $N_{diff} = N_1^* \cup N_2^*$*; it is possible in this case to generalize the* DIFF *operator for n resources* $(r_1.diff(r_2, \ldots, r_n))$*; thus:*

$$\textit{if} \;\; \tau(\alpha_1)_{r_1} \neq \tau(\alpha_1)_{r_i} \; \forall i \in [2, n], \; \textit{then} \;\; N_{diff_{1,(2,\ldots,n)}} = \bigcup_{i=1}^{n} N_i^* \subset \Omega$$

- DIFF *on descriptors:*
  - *if* $N_{diff} = \emptyset$*, then obviously, the operator returns:* $T_{diff} = \emptyset$
    This case implies $(N_1, T_1) = (N_2, T_2)$*, and means that* $R_{r_1} = R_{r_2}$
  - *if* $N_{diff} \neq \emptyset$*, we have to take into account a property of RCT; indeed, since a descriptor can appear in several RCT's labels, we do not only consider the non-similar labels to look for redundancies. This is the reason why we check the common descriptors between* D_1 *and* D_2*:*

$$T_{diff} = \left\{ <\tau(\alpha_i) \setminus \{\delta_{i,j}\}>, \forall \alpha_i \in N_{diff} \;\; | \, \exists \delta_{p,q} = \delta_{i,j}, \; \delta_{p,q} \in D_1 \cap D_2 \right\}$$

Operator 5 (resources Intersection) $r_1.\text{INTER}(r_2)$*: As mentioned earlier, the* INTER *operator has the same structure as* DIFF*. This time, the two-steps analysis is not required to identify different cases, but it is interesting to perform a test on the second label in order to save some processing time. The* INTER *operator returns a tuple* (N_{inter}, T_{inter})*:*

- INTER *on nodes:*
  - *if* $\tau(\alpha_1)_{r_1} \neq \tau(\alpha_1)_{r_2}$*, then obviously the operator returns:* $N_{inter} = \alpha_0$
  - *if* $\tau(\alpha_1)_{r_1} = \tau(\alpha_1)_{r_2}$*, then the operator returns:* $N_{inter} = N_1 \cap N_2$
- INTER *on descriptors:* $T_{inter} = \left\{ <\tau(\alpha_i), <\delta_{j,k}>> \;\; \middle| \; \alpha_i \in N_{inter}, \right.$
  $\left. \exists \alpha_j \in N_1 \cup N_2 \setminus N_{inter} \;\; | \, \delta_{j,k} \in D_1 \cap D_2 \right\}$

The similarity between any two documents may be evaluated as the cosine product of the associated vectors. Alternatively, a user's existing term *profile* can be mapped into the vector space [WZW85] and then the similarity evaluated. In this way, the most relevant documents for a user can be determined and retrieved. The main limitation of this approach is related to synonymy and polysemy. Fortunately, our *profile* enhancement and the community involvement reduces these problems. We are currently considering term weighting schemes [SB88] in order to improve the relevance of *resources* comparison.

From our both previous operators, we evaluate the similarity between two *resources* (by matching terms within categories):

Operator 6 (resources Similitude) $r_1.\text{SIM}(r_2)$*: This operator is based on the operators* DIFF *and* INTER *(notation:* $Card(T_a)$ *is the number of descriptors contained in the ordered family of labels* T_a*); it returns:*

$$\rho = (\rho_N, \rho_T) \in [-1, 1]^2$$

with:

$$\rho_N = \frac{Card(N_{inter}) - Card(N_{diff})}{Card(N_1 \cup N_2)} \qquad \rho_T = \frac{Card(T_{inter}) - Card(T_{diff})}{Card(T_1 \cup T_2)}$$

where:

- ρ_N *gives a global idea about the similarity between the types of* r_1 *and* r_2*.*
- ρ_T *gives a more precise evaluation about* r_1 *and* r_2 *similarity. It also allows us to find similarities between documents having different types.*

It is clear that the SIM operator provides an interesting support for advanced indexing of *resources* as it allows us to record relationships between *resources* each time a new entry is performed in the repository. We plan to add some variables in the SIM operator in order to record the descriptors occurrences and then to return a weight related to the number of occurrences.

***Example* 3** *We propose here, as an illustration of the* SIM *operator, three typical cases, which indicate what kind of values to expect according to the size of the common path on the RCT for two different resources; let us consider* SIM *applied to:*

1. *two resource descriptions having the exact same path on the RCT of a specific community. It will return the tuple (1,1), which indicates that the two resources have the exact same type with a granularity that is as reduced as the RCT extension of the considered community goes deeper into details.*
2. *two resources having about half of their paths in common (the first one of course). Then,* ρ_N *will be negative, whereas* ρ_T *value remains indeterminate and will especially depend on redundancies among the descriptors in the extension.*
3. *two resources only having the first node in common will generate a tuple that tends towards (-1,-1).*

In the following example, we demonstrate that the SIM operator is not sufficient for evaluating the similarities between resources; indeed, we stress that SIM only evaluates the resources' types similitude:

***Example* 4** *We have to point out that two resources having very similar contents might be considered as very different by* SIM *operator. Indeed, the result of* SIM *applied to a picture of a painting and to an audio excerpt of a specialist interview describing the same painting will tend towards (-1,-1).*

These operators provide a strong set of tools which allows us to manipulate resources. It is obvious that an appropriate GUI has to be created for non computer scientists in order to make users' access to the data easier. As an example, for the operator INSERT, the GUI would display all the *descriptors* $\delta_i \in T_r$ contained in the related *resource categorization* R_r and propose a field to fill in for all the empty values σ_j.

2.3.3 Resource Description Update

Annotation is usually captured during the file creation process; it is also possible for users to perform updates on the annotation they initially provided or that has been produced automatically by applications. The other possible case for annotation insertion is related to users who disagree with annotation content and want to modify it.

For consistency reasons, it is vital to ensure that updates are performed safely. Starting from the postulate that a *resource*, once adhering to IMAM, i.e. well annotated, validated, and dispatched (see Sect. 9), cannot be modified, we assume that any modification or add-on on the annotation will generate the creation of a new *resource description* that will have to follow the same steps as a new *resource* would have had. We present here the full update structure and process. The information related to changes that are carried out at once to a *resource description* by a user are contained in a structure called $Delta\text{–}D_r$:

Definition 6 ($Delta\text{–}D_r$**)** *$Delta\text{–}D_r$, denoted ΔD_r, is a tuple $(D_r, \Delta_{i,j}, \pi_u)$, where D_r is the resource description to be updated, the tuple $\Delta_{i,j} = (<\alpha_i>, <\delta_{i,j}>, <\sigma_{i,j}>)_{\substack{i=0,\ldots,d \\ j=1,\ldots,d_i}}$, called changes container, gives the lists of d nodes and d_i descriptors of the i^{th} node where changes apply, followed by changes themselves as a list of descriptors values, and finally π_u is the profile of the user who performs the update.*

We use the full user *profile* because we do not need the user identifier alone, but also the user's rights in order to ensure that he is qualified to perform updates on *resource descriptions*.

Proposition 2 *$\Delta_{i,j} \in \Lambda$. Thus IMAM operators can be applied to changes container (this property is interesting for the update process described below).*

Proof. Since the *changes container* describes a path (even incomplete) of the RCT, it is a *resource description* subset.

As they necessary have to deal with update management, standard DBMS typically consider two types of *descriptor values* as part of the changes container:

- On the one hand, if a *descriptor value* only adds new content without modifying the existing value (case of an empty or incomplete field).
- On the other hand, if a *descriptor value* brings some modification to the existing value (case of fully or partially overwriting a non-empty field).

Then, the common way to manage the update, according to the case which occurs, would be to adopt a complicated and costly versioning strategy. In order to avoid heavy consistency policies related to versioning and to make the update propagation easier, we provide a very simple and powerful update manager based on $Delta\text{–}D_r$ (see following operator); note that the list of *descriptor values* $<\sigma_{i,j}> \subset \Delta_{i,j}$ contains full values to be inserted (i.e. the complete new value to be assigned to corresponding *descriptors*). This is done through the input interface where the previous value is edited and where the user directly apply the changes.

UPDATEDR$(D_r, \Delta_{i,j}, \pi_u)$
1 **if** $write \in$ GETVAL$(\pi_u, rights)$
2 **then if** GETVAL$(D_r, authorID) =$ GETVAL(π_u, ID)
3 **then** PUTVALUE$(D_r, \delta_{i,j}, \sigma_{i,j})$
4 $D'_r \leftarrow \emptyset$
5 **else** CREATEDR(D'_r)
6 $D'_r \leftarrow \{D_r \setminus \sigma_{a,b} \mid \delta_{a,b} = ID\}$
7 PUTVAL$(D'_r, \delta_{i,j}, \sigma_{i,j})$
8 **else** $D'_r \leftarrow \emptyset$
9 **return** (D_r, D'_r)

Figure 2.5: The *resource description* update pseudo-algorithm

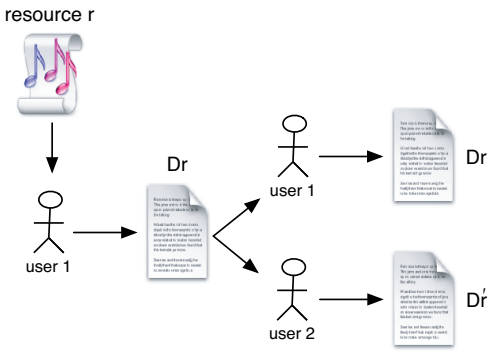

Figure 2.6: Resource description update cases.

Operator 7 (Resource Description Update) *This update operator, denoted* UPDATEDR*, applies the function* μ_{D_r} *and is processed locally on users' devices:*

$$\begin{aligned} \mu_{D_r} : \Lambda^2 \times \Pi &\longrightarrow \Lambda^2 \\ \Delta D_r &\longmapsto (D_r, D'_r) \end{aligned}$$

The pseudo-algorithm of function UPDATEDR *on Fig. 2.5 fully describes operator* μ_{D_r} *and Fig. 2.6 illustrates the considered processes.*

Let us describe the pseudo-algorithm given on Fig. 2.5: after checking if the user has the rights to perform an update on a *resource description* (line 1), two possibilities have to be considered: in the first case (lines 3 and 4), if the user has been the author of the existing D_r, the update process directly acts on D_r. Whereas on the second case, if the user is not the author of D_r, the update process generates a new *resource description* D'_r (line 5), copies the content of D_r in it (except the ID of course) on line 6, and finally proceeds the changes on this new *resource description* (line 7). We describe here the functions appearing in the pseudo-algorithm on Fig. 2.5:

- CREATEDR(D_r) creates an empty instance of *resource description* with a unique identifier.
- GETVAL(X, δ) [X being a *resource description* D_r or a *profile* π] returns the value assigned to *descriptor* δ in the entity X.
- PUTVAL(X, δ, σ) assigns the value σ to the *descriptor* δ of the entity X.

2.3.4 Update Correctness and Propagation

We assume that *resource description* updates are atomic. Especially, two *resource description* updates or a *resource description* update and a read on this same *resource description* are processed in a serial order. Let us call a serial history of all the updates done on a *resource description* $\Delta D_r.history = \{\Delta D_{r_1}, \ldots, \Delta D_{r_n}\}$ according to the order in which the *resource description* updates were executed.

Definition 7 (Inconsistent Resource Description) *Let D_{r_1} and D_{r_2} be two replicas of the resource description D_r. D_{r_1} and D_{r_2} are said to be inconsistent if there are updates ΔD_{r_i} and ΔD_{r_j} such that ΔD_{r_1} reflects update ΔD_{r_i} but not ΔD_{r_j}, while ΔD_{r_2} reflects ΔD_{r_j} and not ΔD_{r_i}.*

Definition 8 (Older and Newer Resource Description) *D_{r_1} is called older than D_{r_2} if the serial history $\Delta D_{r_1}.history$ is the prefix of the serial history $\Delta D_{r_2}.history$. A replica of a resource description is called obsolete if there is a newer replica of the same resource description in the community environment.*

As mentioned in the declaration of Operator 1, the update process is initially applied on the user's device. Once it has been validated, update propagation must be performed in the case a *resource description* has been modified (i.e. D'_r returned by UPDATEDR is empty); it first applies the update to the servers, which will then dispatch the update to the every device that contained D_r. We assume the following correctness features of the update propagation function (denoted UPDATEPROPAG):

- Inconsistent replicas of the *resource description* must be eventually detected.
- Update propagation cannot introduce new inconsistency. *resource description* replica D_{r_i} should acquire updates from D_{r_j} only if D_{r_j} is newer replica.
- Any obsolete *resource description* will eventually acquire updates from a newer replica.

As a conclusion for this chapter, we want to point out that the management of annotations within metadata sets through IMAM (i.e. as part of the resource description and profile structures) needs to be centralized in order to fit the architecture described on Fig. 1.2 and not to be endangered by the limitations of most of the devices. Therefore, each resource description and profile must be stored on the server. Indeed, any update (from a user or an automated process) made to these structures is first applied on the device where it has been performed. Then, the update is reported to the server, which applies the update, and finally the server dispatches the update to all the devices where the considered resource description or profile is located (this motivates the fact that each resource description and profile keeps a record of all the devices where it has been copied).

Chapter 3

IMAM Implementation

> "Science is built up of facts, as a house is with stones. But a collection of facts is no more a science than a heap of stones is a house."
>
> - Henri Poincaré (1854-1912)

Standards for the manipulation of annotations through metadata are existing and are widely used. However, they tend to be too generic and complex to become very efficient. Bringing some restrictions to the usage of these standards and to metadata in general makes it possible to handle data and knowledge safely and properly. This is of course the aim of IMAM, which becomes even more attractive when associated with standards such as XML based metadata management frameworks. In addition, metadata needs to be exchanged efficiently; in our case, it is vital for devices to access and send annotations since it is the condition for our services to be performed.

In this chapter, after giving our motivations for the use of some standards in a specific framework, we present the architecture that fits IMAM requirements and defines the constraints on its simple components that have been described in the previous part. We finally show examples of implemented profiles.

This chapter is organized as follows:

- In Sect. 3.1, we motivate and describe the architecture we are proposing in order to apply and to support IMAM in the most efficient way.
- In Sect. 3.2, we provide implementation examples of IMAM components based on imports from annotations populated with Protégé.
- In Sect. 3.3, we summarize the benefits that can be obtained by using IMAM.

3.1 Architecture

3.1.1 Technical Choices

Semantic Management

As we explained it in Sect. 2.1, thesauri and ontologies are definitely the best solutions for structuring and relating the knowledge that is available about any *resource*. It is then important to enable the manipulation of this categorized information for systems through modeling such as IMAM which must be associated to a specific data model (in our case with XML).

In the context of the Semantic Web, it is necessary to provide data models to describe Web resources (e.g. Web pages) with application-independent languages such as the Resource Description Framework[1] (RDF) which are also applicable to multimedia resources. RDF is a syntax for representing metadata about resources. RDF is most commonly represented in XML, therefore benefiting from platform-independence and other XML advantages. RDF is an exciting technology that will eventually allow online agents (e.g. automatic spidering applications) to gain and infer knowledge, such as date, subject and relationships, about online resources and services. RDF, with the Web Ontology Language[2] (OWL), forms the cornerstone of the World Wide Web Consortium's Semantic Web activity and projects aimed at the embedding, gathering and automated understanding of metadata. An Example of the usage of OWL as a more abstract modeling layer on the top of XML data sources (described by XML Schema) has been given [LF04] and shows how the semantic relationships provided by OWL can be used for mapping heterogeneous data sources to a common global schema.

The vocabulary used to describe documents can be specified in terms of ontologies, where each description term and its semantic relation to other terms are defined. Thus ontologies facilitate the sharing and exchange of information about multimedia between applications and users. Accordingly, a multimedia ontology comprises a shared vocabulary to describe multimedia documents and their organization in a structured way such that users and applications can process the descriptions with reference to a common understanding specified in ontologies. All this knowledge structure and content can be quite easily managed within RDF, once it has defined. RDF and OWL are definitely interesting tools for possible implementations of IMAM.

Distributive Environment

Although the solutions presented in the previous section are very attractive and powerful, we agree with Jim Gray [Gra04] who claims that it is not possible to provide a generic knowledge structure that would be applicable to the whole world wide web. Indeed, essentially, the Web is fully heterogeneous and will remain so. Thus, it is imperative to apply some restrictions to the architecture our knowledge management is applied to. Since we decided to rely on metadata, it is vital to ensure the reliability of the annotations and their

[1] http://www.w3.org/RDF/
[2] http://www.w3.org/2001/sw/#spec

categorization. Hence, communities appear to be the best environment where collaborative information and metadata can be gathered.

Distribution of data within communities has been a very controversial topic for some years already. Indeed, with the advent of efficient data sharing systems based on P2P protocols, looking for and downloading multimedia documents through internet as a member of a community (which provides such documents) is easy; as a matter of fact, both music and movies majors are currently facing a uncontrolled distribution of copies of their products. The success of P2P computing is attributable in part to the rise of XML as a common descriptor language, as well as the arrival of standards-based solutions that ensure the integrity of data and services accessed by P2P software. To be able to build a system relying on P2P with XML, it is necessary to consider several properties that are required by distributed systems:

- Manageability: it is necessary to manage complex systems (updating, repairing, and logging).
- Information coherence: audit-ability and consistency are aspects of information coherence.
- Data access: peers need simple and efficient querying solutions for retrieving data.
- Extensibility: it is the ability of the system to be grown and to get new resources.
- Security: it prevents people from taking over the system, injecting bad information...
- Scalability: it indicates the limitations for the system to become larger.
- Legal Issues: it is vital to know the aim of the system and the ways people are going to use it...

It is easy to notice that regarding some of these criteria, decentralized systems are not always better than centralized systems. The simplicity of centralized systems makes them easier to manage, control, and perform queries in distributed environments (e.g. Mariposa [SAL96]) even if some decentralized systems are providing effective querying support in P2P networks such as DBGlobe [PAP03] or REMINDIN' [TSW04] which targets *row* information stored in file systems; moreover, decentralized systems grow better and are more resistant to failures or shutdowns as it has been pointed out in [RL03]. However, scalability of decentralized systems is hard to evaluate; it remains an active research topic. Therefore the choice depends entirely on the needs of the applications. Using hybrid topologies covers many of the drawbacks of both system types. Indeed, different topologies can be chosen for different parts of a system to get the best of the strengths without the weaknesses. Thus, Client/Server distribution model can still be useful to compensate the weaknesses of the peers (such as fragility, de-connectivity, battery limitations...).

3.1.2 Framework Description

Hybrid Architecture

As the use of the term *pervasive* is still quite exaggerated in order to describe the digital world we are living in, it is not possible to consider every device as reliable and powerful enough to support a fully decentralized system. Indeed connectivity and capacity of most devices, that are commonly used as we are writing

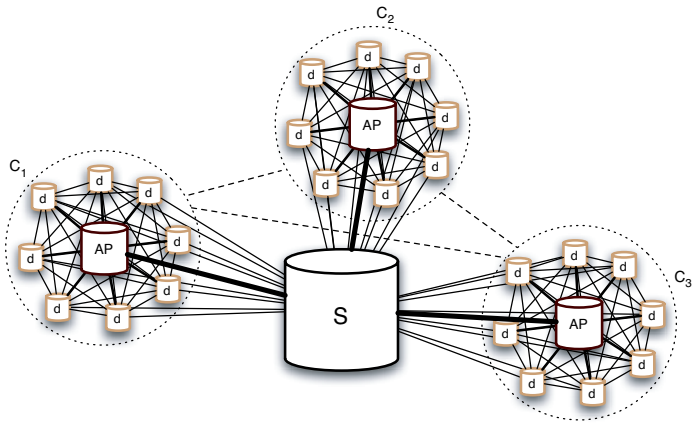

Figure 3.1: IMAM deployment architecture

this dissertation, remain limited; we cannot assume yet that devices (especially mobile ones) have enough processing power and network access so they could support the whole servicing a model such as IMAM is proposing.

Therefore, we adopt and recommend a hybrid centralized/decentralized approach for the deployment of IMAM with servers, *access points*, and devices (respectively denoted S, AC, and d on Fig. 3.1). This strategy makes it possible to take advantage of the strong experience and reliability centralized systems have acquired and of the flexibility and efficiency of decentralized systems.

The framework, which IMAM is using, is based on the concept of project (such as DSR), i.e. groups of people that share some interests, allocate some computing resources, and spend some time on collaborative activities. Then, within a project, several communities can be represented (e.g. DSR with communities focusing on architecture, religions or sociology...). Note that there might be some common interests between the communities (this is in fact obvious as they are part of the same project)

In order to support IMAM efficiently, the framework has to handle the physical entities that appear on Fig. 1.1 and to respect the architecture described on Fig. 1.2. IMAM deployment architecture is shown on Fig. 3.1 and is made of the following elements, which are all required to have a memory space fully dedicated to the projects using IMAM on their disk:

- Server (S): In our architecture, the server acts like a repository that is performing services dedicated to all the devices that are involved in the communities registered in a project. it provides typical database services (such as back-up, safety of the data...) and offers computing power for costly indexing and operative tasks (tasks that are clearly emphasizing the centralized client/server structure). *Resources*, which are inserted by a member of a community into the shared environment, are initially stored on the server from where services can be initiated. As shown on Fig. 1.1, the server side can be made of several servers that have hierarchical roles.

- *Access Point* (AC): the motivation for this special device is to make the balance in our framework between the server side and the other devices; it is a device that can be used as a sub-server, and must have sufficient computing power and network bandwidth access (typically desktops that are rarely shutdown). One *access point* is required for each community, and is used to store *resources* that are relevant to the community. In Chap. 9, we will use the *access points* for the data placement as a first layer for the adaptability dedicated to the communities.

- Device (d): they are devices that have input ports (ideally network connection) and are compatible with at least one type of multimedia document. They must be registered as used by at least one community member in order to benefit from the community services. Devices are seen as simple clients from the server side, but they are seeing each-others as peers as soon as they are involved in a same community (as it appears on Fig. 3.1. The main advantage for this P2P structure is to make the access to the resources easier and faster. NB: a user can be involved in several communities and a device can be shared by different users being involved in different communities.

In cases when project are quite small (e.g. with only one community), the architecture can be simplified. A family sharing data such as pictures, agendas... can use a simple PC as a server. It will of course generate some basic administration tasks such as ensuring there is enough available disk space, or checking the rights users have on the system; anyway, it is far more easier than managing a fully dedicated server such as the one used for DSR.

IMAM Settings and Strengths

After the creation of a new community, there is an initialization phase; during this period, the services are less selective as the comparisons between resources to find the most relevant ones are less competitive. This period ends when the amount of resources that have been inserted within the community (i.e. on the server) becomes big enough. From the early time of a community, the main activity for the members is to populate the repository with new resources and their resource descriptions. This is a necessary condition for the community to exist.

Then occur the update operations; they can be applied to resources description and profiles. The resource description update process has been well defined in Sect.2.3.3. We can mention here that the process, which include the propagation of the update (resource descriptions and profiles are easily copied on several devices, and updates, once validated, must be reported to all occurrences of the updated entity) is centralized: each modification is first reported to the server which then propagate the update on all the occurrences of the modified item that are dispatched within the community. This is why the Resource Description and profile keep a record of all the locations where they have been copied (resource descriptions also keep the record of the resource locations). This propagation strategy is definitely centralized and complies with the operator defined in sections 2.3.3 and 2.3.4.

IMAM services are basically performed on the server side; this is indeed the only solution that ensures

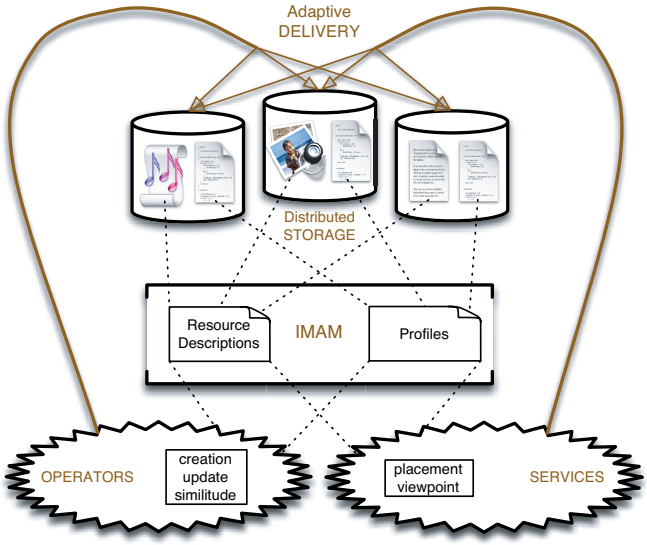

Figure 3.2: Adaptive delivery of resources through IMAM's operators and services

a quality of services as it does not rely on weak devices. This strategy is nothing but common, whereas the services, in themselves, are innovative; indeed, there are no equivalent processes proposed by systems that manage collaborative data. We briefly show some qualities of these services based on IMAM that will described in details in Part III:

- placement: as we mentioned it in the previous section, [RL03] provides an interesting management of replicated data using metadata; however, the proposed algorithm does not consider the users' characteristics when performing replica reconfiguration, and so misses the opportunity to offer a more relevant data placement.

- viewpoint: REMINDIN' [TSW04] is managing data on P2P networks that is contained in predefined locations (so do most of P2P files sharing systems); the REMINDIN' peer behavior for answering queries is based on peers which successfully answered these queries. The query management proposed with IMAM is more traditional from first sight (as it is simply based on key words search) but the viewpoint provided in addition, by considering the querying environment, makes the query answer more relevant to users.

A more complete and consistent review and comparison of the work related to IMAM services is proposed in Sect. 7.3. The framework in which IMAM is applied is described on Fig. 3.2 where the distributed storage is the architecture represented on Fig. 3.1 and is considered as an entity where IMAM operators and services are performed from the server side.

3.2 IMAM Knowledge Entities

In this section, we present one source of annotations we are currently manipulating. We describe the metadata management used for this project and explain how users are inserting the annotations with Protégé. We finally introduce our strategy for correctly extracting the useful annotations depending on constraints applied to the profiles descriptor values.

3.2.1 Ontology-based Metadata Management

A key feature of this system is the multilingual ontology-based metadata support. Our platform follows a promising research approach based on the usage of metadata and ontologies. Metadata is any information which characterizes instance data, and which describes its relationship. Metadata is used to provide an effective use of data, in order to facilitate any data management, any data access, and data analysis. An ontology is an explicit specification of the conceptualisation of a domain. Ontologies enable domain experts to create an agreed-upon vocabulary and semantic structure for exchanging information about that domain. Ontologies facilitate cataloguing and sharing knowledge, as domain expert are able to contribute to a shared, worldwide, but well-organized knowledge base of technical information. We considered a metadata management architecture and designed multi-layer ontologies to classify and describe resources. It is based on Protégé 2000 . Each ontology is related to one field such as history, geography, architecture, and art... However, possibility of overlapping exists as different ontologies may have equivalent concepts, or may contain subsets of separate ontologies within themselves.

This problem of ontology integration has been solved by classifying and reorganizing ontologies in a logical and semantic sense according to metadata. This points to a need for a formal model for ontology-based metadata management. Ontology is the formal and explicit conceptualization of a particular domain. It includes a set of concepts and their relationships. Based on Protégé 2000, we defined our ontology structure as a 6-tuple: $O := \{C, P, A, HC, prop, att\}$ where C represents a domain-based set of concepts, P a set of relation identifiers, and A a set of attribute-value relations.

***Example* 5** *Let us consider a subset of our ontology structure related to spirituality:*
$C :=$ {*SPIRITUALITY, RELIGION, LANGUAGE, OBJECT, LANGUE, BOUDDHISME*}, $P :=$ {*EXPRESS, CREATE*} *, and A defines the relations EXPRESS(RELIGION, LANGUAGE) and CREATE(RELIGION, OBJECT).*

HC is a Hierarchy of Concepts that are linked together through relations (e.g. specialization, generalization). $H_c \in C.C$ is a directed transitive relation called concept taxonomy; function $prop : P \rightarrow C.C$ relates two concepts non-taxonomically; function $att : A \rightarrow C$ introduces the relationship between concepts and literal values.

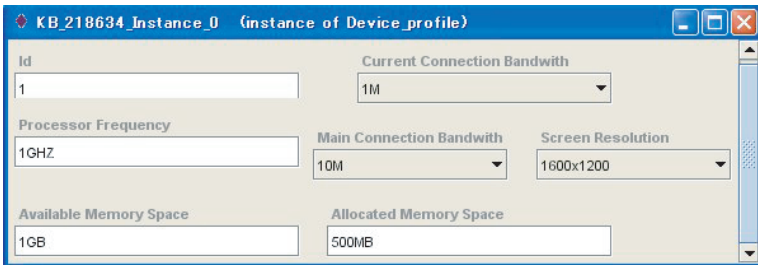

Figure 3.3: Interface for device profile (extract)

3.2.2 Resource Entry

One solution that has been proposed for DSR was to use a metadata management based on the framework described in the previous section. This strategy is moreover currently used as part of SPP (Science Partnership Program) from the Japanese Ministry of Education, which establishes a cooperation program between teachers and researchers related to science. The objective of this project is to initiate students to the usage of digital contents, to the creation and manipulation of metadata related to these contents, and to the annotation process.

Resource entry is performed via a web browser interface similar to that used for querying. Users who enter resources need to log on. Write, modification and suppression rights can be assigned and controlled by the system administrator for each user; some predefined types of user provide community and group management abilities. Information such as the identification profile of the user and date of the entry are automatically filled in by the system. To maintain the integrity of the resource being entered into the system, the controlled lists of relevant vocabulary within the thesaurus are used for each translatable field. When uploading resource via the web interface, users are required to enter some preliminary metadata related to the resource. When a resource is saved into the main database, the metadata is translated into a language independent code representation. The creation of metadata profile is done according to the metadata category such as structural metadata, content metadata and contextual metadata avoiding overlapping between attribute sets. The following screen shots describes the profile management application: Fig. 3.3 for the device profile, Fig. 3.5 for the community profile, and Fig. 3.4 for the user profile.

3.2.3 Conceptual Structures

Once the annotation has been inserted by a user or an automated process into Protégé, it is very simple to export information from the structures presented above into XML file (or even RDF). However, it is important to make sure that all the requirements that IMAM is bringing, are followed by the profiles edited via Protégé; this is indeed the step, which needs some corrections when exporting the profiles into XML. In Table 3.1, we show the constraints that are applied to the descriptor values of the different profiles (Note that the descriptors' numbers are the same ones as in Table 2.2):

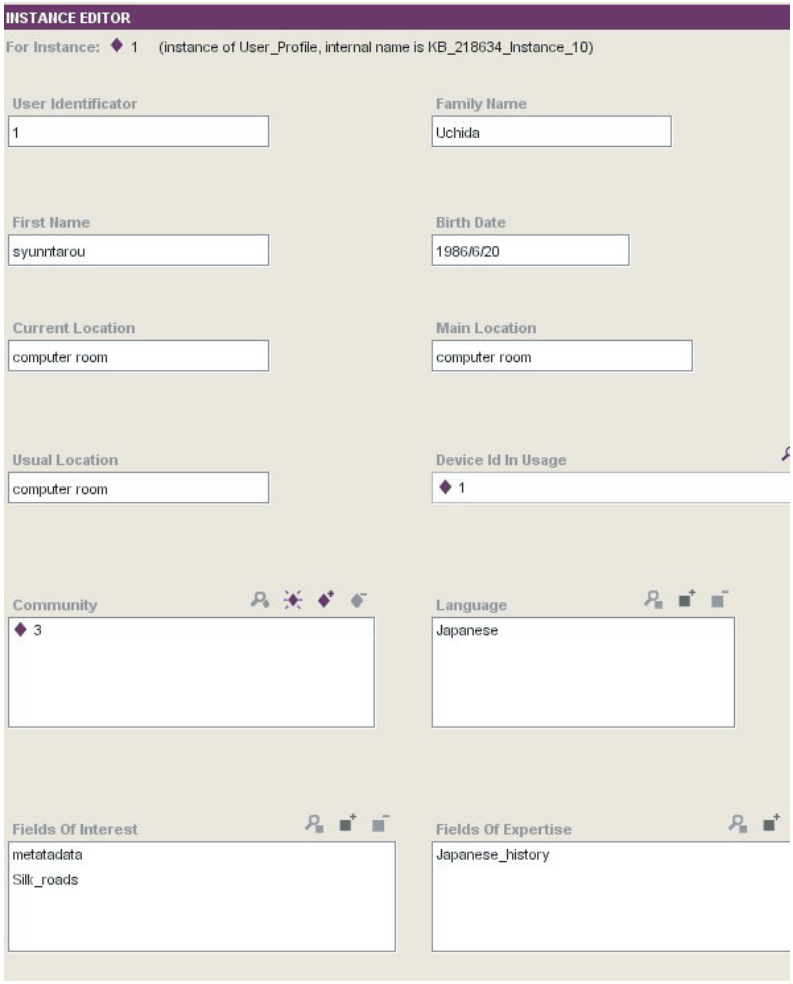

Figure 3.4: Interface for user profile (extract)

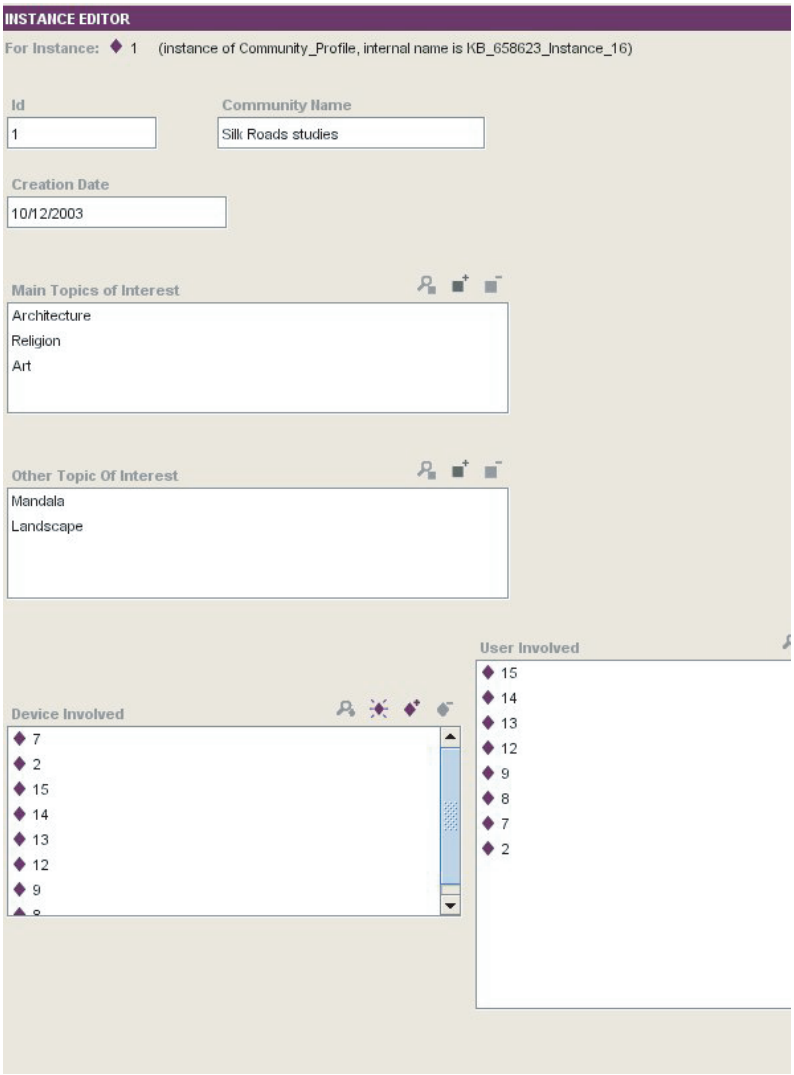

Figure 3.5: Interface for community profile (extract)

Table 3.1: Constraints on profiles' descriptor values

descriptor	required	automated	unique value	list of terms	predefined list	IDs list	predefined format	boolean
user profile								
1	√	√		√				
2	√							
3							√	
4			√					
5				√				
6		√	√					
7				√				
8					√			
9					√			
10				√				
11		√				√		
12		√				√		
device profile								
1	√	√		√				
2	√				√			
3	√		√					
4	√	√						
5		√	√					
6		√	√					
7		√	√		√			
8			√		√			
9		√	√					
10								√
community profile								
1	√	√						
2	√			√				
3	√		√		√			
4					√			
5	√					√		
6						√		

Simple examples of the possible results of profiles in XML exported from Protégé are displayed in Sect. A).

The resources descriptions are basically following a path of the RCT; therefore, once the RCT is fully defined and implemented, it becomes trivial to extract the nodes and descriptors that are needed by a specific resource. In Sect. B.1, we provide an extract of the core RCT implementation in RDF (see Fig. B.5) with an extract of the related schema (see figures B.1- B.4).

3.3 Conclusions

In this part, we introduced the motivations of the work presented in this dissertation; the first chapter described the issues, which management of multimedia resources is facing. We showed that the collaborative frameworks, and in particular communities, provide attractive characteristics for capturing environmental

knowledge. Thus, we gave an overview of the architecture that we are proposing in order to offer innovative services to communities' members, who can enjoy a better access to the resources. We then defined, in Chap. 2, the first and main component of this architecture, which is a modeling that enables to represent and categorize any knowledge about entities involved in the sharing of resources by communities. This modeling, based on a knowledge categorization tree, is made of two main components:

- The *resource description* is an extended subset of the RCT, which is a tree structure that categorizes the information describing resources and their content.
- The profile is a subset of the *resource description* that aims at containing and categorizing the available and useful knowledge about users, communities, and devices.

These two entities make it possible to capture any environmental knowledge that might be involved in the sharing of resources, and moreover in the processing of automated services dedicated to the adaptive delivery of resources. We designed some simple operators that enable us to manipulate the resources and their resource descriptions, to compare resources types, and to ensure the consistency of data that might be often updated and copied. In this chapter, we finally described the structure of the architecture that fits IMAM and gave some illustration of profiles.

Part II

Information Management with XML

Chapter 4

Foundations for a Collaborative Information Model

> "We can be knowledgeable with other men's knowledge but we cannot be wise with other men's wisdom."
>
> - Michel de Montaigne (1533-1592)

In Part I, we described the information modeling for adaptive management, called IMAM, which offers a set of high-level operators for handling resources and related metadata. IMAM is definitely influenced by XML; the tree structure with ordered set of attributes indeed fits perfectly the Extensible Markup Language. Moreover, XML offers an amazing opportunity to extend and personalize its structure, and also to easily reuse and improve existing extensions or tools. This explains why XML has been very popular, and is widely used as a standard data model.

In any collaboration scenario, document metadata play an important role for indexing and retrieving documents in jointly used archives. Yet, one critical problem concerning metadata is the reliability of user provided data. Incompleteness and inconsistency of metadata are frequent; incompleteness results from partially skipping of information, data being incorrect cause inconsistency. Using a well defined knowledge structures such as RCT, *resource description*, and profiles may help, but this demands additional effort to check data correctness. Thus, it is important to evaluate all criteria that determine correct data, in our case XML. We are also confronted to encoding issues as people involved in communities relying on IMAM might use different languages and so manipulate various sets of characters. XML once again provides an interesting solution with Unicode support. However, the representation and management of different languages within communities has to be taken care of.

In this part, we motivate some requirements that we applied to define IMAM (such as versioning or update) and describe XML-based tools that are relevant to our modeling. Then we focus on database management of XML (namely storage and indexing) and provide a scheme that aims at handling the contextual information we are dealing with.

In this chapter, we consider XML as our data model and investigate interesting properties and useful extensions. Some of these elements simply have to be integrated in our framework in order to achieve our goals. However, manipulating XML implies to be very careful with data correctness; we point out each step that has to be performed in order to ensure data consistency. We finally overview the mixed transactional layer (containing distribution & DBMS transaction processing) that is necessary for adaptive collaborative services to exist.

This chapter is organized as follows:

- In Sect. 4.1, we describe a set of interesting XML extensions and tools.
- In Sect. 4.2, we provide directions that allow us to ensure data consistency.
- In Sect. 4.3, we review XML support for transactions and give directions for the protocol to be used in IMAM services.

In Chap 5, after considering the existing solutions for XML storage, we provide a new XML storage management scheme based on an advanced graph data model. Chap. 6 describes an efficient structural indexing support for this storage strategy dedicated to contextual information within XML.

4.1 Contextual Information Representation

The information model definition is based on the description of the data we want to handle. To keep a generic approach, we consider cultural multimedia documents. Therefore, to be able to deal with users' relevant information and offer multi-viewpoints, we focus on an appropriate management of multidimensional data. As we explained in Sect. 3.1, XML is considered as a language for both data and transactions management. The main aspects presented here aim at bringing context dependency to the information model and to handle XML extensions or tools that can be helpful in order to implement IMAM framework.

4.1.1 XML Comparison

There is a critical need for XML files comparison; indeed, a differencing utility is an especially important tool for people working on collaborative projects, because it allows them to quickly identify which parts of the code have changed, saving time and effort in editing, troubleshooting, and versioning. Moreover, evaluating similarities (both structural and semantic) enables systems to build indexes providing useful information for information filtering processes.

Finding what has changed in your XML can be hard. Representing that change so that it can be processed is harder[1]. XMLSpy[2] includes a visual XML differencing utility that allows users to easily compare and merge XML documents and directories in a XML-aware manner; it only offers structural comparison support.

XMLdiff[3] is a python tool that figures out the differences between two similar XML files, in the same way the diff utility does it for text files. It is using a XSL transform, that when applied to the master file, generates another XSL transform, which confirms the existence and positions of nodes from the master file in the second file. It basically checks whether the files are equivalent from the XPath standpoint. Even if the second file has more nodes than in the input file, it is irrelevant as long as the XPaths are still the same.

Existing solutions are not sufficient for *resources* comparison (difference and intersection) and similarity evaluation. Our proposal is described in Sect. 2.3.2 and is applied in an appropriate form for data placement processing in Chap. 9.

4.1.2 GIS Sets

Since we have to manage geographical data, it is important to be able to handle typical GIS features. Moreover, the geographical information layers, which are most of the time represented through maps like in [LH04], can be a very convenient gateway to link and access any kind of data.

The Open GIS Consortium[4] has released OpenGIS Geography Markup Language (GML) Implementation Specification V3.0 featuring a new modular design. GML is an XML grammar written in XML Schema for the modelling, transport, and storage of geographic information; it provides a variety of kinds of objects for describing geography including features, coordinate reference systems, geometry, topology, time, units of measure and generalized values.

The G-XML project [AK00] is a Japanese project involving universities and companies (from July 1999, funded by the Ministry of International Trade and Industry). It aims at establishing a standard XML based protocol for spatial data exchange (i.e. for GIS applications). The approach is market oriented; it is focused on end-consumers (e.g. mobile phones). In fact, there are two objectives: establishing the protocol and developing some prototype systems in order to validate the protocol. The G-XML protocol is only a set of DTD. There are seven G-XML prototypes (e.g. viewer, wrapper for SVG...). It is offering license-free software components. The first period (definition of the protocol and prototypes) is over (March 2000). A new project is running in order to upgrade G-XML and make it become an international standard. The main quality of G-XML is to be divided into four sub-protocols: Real World G-XML (RW-GXML), Point & Direction Based G-XML (PD-GXML), Semantic G-XML (S-GXML) and Graphics Based G-XML (G-GXML). This approach allows developers and users to get easily what they are looking for. The only programming language used is Java.

The only real standard is actually GML; it is used by all the main GIS companies and organizations in the

[1] http://compare.deltaxml.com/
[2] http://www.altova.com/products_ide.html
[3] http://www.logilab.org/projects/xmldiff
[4] http://www.opengis.org

whole world. However, it is important to keep an eye on the Japanese GXML that could have an increasing impact in Asia. Finally, we must point out that GIS related data often contains multidimensional structures and relies on contextual information such as localization. The context-dependency as been taken into account for the design of IMAM in the previous chapter. The multidimensional aspect is an important issue to be considered for storage and indexing matters; we provide relevant solutions in Chap. 5 and Chap. 6.

4.1.3 XML-based Multilingual Support

The elements we described above are very relevant for the management of large amounts of XML multimedia documents, and moreover when they are linked to geographical data. This is the case with the Digital Silk Roads project. DSR data is presenting another important characteristic that has to be handled by IMAM: it manipulates cultural resources form people of different countries and nationalities. As it has been mentioned in Sect. 2.2.1, RCT is a monolingual structure; however, some metadata, for some predefined *descriptors*, can provide multilingual content in *resource descriptions*. Therefore, it is imperative to be able to manage multilingual annotations for *descriptors values* within XML files.

In this section, we will consider thus an important aspect for IMAM: the multilingual support. We present here the management of multilingual information within XML (which is supporting Unicode). Note that we do not give an example of the use of MXML for multilingual management here since it will be well described in Sect. 4.2.3.

Using XSLT

When using XML attributes, it is necessary to utilize the `xml:lang` attribute in the XML documents; in other words, it requires to repeat the text in the multiple languages as needed and mark them accordingly. For example, Fig. 4.1 shows the multilingual ability of XML:

```
<MyText>
   <Text xml:lang="ja">日本語はここ</Text>
   <Text xml:lang="en">English goes here</Text>
   <Text xml:lang="it">Italiano e' qui</Text>
</MyText>
```

Figure 4.1: Extract of a multilingual XML file

Next, an XSLT[5] stylesheet has to be created to select the proper element based on the language; see example on Fig. 4.2.

XLIFF

XLIFF[6] is a format to store extracted text and carry the data from one step to another in a localization process. An XLIFF document is composed of one or more <`file`> elements, which correspond to original files or

[5] W3C (1999) XSL Transformations (XSLT): www.w3.org/TR/xslt

[6] Founded: Sept 2000; Founding Members: Novell, Oracle and Sun. http://www.opentag.com/xliff.htm

```
<xsl:stylesheet version = '1.0'                                          1
     xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>                   2
<xsl:template match="Text">                                              3
     <MyText>                                                            4
          <xsl:choose>                                                   5
               <xsl:when test='lang("ja")'>                              6
                    <xsl:Text>Japanese version: </xsl:Text>              7
               </xsl:when>                                               8
               <xsl:when test='lang("it")'>                              9
                    <xsl:Text>Italian version: </xsl:Text>              10
               </xsl:when>                                              11
               <xsl:when test='lang("en")'>                             12
                    <xsl:Text>English version: </xsl:texT>              13
               </xsl:when>                                              14
          </xsl:choose>                                                 15
          <xsl:value-of select="."/>                                    16
     </MyText>                                                          17
</xsl:template>                                                         18
```

Figure 4.2: Language selection through XSLT stylesheet

sources (i.e. database table in the case of RDBMS) identified by original attributes. A <file> contains the source of the localizable data and, once translated, the corresponding localized data for one, and only one, locale. Localizable data are stored in <trans-unit> elements. The <trans-unit> element holds a <source> element to store the source text, and a <target> element to store the latest translated text. The multilingual management with XLIFF is quite obvious. The <target> elements are not mandatory, and it is also perfectly correct, if desired, to duplicate the original source text in the <target> element at the beginning of the process (with the xml:lang attribute set to a target language). The following XML extract shows a detailed example of translation management with XLIFF (see Fig. 4.3).

4.2 Data Consistency

We have to point out here that managing XML files implies to ensure that XML files are valid and well-formed before insertion; i.e. to use a cleaning method in case of recuperation [GMA02], and a correct process for the creation of new data. The encoding correctness is moreover important since we are working with multilingual data. This is a necessary condition for any XML file to be used effectively.

4.2.1 XML Correctness

One reason to use XML is a well-defined validation mechanism in XML Schema; no such mechanism exists for flat file formats. The first point which has to be considered, is the strictness of XML when writing code. The whole idea of XML[7] is that it should be independent of the platform it is running on. The same code should run the same way on a PC, a Mac, a mobile phone and even a toaster. As XML does not actually do anything (it is just a language for defining data), it is up to software developers to make software to use this data on a particular platform. This means that it is important that all XML code is structured the same way,

[7] see XML specifications: http://www.w3.org/XML/Core/#Publications

```
<?xml version="1.0"?> <xliff version="1.0">
 <file original="Graphic Example.psd"
  source-language="en" target-language="ja"
  tool="Rainbow" datatype="photoshop">
  <header>
   <skl>
    <external-file uid="3BB236513BB24732" href="Graphic Example.psd.skl"/>
   </skl>
   <phase-group>
    <phase phase-name="extract" process-name="extraction"
     tool="Rainbow" date="20030214T152258Z" company-name="NII" job-id="123"
     contact-name="Jerome Godard" contact-email="jerome@grad.nii.ac.jp">
     <note>Make sure to use the glossary I sent you yesterday. Thanks.</note>
    </phase>
   </phase-group>
  </header>
  <body>
   <trans-unit id="1" maxbytes="14">
    <source xml:lang="en">Quetzal</source>
    <target xml:lang="ja">Quetzal</target>
   </trans-unit>
   <trans-unit id="3" maxbytes="114">
    <source xml:lang="en">application handling and processing XLIFF</source>
    <target xml:lang="ja">XLIFF 文書を編集、または処理するアプリケーションです</target>
   </trans-unit>
   <trans-unit id="4" maxbytes="36">
    <source xml:lang="en">XLIFF Data Manager</source>
    <target xml:lang="ja">XLIFF データ・マネージャ</target>
   </trans-unit>
  </body>
 </file>
</xliff>
```

Figure 4.3: Translation management with XLIFF

so that software can easily be developed. Because of this requirement for correct code, it has been decided (and is a standard) that if any mistakes (for example incorrectly nested tags) are found in XML code, it will not execute, and will just give an error message. This means that when writing XML, developers must be very careful about correct syntax.

4.2.2 Encoding Problems

By definition, an XML document can contain any Unicode character except some of the control characters. Unfortunately, many databases offer limited or no support for Unicode and require special configuration to handle non-ASCII characters. If data contain non-ASCII characters, it has to be ensured that database and data transfer software handle these characters. A cleaning process is very often needed to make pure Unicode files. As an example, a rigorous strategy is described in [GMA02] in the case of multilingual lexical data contained in XML files.

4.2.3 Versioning

Versions management is a dramatical issue in collaborative environments. The main problem brought by data distribution within communities is the heterogeneity of users and behaviors, in opposition with the traditional databases where the number of users is well known and the behaviors quite homogeneous. Another problem

is to integrate data based on the same conceptual entity with different contents from various sources. These issues are particularly true for IMAM as it aims at associating automated and human processes for annotation and distribution matters.

In order to clear up those difficulties, Multidimensional XML (MXML) enables the context-dependency of XML documents. MXML was first introduced in June 2000 [SGR00] as an extension of XML. The aim of this language is to offer various contents from the same source according to what the user is looking for and to what kind of user he is (depending on his background, behavior and preferences). MXML moreover avoids keeping many files that have the same content in different languages for example. In this way, it enables manipulating easily several views of one document. A precise syntax [SGM00] shows how to include dimensional dependencies into XML and some formal definitions (multidimensional properties, generic aspect) and structural features (multidimensional DTD support and graph data model) [GSK01] give a more precise and complete view of MXML. Some interesting properties are then described: the value of one dimension does not depend on the values of others dimensions; so are the dimensions said to be orthogonal. The property of inheritance of contexts throughout the paths of the document graph is assumed: the reduction into a conventional XML document of an MXML document in a specific world is consistent with the fact that MXML is a subset of XML. One specific illustration of the use of MXML has been described [MGS01] to represent time-dependent information. Using the properties of Time domain, MXML integrates elements and attributes having time-period dependency. Several dimensions can be used within time subset. E.g. Day/Month/Year is a three dimension space; that can itself be a subset of the dimension date. Giving a formal approach of a multidimensional logic, [WLO01] defines a set of possible contexts and its properties that seem to be very convenient for MXML. The dimension operators based on the multidimensional logic called ML(ω) that are very effective to get a multi-viewpoint perspective of data (the data model used is hypercube-based). It then describes the prototype of a multidimensional, XML database system.

Treating XML as normal text in an information management system can work for roll-back and basic version control. However, the delta files will often be unnecessarily large, for example recording changes to attribute order, which is not a significant XML change. Critically, to process a difference report, it is necessary to work with text-based, line-oriented results, which do not map well to the XML originals, making post-processing very difficult. Because aline-by-line text-based systems require extensive custom (i.e. domain-specific) extensions of text-based algorithms to cope with tree-structured XML files, such solutions have proven unreliable and often simply will not work. DeltaXML[8] provides a generic solution for merging data that has been edited or amended by several people or systems independently. When looking for XML document management system which has the feature of version management, some people even use systems such as CVS (an open source source code management system) but it is not XML-aware directly.

Finally, we have to mention XML Namespaces. An XML namespace is a collection of element type and attribute names. The namespace is identified by a unique name, which is a URI. Thus, any element type or attribute name in an XML namespace can be uniquely identified by a two-part name: the name of its

[8] http://www.deltaxml.com/

XML namespace and its local name. This two-part naming system is the only aspect defined by the XML namespaces recommendation. XML namespaces are declared with an `xmlns` attribute, which can associate a prefix with the namespace. The declaration is in scope for the element containing the attribute and all its descendants. Namespaces have two purposes in XML:

- To distinguish between elements and attributes from different vocabularies with different meanings and that happen to share the same name.
- To group all the related elements and attributes from a single XML application together so that software can easily recognize them.

The first purpose is easier to explain and to grasp, but the second purpose is more important in practice. However, Namespaces have sizeable drawbacks: relative namespace URIs in particular in the intersection between RDF and XML.

IMAM versioning scheme is provided through the *resource description* update management, which has been described in Sect. 2.3.3. Our approach is indeed more restrictive than versioning schemes reviewed in this section. However, it perfectly fits collaborative management of shared *resources* (as defined in the previous chapters) and is less costly.

4.3 Data Distribution

4.3.1 Transactional Issues

Distributed Systems

The most significant changes brought about by XML have been in the way distributed systems store and exchange information; the main area where XML contributes is P2P computing. Among the compelling applications that are emerging from it:

- Collaboration. Applications (such as instant messengers) enables users to form ad hoc virtual workspaces, where individuals can share schedules and documents, conduct voice, video and text conversations, and perform other tasks.
- Distributed processing. More and more communities (firms, labs, groups of individuals) are using P2P software to tap unused processor cycles to create virtual supercomputers over the network.
- Content distribution. P2P software (such as BitTorrent[9]) makes it possible to cut down WAN traffic by allowing systems to seek files from other systems on local networks. The result: faster downloads, reduced WAN bandwidth usage, and lower operating costs.

[9] http://bittorrent.com/

- Knowledge management. P2P knowledge management applications (such as Kazaa[10]) simplify information handling, using intelligent agents to sift through messages, Web sites, and other data sources according to the individual's profile.

DBMS Transactions

A transaction is a set of one or more statements that are executed together as a unit, so either all of the statements are executed, or none of the statements is executed. Typical problems DBMS designers have to solve are related to transactions; it is indeed vital to ensure that concurrency and update processes are safe, otherwise the data and the information contained in DBMS cannot be reliable. In databases, the ability to handle transactions allows the user to ensure that integrity of a database is maintained. ACID properties for transactions have been defined and are widely used in order to tackle these issues:

- Atomicity: the entire sequence of operations in a transaction must be either completed or aborted.
- Consistency: the transaction takes the resources from one consistent state to another. If the transaction cannot achieve stable end-state, it must return the system to its initial state.
- Isolation: a transaction's effect is not visible to other transactions until the transaction is committed.
- Durability: the results of a committed transaction are permanent and must survive system failure.

A distributed transaction is a database transaction that must be synchronized among multiple participating databases which are distributed among different physical locations. A common algorithm for ensuring correct completion of a distributed transaction is the two-phase commit. As our work aims at providing enhanced management of XML files, it is imperative to consider and apply the DBMS concepts.

4.3.2 XML-based Protocols

SOAP[11] is a lightweight protocol for the exchange of information in a decentralized, distributed environment. It is an XML-based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined data types, and a convention for representing remote procedure calls and responses. SOAP can potentially be used in combination with a variety of other protocols; however, the only bindings defined in this document describe how to use SOAP in combination with HTTP and the HTTP Extension Framework.

JXTA[12] is a set of open, generalized peer-to-peer (P2P) protocols that allow any connected device on the network (from mobile phone to PDA, from PC to server) to communicate and collaborate as peers. The JXTA protocols are independent of any programming language, and multiple implementations exist for different

[10] http://www.kazaa.com
[11] W3C (2000) Simple Object Access Protocol (SOAP): http://www.w3/TR/SOAP/
[12] http://www.jxta.org/

environments. Because the protocols are independent of both programming language and transport protocols, heterogeneous devices with completely different software stacks can interoperate with one another. The Project JXTA team designed a set of six protocols based on XML messages. Each of the JXTA protocols addresses exactly one fundamental aspect of P2P networking:

- Peer Resolver Protocol.
- Peer Discovery Protocol.
- Peer Information Protocol.
- Pipe Binding Protocol.
- Rendez-Vous Protocol.
- Endpoint Routing Protocol.

The JXTA Framework is a very good basis to implement a specific protocol set for our distributed mobile information system. Its layers are described on Fig. 4.4. This architecture fits our need very well and already supports all types of processes we want to provide as part of adaptive services.

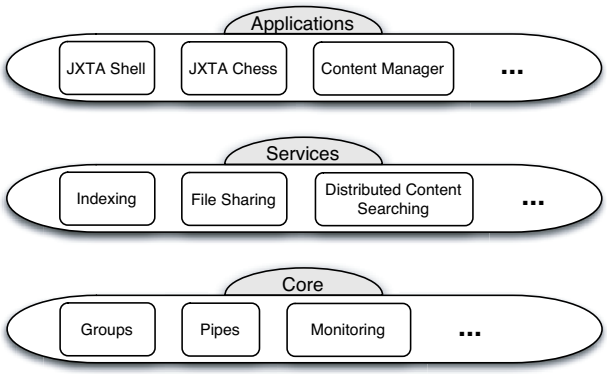

Figure 4.4: The 3 Layers of JXTA Services

A technical choice has to be done in order to choose what protocols to adopt. It mainly depends on the global architecture we decide to use. Since Peer-to-Peer approach is perfectly adapted to the needs described in Sect. 3.1.2, JXTA is an appropriate basis for a future implementation of delivery services using IMAM. However, as said previously, the fully decentralized approach will not be sufficient for at least some years; then the remaining issue to investigate will be the optimization of the transactions' performances, including the data compression.

Chapter 5

Multimedia-enabled XML Management

> "To me style is just the outside of content, and content the inside of style, like the outside and the inside of the human body. Both go together, they can't be separated."
>
> - Jean-Luc Godard (b. 1930)

While file systems, relational, and object-oriented database management systems have met people's needs for several decades in order to store and retrieve data, XML imposes new requirements on how that information needs to be stored so that it can be retrieved in a structured, hierarchical manner. Indeed, XML has gone from humble beginnings to widespread implementation in a comparatively short period of time. Initially popularized in the Web publishing industry as a document sharing technology, XML has evolved into an industry wide data communication and storage medium. Whereas multimedia documents once consisted of little more than text and images, these documents, now based on XML, have become the medium of choice for delivering data pulled from databases in the back end to applications and documents. XML itself is moving into the database, and is becoming the basic data storage structure. This chapter is devoted to management of multimedia related XML files. We focus on the database side of data management as we are considering XML data storage strategies and investigate the best approach to store IMAM related data.

The chapter is structured as follows:

- In Sect. 5.1, we present the possible strategies for the storage of XML files and give an overview of the existing solutions.
- In Sect. 5.2, we investigate the management of multimedia documents through XML.
- In Sect. 5.3, we describe XML mapping to an innovative graph data structure and show that this strategy is adapted to IMAM requirements.

5.1 Storage Strategies

As metadata favorite capture and manipulation language, XML content can be dedicated to processes (structural metadata), or to human consumption (guide metadata). There are indeed two main kinds of XML files[1]: data-centric and document-centric. Data-centric documents are using XML as a data transport. They are characterized by fairly regular structure, fine-grained data and little or no mixed content. Document-centric documents are usually documents that are designed for human consumption. They are characterized by less regular or irregular structure, larger grained data, and lots of mixed content. In fact, the document-centric view is resulting from SGML. As a general rule, data type is stored in a traditional database, such as a relational, object-oriented, or hierarchical database. Furthermore, document type can be also stored in a native XML database (a database designed especially for storing XML). But in the real world, with complex and heterogeneous XML files such as multimedia ones, it is very difficult to define the limit between those two types. That is why it is necessary to use hybrid systems to manipulate XML in order to keep its advantages. In this part, we will give an overview of the main and more interesting ways to store and retrieve XML documents.

Basically, the most common schemes applied today for XML in DBMS are the following:

- Decomposition (shredding).
- Storage of the entire document.
  - Storage of information about document structure in a database.
  - Storage of document content in the database, or store externally with pointers in the database.
  - If the former, provide full text searches.
  - Index on tagged fields.

5.1.1 Storing XML Data in a Traditional DBMS

In order to transfer data between XML documents and a database, it is necessary to map the XML document structure (DTD, XML Schema, RELAX-NG) to the database schema; many strategies have been proposed for mapping XML data to relational data [FK99, RP02, SKW00, STZ99]. The structure of the document must exactly match the structure expected by the mapping procedure. Since this is rarely the case, products that use this strategy are often used with XSLT. That is, before transferring data to the database, the document is first transformed to the structure expected by the mapping; the data is then transferred. Similarly, after transferring data from the database, the resulting document is transformed to the structure needed by the application. A way to move or correct the structure of multilingual lexical XML data is described in [GMA02].

One of the main weaknesses that imply the use of traditional DBMS is the waste of memory space. Implementation done in [SYU99] shows that from 7,65 MB of XML documents, it requires 11.42 MB as a

[1] XMLdev: http://www.xml.org/xml/xmldev.shtml

table-based structure; which is 50% more. Another drawback of using RDBMS is that it lowers performance since a mapping from XML data to the relational data may produce a database schema with many relations. Moreover, queries on XML data when translated into SQL queries may contain many joins, which implies expensive queries' evaluation and optimization.

Most relational database vendor are trying to make their database suitable for storing XML, and adding XML views of their relational data. But there is a variety of implementations and models for storing XML, and a lot of innovation is still going on.

5.1.2 Storing Data in a Native XML Database

This solution can be valuable when the data is semi-structured, i.e. has a regular structure. In this case, mapping XML to a relational database often results in either a large number of columns with null values (which wastes space) or a large number of tables (which is inefficient). A second reason that makes the use of traditional DBMS unattractive, is retrieval speed. Some storage strategies used by native XML databases store entire documents together physically or use physical (rather than logical) pointers between the parts of the document. This allows the documents to be retrieved either without joins or with physical joins, both of which are faster than the logical joins used by relational data-bases. For example, an entire document might be stored in a single place on the disk, so retrieving it or a fragment of it requires a single index lookup and a single read to retrieve the data. A relational database would require several index lookups and at least as many reads to retrieve the data.

Speed is increased only when retrieving data in the order it is stored on disk. Indeed, retrieving a different view of the data would probably bring worse performance than in a relational database. In [YIN00], the storage management is done through an offset space, which is an ad-dress space in secondary memory. This is an efficient way to store structures such as trees and avoids using multiple relations. An offset space is very similar to a main memory space and offers the same characteristics than UNIX file system does. We must here point out that this approach has been used from the beginning in the Phasme project [ABO96]. Another problem with storing data in a native XML database is that most native XML databases can only return the data as XML. Moreover, using version control systems such as CVS brings the possibility to have simple transaction management.

There is a bunch of native XML databases both from academic and corporate worlds. It is very difficult to compare these products since they usually have to be customized and optimized according to the application they are connected to. We present here some of the main existing solutions.

- Wolfgang Meier's eXist[2] is an open source native XML database [Mei02], featuring index-based XQuery processing, and XUpdate support. The current release benefits from a lot of testing done by other projects, and fixes many instabilities and database corruptions that were still present in the previous version. In particular, the XUpdate implementation should now have reached a stable state.

[2] http://exist-db.org

Concurrent XUpdates are fully supported. The XQuery implementation has matured, adding support for collations, computed constructors, and more. Module loading has been improved, allowing more complex web interfaces to be written entirely in XQuery (see new admin interface). Finally, there's a new WebDAV module, a reindex/repair option and support for running eXist as a system service.

- dbXML[3] is a native XML database that supports four different data stores. The first of these is a proprietary data store that uses B trees. The second is an in-memory data store, which is used for temporary storage and whose contents are deleted when the database is stopped. The third is the file system. And the fourth is a mapping to a relational database (it is not known what mapping is used). dbXML is capable of storing and indexing collections of XML documents in both native and mapped forms for highly efficient querying, transformation, and retrieval. In addition to these capabilities, the server may also be extended to provide business logic in the form of scripts, classes and triggers.

- Tamino[4] XML Server is Software AG's XML server [Sch01] for storing, managing, publishing and exchanging XML documents in their native format, based on open-standard Internet technologies. The XML engine uses the Data Map, which describes where the data in a given XML document is stored. This allows individual XML documents to be composed of data from multiple, heterogeneous sources, such as the native XML data store, relational databases, and the file system. Since the connections to external data (made through the X-Node module) are live and bidirectional, Tamino may thus be used to perform heterogeneous joins and updates.The latest Tamino XML Server 4.2 includes enhanced enterprise-scale features and is available on major operating system platforms.

A comprehensive list of native XML database products is maintained by Ronald Bourret[5].

5.1.3 Mixed Approach: Hybrid System

Since most of data become more and more complex and heterogeneous (especially in the multimedia area), it seems interesting to mix both storing approaches described above. First, the major reason is storage constraints (memory space, access time, declustering...), and secondly it enables to use efficient retrieving methods (indexing, query management...). In the case of a hybrid system (this the name commonly used to describe the mixed approach), it is first necessary to look at the physical organization used to store the data. [KM00] introduces a hybrid system called Natix that has a physical record manager that is in charge of the disk memory management and buffering. Of course, it uses a tree data model. Then it handles methods to dynamically maintain the physical structure. [Shi01] describes how XML documents can be indexed and how the text retrieval process can be improved with the use of a mixed storage model: attributes are stored in a DBMS and the element contents and their indices are saved in files. It seems this hybrid approach is a good trade-off between performance and cost in indexing and retrieval.

[3] http://www.dbxml.com/
[4] http://www1.softwareag.com/Corporate/products/tamino/default.asp
[5] http://www.rpbourret.com/xml/ProdsNative.htm

In fact, the major players on the database market (Oracle, DB2, MS SQL-Server) provide specific XML management within their existing systems; and it is quite obvious that they are all working very hard to improve it. The next version of Microsoft SQL Server, for instance, initially code-named Yukon and finally called SQL-Server 2005, adds native XML data storage to the DBMS through a new native XML data type. The introduction of this native XML data type is coupled with the emerging industry standard XQuery language.

5.2 Multimedia Documents Through XML

5.2.1 Current Approaches

The desire to integrate XML with pre-existing data formats has been a longstanding and persistent issue for the XML community. Users often want to leverage the structured, extensible markup conventions of XML without abandoning existing data formats that do not readily adhere to XML 1.0 syntax. Often, users want to leave their existing non-XML formats as is, to be treated as opaque sequences of octets by XML tools and infrastructure. Such an approach allows widely used formats such as JPEG and WAV to peacefully coexist with XML. As XML is increasingly used as a message format (e.g. SOAP), the interest in integrating opaque data with XML has increased to the point where there are at least two competing proposals for doing so (SOAP with Attachments, denoted SwA, and WS-Attachments).

5.2.2 Embedding Multimedia Documents

Traditionally, two techniques for dealing with opaque data in XML have been used; *by value* or *by reference*. The former is achieved by embedding opaque data as element or attribute content. XML supports opaque data as content through the use of either base64 or hexadecimal text encoding. This approach is codified by XML Schema's two binary data types, `xs:base64Binary` and `xs:hexBinary`. The lexical representation of the `xs:hexBinary` is a simple hexadecimal character sequence; the lexical representation of `xs:base64Binary` uses the base64 algorithm as defined by RFC 2045[6]. The underlying value space of both types is identical: an ordered sequence of octets.

An XML instance demonstrating the use of base64 in simple XML document is given in Fig. 5.1.

```
<m:data xmlns:m='http://example.org/people'>
  <photo>/aWKKapGGyQ=</photo>
  <sound>sdcfo2JTiXE=</sound>
  <hash>Faa7vROi2VQ=</hash>
</m:data>
```

Figure 5.1: Use of base64 in XML

[6]"Base64 Content-Transfer-Encoding," RFC 2045, Section 6.8, IETF Draft Standard, November 1996; http://www.ietf.org/rfc/rfc2045.txt

In the next example (see Fig. 5.2), the photo, sound, and hash elements each contain a base64 string (i.e. a sequence of characters) that represents the following octet sequences:

```
fd a5 8a 29 aa 46 1b 24 (photo)
b1 d7 1f a3 62 53 89 71 (sound)
15 a6 bb bd 13 a2 d9 54 (hash)
```

Figure 5.2: Octet sequences represented in base64 strings

The fact that the children of the photo, sound, and hash elements are encoded as base64 is implicit (although discoverable through an XML Schema or RELAX-NG schema), but can be made explicit using xsi:type or an application-specific annotation.

It is well known that base64 encoded data expands by a factor of 1.33 original size, and that hexadecimal encoded data expands by a factor of 2 (assuming an underlying UTF-8 text encoding in both cases; if the underlying text encoding is UTF-16, these numbers double). Also of concern is the overhead in processing costs (both real and perceived) for these formats, especially when decoding back into raw binary. When comparing base64 decoding to a straight-through copy of opaque data, the throughput of at least one popular programming system decreased by a factor of 3 or more.

These performance concerns have discouraged many developers from using embedded data in XML. It is interesting to note, however, that XML Schema defines the value space of the base64Binary and hexBinary data types as the actual octets. This makes it is possible to reduce or eliminate the size and performance costs of base64/hex decoding in many common scenarios (e.g. in-memory DOM trees, SAX pipelines). However, this is not the case when the XML is serialized as UTF-8 or equivalent due of the nature of XML 1.0.

5.2.3 Referencing Multimedia Documents

XML 1.0 explicitly supports referencing external opaque data as external unparsed general entities. Considered a fairly esoteric feature of XML, unparsed entities are not widely used. The primary obstacle to using unparsed entities is their heavy reliance on DTDs, which impedes modularity as well as use of XML namespaces. They are also not available to SOAP, which explicitly prohibits document type declarations in messages.

A more common way to reference external opaque data is to simply use a URI (Uniform Resource Identifier, see Table 5.1) as an element or attribute value (see Fig. 5.3). XML Schema supports this explicitly through the `xs:anyURI` type (see Fig. 5.4).

```
<?xml version="1.0"?>
<data>
  <photo data="http://example.org/me.jpg" />
  <sound data="http://example.org/it.wav" />
  <hash data="http://example.org/my.hsh" />
</data>
```

Figure 5.3: Documents refereed through URI

Table 5.1: Generic URI description.

URI	
elements	description
Scheme	e.g. http, ftp, rtp…
Authority	host name, port number, optionally user information
Path	usually hierarchical, absolute or relative, segments separated by slash (/)
Query string	(optional)
Fragment	position within resource

An XML schema can describe the content of the data attribute (see Fig. 5.4):

```
<xs:attribute name="data" type="xs:anyURI" use="required" />
```

Figure 5.4: XML Schema support

As can RELAX-NG (see Fig. 5.5):

```
<rng:attribute name="data"
     datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
  <rng:data type="anyURI" />
</rng:attribute>
```

Figure 5.5: RELAX-NG support

Referencing opaque data avoids some of the performance and bloat issues associated with base64/hex encoding, but introduces its own problem; because the data is external to the document, it isn't part of the Infoset. We addressed here the drawbacks that the current approaches to include multimedia documents in XML present. Our strategy to overcome these problems is described in Sect. 5.3.3.

5.3 Our Proposal

In Chap. 2, we described the *Resource Categorization Tree* (RCT) as a key structure of the domain metadata management under IMAM control. In order to organize stored data (e.g. ranging from scanned-in text documents to multimedia data and accumulated annotations) in a logical way that supports complex knowledge-intensive tasks users can perform in IMAM-based environments, let us in the following study in depth the physical layer of the digital data storage in such a context.

Multimedia documents include text, sound, pictures, and video information. All those different components are combined and integrated within the same framework: obviously the Extensible Markup Language (XML). Indeed, XML provides a standardized support for including semantic information within documents describing semi-structured data. There are many types of applications manipulating this kind of data (e.g. GIS systems, educational systems, XML databases) and they more and more need to take users' background

and characteristics into account. To be able to deal with users relevant information, one very promising solution is to manage appropriate multidimensional data. After having presented several directions we are very interested in, we propose a data structure and mapping that fits our needs.

5.3.1 From Data to Information

A data model is a mathematical formalism with two parts: a notation for describing data and a set of operators to manipulate that data [Ull88]. Many developers tend to consider XML as a data model. In fact, it is possible to go when step further by saying[7]:

***XML** is more than a data model, it is an **information model**, where:* **data + context = information**

However, this sentence is exaggerating; another layer must indeed be provided in order to capture and support the whole information content. An interesting technology based on XML makes it possible to enhance XML ability to be an information Model: Topic Maps[8]. The purpose of Topic Maps is to support the distributed management of information and knowledge by linking two layers:

- the information layer: the lower layer containing the content (any kind of file format).
- the knowledge layer: the upper layer consisting of topics (subjects the information is about) and associations (relationships between two subjects); it basically contains indexes.

These two layers are linked through occurrences which indicate information that is relevant to a given knowledge topic. Topics in the knowledge layer have relationships (called associations) that are multidirectional and N-ary. Syntaxes for all the elements is well defined and some advanced are proposed. One very interesting feature brought with Topic Maps is the concept of scope. the aim of scope is to represent contextual aspects, and doing so, to make it possible to express multiple viewpoints from one knowledge source. We have to indicate here that scope and its syntax to represent context-dependent data is quite similar to MXML (see Sect. 4.2.3). The structure we are proposing to manage the storage of XML multimedia documents presents some similarities with Topic Maps. Anyway, Topic Maps are definitely web-oriented (e.g. locators for resources are mainly url). The data structure we present in the following section brings the power and the effectiveness of DBMSs and enhances the ability to relate information entities within XML.

5.3.2 Extended Binary Graph

Extended Binary Graph (EBG) data structure is the basis of the work done [AO98a] on Application Oriented DBMS (AODBMS). It combines three strong concepts: DBgraph [Th89], Decomposition Storage Model [VKC86], and the Graph Data Model [Kun90]. The Extended Binary Graph structure is a graph between OIDs and values and between OIDs and OIDs (see Fig. 5.6):

[7] Quote from xml-dev: http://lists.xml.org/archives/xml-dev/
[8] XML Topic Maps 1.0, 2001: http://www.topicmaps.org

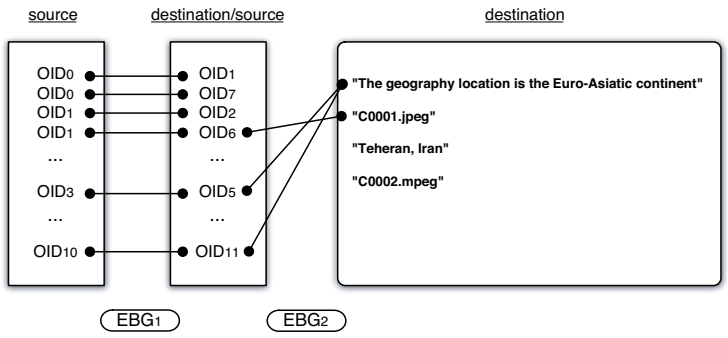

Figure 5.6: XML to EBG mapping

Definition 9 (Extended Binary Graph) *An Extended Binary Graph (EBG) is graph $G(X, A)$ where $X = (S, D)$ is the set of vertices of G, A is the set of edges of G; S is the Source set and D the Destination set. The Edge $(s, d, S.k) \in A$ iff $s \in S$, $d \in D$, and $s_{S.k} = d$ where $S.k$ represents the k^{th} item of the Source set. $s_{S.k}$ corresponds to the value in the Destination set.*

The DBgraph avoids value duplication for complex data type. The Decomposition Storage Model (DSM) introduces clustering and vertical partitions. The DSM combined with horizontal partitions provide efficient cache management. The Graph Data Model facilitates hierarchical relationships between objects. EBG is the core structure of the Phasme prototype [AO98b], which is an innovative information engine kernel. We proposed a data model [GAO02] based on the EBG structure that handles XML and described the mapping process that ensures an effective storage management of XML data through EBG (see Sect. 5.3.3).

5.3.3 XML Support in our Framework

Architecture

The architecture of Phasme prototype is provided on Fig. 5.7. XML is supported under the XML plug-ins service including the document management functions (creation, manipulation, suppression, indexing). The core of the system is the execution reactor, which mediates the requests coming from external applications (XSQL query or direct document manipulation). The document-type support includes the meta-data[9] associated to each document. Phasme prototype is being extended to support DTDs and XML Schema. The latter support will allow mapping directly XML representation information such as structural properties of documents into Extended Binary Graphs. All the vertical XML support depends heavily on the many-sorted algebra that defines XML manipulation functions. For this reason, a plug-ins defines a set of functions based on the Phasme Internal Language. A major goal in this project is to extend Phasme prototype customizability to XML support and to optimize the implementation of such an XML support plug-ins.

[9]The Dublin Core Metadata Initiative: http://dublincore.org

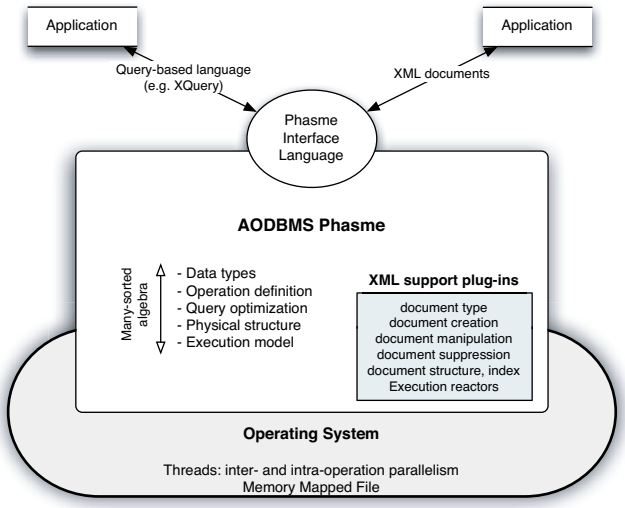

Figure 5.7: XML support inside Phasme engine prototype

XML to EBG Mapping

Starting from annotations within XML related to any kind of *resource*, we need to provide a full scheme that maps the information contained in the tagged tree to EBG. The processes to be applied for XML data to be stored efficiently are described on Fig. 5.8.

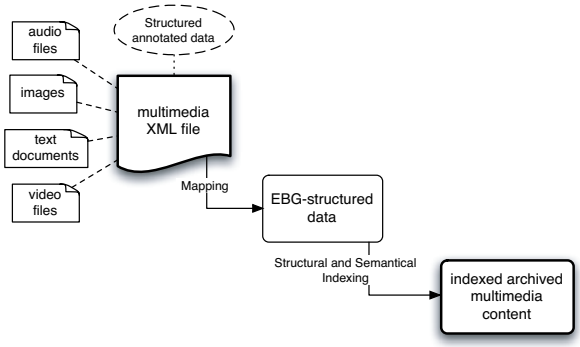

Figure 5.8: Processes to store XML through EBGs

The EBG structure is a combination of three concepts (see Sect. 5.3.2). It ensures a compact data structure to maximize the probability that the hot data set fits in main memory.

The following XML extract (see Fig. 5.9) is taken from the Digital Silk Roads project. The content is related to buildings called caravanserai that are typical along the silk roads. It is interesting to notice that

this has references to multimedia files (image and video); let us also point out the fact that this XML extract contains the same text element twice (lines 12 and 23).

```
<inventory>                                                                                                    1
  <object id=C0001>                                                                                            2
    <description>                                                                                              3
      <section>                                                                                                4
        <section>                                                                                              5
          This study is focusing on the creation of the metadata related to the caravanserai object or         6
          "Khan" object.This object has been used in the history to host the caravan (e.g. men, goods,         7
          animals) on the economic exchange roads between Europe and Asia. The caravanserai is also            8
          used as hostel on the pilgrimage's West-East and North-south routes.                                 9
        </section>                                                                                             10
        <section>                                                                                              11
          The geography location is the Euro-Asiatic continent.                                               12
        </section>                                                                                             13
      </section>                                                                                               14
    </description>                                                                                             15
    <image> C0001.jpeg </image>                                                                                16
  </object>                                                                                                    17
  <object id=C0002>                                                                                            18
    <location> Teheran, Iran </location>                                                                       19
    <video> C0002.mpeg </video>                                                                                20
    <description>                                                                                              21
      <section>                                                                                                22
        The geography location is the Euro-Asiatic continent.                                                  23
      </section>                                                                                               24
    </description>                                                                                             25
  </object>                                                                                                    26
</inventory>                                                                                                   27
```

Figure 5.9: Extract of Digital Silk Roads project XML file

Figure 5.10 gives the associated syntax tree; both representations present equivalent structural properties. We simplified the syntax tree avoiding the CDATA nodes. Furthermore, each node has an OID (Object IDentifier) assignment (notation $\text{OID}_{(value)}$); e.g. OID_0 for the node "inventory". Figure 5.10 moreover illustrates one of the advantages, that EBG is offering for the management of *resources* through XML: repetitions of references to *resources* (such as the one in Fig. 5.9, lines 12 and 23) do not imply a redundant storage of the *resources*.

Figure 5.6 shows how pieces of data in XML are stored as EBGs (e.g. EBG_1, EBG_2). The left column is referred as source, the right column as destination. An EBG is a set of non-oriented arcs between items that are either OIDs or values. EBGs contain either fixed-size item values or variable-size item values. Each value is stored only once so data values are shared between OIDs when values belong to at least two different objects (e.g. OID_5 and OID_{11} share the same value). Here, we do not include the description of the tag set for the values themselves. The semantic tagging issues are tackled under the Linguistic DS cooperation [HAB01].

Persistency is managed in an orthogonal way from the data structure point of view. So in our case, any persistent index or persistent data structures are stored directly inside EBGs. EBGs map the graph contents of documents into the main-memory. Phasme prototype uses the *mmap* file mechanism [Sil00], which enables to have the same data image on disks and in memory. This mapping enables to tune the granularity of the retrieval mechanism.

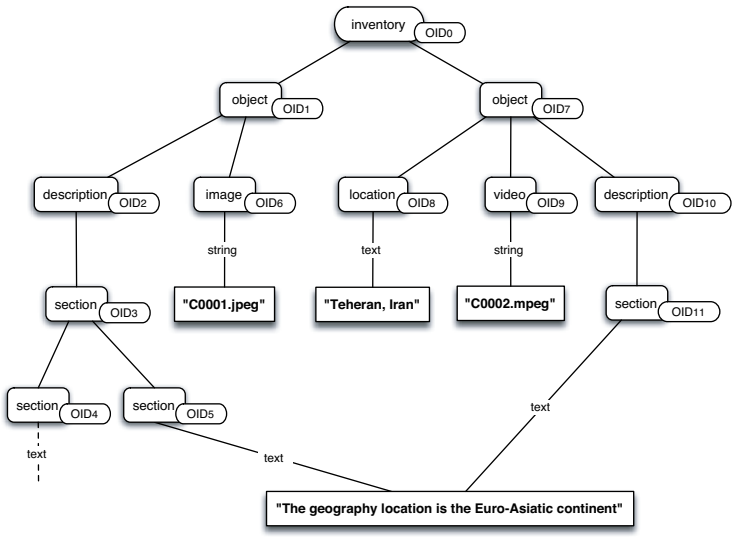

Figure 5.10: XML extract graph-view

Qualitative Assessments

XML document processing can be done either in a pipeline-based way or in a set-based way. This section will compare the alternatives. We assume that intermediate results of document querying cannot fit in memory.

Complex data operations are often expressed by composition of traversal operations to optimize the data access in the EBG structure. Also it has been demonstrated in [Gib85] that pipeline-based and set-based processing strategies are equivalent for graph-based operations. Depth-first search inside the EBG has a complexity of $O(max(card(X), card(A))$ which is similar to breast-first search complexity equal to $O(card(A))$. The processing strategies are chosen according to the index support and to the management of intermediate results. Intermediate results are inside Phasme prototype, either materialized or either transferred in pipeline. The main advantages of the materialization are to exploit share common results and to optimize multiple accesses to the XML document set. Multi-query optimization issue is often seen as a NP-hard problem so heuristics are necessary. This issue will be the object of a specific investigation in the context of XML document set.

The query processor of Phasme prototype includes a dynamic query optimization and execution optimization at run-time as it has been described in [AO01]. Phasme prototype processing is based on the many-sorted algebra approach [Gut89] for query processing and optimization processing directly applied on the EBG structures. It gives a high performance layer that is customizable accordingly to workloads and to users. W3C XML algebra[10] is one issue of improvement for our system. Here, we omit a technical presentation of the EBG query processing and refer the interested reader to [AO98b] for a comprehensive overview.

[10]W3C XML algebra: http://www.w3.org/TR/query-algebra/

Chapter 6

Indexing Strategies

"The most merciful thing in the world, I think, is the inability of the human mind to correlate all its contents."

- H. P. Lovecraft (1890-1937)

Indexes are the main support for information retrieval in databases. Just like indexing in databases, indexing of XML data makes it faster to execute certain queries. The issue, as with databases, is to create indexes that reflect the commonly used search patterns. However, since the type of XML data is not fixed, traditional queries are not always sufficient to locate the desired data. Consequently, regular expression queries are commonly supported by XML query languages. However, it is not clear that simple path access structures developed for Object-oriented databases will be suitable in this domain. This chapter proposes a review of structural indexing strategies and comments the relevance to XML files related to multimedia documents. Then, it describes our indexing scheme dedicated to the storage management introduced in Sect. 5.3.

The chapter is structured as follows:

- In Sect. 6.1, we introduce the usefulness of indexes dedicated to XML and the need for new strategies.
- In Sect. 6.2, we investigate the existing structural indexing methods.
- In Sect. 6.3, we describe our proposal for a multidimensional indexing operator based on EBG.

6.1 Introduction

There are many ways to index XML documents in order to use their content in a database. We have to underline here XPath[1] that offers some strong and well-defined opportunities to describe XML documents without dealing directly with a tree structure.

[1] W3C (1999) XML path language (XPath): www.w3.org/TR/xpath

Indexing semistructured data has been quite deeply investigated. Most of the attention is now focused on XML. With new needs in the area of data management appear new indexing requirements. This is particularly true with XML. One of the main challenges when indexing XML is to deal both with data structure and information content. Indexing strategy must be very relevant for the data used and the environment it is applied to. It also has to combine several indexing capabilities [MWA98]. Most of the database systems use B-tree [BM72] based structures to access information. Many tree structures have been proposed (within several families [FJ02]). Some methods have been designed, very often using the B^+-tree, to index semistructured data and especially XML. We describe here methods that present some interests regarding our goals. It has to be clear that our aim is to propose an indexing method for XML that supports efficiently the multidimensional features presented in Sect. 4.2.3. We will not define a new *Our*-tree method since we consider that we can use and associate good ideas from existing ones.

We investigate the generic issues concerning the achievement of flexible support of XML multidimensional data in a information engine system. We then focus on the indexing part in order to define a new operator to fill the gap concerning XML management which depends on users' characteristics.

6.2 Structural Indexing

6.2.1 BUS-based Indexing

The BUS (Bottom Up Scheme) indexing method [SJJ98] has been developed to improve storage requirements and query processing for structured documents. To reduce the storage, a general identifier (GID) is allocated to each element in order to index only the leaf nodes (with the text content) on a B^+-tree structure. Various improvements have been brought to consolidate this method. We present here only the more recent ones.

XML documents indexing and text retrieval processes can be improved by the use of a mixed storage model: attributes are stored in a DBMS whereas the element contents and their indices are saved in files. XRS-II (XML Retrieval System) architecture [Shi01] is based on this combination: database search and full-text search engine. The text retrieval uses BUS technique. Attributes are stored in databases; a table is allocated for each element type if it has an attribute list. The two engines work in parallel and their results are merged together to produce the final results. This hybrid approach is a good trade-off between performance and cost. Using a database in processing attributes makes it possible to handle a set of comparison and to join operators efficiently. But the structures of the tables where the attributes are stored depend on the XML structure (DTD or Schema). So that the query must be very precise and should respect the attributes definition because the database search engine manages only exact matching.

Presenting the Modified Bus method (MBM), [OSS01] brings some improvements to the BUS method. To minimize the overhead due to indexing, the number of terms to be indexed is reduced according to a user-defined dictionary. It implies to define specific domain knowledge for each application, which is a lot easier and effective than designing a complete ontology. User information knowledge could be integrated in this domain specific indexing model as a multidimensional feature. Another improvement is to avoid the

reorganization of the indexed tree after insertion updates (when the maximum number of children of the BUS method has been reached) by adding the parent ID to each element GID.

6.2.2 Path-based Indexing with Numbering Schemes

Because of the XML tree structure, path-based indexing is frequently used in order to manage XML file content. A common way to improve the path-based indexing method is to assign numbering schemes (most of the time a pair of numbers) to the nodes that enable us to have information about a node's position in the tree and its genealogy. There are various algorithms to evaluate these attribute numbering schemes.

[SYU99] shows how to use path-based indexing to move out the tree structure into a relational table-based structure. To each node is added regional information that consists in some parameters defined by the position in the tree and its hierarchy. Then the retrieval tasks use XQL with path operators. Doing so, database schemas for storing XML documents are independent of the XML files structure (DTD or schema). Unfortunately, this method creates a large memory overload.

XISS [LM01] is a system indexing XML data in order to process regular path expression queries. It uses a numbering scheme for elements and attributes, with a good ability to handle the parent-child relationships between the nodes of the XML tree. This technique is based on tree traversal order and brings improvements with an extended preorder and a range of descendants. This system supports search by element or attribute name and structure; it uses three indexes to make these operations possible. The element and attribute indexes are B^+-tree based. The $\varepsilon\varepsilon$-Join algorithm offers powerful search ability for very long unknown path. The main drawback of XISS's numbering scheme is that it reaches its limit when all the reserved spaces (extended preorder) are used. This implies a global reordering when a new insertion occurs.

6.2.3 Multidimensional Indexing

Indexing multidimensional data is a wide and complicated research topic. As it has been observed in Gaede and Günther's survey [GG98], comparing multidimensional access methods is a very difficult task. Indeed, evaluating multidimensional index implies to define very precisely the kind of data to be managed and the type of usage to be applied to (through queries). We present here the most relevant multidimensional indexing strategies that fit our needs.

One of the most promising ways to index multidimensional XML is to handle multidimensional hierarchies in logarithmic time. The Universal B-tree (UB-tree) [Bay96] is a generalization of B-trees for multidimensional data. It keeps the advantages of B-trees (balanced and guaranteed performances), and in addition to the linear space requirements for storage and the logarithmic time, it preserves the clustering of objects with the Cartesian distance. A strong formal structural approach has been given with the UB-tree, based on a cubic decomposition. The UB-tree access method is powerful to index multidimensional data for two main reasons. The logarithmic time ensures an effective behavior with large amounts of information with deep structure (e.g. geographical databases). And a single UB-tree enables the replacement of a group of

secondary indexes, which are usually used for each dimension in a multidimensional search.

Some other recent work has been done based on same ideas. Another index called a Skip Tree [WLO01] keeps the linked list representation. This method to find a node is based on a logarithmic time approach to find node positions. The skip factor indicates the number of objects that are contained in a sequence which is linked to the leaf nodes. Whereas a B-tree gives an independent key to each node, a Skip Tree allocates to a node a key that is equal to the sum of its children's keys. But no detailed information has been given about how index management is done with Skip Trees (how to set the skip factor, how to deal with externally stored data...).

Another strategy, though using RDBMS, proposes an interesting approach to XML multidimensional indexing. The multidimensional hierarchical clustering (MHC) [BRB02] and its use mixed with UB-trees enables the manipulation of multidimensional index-structures. This is based on three dimensions: path, value and document identifiers. Each problem becomes a two-dimensional problem as a restriction of the three-dimensional universe. According to this work, the most promising indexing solution is to combine two compound B-trees.

But dealing with balanced trees for multidimensional indexing has been criticized: originally presented to overcome the lack of worst-case predictability of the R-tree [Gut84] and R^+-tree [SRF87], the BV-tree [Fre95] is an unbalanced tree. However, it becomes a B-tree in the one-dimensional case: it recursively partitions the data space into sub-spaces. So the properties of the B-tree are preserved (so far as it is topologically possible). Unfortunately, the BV-tree has multiple-page sizes in the index, which becomes a handicap in the case of very large multidimensional XML files.

6.3 Efficient Indexing Support

The Phasme prototype XML Indexing is based on the EBG structure. The indexing mechanisms are those available in Phasme prototype as plug-ins, so it gives a set of indexing mechanisms and strategies available according to the characteristics of the XML contents. Indexing support follows the EBG structure's fast access. It includes the support of multi-dimension indexing such as SR-tree [AOS00] or signature file indexing [ABO96]. We intend to extend the indexing to support UB-tree. [RMF00] and to improve this indexing accelerator according to EBG features. Though indexing processing in the context of EBGs is different from traditional model for XML support, it enables to tune and to customize the usage of indexing strategies according to the XML-based application requirements and workloads. The tuning/customization issue is another key point to be addressed by this project where the knowledge and the environment mining processing are two relevant domains to be investigated.

6.3.1 Multidimensional Operator Definition

In this section, we discuss the major points related to the multi-dimensional support inside the Phasme information engine prototype. Using the Phasme prototype kernel to support a multi-dimension retrieval operator

poses the following challenges for its design and implementation:

- efficiently dealing with huge amounts of XML documents, while keeping low overhead.
- combining vertical customizability and optimization.

On the first point, our approach is to use the Phasme Information Engine prototype as a customizable XML multimedia document storage, indexing and retrieval system. On the second point, vertical customizability provides several open layers (e.g. data definition, operation definition, query language support, data structure, query optimization, execution model). So from the XML processing point of view, such an open system enables higher efficiency due to the close relationship between XML content retrieval algorithms, customized XML data structures, customized multi-dimensional approaches and the data itself. Optimization is customizable in some way due to the use of neural networks to integrate cost models. The multi-dimensional access method is designed as a many-sorted algebra and optimized to reduce the amount of data accessed for XML retrieval queries. In this access method, XML documents are indexed by an extension of the UB-tree [Bay96] for EBG management.

6.3.2 Many-sorted Algebra

The layers of the Phasme system prototype are based on the concept of Many-sorted Algebra. Phasme prototype uses many-sorted algebra at any layer of its architecture as a query language, as a language to define new data structures and related indexes, and also as an executable language to describe query plans (access plans) and related query optimization. Such an algebra enables the Phasme system prototype to be customizable at any layer for end-users requirements related to XML document multi-dimensional support, or optimization rules.

A many-sorted algebra [GTW78] is a collection of sets and functions applied on these sets. It is described by an S-sorted signature Σ where S is a set of sorts (names for the sets) and Σ a family of sets$_{w,s}$ of operator symbols (names for the functions) where $w \in S^*$ and $s \in S$ describe the functionality of operators in $\Sigma_{w,s}$.

For example, let us assert that the set of sorts are $S = s_1$ with s_1 = TREE. The major operations associated to the TREE structure are shown as follows:

- TREE→TREE select, project filter _ #
- TREE×TREE→TREE valuejoin, union, intersection, difference _ _ #

where "_" denotes an operand and "#" the operator. Each application customizes the system according to its requirements. It builds an application-defined many-sorted algebra.

6.3.3 XML Dimension Description

This model is based on the multidimensional extension of XML. A sample of an XML file with cultural and geographic content (from the Digital Silk Roads project) is given in the following example (see Fig. 6.1):

text related to Silk Roads:

Caravanserais are buildings that were especially constructed to shelter men, goods and animals of caravans on trade or pilgrimage's routes between the East and the West and also between the North and South of the Euro-Asiatic continent.

```
1  <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001" xml:lang="en">
2    <Description xsi:type="ContentEntityType">
3      <MultimediaContent xsi:type="LinguisticType">
4        <Linguistic>
5          <Sentence id="b.GEN.1.1" type="culture">
6            <Phrase>
7              <Phrase id="CARAVANSERAIL">Caravanserais</Phrase>
8              are
9              <Phrase>
10               <Phrase> buildings </Phrase>
11               that were especially constructed to
12               <Phrase>
13                 shelter
14                 <Phrase id ="MEN">men,</phrase>
15               </Phrase>
16               ...
17             </Phrase>
18           </Phrase>
19         </Sentence>
20       </Linguistic>
21     </MultimediaContent>
22   </Description>
23 </Mpeg7>
```

Figure 6.1: Data example: extract of Digital Silk Roads project XML file, from the short text given above

Dimensions are declared using the dimension declaration. For example, the declaration shown on Fig. 6.2 provides an extract of MXML which denotes that `Silk_Road_Season` is a dimension name and constrains its possible values to elements of the set summer, winter which will be related to the summer silk road and to the winter silk road.

```
1 <!DIMENSION Silk_Road_Season {summer, winter}>
```

Figure 6.2: Dimension declaration example

6.3.4 Multidimensional EBG Mapping

A dimension is defined by an assignment of a specific value to the OID part of an EBG and to its related descendants. Dimensions can be applied in term of XML context or in term of XML attributes. The EBG graph supports multidimensional nodes by the usage of context edge linking in a context binary graph (called CBG) and EBGs. The extract of XML file (extended as MXML) shown on Fig. 6.3 gives an example of the context dimension `Season` with the values `Summer` and `Winter` on the Silk Roads:

Any Context Binary Graph is a set of arcs where each context is a map from arc-names to any related data. Each arc-name is a non-empty string. In the previous example, `Season=summer` (on line 4) and

```
<silk_road>                                                                    1
  <route name="bilateral trade">                                               2
    <@comment>                                                                 3
      [Season=summer]                                                          4
        <comment>                                                              5
          the road went through Taklamakan desert                              6
        </comment>                                                             7
      [/]                                                                      8
      [Season=winter]                                                          9
        <comment>                                                              10
          the other branch headed south to Khotan and Yarkand                  11
        </comment>                                                             12
      [/]                                                                      13
    </@comment>                                                                14
  </route>                                                                     15
</silk_road>                                                                   16
```

Figure 6.3: Example of contextual information management

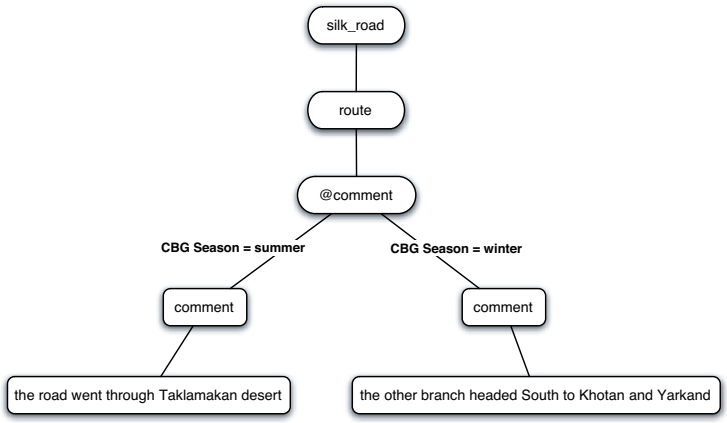

Figure 6.4: Context Binary Graph (CBG) Example

`Season=winter` (on line 9) are arcs of CBG (as shown on Fig. 6.4). On the other hand, tags <`silk_road`>, <`route`> and <`comment`> are represented with EBG.

A naming network of arbitrary topology can be built by grouping all the contexts. Contexts could be events, facts, comments, locations (see example on Fig. 6.5)...

The EBG multi-dimensional type support is based on an extension of the UB-tree in the context of the Extended Binary Graph (EBG) main memory structure. The UB-tree innovation is the concept of Z-Regions to create a disjunctive partitioning of the multidimensional space. A Z-region [a:b] is the space covered by an interval on the Z-Curve and is defined by two Z-Addresses a and b. Let us remind that the physical structure of the EBG is based on memory-mapped file so the mapping of the UB-tree structure is the same on the disk and in memory.

```
<caravanserail>
  <inventory>
    <name = "..."  location = [urban] "yes" [/] "no">
      <specification>
        <ENO> ... </ENO>
        ...
      </specification>
    </name>
  </inventory>
  <control authority code = "...">
    <name> ... </name>
    <address> ... </address>
  </control authority>
</caravanserai>
```

Figure 6.5: Example of attribute dimension, type of caravenserai

6.3.5 EBG Bit Interleaving Issues

The UB-tree relies on the Z-address calculation [OQ97]. For EBG values, the bit interleaving algorithm uses the binary representation of the EBG. The pseudo-code in Fig. 6.6 provides the EBG-based bit-interleaving algorithm (bit interleaving processing regarding the EBG structure is shown in sect. 6.3.7).

```
UBKEY(ebg, steplength, d)
1  bs ← TRANSFORMATIONEBG(ebg)
2  bp ← 0
3  for s ← 1 to steplength −1
4       do for i ← 0 to d −1
5            do addr[bp] ← bs[i][s]
6               bp ← bp +1
7          return addr
```

Figure 6.6: EBG-based UB Key Function

The inputs of the UBKEY function are an *ebg* of EBG type (memory mapped file), *steplength* which is the maximum size of the bit string allowed, and *d* the ebg dimension (values domain).

Among the he variables used, *bs* is a bitstring representation of the attribute values represented as a table of size [dimno]; it gets the transformation of the *ebg* key values on line 1 in Fig. 6.6. *bp* indicates the bit position in the Z-value; it is initialized as the first bit of the bit-interleaving on line 2. Note that the bp^{th} bit of the Z-value is set to the s^{th} bit of the i^{th} bit string.

UBKEY returns *addr* of Z-value type.

6.3.6 EBG Insertion inside the UB-Tree

The low level of the algorithms of the UB-tree support for the EBG data structure is based on the adaptation of the main-memory representation of B-trees. Processing an EBG insertion (see Fig. 6.7), deletion or update includes the Z-value calculation to determine the storage location of EBG inside the B-tree.

UBTREEEBGINSERTION(ebg)
1 $a \leftarrow$ UBKEY(ebg)
2 **return** BTREEINDEXINSERT(a, ebg)

Figure 6.7: EBG Insertion

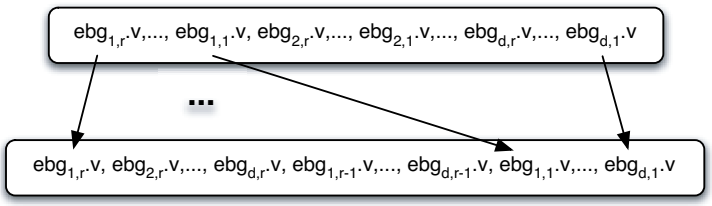

Figure 6.8: Address Calculation via the Bit Interleaving

6.3.7 Address Calculation via the Bit Interleaving

Any EBG can be accessed as a multi-dimensional structure. The cardinality (e.g. number of values (ebg.card)) of its d-dimension is equal to 2r (EBG's property [AO98b]). So we can consider any EBG_i as a sequence of bits:

$$\{ebg_{i,r}.value; \ldots; ebg_{i,l}.value\}$$

where i is the EBG number. Bit-interleaving creates an r-dimensional sequence out of d-dimensional EBGs by rearranging the bits of the EBGs used for the index keys in the following way (notation: we use $ebg_{i,r}.v$ instead of $ebg_{i,r}.value$):

$$\begin{aligned} &interleave^{d,r}(ebg_{1,r}.v, \ldots, ebg_{1,1}.v, ebg_{2,r}.v, \ldots, ebg_{2,1}.v, \ldots, ebg_{d,r}.v, \ldots ebg_{d,1}.v) \\ &= (ebg_{1,r}.v, \ldots, ebg_{d,r}.v, ebg_{1,r-1}.v, \ldots, ebg_{d,r-1}.v, ebg_{1,1}.v, \ldots, ebg_{d,1}.v) \end{aligned}$$

where $ebg_{i,j}.v$ is a pointer on the value of the i^{th} EBG of the j^{th} dimension (see Fig. 6.8). Inverse function of the EBG access can be provided directly in applying $interleave^{d,r}$ on inverse ebg structure (ebg.reverse()).

The algorithm complexity of bit-interleaving is $O(d * r)$, where is d is the number of EBGs which participate in the index and r is the length of each EBG value in bits. This approach enables us to switch easily from EBG graph representation to EBG vector (address) representation.

6.4 Conclusions

In this part, we first reviewed XML-based technologies that present some interest for IMAM possible implementations. There is no doubt about the fact that XML is very convenient to handle metadata and to contain

references to multimedia documents. However, XML also has some drawbacks; it is quite heavy to process and its Schema and numerous extensions make it quite complicated. Storing XML is another issue that we have investigated. We proposed a hybrid approach based on a data model, which makes it easy to store XML after a simple mapping. We then designed an indexing strategy for multidimensional data contained in XML (and in particular MXML) and provided a simple, but yet interesting solution.

Unfortunately, this design has not been implemented yet, and this for two reasons:

- First, the DBMS prototype, on which the EBG structure is implemented, is currently not available. It is in fact being updated in order to solve some important issues related to Unicode management on the target machine (Windows) and platform evolutions (Solaris).
- Secondly, while the design of the indexing strategy has been proposed in 2003, the definition of IMAM was still under progress. Therefore, IMAM consistency policy (i.e. the update management presented in Sect. 2.3.3), which has been finalized in 2004, made the use of multidimensional XML for the versioning management of resource descriptions and profiles inaccurate.

However, the design presented in this part remains very attractive for handling multidimensional data through the EBG structure. It is in fact planned to add the mapping proposed in Sect. 5 and the indexing strategy described in Sect. 6 within the Phasme prototype as soon as its kernel will have been updated and its encoding problems solved.

Part III

Adaptive Services

Chapter 7

Adaptive Services Overview

> "We are all special cases."
> - Albert Camus (1913-1960)

Storage and indexing of data are not the only services that an innovative information management system should offer to its users. Indeed, data must be accessible by many users at the same time. Moreover, the access has to be operated smoothly with simple and few actions from users, in order to ensure that the system is attractive. Personalization of data access, based on techniques such as information filtering [Loe92], is a major feature that can improve the management of resources within communities. Furthermore, services processed without human action, such as scheduled data placement [KL03], can decrease the network consumption generated by the distribution of data and optimize the computing costs as they reduce latencies when transferring data.

Distributed adaptive and automated services require to exploit all the environmental knowledge that is available about the elements involved in the processes [PJF04]. An important category of this knowledge is related to devices' states; indeed, knowing if a device is on, in sleep mode, off, if its battery still has an autonomy of five minutes or four days, or if it has a wired or wireless connection, etc. helps adapting services that can be delivered to this device. For each device, we consider a state control that is part of the device's *profile*. Then, of course, we use the information contained in communities' and users' *profiles*. The information that can be gathered in collaborative environments (i.e. people sharing interests and resources) shall increase the ability to create new kinds of services. Personalized services rely on user-related contexts such as localization [CLM03], birth date, languages abilities, professional activities, hobbies, communities' involvement, etc. that give clues to the system about users' expectations and abilities. All this information is quite easy to extract and to manipulate through IMAM.

In the remainder of this part, we tackle the goal of this dissertation, which is to improve the access to *resources* for users involved in communities. We present the two main adaptive services based on our model

that aim at giving users more satisfaction about the *resources* management and their end-user vision. This chapter motivates the need for adaptive services within collaborative environments and reviews the existing strategies that aim at providing to users innovative and more relevant services.

The chapter is structured as follows:

- In Sect. 7.1, we introduce the motivations for adaptive services.
- In Sect. 7.2, we show the interest of such services with some case studies.
- In Sect. 7.3, we survey the literature that motivated the services definitions, algorithms, and optimizations presented in Chapters 8-11.

In subsequent chapters, we describe automated and personalized services that take advantage of the collaborative knowledge management provided by IMAM. We start with query optimization (Chap. 8), and afterwards propose an authoritarian data placement (Chap. 9). We finally introduce IMAM adaptive merged services (Chap. 11).

7.1 Preamble on Personalization of Services

Personalization has been a very popular and fashionable topic some years ago; a consortium was even created about it[1]. And then it quickly became outmoded (see a report[2] by Jupiter Research entitled "Beyond the Personalization Myth") for obvious reasons: researchers and mainly developers were expecting that personalization of graphic interfaces and web pages would make their product more attractive whereas they did not work on content management and navigation design. However, personalization is again becoming attractive; this time, research is done more discreetly and also more seriously as it involves the benefit and income of most major players of the software market (Microsoft, Google, IBM...).

Personalization can be described as the ability to deliver content and provide services tailored to individuals. The ultimate aim of personalization is in fact user satisfaction (when not performed for commercial reasons...). It is motivated by the recognition that a user has needs, and meeting them successfully is likely to lead to a satisfying relationship and re-use of the services offered. Beyond the common goal, however, there is a great diversity in how personalization can be achieved; the two most common and recognized approaches are explicit personalization (user explicitly gives information, usually by filling in forms), and implicit personalization (requires more sophisticated techniques for the analyze and adaptation to users activity history).

Good personalization requires the system to know a lot about the user [Sei98]. Information about the user can be obtained from a history of previous sessions, or through interaction in real time. *Needs* may be those stated by the customer as well as those perceived by the business. Once the user's needs are established, rules and techniques, such as collaborative filtering, are used to decide which content might be appropriate.

[1] http://www.personalization.org
[2] http://www.jupiterdirect.com/bin/report.pl/94553/1015

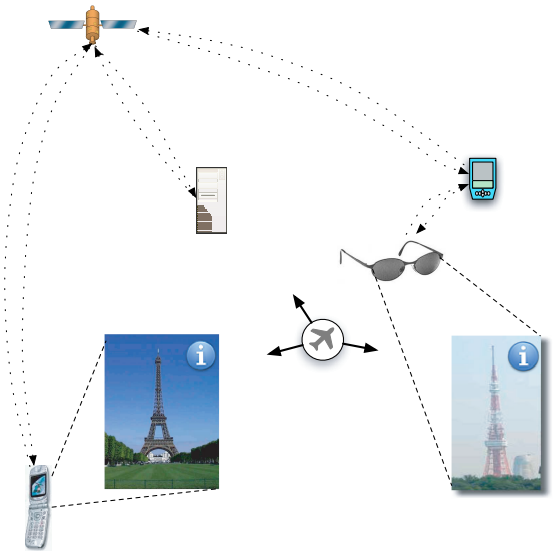

Figure 7.1: Collaborative data delivery with contextual information

Services to be proposed to users cover usual database functions, personalized automated processes, and management of transactions in order to ensure the capability of systems to work in heterogeneous distributed mobile environments. All these aspects have to be taken into account for the design of services that fit users needs. In fact, more and more people are showing strong interests in peer-to-peer [Abi03] as a foundation for creating advanced distributed applications; moreover, innovative sharing strategies are implemented and used in peer-to-peer [LOP02, MEA02, AEM03] and mobile systems [KM03, GTB02]. But they are generally lacking in a unique generic basis for knowledge management that would allow us taking fully advantage of these powerful distributive environments.

Finally, let us mention that the term personalization is too restrictive for the work presented here as it is usually applied to users only; the services we want to provide shall be dedicated to communities and then to users. Devices, moreover, can be the target of specific services. Hence, we prefer to discuss about adaptability of services.

7.2 Case Studies

In this section, we give three illustrations of what adapted services based on IMAM can perform:

1. **DSR communities**; let us consider a class studying caravans in Iraq with a focus on the 14^{th} century and looking for *resources* on pilgrims exchanging specific products. It would be interesting and useful for the students to get on their laptop maps, pictures, videos that are related to their topic. This could be done by creating the community some time before the class starts this lesson. The automated data

distribution would be then restricted by data filtering and *resources* would be re-formated according to the characteristics of the devices that students are using. These operations could be scheduled in order to be performed sometime before a class for student to prepare the lesson or just before a precise time for an examination. IMAM services for the management of *resources* seems quite promising for educational support in pervasive digital environments.

2. **Fig. 7.1**; if we consider cultural and touristic activities, mobile devices should be able to offer attractive adaptive services to people traveling. For instance, two persons registered as members of a group dedicated to the famous capitals of the world, might appreciate to share and access data that has been produced by one another. Let us consider that the first user has taken a picture of Eiffel Tower in Paris with his mobile phone and has added some annotation to his picture during his tour in the building. Then, the second user, when watching Tokyo Tower with his glasses (being in fact wireless head-mounted displays), can choose to access data about the Eiffel Tower as the two towers look quite similar. He might get nice pictures from the first user with relevant comparative information (e.g. tower's height, weight, opening date). This example shows that augmented reality management could take advantage of IMAM.

3. **Papillon project**[3]; in this project, active members are inserting and validating entries for monolingual lexical bases that are related through a pivot architecture. Papillon already proposes some adaptability to users such as interface plasticity [Th02a]; i.e. GUI and contents are adapted to fit users devices screens. IMAM makes it possible to offer other kinds of adaptive services to Papillon users. The distribution of entries to be validated could be processed automatically by adaptive services: the selection of entries would be done by term matching analysis with thesaurus related to users fields of interests and language capabilities. This data distribution could in fact improve the management of this kind of collaborative work by avoiding users to perform the time consuming and repetitive selection tasks, and so adaptive services based on IMAM would reinforce community members involvement.

7.3 Related Work

Service personalization is well tackled in [RJR02] where the authors focus on content network with a framework based on a service manager and an authorization server that automate the tailoring of the services. The adaptation of data access relies on a rule-based service execution and content responses are encapsulated in a call-out protocol such as ICAP (Internet Content Adaptation Protocol). Although they provide an interesting approach, Ravindran et al. do not give a clear idea of application for their framework and do not address the evolution of services. The importance of dynamic approach for the personalization of access to *resources* has been pointed out in [UKT03]. Moreover, it gives much importance to the structure without focusing on the data itself. Therefore, the design of personalized access to data is very clear and robust. Dissemination service proposed in ONYX [DRF04] integrates XML rich functionality of transformation for query result

[3] http://www.papillon-dictionary.org

customization. We are addressing a more challenging problem as support for *resources* through XML and similarity evaluation leads to increased complexity. Indeed, whereas ONYX architecture benefits from its solid database approach, it is also restricted since it does not consider the available semantic about *resources* and environments.

Data placement has been clearly recognized [VRT01] as key feature for the consolidation and distribution in systems that have to support the management of large amount of heterogeneous data. Veitch et al. moreover consider that automated placement of data in this context is the best solution for systems to be able to adapt to new applications as they arise and evolve. Unfortunately, their vision was missing the growing role that small devices were going to play in the design and management of services. Data placement has also been much investigated in the grid community; however, existing strategies focus either on restrictive cases such as intermittently available environments [HV02] or on specific data types [KL04]. They moreover too often consider too restrictive distributive scenari.

We agree that distributed knowledge management has to assume two principles [BBC02] related to the classification: autonomy of classification for each knowledge management unit (such as community), and coordination of these units in order to ensure a global consistency. Having a decentralized peer-to-peer knowledge management, the SWAP platform [PSS04] is designed to enable knowledge sharing in a distributed environment. Pinto et al. provide interesting updates and changes support between peers. However, vocabularies in SWAP have to be harmonized; which implies to have some loss of knowledge consistency. But even if we share the approach of core knowledge structure that is expendable, the vocabulary, in our case, is common and fully shared by the community, so the knowledge evaluation and comparison can be more effective. Moreover, SWAP provides some kind of personalization (user interface mainly) but does not go as far as IMAM aims at. From our point of view, SWAP definitely lacks environmental knowledge management that is required to perform advanced services; on the other hand, DBGlobe [PAP03] is a service-oriented peer-to-peer system where mobile peers carrying data provide the base for services to be performed. Its knowledge structure is quite similar to IMAM as it is using metadata about devices, users and data within *profiles*; moreover, communities are also focused on one semantic concept. DBGlobe relies on AXML [ABC04] in order to perform enbedded calls to Web services within XML. Thus, it provides a very good support for performing services but does not focus on users and environments knowledge in order to offer optimized authoritarian adaptive services. Described as a P2P DBMS, AmbientDB [FB04] relies on the concept of Ambient Intelligence, which is very similar to our vision of adaptive services with automatic cooperation between devices and personalization. However, although AmbientDB is using the effective Chord Distributed Hash Table to index the metadata related to *resources*, it lacks the environmental knowlege managenent provided by IMAM that is necessary to achieve adaptive collaborative distribution and personalized query optimization.

Chapter 8

Query Optimization

> "Nothing is lost, nothing is created, everything is transformed."
> - Antoine Laurent de Lavoisier (1743-1794)

Query optimization is an important issue for distributed systems. It has been investigated for decades by the database community, and still new strategies and improvements appear regularly. The core of query processing in traditional databases relies on relations; it makes it (quite) simple to extract information that shows correlations with conditions expressed in queries. The strength of relational databases is to enable queries providing results that are certain with ensured performances; but it also makes them weak when dealing with knowledge that cannot be expressed with pure relations.

Through the collaborative management of heterogeneous resources, we are confronted with complicated issues related to queries which do not fit the relational approach. Annotations, as they are used within P2P data exchange systems for instance, make it possible to perform simple queries based on limited attributes. These queries, which are using term matching techniques, are relevant for restrictive sets of resources. But they are not sufficient for optimizing the queries results.

As we want to provide automated processes that would help community members to access more relevant data without performing more complex queries, we need a new scheme to refine queries and to adapt the results to the querying environment. IMAM enables us to identify clear and reliable knowledge representations with the *resource description* and profiles. The information we have about users, communities, and devices must be exploited to optimize user satisfaction. This chapter addresses this problem by proposing a new approach that aims at improving the selection of resources within typical query results (i.e. sets of resources). This service has been partially introduced to the XML management community at EDBT'04 DataX workshop [GAG04b]. The chapter is structured as follows:

- In Sect. 8.1, we present our vision for an innovative query management relying on IMAM and introduce the *viewpoint* service dedicated the enhanced querying personalization.

- In Sect. 8.2, we give an illustrative scenario that clarifies the components of the *viewpoint*.
- In Sect. 8.3, we point out the benefits of the service and show some examples.

8.1 Offering Multi-viewpoint

The word *viewpoint* has various interpretations. It can be *a perspective of interest from which an expert examines a knowledge base* [MRU90] or *an interface allowing the indexation and the interpretation of a view composed of knowledge elements* [RD02]. Our approach is slightly different. Indeed, we focus on the interaction between the data and the querying environment: we use all available knowledge to extract and to provide the most relevant set of data for the user. The *viewpoint* becomes the characterization of an association *resource-environment*; indeed, the *viewpoint* has to be a complete knowledge basis for an accurate information access and representation. The ability to provide an optimized *viewpoint* depends on the available amount and quality of information about users and resources. Then, it becomes easily obvious that the *viewpoint* is the result of comparative tests between environment and resource characteristics:

Operator 8 (Viewpoint) *Viewpoint is expressed as a function returning an ordered set of Resource Descriptions. We use ν to denote the viewpoint:*

$$\begin{array}{rrcl} \nu = p \circ t \circ g : & \Lambda^p \times \Pi & \longrightarrow & \Lambda^q \\ & \left(<D_i>_{i=1,\ldots,p},\, \pi_e\right) & \overset{\Psi\circ\Theta\circ\Gamma}{\longmapsto} & <D'_k>_{k=1,\ldots,q} \end{array}$$

$$\begin{array}{rrcl} with \quad g : & \Lambda^p \times \Pi & \rightarrow & \Lambda^q \times \Pi \\ & \left(<D_i>_{i=1,\ldots,p},\, \pi_e\right) & \overset{\Gamma}{\mapsto} & \left(<D_j>_{j=1,\ldots,q}, \pi_e\right) \\ t : & \Lambda^q \times \Pi & \rightarrow & \Lambda^q \times \Pi \\ & \left(<D_j>_{j=1,\ldots,q},\, \pi_e\right) & \overset{\Theta}{\mapsto} & \left(<D'_j>_{j=1,\ldots,q}, \pi_e\right) \\ p : & \Lambda^q \times \Pi & \rightarrow & \Lambda^q \\ & \left(<D'_j>_{j=1,\ldots,q},\, \pi_e\right) & \overset{\Psi}{\mapsto} & <D'_k>_{k=1,\ldots,q} \end{array}$$

where p is the number of considered Resource Descriptions and q the number of returned Resource Descriptions ($q \leq p$), π_e is the profile of the environment e (with $\pi_e = \pi_u \cup \pi_d$, u denotes a user, and d a device), and Γ, Θ, and Ψ are three sets of selective rules:

- *Γ contains acceptation rules denoted γ. If a descriptor value of the resource description D_λ does not respect a rule $\gamma_i \in \Gamma$, then the set returned by g does not contain D_λ.*
- *Θ contains transformation rules denoted θ. If a descriptor of the resource description D_λ is involved in any rule $\theta_i \in \Theta$ and if the corresponding value σ does not satisfy this rule, a new resource λ' (with*

the related $D_{\lambda'}$) will be created as the result of a modification applied to the resource λ by t according to instructions contained in θ_i.

- *Ψ contains re-ordering rules denoted ψ. If a descriptor value of the resource description D_λ does not respect a rule $\psi_i \in \Psi$, then the position of D_λ in the set of resource descriptions returned by p. This re-ordering depends on the result of the rules and then on the existing order in the original set of resource descriptions: any D_λ that does not respect rules in Ψ will be pushed behind the resources descriptions that respect all or more rules than D_λ (i.e. p rearranges in order the resource description sets by classifying decreasingly the elements respecting the larger amount of rules).*

Each set of rules is deeply dependent on the domain the viewpoint is applied to. It is obvious that rules must be defined according to communities' and users' interests. Moreover, the rules rely on the available applications (especially for transformation rules). Each rule used by the viewpoint is a test on a pair of descriptor values; one from the *resource description* and the other one from the profile. Thus the syntax for each rule is very simple and relies on the fact that the rule is true or false for each test. We provide a summary of notations used for the viewpoint in Table 8.1.

Table 8.1: Notations used for the selective functions of the *viewpoint*

type	function	set of rules	rule
acceptation	g	Γ	γ_i
transformation	t	Θ	θ_j
re-ordering	p	Ψ	ψ_k

Note that it is imperative to keep the order of the compound functions when applying ν; indeed, another order would generate inconsistencies in the management of resources as it might for instance create new resources and reject them afterwards.

8.2 Illustrative Scenario

We consider the scenario of an researcher being a member of DSR, who looks for resources that contain maps of the historical silk roads. A typical query in that case would be a set of terms such as $<$ *maps silk roads* $>$; the query is directly sent to the server with the IDs of the user and of the device, so the server can select their profiles from its own memory. This kind of query on a repository which is dedicated to the silk roads would of course return a very large set of resources. Let us just consider a small set of resource descriptions $S = < D_{r_1}, D_{r_2}, D_{r_3} >$ (in order to make the example simple and short) where:

- r_1 is a high resolution map covering the whole Asia and showing the main historical silk roads with comments written in English.
- r_2 is a movie file containing a short documentary on the silk roads in Iraq.

- r_3 is a low resolution satellite picture of Iraq where silk roads have been drawn with comments written in Arabic.

As it has been explain above, the viewpoint is a compound of three functions so the refined selection process is done in three steps:

1. Selection. The first set of rules applied by function g might remove resources descriptions which do not respect at least one rule. The rules are considering descriptors in the environmental profile that are involved in at least one rule. In our example, one rule checks the size of the resources and the tuple (available memory space, bandwidth) from the profile. The movie, with a size of 30MB, exceeds both limits from the tuple as the user is processing his query from his mobile phone. So g returns the set: $< D_{r_1}, D_{r_3} >$ (note that it means that both other resources passed the tests).

2. Modification. The second set of rules applied by function t can modify some resources (and then creates a new resource description) if a resource of a certain type exceeds a threshold defined in the rule or in the profile for this type of resource. Then the rule can call another application which is able to modify the resource so it would not exceed the threshold anymore. This is the case with r_1 which resolution is very high and exceeds the resolution of the mobile phone screen. Then t calls an application that reduces the resolution of r_1 until it reaches the screen resolution, and a new resource r'_1 is created and replaces r_1. t returns $< D'_{r_1}, D_{r_3} >$.

3. Reordering. Finally, the last set of rules applied by p can reorder the set if it considers that a resource with a higher priority (or value) is behind a resource with a lower priority. For this kind of rule, a very appropriate evaluation relies on the languages the user can understand. Here, as the researcher is an Iraqi, and so is fluent in Arabic whereas he has a poor English level (reminder: the values for the language descriptor are ordered). Then, p will return $< D_{r_3}, D'_{r_1} >$, which will be the result of the *viewpoint*.

8.3 Adaptive Query Management

The sets of rules, which the viewpoint is using (Γ, Θ, and Ψ), are contained in two different categories: we can consider Θ's rules results as commands for *resources* themselves to be adapted, when the other sets of rules adapt the already returned sets of resources; for instance, an image, that has a bigger resolution than the one of the user's screen, would be reduced to the screen resolution. This strategy is very useful for distributed systems and heterogeneous environments since it reduces the bandwidth consumption and fits devices characteristics (especially mobile devices).

***Example* 6** *We give examples of rules that can be applied for multi-viewpoint support: Tables 8.2, 8.3, and 8.4 respectively propose descriptions of possible rules for the sets* Γ, Θ, *and* Ψ.

Table 8.2: *Viewpoint* **acceptation** rules example

Γ	
name	description
γ_1	Size constraint: If the size of the *resource* is superior than a threshold defined in the rule, then the *resource* will be rejected.
γ_2	Video encoding: If a video file requires a codec that is not available on the device involved in π_e, then the video will be rejected.

Table 8.3: *Viewpoint* **transformation** rules example

Θ	
name	description
θ_1	Monochrome monitor: If a *resource* has chromatic content whereas π_d indicates the screen is monochrome, then θ would return a new *resource* in black and white.
θ_2	Resource resolution: If the resolution of a *resource* exceeds the screen resolution sr appearing in π_d, then a new *resource* will be created with a resolution being equal to sr.

As said in the previous section, it is imperative to define the transformation rules according to the server software environment; in the DSR case, we use some applications providing image management, text summarization... Then it becomes trivial to manage the information, and to apply the modifications depending on the descriptor values.

A major benefit of the RCT is to allow us giving an appropriate *viewpoint* to each user for a same set of *resources* (taking the user's characteristics and environment into account). In fact, our *viewpoint* can be seen as a query optimizer, since it clears and modifies an initial set of *resources*. It has initially been defined in [GAG04b] with only two sets of rules (re-ordering was integrated within the two other sets). This strategy was lighter and seemed more optimized. However, after simulating some simple tests, we realized that the two steps approach might generate incoherent resources management. Therefore, we added to the *viewpoint* a third function using re-ordering rules only.

Ideally, our query optimization strategy, as a distributed and decentralized operation, would require a large CPU contribution from the servers and devices as they have to apply the *viewpoint* on all the *Resource Descriptions* they are receiving from other devices. This would be especially true for re-ordering rules as they compare each rersource description to all the others in the considered set and require to add some temporary factors (e.g. the number of rules, which the resource description follows). Moreover, since we are dealing with high resolution multimedia *resources* and as we are reasonably convinced that portable devices processing capacity will soon increase much, we claim that the benefits of the *resources*' selection worth the overload on devices CPU and primary memory. The needs for such a distributed query management to be designed will be investigated in Chap. 11.

Table 8.4: *Viewpoint* **re-ordering** rules example

Ψ	
name	description
ψ_1	Language Preferences: If the *resource* contains information in a language that is not mentioned in π_u, then the *resource* will be pushed behind resources being more relevant from a linguistic point of view.
ψ_2	Focus point: If the location of a user, which is provided in π_u and can be identified with GIS-based annotations, appears on a map that is considered by p, then ψ would push ahead (in the returned set) maps that are focused on the area the user is located in.

However, the architecture we are proposing on Fig.1.2 relies on a centralized strategy for the processing of queries; therefore, the ability to perform a reliable query optimization (regarding time consumption) depends of course on the complexity of the rules that are used by the *viewpoint* but the main aspect here is the capacity of the server to process the operations. Thus, the specifications of a community's server dedicated to the usage of IMAM and its services must be able to cover the operative costs that the viewpoint requires.

Chapter 9

Data Placement

> “Real knowledge is to know the extent of one’s ignorance.”
> - Confucius (551-479 BCE)

As it has been presented in Chap. 7, automated data placement is an interesting strategy for the distribution of data that can offer to individuals a better access to *resources*. We in fact consider that data placement aims at improving data access which is far too often too difficult and disappointing for end-users.

The definition and design of authoritarian placement of data for communities is one of the main contribution of this dissertation and represents an innovative approach to handle *resources* distribution within communities by automatically delivering interesting and appropriate content to people. From our point of view, automated placement processes must fill some gaps in *resources* for users; it considers and distributes available information users’ are hypothetically or potentially missing.

This chapter proposes a full description of our automated placement strategy based on IMAM for communities sharing access to environments and *resources*. After introducing the interest for such kind of service, we show how we evaluate *resources* relevance to environments (devices used by users being involved in communities). We finally give a precise and consistent overview of the placement processing and address the global management of the data placement which automatically dispatches on appropriate devices *resources* that seem to be very relevant to a user or a community. The work described in this chapter has been partially presented to the DELOS community [GAA04]. The chapter is structured as follows:

- In Sect. 9.1, we introduce the placement service.
- In Sect. 9.2, we present our relevance evaluation strategy.
- In Sect. 9.3, we describe the operator with its algorithm.
- In Sect. 9.4, we address some remaining issues and some improvements that can be applied to placement operator.

9.1 Service Description

Most of the systems dealing with automated distribution of data are focusing on scheduled [KL03] or load balancing [GSB04] data placement. Some recent works address other types of adaptation: Stork [KL04] enhances the interaction between peers by evaluating environmental states (devices, users), and placement in [WL04] is performed depending on storage capacity for streaming processes. But there is no tentative to apply a selective authoritarian placement of all the *resources* and knowledge, which a community is manipulating.

The scope of our data placement is to automatically copy *resources*, which seem to be very relevant to a user or a community, on the appropriate devices. Our placement is based on a specific hierarchical network architecture defined within IMAM (mix of client/server and P2P) where servers are central repositories getting and providing data to communities in which devices can then act as peers.

The architecture of the services based on IMAM deals with communities of users. The data is basically stored on a main central server, with back ups on local servers, and is then accessed from any kind of device. In a project such as DSR, since most of the countries that are involved in the project have low computing and bandwidth capacities, it is important to optimize the distribution of the *resources*; this is the aim of the data placement. Therefore, using automated processes, we can dispatch efficiently and accurately the *resources* for communities and users.

We have here to remind that IMAM framework has a 3-layers architecture: *servers*, *access-point*, and *devices*. Indeed, each community has a device called *access-point*, i.e. a machine that has enough computing power, storage capacity, and connect-ability to be a kind of sub-server for the other devices of the community. This architecture requires the information about a layer to be kept on the upper layer. In fact, the *server* must contain a record of all *access-points*'s profiles, and an *access-point* has details about all the *devices* and users involved in the community that the *access-point* is representing.

Example 7 *It is easy to notice the interest of data placement when considering a DSR member who is an architect, and therefore is part of the architect community within DSR. This person would get an easier and faster access to the data related to architecture from the access-point (e.g. many resources containing the buildings label). Then, according to his own profile (location, other topics of interest...), he would receive on his device some resources that are relevant to him.*

9.2 Resource Relevance Evaluation

Each time a *resource* is added to the system on the main server (in the case of DSR, NII server), its *resource description* is used for analyzing the possible correlations with the communities and users interests. The strategy we are using to evaluate the significance of a *resource* placement on a device is quite similar to the one used for operator SIM (which evaluates the similarity between two *resources*, see Sect. 2.3.2). But in the case of the placement, the *descriptors* are replaced by the *descriptor values*. We extract the ratio of common *descriptors values* by using the function ρ_D (Note that for readability's sake, the function description only

considers the *descriptor values* as atoms; the implementation has to consider each term within the *descriptor values* as a comparative entity):

$$\rho_D(A,B) = \frac{\mid \textsc{Tinter}(A,B) \mid}{\mid \textsc{Tunion}(A,B) \mid} \in [0,1]$$

with:

- $\textsc{Tinter}(A,B) = \{<\sigma_{inter}> \mid \sigma_{inter} = \sigma_{i,j} = \sigma_{k,l},\ \ (\sigma_{i,j} \in A) \wedge (\sigma_{k,l} \in B)\}$
- $\textsc{Tunion}(A,B) = \{<\sigma_{union}> \mid (\sigma_{union} \in A) \veebar (\sigma_{union} \in B)\}$

where A and B contain ordered families of labels, which are lists of descriptors with related values (there can be only one label, in the case of a profile for instance). $\veebar$ denotes operator *exclusive-or*.

However, this generic approach presents some shortcomings:

- Computational scalability. We can claim that the ratio of common descriptor values would be quite low in most of the cases; this implies unnecessary heavy computation. But considering that these operations are performed on the server side (not on the fly, it can be managed through FIFO queuing), reasonable recall can be expected.
- Inadequate similarities. Irrelevant correlations between some descriptor values might occur; a resource and a community, for instance, having the same identifier value. This does not imply that the community would be interested in this resource. We clearly need to apply a selection to skip non relevant descriptors.
- Similarity scale. With our approach, similarity is most of the time based on a few occurrences among many descriptor values and so it makes it difficult to compare and evaluate relevance.

This last point is the main issue we are confronted with. We therefore have to bring some improvements to our strategy in order to make automated data placement more relevant.

9.3 Placement Processing

In this section, we precisely describe IMAM's *resource* placement operator called Dispatch (denoted DISP), which is applied to the *Resource Description* of any new *resource r* added (or updated) on the servers. This operator relies on memory spaces that are allocated on each device for the server to place the data and on exact free memory that remains available on this space in real time. Note that it is important to keep records on the servers of the exact set of resources dispatched to a device and a user for:

- possible recommendation functionalities,
- context of feedback,
- or accounting purposes.

The DISP operator first applies the function ρ_D to communities. Depending on two threshold values s_{c_1} and s_{c_2} ($s_{c_1} > s_{c_2}$), we need to decide if the resource has to be placed on all the devices used in the community (Case 1 on Fig. 9.3) or only on the community's *access point* (Case 2 on Fig. 9.3); the operator dispatches the *resource* on all devices of a community for which the value returned by ρ_D is higher than s_{c_1}, and if the *resource* seems to be quite relevant only for a community (i.e. the returned value is between s_{c_1} and s_{c_2}), the operator copies the *resource* on the *access point* only. The last option for DISP, when the *resource* does not seem to be relevant for a whole community (Case 3 on Fig. 9.3), is to apply ρ_D on each user in this community; again, this is done by using a threshold value s_u. If the value returned by the function is higher than s_u, then the resource is placed on the user's device that is the most able to get it. The selection of the device is processed by the function SELECTDEV(i, j), i and j being integers, the function returns the device (profile) used by the j^{th} member of the i^{th} community that has the largest storage capacity on its placement area (see Fig. 9.1).

```
SELECTDEV(i, j)
1  device ← ∅
2  a ← 0
3  for k ← 1 to K_{i,j}
4      do if FSPACE(π_{d_{i,j,k}}) > a and STATE(d_{i,j,k}) = TRUE
5          then a ← FSPACE(π_{d_{i,j,k}})
6               device ← π_{d_{i,j,k}}
7  return device
```

Figure 9.1: The device selection function pseudo-algorithm

We have to mention that each time a *resource* is supposed to be placed on a device, DISP first checks the ability of the device to store the *resource* and if there is not enough space for it, the operator compares the new *resource* to the *less interesting resource* that is on the placement area of the device. If the new *resource* is more *interesting*, then it shall replace the other one. This is recursively done by the function PROEMIN(D, π, ρ), D being a resource description, π a device profile, and ρ a value between 0 and 1 (see Fig. 9.2).

```
PROEMIN(D, π, ρ)
 1  proemin ← TRUE
 2  if STATE(π) = TRUE
 3     then if ASPACE(π) > SIZE(D)
 4             then PUT(D, π)
 5             else t ← GETWR(π)
 6                  if ρ > t[1, 1] and FSPACE(π) > SIZE(D)
 7                     then DELETE(t[1, 2], π)
 8                          PROEMIN(D, π, ρ)
 9     else proemin ← FALSE
10  return proemin
```

Figure 9.2: The Proemin function pseudo-algorithm

Before uploading a *resource* on a device, DISP checks if the device is online and if it has enough free space on its placement area (limited predefined space) for the *resource* to be stored. The storage capacity (full capacity and empty space) of a device is defined in order to ensure limits (depending on a minimum and a ratio) that cannot be passed over; the available space dedicated to automated services on the device must be precisely defined (default ratio or user's choice) in order to keep enough memory space for the user's *manual* activities. DISP gets this states' information from the device *profile* via several functions:

- FSPACE(π_d) returns the full space allocated for placed data on the device d (in KB).
- ASPACE(π_d) returns the available space in the placement area on d (in KB).
- SIZE(D_{r_i}) returns the size of Resource r_i (in KB).
- STATE(π_d) returns FALSE if the device d is off, and TRUE if it is on.

Thus appears the update problem: variables we need to handle can change at any time very irregularly (frequency might anyway be taken into account); for instance, it is imperative to record the new locations of the *resource* in its *resource description* and devices' states. Indeed, as it has been explained in Sect. 2.3.4, in order to be able taking advantage of the dispatched *resources* for the query management, we have to keep a record of all the locations a *resource* is stored at. So each PUT and DELETE (see below) implies that the *Resource Description* (which contains all these locations within the *locations descriptor*) is updated. The new version of the *Resource Description* is first saved on the server, and then it overwrites the other copies that are on the devices containing the *resource*. The updates processes have to take into account the possibility for a device to be offline, and so to ensure that the update can be performed as soon as the device becomes available. Following the same strategy, when a device is switched on, it updates its IP address in its *profile*, which is copied on the server and related *access points*. We also have to consider the creation of new communities: each time a community is created, the placement operator must be applied on the server to check what *resources* should be dispatched on the devices of this community. The function UPDATEPROF() provides the support described above for every *Resource Description* and *profile* that has to be updated; it is basically an extension of the UPDATEDR operator described in Sect. 2.3.3.

We finally declare all the functions that DISP uses in order to manipulate the *resources* and their *profiles*:

- PUT(r, d) accesses the placement area on the device d and pastes the Resource r there.
- GETPROFCOM(x) (x being the number (*i*) of the i^{th}community, or the community's identifier *comID*) returns the profile of the related community. GETPROFUSE(x) works the same way for a user.
- GETAC(π_c) returns the profile of the *Access Point* of the community π_c.
- DELETE(x, π_d) deletes the resource identified by x on the placement area of the device d.
- Each device's profile contains a table $[r_i, \rho_i]_{i=1...n}$ made of n columns (n being the number of resources stored on the device) and two rows (resource identifier and related ρ_D values) such as ρ_D values are increasingly ordered. The function GETWR(π_d) returns this table for the device d.

- SELECTDEV(i, j) i and j being integers, the function returns the device (profile) used by the j^{th} member of the i^{th} community that has the largest storage capacity on its placement area (see Fig. 9.1).

```
DISP(D_r)
 1  disp ← FALSE
 2  for i ← 1 to C
 3       do π_{c_i} ← GETPROFCOM(i)
 4          ρ_1 ← ρ_D(D_r, π_{c_i})
 5          device1 ← GETAC(π_{c_i})
 6          if ρ_1 ≥ s_{c_1}
 7            then for j ← 1 to U_i                                          ▷ Case 1
 8                      do π_{u_{i,j}} ← GETPROFUSE(i, j)                   ▷ Case 1
 9                         device2 ← SELECTDEV(i, j)                        ▷ Case 1
10                         if PROEMIN(D_r, device2, ρ_1) = TRUE             ▷ Case 1
11                           then UPDATEPROF()                              ▷ Case 1
12                                disp ← TRUE                               ▷ Case 1
13          elseif s_{c_2} ≤ ρ_1 < s_{c_1} and PROEMIN(D_r, device1, ρ_1) = TRUE   ▷ Case 2
14            then UPDATEPROF()                                            ▷ Case 2
15                 disp ← TRUE                                             ▷ Case 2
16          else for j ← 1 to U_i                                          ▷ Case 3
17                      do π_{u_{i,j}} ← GETPROFUSE(i, j)                   ▷ Case 3
18                         ρ_2 ← ρ_D(D_r, π_{u_{i_j}})                      ▷ Case 3
19                         device3 ← SELECTDEV(i, j)                        ▷ Case 3
20                         if ρ_2 ≥ s_u and device3 ≠ ∅                     ▷ Case 3
                           and PROEMIN(D_r, device3, ρ_2) = TRUE            ▷ Case 3
21                           then UPDATEPROF()                              ▷ Case 3
22                                disp ← TRUE                               ▷ Case 3
23  return disp
```

Figure 9.3: The placement pseudo-algorithm

NB: some variables are shared and are accessible from all the functions that are dedicated to the services; this set consists in all the profiles (communities ($< \pi_{c_j} >_{j=1,...,\mathcal{C}}$), users ($< \pi_{u_{j,k}} >_{k=1,...,\mathcal{U}_j}$), and devices ($< \pi_{d_{j,k,l}} >_{l=1,...,\mathcal{L}_{j,k}}$)), sets' number of elements ($\mathcal{C}$ is the total number of communities, $\mathcal{U}_j$ is the total number of users involved in the j^{th} community, $\mathcal{K}_{i,j}$ is the total number of devices used by the j^{th} user of the i^{th} community), and threshold values (s_{c_1}, s_{c_2}, s_u).

The DISP placement operator has been introduced in [GAG04b]; we propose here a full description of the pseudo-algorithm (see Fig. 9.3) that moreover takes into account new features such as checking devices activity and storage capacity.

Algorithms of operator DISP and functions it is using have been given in [GAA04]. They are completed with the update and consistency policies that have been defined in Sect. 3; they extend the update propagation by triggering the operator DISP each time a creation of a new *resource* or an update of an existing *resource* occurs. Moreover, DISP must check if a *resource* is not already on the device (with rID) in case the *resource description* is the result of an update (so the device can avoid to store copies of a *resource* having several *resource descriptions*).

9.4 Possible Placement Enhancements

Scheduling

De-connections of devices is an important issue for the management of transactions in distributed systems. In fact, as it appears for the functions SELECTDEV and PROEMIN respectively on line 4 of Fig. 9.1 and on line 2 of Fig. 9.2, loss of connection might bother services such as automated data placement. The only solution to avoid troubles in most of the cases is to add some scheduling to the operator.

The simplicity of electronic scheduling is attractive: it automates specific processes with which everyone is familiar and does not require complex vocabulary to describe its workings or benefits. The time domain aspect affects the entire applications range from the user interface down to the underlying architecture. Much work has been done to define optimal data placement for specific storage systems such as large scale storage servers [CTZ97] or more precisely tape jukeboxes [HRS99].

It is clear that referencing the activity of users and analyzing behavior history allows administrators to build some statistical strategy for processing the data placement. In addition, considering simple elements such as time slots (for communities that are spread in several countries), day time, or time-off makes it possible to optimize the data distribution. Indeed, by balancing the dispatching operator according to users activity, scheduling can avoid (or at least reduce) servers overload and traffic congestion

However, the scheduling is deeply dependent on the kind of community IMAM is applied to. So a scheduling strategy has to be defined and applied according to each case.

Values Comparison

The approach for the resource relevance evaluation based on the function ρ_D, which we proposed in Sect. 9.2, can easily be improved through implementation choices. Indeed, the selection of restricted vocabularies allows users to choose terms that are used for the categorization and description of both resources with their resource descriptions, and devices with their profiles. Then, attributes sharing same vocabularies make the quality of the similitude evaluation between the knowledge entities higher. Thus, we included some restrictions for the descriptors values that are presented in Table 3.1. The usage of predefined lists of terms is in fact a necessary condition for the placement service to be relevant, as it moreover solves the issue caused by partial matching (i.e. the matching between a word alone and a word in a sentence).

The vocabularies must be of course related and dedicated to each community. This does not necessary require community members to establish and maintain these vocabularies; indeed, many communities are sharing the same interests and can use vocabularies defined by other communities; then they have the possibility to adapt and improve the lists of terms. We do not here investigate the management of shared vocabularies for communities as it is being studied in our research team.

Chapter 10 will investigate the validity of our strategy and will provide the decision elements for the possible need of refinements in ρ_D.

Chapter 10

Services Implementation

> "The whole is more than the sum of its parts."
> - Aristotle (384-322 BCE)

Aggregating simple processes in complex heterogeneous environment is not an easy task. However, IMAM and its clear knowledge structure and operators makes it possible to implement simple functions that produce very useful results (such as the function ρ_D). In order to fit projects such as DSR illustrated on Fig. 1.1 and the architecture we are proposing on Fig. 1.2, the deployment and usage of the services described in chapters 8 and 9 must rely on (at least) a server that performs the costly tasks.

Furthermore, IMAM services deployment requires a favorable environment in order to be useful: collaborative groups of users sharing data. As we are writing these lines, the projects we are involved in are not yet providing us well defined and usable communities. Therefore, the evaluation we are able to present is quite limited.

This chapter proposes a description of a framework we proposed for a preliminary implementation of IMAM and related services. The chapter is structured as follows:

- In Sect. 10.1, we describe the architecture of the platform that is currently being used for the deployment of IMAM services.
- In Sect. 10.2, we consider the evaluation policy for IMAM and define a simple set of test protocols..
- In Sect. 10.3, we briefly enumerate and describe the main components we are using for implementing our modeling.
- In Sect. 10.4, we present some preliminary results from experiments that are being conducted.
- In Sect. 10.5, we summarize the part dedicated to adaptive services based on IMAM by showing the benefits that communities' members get from the viewpoint and the authoritarian placement, and by underlining the remaining improvements that have to be made.

10.1 Framework Used for Implementation

The implemented platform architecture is based on open source components including a storage layer (Dspace), an ontology based metadata management, the query interface and resource entry service, and multi-resolution resource viewing. The system limits the access to data according to users rights to indoor users (Intranet), outdoor users (Extranet) and to the Web users.

10.1.1 Query Interface

Queries are performed via a web browser based interface. Screens for simple or advanced queries can be easily created and the fields to be viewed customized by the system administrator. In addition, date or numeric size fields can be searched by specifying a range of dates or sizes between which searches are performed. Users are able to select the working language and the domain of interest as well as the number of results returned and whether resource results are shown. The interface is divided in three parts:

- Historical and material resources related to artifacts.
- Technical or management information related to photographic resources.
- Technical or management information related to document resources.

Where applicable, the user can choose technical terms from a list of relevant terms classified alphabetically, or can type something directly in. Ontologies in 21 languages will be able to be consulted on line. Full text searches can be made within each field. The display or the output format of the results (e.g. HTML, XML, plain text, formatted tabular, list of images, graphical, statistical analyses etc) is independent of the storage structure in order to optimize the delivery process. It typically follows a methodology based on context-dependent cultural resource accesses.

10.1.2 Multi-Resolution Resource Viewing

Another key component of the resource management system is the capability to remotely view multi-resolution resources including high resolution images of both 2D paintings and 3D objects. Each image resource is stored as both a JPEG thumbnail for rapid previewing and in tiled pyramidal TIFF format for high-resolution viewing. A java applet permits multi-resolution viewing in conjunction with the storage layer. This viewing system is based on the Internet Imaging Protocol. The viewer works by requesting only the tiles at the appropriate resolution required for viewing a particular part of the image. The requested tiles are then dynamically JPEG encoded by the server and sent to the applet. In this way, images of any size can be viewed quickly across the internet.

10.1.3 Multilingual Ontology-based metadata

Multilingual support is becoming very critical in the cultural domain. This can be accounted by: (i) the increasing share of cultural contents accessed over internet, (ii) efforts to develop standards for cultural data from diverse fields for the purpose of digital archiving and research sharing, and (iii) the increase in use of tools to extract semantic from cultural digital data. Furthermore, multi-lingual Ontology-based metadata approach enables searching by semantic and by contextual content as it relies on multi-lingual annotated documents and features extraction processes. Each set of ontologies is based on an object-identifier bridge and mono-lingual Unicode (UTF-8) encoding ontologies. Controlled vocabularies of technical terms (e.g. Art and Architecture Thesaurus AAT, Library of Congress Authorities) from each ontology as well as the free text information fields (such as the titles) have been translated with the support of domain experts.

10.2 Evaluation Policy

10.2.1 Testing Adaptive Services

Measuring the effectiveness of a personalization service involves defining metrics and feedback techniques. To measure success, it is first necessary to understand what success means. Success is related to the goals of the service; relevant questions are what type of user is being targeted, what should the user be able to do, and what does the service want the user to do? In e-business the click-through metric measures the frequency of clicks on a link after it is displayed (e.g. an ad banner); the look-to-buy metric measures outcomes that result from display (e.g. sales). The use of some of these techniques is described in [SCH00].

Features of specific adaptive services can be compared and contrasted on how well they support some personalization rules: how much does one system require explicit user profile setting, how does the system support vocabulary level personalization, how easy is it to automatically populate content profile values, what features do the personalization system include to help manage controlled vocabularies?

10.2.2 Test Protocols

Since DSR and ASPICO are not yet able to perform tests (communities are not yet defined and the amount of available *resource descriptions* is not yet sufficient), we initially propose a limited set of test protocols.

Let us introduce the first testing environment we are using: we define two sets of tests dedicated to lectures and best practice that are given at University of Tokyo, department-of-education attached secondary education school (東京大学教育学部附属中等教育学校). The environment is made of a class of 30 students divided in 5 groups who have to retrieve multimedia documents (they initially focus on maps and pictures) according to requirements related to the subject of the lecture, e.g. the Belief Systems of the Silk Roads.

The Silk Roads encompassed a diversity of cultures embracing numerous religions and world views from a vast region stretching from Venice, Italy, to Heian (present day Kyoto), Japan. Between these two geographic endpoints, represented belief systems are Buddhism, Confucianism, Christianity, Daoism, Hinduism,

Islam, Judaism, and Shinto. During the height of Silk Road trading in the 8th century, Buddhism, Islam, and Nestorian Christianity were the dominant religions.

This activity asks students to think about similarities and differences among belief systems related to mandala and associated symbolisms. Having access to resource sets, students will be asked to organize them into broad categories of essential concerns; some dimensions to be considered are the location, period, related topic of interest. . .

A collaborative memory of digital contents involves access from various kinds of users including experts, and students (University, High school, Primary school, etc.). Furthermore, collaborative support for annotation requires different layers of points of view. Annotation means comments, notes, explanations, or other types of remarks that are attached to each individual resource. When the user accesses the resource, the user can also produce annotation related to one resource with his/her own words. Also the end user can load annotations attached to it from a selected annotation server, or several servers, and see what group thinks. The point of view including education, and cultural backgrounds influences the annotation of contents.

The aim of the four lectures/Practices that took place between November and December 2004 is to introduce the key background related to the Digital Silk Roads project according to historical, geographical, architectural digital dimensions. The case study will demonstrate two functions: the creation and the search (see Fig. 10.1) of annotation/metadata of mandala done by high school students.

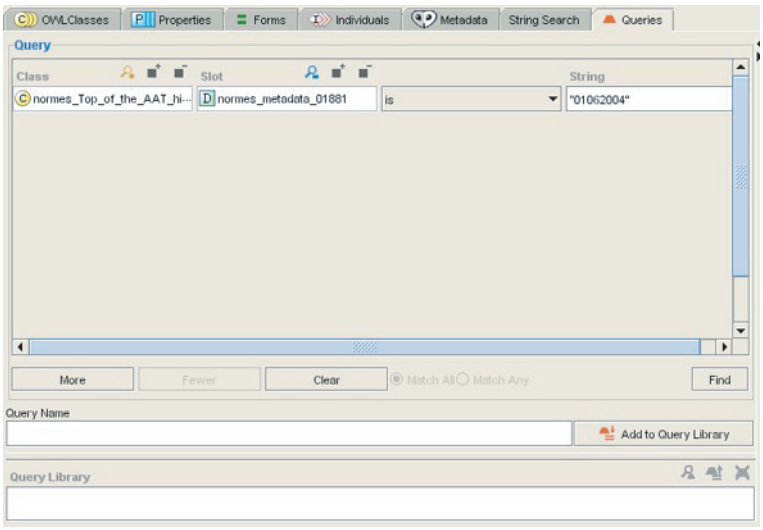

Figure 10.1: Interface for querying

We provide a profile to each community (several communities have been set: e.g. one related to art and the other to religious texts, with some more detailed characteristics of course) such as the one displayed on Fig. 10.3, and obviously, every user also has one (examples of profiles used are shown in Sect. A and a specific profile used for this test session is given on Fig. 10.2). Note that we do provide a profile for the devices, but

as all students are using the same type of computers, the role of these profiles is not preponderant in this set of tests (the profile for the device is given on Fig. 10.4). We of course also provide finite sets of *resources* with *resource descriptions*. Then, depending on the environmental constraints (i.e. computing power and time dedicated to the tests), we apply some requirements:

- To define the space that is allocated for the usage of IMAM services on each computer.
- To define the set of resources to be used (how many, what topics, how many categories...)

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE rdf:RDF [
         <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
         <!ENTITY kb 'http://protege.stanford.edu/kb#'>
         <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
]>
<rdf:RDF xmlns:rdf="&rdf;"
         xmlns:kb="&kb;"
         xmlns:rdfs="&rdfs;">
<kb:User_Profile rdf:about="&kb;KB_218634_Instance_11"
         kb:Family_name="tsujimoto"
         kb:Language="Japanese"
         kb:birth_date="1986/4/5"
         kb:current_location="computer room"
         kb:fields_of_expertise="Japanese_history"
         kb:first_name="mariko"
         kb:id="2"
         kb:main_location="computer room"
         kb:usual_location="computer room"
         rdfs:label="2">
        <kb:inverse_of_user_involved rdf:resource="&kb;KB_658623_Instance_16"/>
        <kb:device_id_in_usage rdf:resource="&kb;evaluation_Instance_0"/>
        <kb:inverse_of_user_involved rdf:resource="&kb;evaluation_Instance_20000"/>
        <kb:fields_of_interest>Buddhism</kb:fields_of_interest>
        <kb:fields_of_interest>Informatics</kb:fields_of_interest>
</kb:User_Profile>
</rdf:RDF>
```

Figure 10.2: User profile test

The set of resources we are using for this session is made of 500 images organized within 40 categories.

In this test protocols, we only consider the placement service; indeed, the viewpoint requires more distributed and heterogeneous environments (for devices mainly) to be applied. Unfortunately, wider communities from the projects we are involved in, and to which we are supposed to apply our model on, are not yet available. Two types of test are to be performed:

A. **Relevance test**: perform placement operator before lesson starts and compare the results (set of resources dispatched) with the set of resources students will select from the server through traditional search and queries.

B. **Performance test**: we check the processing time of the placement operator with different structures of *resource description*. Several formats will be compared such as OWL and RDF.

Then it is important to perform the placement for two groups (or more) having the same profiles but with different threshold values.

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE rdf:RDF [
         <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
         <!ENTITY kb 'http://protege.stanford.edu/kb#'>
         <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
]>
<rdf:RDF xmlns:rdf="&rdf;"
         xmlns:kb="&kb;"
         xmlns:rdfs="&rdfs;">
<kb:Community_Profile rdf:about="&kb;KB_658623_Instance_16"
         kb:birth_date="10/12/2003"
         kb:id="1"
         kb:main_topic_of_interest="Religion"
         kb:name="Silk Roads studies"
         rdfs:label="1">
        <kb:user_involved rdf:resource="&kb;KB_218634_Instance_11"/>
        <kb:user_involved rdf:resource="&kb;KB_218634_Instance_16"/>
        <kb:user_involved rdf:resource="&kb;KB_218634_Instance_17"/>
        <kb:user_involved rdf:resource="&kb;KB_218634_Instance_18"/>
        <kb:user_involved rdf:resource="&kb;KB_218634_Instance_20"/>
        <kb:user_involved rdf:resource="&kb;KB_218634_Instance_21"/>
        <kb:user_involved rdf:resource="&kb;KB_218634_Instance_22"/>
        <kb:user_involved rdf:resource="&kb;KB_218634_Instance_23"/>
        <kb:device_involved rdf:resource="&kb;evaluation_Instance_0"/>
        <kb:device_involved rdf:resource="&kb;evaluation_Instance_11"/>
        <kb:device_involved rdf:resource="&kb;evaluation_Instance_12"/>
        <kb:device_involved rdf:resource="&kb;evaluation_Instance_13"/>
        <kb:device_involved rdf:resource="&kb;evaluation_Instance_14"/>
        <kb:device_involved rdf:resource="&kb;evaluation_Instance_6"/>
        <kb:device_involved rdf:resource="&kb;evaluation_Instance_7"/>
        <kb:device_involved rdf:resource="&kb;evaluation_Instance_8"/>
        <kb:main_topic_of_interest>Architecture</kb:main_topic_of_interest>
        <kb:main_topic_of_interest>Art</kb:main_topic_of_interest>
        <kb:other_topic_of_interest>Landscape</kb:other_topic_of_interest>
        <kb:other_topic_of_interest>Mandala</kb:other_topic_of_interest>
</kb:Community_Profile>
</rdf:RDF>
```

Figure 10.3: Community profile test

10.3 Operators Development

Java is the most appropriate language for the backbone of our implementation; indeed, it fits our environment and provides many relevant development tools. In fact, it enables us to work on heterogeneous platforms and to be quite OS independent. We are using the following elements:

- Java 2 platform Micro-Edition (J2ME); several versions and implementations: MIDP2 (convenient GUI, easy to program), JEODE (implementation of Personal Java, kind of light JDK 1.1.8), and Personal Profile which is almost 1.3 compliant (compiles 1.3, see also RMI); It includes Swing (free implementation from Sun for Zaurus PDA, which is the main device used for our implementation) and is made of several packages: Personal Profile, Personal Basis Profile, CDC Profile (RAM & ROM requirements), CLDC Profile (weak devices).
- XML management: XML Processing with Java (JAXP) with SAX, DOM, XSLT... Plenty of *JAX* (Java API for XML): JAXB, JAXM, JAXR...

All the operators described in chap. 2.3 can be implemented in Java (several of them already are) or in Perl. For comparison matters, we use an application based on Java parsers for XML. We map the XML

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE rdf:RDF [
         <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
         <!ENTITY kb 'http://protege.stanford.edu/kb#'>
         <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
]>
<rdf:RDF xmlns:rdf="&rdf;"
         xmlns:kb="&kb;"
         xmlns:rdfs="&rdfs;">
<kb:Device_profile rdf:about="&kb;KB_218634_Instance_0"
         kb:allocated_memory_space="5G"
         kb:available_memory_space="40GB"
         kb:current_connection_bandwith="40M"
         kb:id="1"
         kb:main_connection_bandwith="100M"
         kb:processor_frequency="860_MHz"
         kb:screen_resolution="1024x768"
         rdfs:label="1">
        <kb:inverse_of_device_id_in_usage rdf:resource="&kb;KB_218634_Instance_10"/>
        <kb:inverse_of_device_involved rdf:resource="&kb;evaluation_Instance_20000"/>
</kb:Device_profile>
</rdf:RDF>
```

Figure 10.4: Device profile test

files, identify similarities (term matching) between these two files and perform the calculation proposed in Sect. 9.2 with the function ρ_D and the complementary relevance evaluation described in Sect. 11.2.2. The prototype we are designing for the *viewpoint* however relies on perl for the processing of the rules. The links to applications (for transformation rules namely) are to be performed through Java calls. Perl is also useful for the randomized generation of resource descriptions and environmental profiles. For the distribution tasks, we are starting from the efficient P2P architecture provided by BitTorrent (initially developed in Python) and consider Java implementations such as Azureus[1].

10.4 Preliminary Results

The tests are to be processed very soon. Indeed, we are still waiting for the resource descriptions of the 500 resources that have been used as an initial set for the sessions described above. Then, as soon as we get them with the query results performed by the students without any external support (i.e. in a quite naive way, which is a very good approach for the validation of a framework such as ours), we can apply the placement relevance evaluation process and publish the comparative results.

We provide here an example of the evaluation process that we are going to perform for the set of communities used in the tests described in Sect. 10.2.2; Table 10.1 will contain the results of the test session denoted i.j where:

- i is the identifier of a set of threshold values (s_{c_1}, s_{c_2}, s_u) which is applied to DISP (see Fig. 9.3).
- j is the size of the memory space allocated on the devices.

[1] Azureus, Java BitTorrent Client: http://azureus.sourceforge.net/

Then we will compare the results depending on i and j. A first set of conclusions will provide information on the type of community that gets better results, on the threshold values that improve the selection (some hypothesis will have to be defined and validated to understand and explain the results and then to generalize them), and on the influence of j (i.e. the allocated space on the devices) on the placement evaluation processing time. An interesting value, which we call deviation (rate in %) defined by $d_{c_k} = \frac{|<D_{rc_k}> \cup <D_{rc_k}>_{\text{DISP}} \setminus <D_{rc_k}> \cap <D_{rc_k}>_{\text{DISP}}| \times 100}{<D_r>_{init}}$ informs us on the scale of the difference between the set selected by the community and the result of ρ_D.

The human judgment also has to be taken into account; the teacher of the class which is doing the tests and the categorization information provided by the RCT make it possible to get a reliable opinion on the relevance of the set selected by a community which has specific querying instructions.

Table 10.1: Placement relevance evaluation sheet

comparative results - session i.j			
	C_1	C_2	$C_3 \ldots$
community's selection	$< D_{rc_1} >$	$< D_{rc_2} >$	$< D_{rc_3} > \ldots$
DISP operator's selection	$< D_{rc_1} >_{\text{DISP}}$	$< D_{rc_2} >_{\text{DISP}}$	$< D_{rc_3} >_{\text{DISP}} \ldots$
DISP processing time	t_1	t_2	$t_3 \ldots$
deviation	d_{c_1}	d_{c_2}	$d_{c_3} \ldots$

Another set of conclusions will be the result of a deeper analysis and of other tests; it will in particular have to answer the following questions: what influence on the comparative results has the ratio of the number of selected resources by a community regarding the initial set (500), e.g. if the community selects half of the initial set, then can the results be relevant enough? What is the variation of DISP operator's selection according to the precision of the profiles (i.e. number of significant terms that are used in the relevance evaluation process)? What is the probability, when a user is requesting a specific resource, for it to have already been placed on the user's device? The following step will be to evaluate the part of the service dedicated to each user separately (lines 16-22 in DISP algorithm shown on Fig.9.3).

10.5 Conclusions

Part III focused on automated services that aim at improving the adaptive delivery of resources within communities. Although we cannot provide results demonstrating the benefits of IMAM services, we must point out that these services are not harmful neither bothering for communities' members that would use them. Indeed, as the space dedicated to the placement on the devices, and the rules used by the *viewpoint* are chosen by the communities, any drawbacks can easily be suppressed by the community itself. Moreover, the placement under conditions defined by scheduling constraints can ensure that the service is not overloading users' devices. The key issue for the services efficiency and relevance is tailoring; the tests described above aim at providing directions for the evaluation and improvement of IMAM services.

Part IV

Collaborative Resources Delivery in Perspective

Chapter 11

Merged Services for Advanced Distribution of Resources

> "Nothing is more fairly distributed than common sense: no one thinks he needs more of it than he already has."
>
> - René Descartes (1596-1650)

The interesting features we are getting from the DISP operator and the *viewpoint* can then be enhanced by using an appropriate query management based on our three-layers architecture (server, *access point*, normal device). Merging transactions (DBMS and Network) is a stimulating concept but remains far from our achievements. However, we are convinced that associating the transactional processes between the devices and the resources would empower IMAM services. This first requires to identify the useful elements that raised with new distributive models (especially mobile and P2P ones). Then we need to ensure that IMAM provides the best relevance evaluation model for resources management; which means that we continuously have to refine our model.

This chapter proposes a first (and not yet decisive) idea of how IMAM services could be improved by integrating a merged management of transactions. The chapter is structured as follows:

- In Sect. 11.1, we review the issues we are facing when considering the deployment of IMAM's services.
- In Sect. 11.2, we present the improvements that IMAM needs in order to support a unified resource relevance evaluation method and to apply it to its adaptive services.
- In Sect. 11.3, we describe the architecture and components of a platform that fits the modeling and the services presented in this dissertation.

11.1 Distributive Issues for Adaptive Services

11.1.1 Peer-to-Peer.

Noticing that current peer-to-peer systems lack efficient knowledge management, Lee and al. [LOP02] propose to improve file sharing systems with interconnected (or linked) files. Indeed, being able to evaluate relationships (based on similarities) between resources enables query processing to be more efficient. Much work done on shared resources in peer-to-peer systems addresses the problem of community-related information representation. Focusing on this issue, U-P2P is an XML-based framework that allows users to easily access knowledge describing resources [MEA02] and communities [AEM03]. Another peer-to-peer architecture using XML-Schema called KEx [BBM02] considers social structures to achieve semantic coordination between peers. Moreover, Kex implementation is build on JXTA, which is a very interesting set of peer-to-peer interoperability framework.

In pure P2P (decentralized), all peers are playing both the roles of client and server. As we explained in Sect. 4.3.1, it is interesting to use hybrid architectures; in this case, some nodes assume the role of a super-peer and the others are considered as leaf-nodes. Each super-peer becomes a proxy for all its neighboring leaves: it indexes all their document and processes their requests.

Transactional processes between peers on a P2P network are due to two types of activities:

- Searching for an object.
- Sharing an object.

The search methods can be categorized as either *blind* (in a pure decentralized system) or *informed* (in a partially centralized system). A comparison of search methods for these two categories is given in [TR03]; the evaluation is based on three criteria: accuracy, bandwidth consumption, and discovered objects. According to the authors, the best strategy is to perform *informed* search that have no costly index updates (such as *DRLP* and *s-APS*).

11.1.2 Mobile Knowledge

Because of mobile devices compact hardware structure, transmission rates, low autonomy, and higher probability to be damaged, it is important to take technical limitations into account when designing mobile applications. It is also valuable to consider the available knowledge about users as it can be helpful for optimizing the use of mobile devices. Some works propose an integrated context-aware knowledge management using ontologies for mobile devices. But as far as we know, these approaches are focused on one field of application, e.g. for sensors acquisition [KM03], and do not take fully advantage of all the available knowledge.

Some investigations on distributed and mobile collaborative systems [DG02] provide interesting overviews of the problems and describe nice frameworks such as a five-layer architecture for *distributed and mobile collaborative* (DMC) systems. [FDK02] even gives some requirements for access control in mobile P2P teamworking systems but does not really describe how to validate them.

An interesting framework description has been given [GTB02] for the management of mobile knowledge. It uses context data in order to capture information about users' location and available resources, or to detect the presence in a same area of people being part of a same community. Our approach is quite similar from the context-awareness point of view but we think that this kind of framework can be generalized. Indeed, it is obvious to us that the first layer to be defined is a strong unified model for the management of knowledge in a distributed and mobile environment. Then it becomes possible to supply coherent and advanced distributed services based on knowledge management.

11.2 Generalized Relevance Evaluation

In order to overcome the shortcomings of our relevance evaluation described in Sect. 9.2, we need to complete our model by enhancing the selection of *very* relevant information.

11.2.1 Existing Techniques

Recommendation systems are basically all using Refined Similarity Evaluation in order to provide personalized information: Amazon[1] is referring to users having similar *tastes* in order to recommend articles (Amazon determines a user's interests from previous purchases as well as ratings given to titles; the user's interests are compared with those of other customers to generate titles which are then recommended during interaction), so do collaborative music recommendation systems[2]. Extracting correlations between heterogeneous entities is a complicated issue. As a matter of fact, existing comparative methodologies based on similarity contents are considering homogeneous knowledge structures (e.g. user *profiles*); *profiles* similarity evaluation, such as in [MSD04, ZL04] for instance, is a quite common task in recommendation systems. These social filtering systems usually compute term frequency or users' ratings in order to evaluate common interests.

Collaborative filtering compares a user's tastes with those of other users in order to build up a picture of like-minded people. The choice of content is then based on the assumption [GNO92] that this particular user will appreciate the elements that people having similar tastes also enjoyed. The preferences of the community are used to predict appropriate content. The users' tastes are either evaluated from their previous actions or else measured directly by asking the user to rate elements. This method has an advantage of speed and efficiency in computation [PHL00], thus delivering rapid feedback. The reliance on a *critical mass* of users can be a problem for collaborative filtering; a small sample population may lead to lower-quality recommendations. The quality of recommendations increases with the size of the user population. Moreover, collaborative filtering may be less important as a technique [KS04] when categories of users and preferences are already well-known and well-defined.

[1] http://www.amazon.com

[2] Music Recommendation System for iTunes: http://music.cs.uiuc.edu/about.php

There are two main equations in the literature for evaluating similarity between two *profiles*, also sometimes regarded as *profile* vectors. Let us consider a rating π_{i_x}, being a value (e.g. an integer value) from an ordered set π_i (i.e. a *profile*). In the following expressions, we use a set-selective notation for the summation bounds, so a *profile* does not have to rate every attribute:

- Mean Squared Difference: $d_{MSD}(\pi_i, \pi_j) = \frac{1}{|P_{\pi_i \cap \pi_j}|} \sum_{x \in P_{\pi_i \cap \pi_j}} (\pi_{i_x}, \pi_{j_x})^2$

  where $P_{\pi_i \cap \pi_j}$ is the set of attributes which both profiles π_i and π_j have ratings for.

- Pearson Correlation Coefficient [RIS94]:

$$r_{Pearson}(\pi_i, \pi_j) = \frac{\sum_{x \in P_{\pi_i \cap \pi_j}} (\pi_{ix} - \overline{\pi_i}) \cdot (\pi_{jx} - \overline{\pi_j})}{\Big(\sum_{x \in P_{\pi_i \cap \pi_j}} (\pi_{ix} - \overline{\pi_i})^2 \cdot \sum_{x \in P_{\pi_i \cap \pi_j}} (\pi_{jx} - \overline{\pi_j})^2 \Big)^{\frac{1}{2}}} \in [-1, 1]$$

  then:

  - $r_{Pearson} > 0$ implies that π_i and π_j are positively related.
  - $r_{Pearson} = 0$ implies that π_i and π_j are not related.
  - $r_{Pearson} < 0$ implies that π_i and π_j are negatively related.

  It is important to note that the correlation coefficient is not transitive; however, profile similarity is, at least to some degree, transitive.

One of the possible way for us to use this kind of strategy would have been to require ratings about the *descriptors* and/or *descriptors values* of the RCT from each user. And then it would have been easy to perform a data placement relying on recommendations based on users having *similar profiles*. However, this would go against the choice we made; indeed, the community layer brings, according to us, much relevance to the information management. In a quite aggressive manner (which shows the importance of this issue), Google is developing many applications dedicated to the relevance of information. Examples of features the company is providing are personalized query results and adaptive advertising service for web pages[3]. The strategy is to adapt information according to users *profiles* and browsing history. IMAM's services go one step further as they consider communities and their precious topic orientation in order to improve the *resources* retrieval and distribution.

11.2.2 Refined Relevance for IMAM Services

We see two possible directions for improving the relevance evaluation that is used within IMAM services:

[3] Google AdSense program:
http://www.google.com/services/adsense_tour/

- Weighting attributes. It is possible to add an attribute to each descriptors and descriptors values; then this attribute would get high weight for elements that seem to be very attractive to users. This weighting strategy can be performed by the user himself, or by automated processes that analyze the user activity history (key word extraction...). [JCS04] provides interesting weighting strategies that are dedicated to recommendations through collaborative filtering. However, this approach cannot be applied to our model. Indeed, our approach must be dedicated to each user, and not only to growing groups of users that are becoming advising entities; as we explained it in previous sections, the notion of relevance must include different types of environmental entities.

- Path selection. The RCT structure of IMAM provides a strong support for selecting types of *resources* that are interesting to a user. Indeed, by selecting part of or full paths that are related to specific topics and multimedia types, users can express their preferences in a simple way. This approach is relying on the extensions communities will have brought to core RCT. In fact, the more detailed the extensions are, the easier it is for user to specify very relevant categories of *resources*. Then, the matching evaluation between two knowledge entities can be improved by applying graph matching algorithms using filters [MGR02] on our operators.

Evaluation of resources relevance to environments can be enhanced by identifying more precisely the information that is interesting for communities and their users. The extensions and improvements we plan to add to our model include a combination a of these two strategies.

11.3 Merged Services

The first requirement for our framework is to enable IMAM services to be platform independent for the management of the *resources*. One possible approach is to define a customized mobile database kernel allowing users to manage easily and efficiently the data (focusing on multimedia data) they are handling on mobile devices. The main originality of this strategy is to focus on the mobile side; much of the work done on the data management for mobile devices is still based on the usual Client/Server architecture and fails on bringing new services for the next generation of communication protocols. Using an hybrid approach mixing C/S and P2P features, as it is done in the DMC architecture described in [DG02], seems to be the best strategy to fulfill the needs of mobile devices. Then, one of the greatest challenge of the DBMS cellular model is to manage itself, i.e. without administrator. So it needs to have high level automated features. To be effective, the mobile DBMS has to be able to run itself without the help of a server. Moreover, with highly dynamic data dependencies, the amount of data circulating would be enormous considering the traditional RDBMS services. An independent system, being able to manage its own communications and taking into account the network capacity, would reduce this drawback. Of course, the micro kernel must be able to query the server version of Phasme prototype, and reciprocally. We initially planned to implement a prototype of this mobile kernel based on EBG and the customization provided in Sect. 5.3. Unfortunately, the update of the EBG core prototype phasme has not been completed yet.

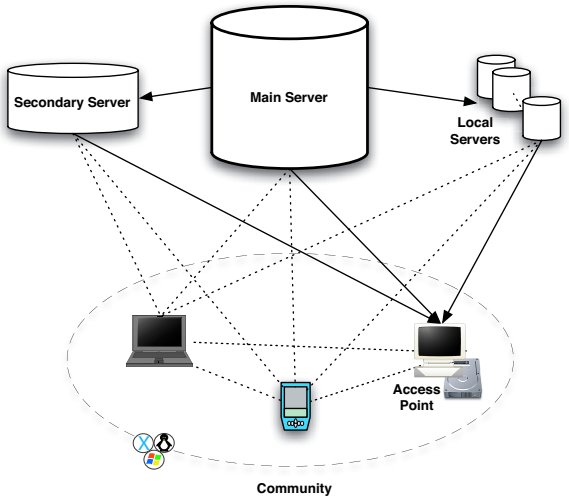

Figure 11.1: Resources distribution

Moreover, from all we have seen in the literature and in the available products, designing micro DBMS kernels for small devices still faces many issues that are far from being solved. In addition, as most of the work done relies on the relational model, the strategies that have been chosen for the existing solutions do not fit IMAM.

Therefore, we propose to perform IMAM services through agents or sets of processes running directly on devices OS. We are currently designing a distributed query manager based on JXTA and the P2P delivery protocol BitTorrent[4]; in fact *resource descriptions* can partially be seen as BitTorrent *trackers*, as they contain all the locations of the *resources*. We now just have to take advantage of IMAM's support to provide appropriate *resources* to users in the best conditions. Following BitTorrent strategy, we can provide distributed query processing by using the placed and indexed data; then a device can access all copies of a *resource* (even not complete ones) through fully distributed and automated collaborative management of resources (see Fig. 11.1).

However, in order to provide a fully-distributed joint processing for IMAM servicing applied to the architecture, which we initially defined in Fig. 1.2, some requirements clearly appear:

- when returning a set $< D_r >$ as a query result, the *viewpoint* might check if some *resources* have not already been *placed* on the querying device. Then, the process can avoid to download the *resource* again and might allocate a higher importance to the concerned *placed resources* by putting them ahead in the returned set or simply inform the user that the access to these *resources* would be faster.

- for improving the interests definition of the users. For instance, the profiles might keep a record of the most used terms in the queries for the placement to integrate them in its relevance evaluation process.

[4]as it is used in the eXeem project: http://www.exeem.com/

Chapter 12

Conclusions

> "And Presently some master brain in the Inner Party would select this version or that, would re-edit it and set in motion the complex processes of cross-referencing that would be required, and then the chosen lie would pass into the permanent records and become truth."
>
> - George Orwell (1903-1950) "Nineteen Eighty Four"

12.1 Summary of Contributions

As communities generate increasing amounts of transactions and deal with fast growing data, it is very important to provide new strategies for their collaborative management of *resources*. In this dissertation, we presented and described a framework based on a generic Information Modeling for Adaptive Management called IMAM, which aims at solving these issues.

In Chap. 1, we first motivated the need for such modeling in order to provide personalized delivery services to users who are involved in communities, and introduced the architecture our modeling is designed for. Then in Chap. 2, we gave an overview of IMAM's formal structure with its operators, focusing on update and consistency policies. Chapter 3 proposed a description of our technical choices and examples for the implementation of IMAM entities. In the second part of the dissertation, Chap. 4 investigated solutions to manage the data itself through our architecture among heterogeneous environments and platforms. Thus, we proposed an hybrid design for the storage (in Chap. 5) and indexing (in Chap. 6) of resources descriptions with consistent links to the resources within XML. This strategy relies on a multidimensional index that has been intuitively adapted to MXML. Part III proposed and defined adaptive services that enable collaborative projects to automatically dispatch *resources* and to make users' queries results more relevant. Chapter 7 investigated the need for adaptive services and existing solutions. The automated services, that we are proposing by using IMAM, provide a query result optimizer called *viewpoint* (in Chap. 8), and an authoritarian data placement (in Chap. 9). Some preliminary descriptions and results of the services implementation, and the

validation policy of IMAM services have been given in Chap. 10. Finally, Chap. 11 reviewed some technical issues that are remaining and proposed open technical challenges for IMAM services to comply with fully decentralized architectures.

The motivation for this work is definitely to improve user's access to information and to reach high satisfaction levels. This directly points out the main issue we are facing; finding the good balance between authoritarian services and user satisfaction. Thus, the main contribution of this dissertation is to provide a generic framework, which, for the first time, can handle any useful information for the automated collaborative management of shared data. This includes the categorization, manipulation, and comparison of resources on which is built an adaptive and relevant data delivery within communities sharing common interests.

In Sect. 12.2, we give some concluding remarks. We finally mention that our work uncovered some specific issues such as the definition of merged services, which are reviewed in Sect. 12.3.

12.2 Concluding Discussions

12.2.1 To Be or not to Be Generic

Our goal is to provide a modeling that can be used by any community and on any platform. However, the consistency of a knowledge representation structure can always be criticized as it can be too precise or too general for some applications. Our approach relies on a core that can be adapted by any group of users who share some interests and data. Therefore, we are confident that our extensible RCT approach makes it possible for communities to get a relevant support as the granularity of the results, that the users will obtain (i.e. their satisfaction), will depend on the precision of the extension and on the services settings (allocated space, rules...) that they will define regarding their own needs.

12.2.2 Ethical Advisory

Automated processes imply for users to loose some control on what is done on their device and with their information. Thus, IMAM must help users to find what is good for them, but must not over-control it. This is a very important difference. Otherwise, even if the services are somehow useful, the danger of making user services-dependent (i.e. for users judgment ability to decrease) is more important.

How much power and authority should users give the artifice over their choices? Indeed, it is important not to mistake a tool for the truth of the results of using the tool. A computer doesn't know how to add 2 + 2. It can be used to simulate that operation and give a repeatable result. If 2 + 2 = 4 for an acceptable number of uses, it is a useful tool. If it hits the one context in which that isn't true, it fails. So it is a dramatic issue to understand in advance what we are committing to and what the bet is.

Another issue is privacy. Computer ought to know from user calendar what kind of, when, with whom and where a person has plans. It is then able to provide personalized services to the user according to the information it could have gathered. This level of personalization is only possible if we surrender every piece

of our lives to the machine. There might in fact come a day when every single step of our lives is recorded in detail (some governments already started...), therefore, what will be done with this data (which is very valuable for many companies) has to be carefully paid attention to. And unfortunately, some usages of this information by web sites and corporations are still to be discovered...

12.3 Remaining Challenges

Efficient XML transmission/parsing is one of those things that only becomes a problem if XML is pervasive (which is in fact one of IMAM's assumptions). XML has grown wildly in its popularity as a medium for exchange of structured data and is now increasingly being used in large scale distributed applications and dedicated messaging protocols. However, XML is quite verbose and sub-optimal for such applications, mostly for the sake of human readability. Compression of XML documents is widely used to lower bandwidth usage and storage capacity for large documents. This is fine as long as people do not have to deal with XML streams and do not use it for devices with low computation power. There are various strategies to *minimize* XML files or entities, e.g. XPRESS [MPC03], XGRIND [TH02b]), Xqueeze[1], or XMill [LS00]. It is hard to figure out what compression method is the best one; indeed, there is no perfect strategy since the results depends on the kind of XML files and on the environment (computing power and usage). But still it is obvious that compression can be an important factor in order to improve XML transmission. And it would be interesting to consider the usage of compressed XML within IMAM services. Another way to reduce the amount of data within XML is to summarize the lexical data. there are interesting issues and solutions proposed in the literature. Anyway, nothing seems to be fully effective yet and investigations are still in progress.

In addition, services processing must be refined and improved. First, relevance evaluation can be enhanced by adding some weights to the descriptors according to users preferences and activities. Thus, mixing passive and active relevance feedback could allow IMAM to be more selective when considering which resources are supposed to be attractive to a user. Then the services distribution processes can be merged within a P2P system that could take advantage of the locating information contained in the resource descriptions and profiles in order to make the uploads and downloads of data faster and safer. This merged service can also contain some scheduling, so it can improve the usage of the available computing resources involved in a community. Ultimately, IMAM knowledge entities and the services processing should be managed in a fully distributed manner, i.e. without any server gathering all the available information; this decentralized strategy still relies on many unsolved issues regarding data consistency and safety.

Our modeling has been quite deeply investigated and received useful comments from many people (especially at DataX and Delos workshops). Unfortunately, this dissertation lacks in experimental results; the tests have indeed been postponed several times and we are just starting to perform the first set of simulation that should validate the placement relevance strategy. However, as we mentioned it before, we know that our services can be considered as not bothering for the users thus we are quite confident in the validation process

[1] http://xqueeze.sourceforge.net/

and results we will get. One of the next step has to be done through experiences in order to evaluate threshold values for the placement and to test and adapt the sets of rules for the *viewpoint*. Of course, we look forward to performing big scale test and evaluation, ideally with large communities that are spread all around the world.

There is no limit for designing and enhancing adaptive services. The main issue (that will in fact remain) is to find the good balance between complexity of the operators (i.e. granularity of the information treatment) and the computing costs.

Appendix A

Profiles

A.1 User

```
<ID> </ID>
<FAMILLY_NAME> </FAMILLY_NAME>
<FIRST_NAME> </FIRST_NAME>
<BIRTH_DATE> </BIRTH_DATE>
<MAIN_LOCATION> </MAIN_LOCATION>
<USUAL_LOCATION> </USUAL_LOCATION>
<CURRENT_LOCATION> </CURRENT_LOCATION>
<LANGUAGE> </LANGUAGE>
<FIELDS_OF_EXPERTISE> </FIELDS_OF_EXPERTISE>
<FIELDS_OF_INTEREST> </FIELDS_OF_INTEREST>
<DEVICE_ID_IN_USAGE> </DEVICE_ID_IN_USAGE>
<COMMUNITY_INVOLVEMENT> </COMMUNITY_INVOLVEMENT>
```

Figure A.1: Structure of user profile

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<User_Profile
    sl1="Family_name" vt1="String"
    sl2="Language" vt2="String*"
    sl3="birth_date" vt3="String"
    sl4="community_involvement" vt4="String"
    sl5="current_location" vt5="String"
    sl6="device_id_in_usage" vt6="String"
    sl7="fields_of_expertise" vt7="String*"
    sl8="fields_of_interest" vt8="String*"
    sl9="first_name" vt9="String"
    sl10="id" vt10="String"
    sl11="main_location" vt11="String"
    sl12="usual_location" vt12="String"
/>
```

Figure A.2: Entries and constraints for the schema of user profile

A.2 Community

```
<ID> </IDr>
<NAME> </NAME>
<MAIN_TOPIC_OF_INTEREST>RELIGION,GEOGRAPHY, HISTORY, ASIA </MAIN_TOPIC_OF_INTEREST>
<OTHER_TOPIC_OF_INTEREST> </OTHER_TOPIC_OF_INTEREST>
<BIRTH_DATE> </BIRTH_DATE>
<USER_INVOLVED> </USER_INVOLVED>
<DEVICE_INVOLVED> </DEVICE_INVOLVED>
```

Figure A.3: Structure of community profile

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Community_Profile
    sl1="birth_date" vt1="String"
    sl2="device_involved" vt2="Instance(Device_profile)*"
    sl3="id" vt3="String"
    sl4="main_topic_of_interest" vt4="String*"
    sl5="name" vt5="String"
    sl6="other_topic_of_interest" vt6="String*"
    sl7="user_involved" vt7="Instance(User_Profile)*"
/>
```

Figure A.4: Entries and constraints for the schema of community profile

A.3 Device

```
<=== REQUIRED FIELD ===>
<ID> </ID>
<=== REQUIRED FIELD PREDIFINED LIST = {PC_DESKTOP, PC_PORTABLE, MAC, LINUX_DESKTOP}===>
<DEVICE_TYPE> </DEVICE_TYPE>
<=== REQUIRED FIELD ===>
<ALLOCATED_MEMORY_SPACE> </ALLOCATED_MEMORY_SPACE>
<PROCESSOR_FREQUENCY> </PROCESSOR_FREQUENCE>
<RAM> </RAM>
<SCREEN_RESOLUTION> </SCREEN_RESOLUTION>
<MAIN_CONNECTION_BANDWITH> </MAIN_CONNECTION_BANDWITH>
<CURRENT_CONNECTION_BANDWITH> </CURRENT_CONNECTION_BANDWITH>
```

Figure A.5: Basic structure of device profile

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Device_profile
    sl1="allocated_memory_space" vt1="Integer"
    sl2="available_memory_space" vt2="String"
    sl3="current_connection_bandwith" vt3="Symbol()"
    sl4="id" vt4="String"
    sl5="main_connection_bandwith" vt5="Symbol()"
    sl6="processor_frequency" vt6="String"
    sl7="screen_resolution" vt7="Symbol()"
/>
```

Figure A.6: Entries and constraints for the schema of device profile

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<project>
  <Community_Profile>
    <id>1</id>
    <name>Silk Roads studies</name>
  </Community_Profile>
  <Device_profile>
    <allocated_memory_space>300000</allocated_memory_space>
    <available_memory_space>2000000</available_memory_space>
    <current_connection_bandwith>1M</current_connection_bandwith>
    <id>2</id>
    <main_connection_bandwith>1M</main_connection_bandwith>
    <processor_frequency>1GHZ</processor_frequency>
    <screen_resolution>1280x768</screen_resolution>
  </Device_profile>
  <Device_profile>
    <allocated_memory_space>500000000<allocated_memory_space>
    <available_memory_space>1000000000</available_memory_space>
    <current_connection_bandwith>1M</current_connection_bandwith>
    <id>1</id>
    <main_connection_bandwith>10M</main_connection_bandwith>
    <processor_frequency>1GHZ</processor_frequency>
    <screen_resolution>1600x1200</screen_resolution>
  </Device_profile>
</project>
```

Figure A.7: Export of device profiles from Protégé

Appendix B

Case Study

B.1 RDF (XML)

B.1.1 Schema

On figures B.1- B.4, we present extracts of the schema used for the tests described in Chap. 10. We use one schema only for all the entities that are involved in the community activity (community, users, and devices characteristics). Restrictions for the descriptors values are defined according to the constraints given in Table 3.1.

B.1.2 Extract of the RCT in RDF

We give a very simple example of the RCT implementation as an extract in RDF on Fig. B.5. In this figure, we do not represent the hierarchical tree structure that has been clearly introduced in Sect. 2.2.1 in order to make it easier to read; we just show the implementation of some descriptors.

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE rdf:RDF [
        <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
        <!ENTITY a 'http://protege.stanford.edu/system#'>
        <!ENTITY kb 'http://protege.stanford.edu/kb#'>
        <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
]>
<rdf:RDF xmlns:rdf="&rdf;"
         xmlns:a="&a;"
         xmlns:kb="&kb;"
         xmlns:rdfs="&rdfs;">
<rdfs:Class rdf:about="&kb;Community_Profile"
         rdfs:label="Community_Profile">
        <rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
<rdfs:Class rdf:about="&kb;Device_profile"
         rdfs:label="Device_profile">
        <rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
<rdf:Property rdf:about="&kb;Family_name"
         a:maxCardinality="1"
         a:minCardinality="1"
         rdfs:label="Family_name">
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;Language"
         a:defaultValues="Japanese"
         a:minCardinality="1"
         a:range="symbol"
         a:values="Japanese"
         rdfs:label="Language">
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <a:allowedValues>English</a:allowedValues>
        <a:allowedValues>Esperado</a:allowedValues>
        <a:allowedValues>French</a:allowedValues>
        <a:allowedValues>Japanese</a:allowedValues>
</rdf:Property>
<rdfs:Class rdf:about="&kb;User_Profile"
         rdfs:label="User_Profile">
        <rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
<rdf:Property rdf:about="&kb;allocated_memory_space"
         a:maxCardinality="1"
         a:minCardinality="1"
         rdfs:label="allocated_memory_space">
        <rdfs:domain rdf:resource="&kb;Device_profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;available_memory_space"
         a:defaultValues="40GB"
         a:maxCardinality="1"
         a:minCardinality="1"
         a:values="5GB"
         rdfs:label="available_memory_space">
        <rdfs:domain rdf:resource="&kb;Device_profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;birth_date"
         a:maxCardinality="1"
         rdfs:label="birth_date">
        <rdfs:domain rdf:resource="&kb;Community_Profile"/>
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;community_involvement"
         a:maxCardinality="1"
         a:minCardinality="1"
         rdfs:label="community_involvement">
        <rdfs:range rdf:resource="&kb;Community_Profile"/>
</rdf:Property>
```

Figure B.1: RDF Schema 1/4

```
<rdf:Property rdf:about="&kb;current_connection_bandwith"
         a:defaultValues="40M"
         a:maxCardinality="1"
         a:range="symbol"
         a:values="40M"
         rdfs:label="current_connection_bandwith">
        <rdfs:domain rdf:resource="&kb;Device_profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <a:allowedValues>100M</a:allowedValues>
        <a:allowedValues>10M</a:allowedValues>
        <a:allowedValues>1M</a:allowedValues>
        <a:allowedValues>40M</a:allowedValues>
</rdf:Property>
<rdf:Property rdf:about="&kb;current_location"
         a:maxCardinality="1"
         a:minCardinality="1"
         rdfs:label="current_location">
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;device_id_in_usage"
         a:maxCardinality="1"
         a:minCardinality="1"
         rdfs:label="device_id_in_usage">
        <rdfs:range rdf:resource="&kb;Device_profile"/>
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <a:inverseProperty rdf:resource="&kb;inverse_of_device_id_in_usage"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;device_involved"
         rdfs:label="device_involved">
        <rdfs:domain rdf:resource="&kb;Community_Profile"/>
        <rdfs:range rdf:resource="&kb;Device_profile"/>
        <a:inverseProperty rdf:resource="&kb;inverse_of_device_involved"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;fields_of_expertise"
         a:allowedValues="Japanese_religion"
         a:minCardinality="1"
         a:range="symbol"
         rdfs:comment="indicates the fields of expertise of the user"
         rdfs:label="fields_of_expertise">
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <a:allowedValues>Japanese_geography</a:allowedValues>
        <a:allowedValues>Japanese_history</a:allowedValues>
</rdf:Property>
<rdf:Property rdf:about="&kb;fields_of_interest"
         a:allowedValues="search_engine"
         a:minCardinality="1"
         a:range="symbol"
         rdfs:comment="indicates the fields of interest of the user"
         rdfs:label="fields_of_interest">
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <a:allowedValues>Annotation</a:allowedValues>
        <a:allowedValues>Buddhism</a:allowedValues>
        <a:allowedValues>Informatics</a:allowedValues>
        <a:allowedValues>Painting</a:allowedValues>
        <a:allowedValues>Silk_roads</a:allowedValues>
        <a:allowedValues>map</a:allowedValues>
        <a:allowedValues>metatadata</a:allowedValues>
        <a:allowedValues>oneline_translation</a:allowedValues>
</rdf:Property>
<rdf:Property rdf:about="&kb;first_name"
         a:maxCardinality="1"
         a:minCardinality="1"
         rdfs:label="first_name">
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
```

Figure B.2: RDF Schema 2/4

```
<rdf:Property rdf:about="&kb;id"
         a:maxCardinality="1"
         a:minCardinality="1"
         rdfs:comment="idenficator of the user"
         rdfs:label="id">
        <rdfs:domain rdf:resource="&kb;Community_Profile"/>
        <rdfs:domain rdf:resource="&kb;Device_profile"/>
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;inverse_of_device_id_in_usage"
         rdfs:label="inverse_of_device_id_in_usage">
        <rdfs:domain rdf:resource="&kb;Device_profile"/>
        <rdfs:range rdf:resource="&kb;User_Profile"/>
        <a:inverseProperty rdf:resource="&kb;device_id_in_usage"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;inverse_of_device_involved"
         rdfs:label="inverse_of_device_involved">
        <rdfs:range rdf:resource="&kb;Community_Profile"/>
        <rdfs:domain rdf:resource="&kb;Device_profile"/>
        <a:inverseProperty rdf:resource="&kb;device_involved"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;inverse_of_user_involved"
         rdfs:label="inverse_of_user_involved">
        <rdfs:range rdf:resource="&kb;Community_Profile"/>
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <a:inverseProperty rdf:resource="&kb;user_involved"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;main_connection_bandwith"
         a:allowedValues="1M"
         a:defaultValues="100M"
         a:maxCardinality="1"
         a:range="symbol"
         a:values="100M"
         rdfs:label="main_connection_bandwith">
        <rdfs:domain rdf:resource="&kb;Device_profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <a:allowedValues>100M</a:allowedValues>
        <a:allowedValues>10M</a:allowedValues>
</rdf:Property>
<rdf:Property rdf:about="&kb;main_location"
         a:maxCardinality="1"
         a:minCardinality="1"
         rdfs:label="main_location">
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;main_topic_interest"
         a:minCardinality="1"
         rdfs:label="main_topic_interest">
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;main_topic_of_interest"
         a:allowedValues="Tibet"
         a:minCardinality="1"
         a:range="symbol"
         rdfs:label="main_topic_of_interest">
        <rdfs:domain rdf:resource="&kb;Community_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <a:allowedValues>Architecture</a:allowedValues>
        <a:allowedValues>Art</a:allowedValues>
        <a:allowedValues>Central_Asia</a:allowedValues>
        <a:allowedValues>Geography</a:allowedValues>
        <a:allowedValues>History</a:allowedValues>
        <a:allowedValues>Landscape</a:allowedValues>
        <a:allowedValues>Painting</a:allowedValues>
        <a:allowedValues>Religion</a:allowedValues>
</rdf:Property>
```

Figure B.3: RDF Schema 3/4

```
<rdf:Property rdf:about="&kb;name"
         a:maxCardinality="1"
         a:minCardinality="1"
         rdfs:label="name">
        <rdfs:domain rdf:resource="&kb;Community_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;other_topic_of_interest"
         a:allowedValues="SIlk_market"
         a:range="symbol"
         rdfs:label="other_topic_of_interest">
        <rdfs:domain rdf:resource="&kb;Community_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <a:allowedValues>Humain_evolution</a:allowedValues>
        <a:allowedValues>Landscape</a:allowedValues>
        <a:allowedValues>Mandala</a:allowedValues>
        <a:allowedValues>Medical_Plants</a:allowedValues>
</rdf:Property>
<rdf:Property rdf:about="&kb;processor_frequency"
         a:defaultValues="860_MHz"
         a:maxCardinality="1"
         a:minCardinality="1"
         a:range="symbol"
         a:values="860_MHz"
         rdfs:label="processor_frequency">
        <rdfs:domain rdf:resource="&kb;Device_profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <a:allowedValues>1_GHz</a:allowedValues>
        <a:allowedValues>860_MHz</a:allowedValues>
</rdf:Property>
<rdf:Property rdf:about="&kb;screen_resolution"
         a:defaultValues="1024x768"
         a:maxCardinality="1"
         a:range="symbol"
         a:values="1024x768"
         rdfs:label="screen_resolution">
        <rdfs:domain rdf:resource="&kb;Device_profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <a:allowedValues>1024x768</a:allowedValues>
        <a:allowedValues>1280x1024</a:allowedValues>
        <a:allowedValues>1280x768</a:allowedValues>
        <a:allowedValues>1600x1200</a:allowedValues>
        <a:allowedValues>1902x1200</a:allowedValues>
        <a:allowedValues>800x600</a:allowedValues>
</rdf:Property>
<rdf:Property rdf:about="&kb;user_involved"
         rdfs:label="user_involved">
        <rdfs:domain rdf:resource="&kb;Community_Profile"/>
        <rdfs:range rdf:resource="&kb;User_Profile"/>
        <a:inverseProperty rdf:resource="&kb;inverse_of_user_involved"/>
</rdf:Property>
<rdf:Property rdf:about="&kb;usual_location"
         a:maxCardinality="1"
         rdfs:label="usual_location">
        <rdfs:domain rdf:resource="&kb;User_Profile"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
</rdf:RDF>
```

Figure B.4: RDF Schema 4/4

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!DOCTYPE rdf:RDF [
         <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
         <!ENTITY a 'http://protege.stanford.edu/system#'>
         <!ENTITY normes 'http://protege.stanford.edu/normes#'>
         <!ENTITY rdfs 'http://www.w3.org/TR/1999/PR-rdf-schema-19990303#'>
]>
<rdf:RDF xmlns:rdf="&rdf;" xmlns:a="&a;" xmlns:normes="&normes;" xmlns:rdfs="&rdfs;">
<normes:DCM1_Type rdf:about="&normes;normes.1_00923"
         normes:DC.Type.DCM1_type_name="Interactive ressource"
         rdfs:label="Interactive ressource">
        <normes:DC.Type.DCM1_type_definition>An interactive resource is a resource which requires
        interaction from the user to be understood, executed, or experienced. For example - forms
        on web pages, applets, multimedia learning objects, chat services, virtual reality.
</normes:DC.Type.DCM1_type_definition>
</normes:DCM1_Type>
<normes:DCM1_Type rdf:about="&normes;normes.1_00924"
         normes:DC.Type.DCM1_type_name="Service"
         rdfs:label="Service">
        <normes:DC.Type.DCM1_type_definition>A service is a system that provides one or more
        functions of value to the end-user. Examples include: a photocopying service, a banking
        service, an authentication service, interlibrary loans, a Z39.50 or Web server.
</normes:DC.Type.DCM1_type_definition>
</normes:DCM1_Type>
<normes:DCM1_Type rdf:about="&normes;normes.1_00926"
         normes:DC.Type.DCM1_type_name="Sound"
         rdfs:label="Sound">
        <normes:DC.Type.DCM1_type_definition>A sound is a resource whose content is primarily
        intended to be rendered as audio. For example - a music playback file format, an audio
        compact disc, and recorded speech or sounds.
</normes:DC.Type.DCM1_type_definition>
</normes:DCM1_Type>
<normes:DCM1_Type rdf:about="&normes;normes.1_00928"
         normes:DC.Type.DCM1_type_name="Physical object"
         rdfs:label="Physical object">
        <normes:DC.Type.DCM1_type_definition>An inanimate, three-dimensional object or substance.
        For example -- a computer, the great pyramid, a sculpture.  Note that digital representations
        of, or surrogates for, these things should use Image, Text or one of the other types.
</normes:DC.Type.DCM1_type_definition>
</normes:DCM1_Type>
<normes:DCQ.date rdf:about="&normes;normes.1_01127"
         normes:DCQ.definition="date of creation of the resource."
         normes:DCQ.name="Created"
         rdfs:label="Created"/>
<normes:DCQ.date rdf:about="&normes;normes.1_01128"
         normes:DCQ.definition="date of formal issuance (e.g: publication) of the resource."
         normes:DCQ.name="Issued"
         rdfs:label="Issued"/>
<normes:DDC rdf:about="&normes;normes.1_01159"
         normes:ddc.code="720"
         normes:ddc.sujet="Architecture"
         rdfs:label="Architecture"/>
<normes:DDC rdf:about="&normes;normes.1_01163"
         normes:ddc.code="755"
         normes:ddc.sujet="Religion and religion symbolism"
         rdfs:label="Religion and religion symbolism"/>
<normes:DDC rdf:about="&normes;normes.1_01165"
         normes:ddc.code="779"
         normes:ddc.sujet="Photographs"
         rdfs:label="Photographs"/>
<normes:CDWA.Classification rdf:about="&normes;normes.1_01208"
         normes:Classification.term="Painting"
         rdfs:label="Painting"/>
</rdf:RDF>
```

Figure B.5: RDF representation of RCT (extract)

Bibliography

[AAO03] Elham Andaroodi, Frédéric Andrès, Kinji Ono, and Pierre Lebigre. "Ontology for caravanserais of Silk Roads: Needs, Processes, Constraints." In *Proc. of the Nara Symposium for Digital Silk Roads*, pp. 361–367, Nara, Japan, December 10-12 2003.

[ABC04] Serge Abiteboul, Omar Benjelloun, Bogdan Cautis, Ioana Manolescu, Tova Milo, and Nicoleta Preda. "Lazy Query Evaluation for Active XML." In *Proc. of SIGMOD*, pp. 227–238, Paris, France, June 13-18 2004.

[Abi03] Serge Abiteboul. "Managing an XML Warehouse in a P2P Context." In *Proc. of CAiSE*, pp. 4–13, Klagenfurt, Austria, June 16-18 2003.

[ABO96] Frédéric Andrès, Jihad Boulos, and Kinji Ono. "Accessing Active Application-oriented DBMS from the World Wide Web." In *Proc. of CODAS*, pp. 171–173, Kyoto, Japan, December 5-7 1996.

[AEM03] Neal Arthorne, Barbak Esfandiari, and Aloke Mukherjee. "U-P2P: A Peer-to-Peer Framework for Universal Resource Sharing and Discovery." In *Proc. of the FREENIX Track: USENIX Annual Technical Conference*, pp. 29–38, San Antonio, Texas, USA, June 9-14 2003.

[AGO04] Frédéric Andrès, Jérôme Godard, and Kinji Ono. "ASPICO: Advanced Scientific Portal for International Cooperation on Digital Cultural Content." In *Proc. of ICT&P*, pp. 190–199, Varna, Bulgaria, June 24-26 2004.

[AK00] Masatoshi Arikawa and Koichi Kubota. "A Standard XML Based Protocol for Spatial Data Exchange - Its Capabilities and Real Applications." In *Proc. of International workshop on Emerging technologies for geo-based applications (Invited Paper)*, pp. 37–45, Ascona, Switzerland, May 21-26 2000.

[AO98a] Frédéric Andrès and Kinji Ono. "Active Hypermedia Delivery System (AHYDS) using the Phasme Application-Oriented DBMS." In *Proc. of ICDE*, p. 600, Florida, USA, February 23-27 1998.

[AO98b] Frédéric Andrès and Kinji Ono. "Phasme: A High Performance Parallel Application-oriented DBMS." *Informatica Journal, Special Issue on Parallel and Distributed Database Systems*, (22):167–177, May 1998.

[AO01] Frédéric Andrès and Kinji Ono. "The Distributed Management Mechanism of the Active Hypermedia Delivery System platform." *Trans. on IEICE*, **E84-D**(8):1033–1038, August 2001.

[AOS00] Frédéric Andrès, Kinji Ono, Shin'ichi Satoh, and Nicolas Dessaigne. "Toward The MediaSys Video Search Engine (MEVISE)." In *Proc. of VDB5*, pp. 31–44, Fukuoka, Japan, May 10-12 2000.

[AOT01] Frédéric Andrès, Kinji Ono, and Hideaki Takeda. "Multimedia Device Information Engine for Symbiotic Applications." *NII Journal*, (3):45–52, November 2001.

[Bay96] Rudolf Bayer. “The Universal B-Tree for Multidimensional Indexing.” Technical report, Technische Universität München, Munich, Germany, November 1996.

[BBC02] Matteo Bonifacio, Paolo Bouquet, and Roberta Cuel. “The Role of Classification(s) in Distributed Knowledge Management.” In *Proc. of KES*, Podere d’Ombriano, Italy, September 16-18 2002.

[BBG01] Massimo Benerecetti, Paolo Bouquet, and Chiara Ghidini. “On the Dimensions of Context Dependence: Partiality, Approximation, and Perspective.” In *Proc. of CONTEXT*, pp. 59–72, Dundee, UK, July 27-30 2001.

[BBM02] Matteo Bonifacio, Paolo Bouquet, Gianluca Mameli, and Michele Nori. “KEx: A Peer-to-Peer Solution for Distributed Knowledge Management.” In *Proc. of PAKM*, pp. 490–500, Vienna, Austria, December 2-3 2002.

[BD03] Louise Barkhuus and Anind K. Dey. “Is Context-Aware Computing Taking Control away from the User? Three Levels of Interactivity Examined.” In *Proc. of Ubicomp*, pp. 149–156, Seattle, WA, USA, October 12-15 2003.

[BL01] Travis Bauer and David B. Leake. “Real Time User Context Modeling for Information Retrieval Agents.” In *Proc. of CIKM*, pp. 568–570, Atlanta, Georgia, USA, November 5-10 2001.

[BM72] Rudolf Bayer and Edward McCreight. “Organization and Maintenance of Large Ordered Indices.” *Acta Informatica*, **1**(3):173–189, 1972.

[BMN03] Boualem Benatallah, Mehregan Mahdavi, Phuong Nguyen, Quan Z. Sheng, Lionel Port, and Bill McIver. “An Adaptive Document Version Management Scheme.” In *Proc. of CAiSE*, pp. 46–62, Klagenfurt, Austria, June 16-18 2003.

[BRB02] Michael G. Bauer, Frank Ramsak, and Rudolf Bayer. “Indexing XML as a Multidimensional Problem.” Technical Report I0203, Technische Universität München, Munich, Germany, May 2002.

[BS01] Paolo Bouquet and Luciano Serafini. “Two Formalizations of Context: A Comparison.” In *Proc. of CONTEXT*, pp. 87–101, Dundee, UK, July 27-30 2001.

[CG04] Augusto Celentano and Ombretta Gaggi. “A Context-Aware Framework For Multimodal Document Databases.” In *Proc. of MDIC*, Salerno, Italy, June 22 2004.

[CLM03] Giacomo Cabri, Letizia Leonardi, Marco Mamei, and Franco Zambonelli. “Location-dependent Services for Mobile Users.” *IEEE Transactions on Systems, Man, and Cybernetics*, **33**(6):667–681, November 2003.

[CMM03] Thomas Chau, Frank Maurer, and Grigori Melnik. “Knowledge Sharing: Agile Methods vs. Tayloristic Methods.” In *Proc. of WETICE*, pp. 302–307, Linz, Austria, June 9-11 2003.

[CTZ97] Stavros Christodoulakis, Peter Triantafillou, and Fenia Zioga. “Principles of Optimally Placing Data in Tertiary Storage Libraries.” In *Proc. of VLDB*, pp. 236–245, Athens, Greece, August 25-29 1997.

[CTZ01] Shu-Yao Chien, Vassilis J. Tsotras, and Carlo Zaniolo. “XML Document Versioning.” *SIGMOD Record*, **30**(3):46–53, September 2001.

[DG02] Schahram Dustdar and Harald Gall. “Towards a Software Architecture for Distributed and Mobile Collaborative Systems.” In *Proc. of COMPSAC*, pp. 674–679, Oxford, England, August 26-29 2002.

[Doe03] Martin Doerr. “The CIDOC Conceptual Reference Module: An Ontological Approach to Semantic Interoperability of Metadata.” *AI Magazine*, **24**(3):75–92, Fall 2003.

[DRF04] Yanlei Diao, Shariq Rizvi, and Michael J. Franklin. “Towards an Internet-Scale XML Dissemination Service.” In *Proc. of VLDB*, pp. 612–623, Toronto, Canada, August 31 - September 3 2004.

[FB04] Willem Fontijn and Peter A. Boncz. “AmbientDB: P2P Data Management Middleware for Ambient Intelligence.” In *Proc. of PerCom Workshops*, pp. 203–207, Orlando, FL, USA, March 14-17 2004.

[FDK02] Pascal Fenkam, Schahram Dustdar, Engin Kirda, Gerald Reif, and Harald Gall. “Towards an Access Control System for Mobile Peer-to-Peer Collaborative Environments.” In *Proc. of WETICE*, pp. 95–102, Pittsburgh, Pennsylvania, USA, June 10-12 2002.

[Fit99] Tom Fitzpatrick. *Open Component-Oriented Multimedia Middleware for Adaptive Distributed Applications*. PhD thesis, Computing Department Lancaster University, September 1999.

[FJ02] Manuel J. Fonseca and Joaquim A. Jorge. “Towards Content-Based Retrieval of Technical Drawings through High-Dimensional Indexing.” In *Proc. of SIACG*, pp. 263–270, Guimaraes, Portugal, July 1-5 2002.

[FK99] Daniela Florescu and Donald Kossmann. “Storing and Querying XML Data using an RDMBS.” *IEEE Data Engineering Bulletin*, **22**(3):27–34, 1999.

[Fre95] Michael Freeston. “A General Solution of the n-dimensional B-tree Problem.” In *Proc. of ACM SIGMOD*, pp. 80–91, San Jose, California, USA, May 22-25 1995.

[GAA04] Jérôme Godard, Frédéric Andrès, Elham Andaroodi, and Katsumi Maruyama. “Towards a Service-oriented Architecture for Collaborative Management of Heterogeneous Cultural Resources.” In *Pre-Proc. of DELOS Workshop on Digital Library Architectures*, pp. 183–194, S. Margherita di Pula, Italy, June 24-25 2004.

[GAG04a] Jérôme Godard, Frédéric Andrès, William Grosky, and Kinji Ono. “Knowledge Management Framework for the Collaborative Distribution of Information.” In *Proc. of DataX (unformal Proceedings)*, pp. 2–16, Heraklion, Greece, March 14 2004.

[GAG04b] Jérôme Godard, Frédéric Andrès, William Grosky, and Kinji Ono. “Knowledge Management Framework for the Collaborative Distribution of Information.” In *Current Trends in Database Technology - EDBT 2004 Workshops (Revised Selected Papers), LNCS 3268*, pp. 289–298, Heraklion, Greece, March 14 2004.

[GAO02] Jérôme Godard, Frédéric Andrès, and Kinji Ono. “Advanced Storage and Retrieval of XML Multimedia Documents.” In *Proc. of DNIS*, LNCS 2544, pp. 64–73, Aizu, Japan, December 16-18 2002.

[GAO03] Jérôme Godard, Frédéric Andrès, and Kinji Ono. “Management of Cultural Information: Indexing Strategies for Context-dependent Resources.” In *Proc. of the Nara Symposium for Digital Silk Roads*, pp. 369–374, Nara, Japan, December 10-12 2003.

[GG98] Volker Gaede and Oliver Günther. “Multidimensional access methods.” *ACM Computing Surveys*, **30**(2):170–231, 1998.

[GGL03] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. “The Google File System.” In *Proc. of ACM SOSP*, pp. 29–43, Bolton Landing, USA, October 19-22 2003.

[GHI01] Steven D. Gribble, Alon Y. Halevy, Zachary G. Ives, Maya Rodrig, and Dan Suciu. “What Can Database Do for Peer-to-Peer?” In *Proc. of WebDB*, pp. 31–36, Santa Barbara, California, USA, May 24-25 2001.

[Gib85] Alan Gibbons. *Algorithmic Graph Theory*. Cambridge University Press, 1985.

[GMA02] Jérôme Godard, Mathieu Mangeot-Lerebours, and Frédéric Andrès. "Data Repository Organization and Recuperation Process for Multilingual Lexical Databases." In *Proc. of SNLP-Oriental COCOSDA*, pp. 249–254, Hua Hin, Prachuapkirikhan, Thailand, May 9-11 2002.

[GNO92] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. "Using Collaborative Filtering to Weave an Information Tapestry." *Commun. ACM*, **35**(12):61–70, December 1992.

[Gra04] Jim Gray. "The Revolution in Database Architecture." Technical Report MSR-TR-2004-31, Microsoft Research, Redmond, WA, USA, March 2004.

[GS03] Manolis Gergatsoulis and Yannis Stavrakas. "Representing Changes in XML Documents using Dimensions." In *Proc. of XSym*, pp. 208–222, Berlin, Germany, September 8 2003.

[GSB04] Vijay Gopalakrishnan, Bujor Silaghi, Bobby Bhattacharjee, and Pete Keleher. "Adaptive Replication in Peer-to-Peer Systems." In *Proc. of ICDCS*, pp. 360–369, Hachioji, Japan, March 24-26 2004.

[GSK01] Manolis Gergatsoulis, Yannis Stavrakas, and Dimitris Karteris. "Incorporating Dimensions in XML and DTD." In *Proc. of DEXA*, pp. 646–656, Munich, Germany, September 3-5 2001.

[GTB02] Matthias Grimm, Mohammed-Reza Tazari, and Dirk Balfanz. "Towards a Framework for Mobile Knowledge Management." In *Proc. of PAKM*, pp. 326–338, Vienna, Austria, December 2-3 2002.

[GTW78] Joseph A. Goguen, Jim W. Thatcher, and Eric G. Wagner. "An Initial Algebra Approach to the Specification, Correctness, and Implementation of Abstract Data Types." In R. Yeh, editor, *Current Trends in Programming Methodology*, volume Data Structuring, pp. 80–149, 1978.

[Gut84] Antonin Guttman. "R-trees: A Dynamic Index Structure for Spatial Searching." In *Proc. of ACM SIGMOD*, pp. 47–57, Boston, Massachusetts, USA, June 18-21 1984.

[Gut89] Ralf Hartmut Güting. "Gral: An Extensible Relational Database System for Geometric Applications." In *Proc. of VLDB*, pp. 33–44, Amsterdam, The Netherlands, August 22-25 1989.

[HAB01] Koiti Hasida, Frédéric Andrès, Christian Boitet, N. Calzolari, T. Declerck, Farshad Fotouhi, William Grosky, Shun Ishizaki, Asanee Kawtrakul, Mathieu Lafourcade, Katashi Nagao, Hammam Riza, Virach Sornlertlamvanich, Remi Zajac, and A. Zampolli. "Linguistic DS." ISO/IEC JTC1/SC29/WG11, MPEG2001/ M7818, 2001.

[HC03] Christopher K. Hess and Roy H. Campbell. "A Context-Aware Data Management System for Ubiquitous Computing Applications." In *Proc. of ICDCS*, pp. 294–301, Providence, USA, May 19-22 2003.

[HRS99] Bruce Hillyer, Rajeev Rastogi, and Abraham Silberschatz. "Scheduling and Data Replication to Improve Tape Jukebox Performance." In *Proc. of ICDE*, pp. 532–541, Sydney, Australia, March 23-26 1999.

[HV02] Yun Huang and Nalini Venkatasubramanian. "Data Placement in Intermittently Available Environments." In *Proc. of HiPC*, pp. 367–376, Bangalore, India, December 18-21 2002.

[JCS04] Rong Jin, Joyce Y. Chai, and Luo Si. "An Automatic Weighting Scheme for Collaborative Filtering." In *Proc. of SIGIR*, pp. 337–344, Sheffield, UK, July 25-29 2004.

[KH98] Thomas Klement and Matthias Hemmje. "Metadata for Multidimensional Categorization and Navigation Support on Multimedia Documents." In *Proc. of ERCIM Database Research Group Workshop on Metadata for Web Databases*, Sankt Augustin, Germany, May 25-26 1998.

[KK04] Magnus Karlsson and Christos Karamanolis. "Choosing Replica Placement Heuristics for Wide-Area Systems." In *Proc. of ICDCS*, pp. 350–359, Tokyo, Japan, March 23-26 2004.

[KL03] Tevfik Kosar and Miron Livny. "Scheduling Data Placement Activities in Grid." Technical Report 1483, Computer Sciences Department, University of Wisconsin, Wisconsin, USA, July 2003.

[KL04] Tevfik Kosar and Miron Livny. "Stork: Making Data Placement a First Class Citizen in the Grid." In *Proc. of ICDCS*, pp. 342–349, Tokyo, Japan, March 24-26 2004.

[KM00] Carl-Christian Kanne and Guido Moerkotte. "Efficient Storage of XML Data." In *Proc. of ICDE*, p. 198, San Diego, California, USA, February 28 - March 03 2000.

[KM03] Panu Korpipää and Jani Mäntyjärvi. "An Ontology for Mobile Device Sensor-Based Context Awareness." In *Proc. of CONTEXT*, pp. 451–458, Stanford, CA, USA, June 23-25 2003.

[KS04] Jon M. Kleinberg and Mark Sandler. "Using mixture models for collaborative filtering." In *Proc. of STOC*, pp. 569–578, Chicago, IL, USA, June 13-16 2004.

[Kun90] Hideko S. Kunii. *Graph Data Model and Its Data Language*. Springer-Verlag, 1990.

[LF04] Patrick Lehti and Peter Fankhauser. "XML Data Integration with OWL: Experiences and Challenges." In *Proc. of SAINT*, pp. 160–170, Tokyo, Japan, January 26-30 2004.

[LH04] Hsiung-Peng Liao and Jung-Hong Hong. "Map Interface Valid Coverage Analysis Based on XML Metadata." In *Proc. of Geoinformatics*, pp. 812–819, Gävle, Sweden, June 7-9 2004.

[LM01] Quanzhong Li and Bongki Moon. "Indexing and Querying XML Data for Regular Path Expressions." In *Proc. of VLDB*, pp. 361–370, Roma, Italy, September 11-14 2001.

[Loe92] Shoshana Loeb. "Architecting Personalized Delivery of Multimedia Information." *Communications of the ACM*, **35**(12), December 1992.

[LOP02] Yugyung Lee, Changgyu Oh, and Eun Kyo Park. "Intelligent Knowledge Discovery in Peer-to-Peer File Sharing." In *Proc. of CIKM*, pp. 308–315, McLean, Virginia, USA, November 4-9 2002.

[LS00] Hartmut Liefke and Dan Suciu. "XMill: an efficient compressor for XML data." In *Proc. of ACM SIGMOD*, pp. 153–164, Dallas, Texas, USA, May 16-18 2000.

[MEA02] Aloke Mukherjee, Babak Esfandiari, and Neal Arthorne. "U-P2P: A Peer-to-Peer System for Description and Discovery of Resource-Sharing Communities." In *Proc. of ICDCS Workshops*, pp. 701–705, Vienna, Austria, July 2-5 2002.

[Mei02] Wolfgang Meier. "eXist: An Open Source Native XML Database." In *Proc. NODe Web- and Database-Related Workshops*, pp. 169–183, Erfurt, Germany, October 7-10 2002.

[MGR02] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. "Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching." In *Proc. of ICDE*, pp. 117–128, San Jose, California, USA, February 26 - March 01 2002.

[MGS01] Theodoros Mitakos, Manolis Gergatsoulis, Yannis Stavrakas, and Efstathios V. Ioannidis. "Representing Time-Dependent Information in Multidimensional XML." In *Proc. of ITI*, pp. 111–116, Pula, Croatia, June 19-22 2001.

[MPC03] Jun-Ki Min, Myung-Jae Park, and Chin-Wan Chung. “XPRESS: A Queriable Compression for XML Data.” In *Proc. of ACM SIGMOD*, pp. 122–133, San Diego, California, USA, June 10-12 2003.

[MPR00] Udi Manber, Ash Patel, and John Robison. “Experience with Personalization of Yahoo!” *Commun. ACM*, **43**(8):35–39, August 2000.

[MRU90] O. Marino, François Rechenmann, and P. Uvietta. “Multiple Perspectives and Classification Mechanism in Object-Oriented Representation.” In *Proc. of ECAI*, pp. 425–430, Stockholm, Sweden, 1990.

[MSD04] Stuart E. Middleton, Nigel Shadbolt, and David De Roure. “Ontological User Profiling in Recommender Systems.” *ACM Trans. Inf. Syst.*, **22**(1):54–88, January 2004.

[MWA98] Jason McHugh, Jennifer Widom, Serge Abiteboul, Qingshan Luo, and Anand Rajaraman. “Indexing Semistructured Data.” Technical report, Stanford University, Computer Science Department, 1998.

[MZL01] Kurt Maly, Mohammad Zubair, and Xiaoming Liu. “Kepler - An OAI Data/Service Provider for the Individual.” *D-Lib Magazine*, **7**(4), April 2001.

[Ono01] Kinji Ono, editor. *Proceedings of the Tokyo Symposium for Digital Silk Roads*, Tokyo, Japan, December 11-13 2001. UNESCO & National Institute of Informatics.

[Ono03] Kinji Ono, editor. *Proceedings of the Nara Symposium for Digital Silk Roads*, Tokyo, Japan, December 10-12 2003. UNESCO & National Institute of Informatics.

[OQ97] Patrick O’Neill and Dallan Quass. “Improved Query Performance with variant Indexes.” In *Proc. of ACM SIGMOD*, pp. 38–49, Tucson, Arizona, USA, May 13-15 1997.

[OSS01] Vincent Oria, Amit Shah, and Samuel Sowell. “Indexing XML Documents: Improving the BUS Method.” In *Proc. of MIS*, pp. 89–98, Capri, Italy, November 7-9 2001.

[PAP03] Evaggelia Pitoura, Serge Abiteboul, Dieter Pfoser, George Samaras, and Michalis Vazirgiannis. “DBGlobe: a service-oriented P2P system for global computing.” *SIGMOD Record*, **32**(3):77–82, September 2003.

[PHL00] David M. Pennock, Eric Horvitz, Steve Lawrence, and C. Lee Giles. “Collaborative Filtering by Personality Diagnosis: A Hybrid Memory and Model-Based Approach.” In *Proc. of UAI*, pp. 473–480, Stanford, California, USA, June 30 - July 3 2000.

[PJF04] Filip Perich, Anupam Joshi, Timothy W. Finin, and Yelena Yesha. “On Data Management in Pervasive Computing Environments.” *Trans. Knowl. Data Eng.*, **16**(5):621–634, May 2004.

[PSS04] Helena Sofia Pinto, Steffen Staab, York Sure, and Christoph Tempich. “OntoEdit Empowering SWAP: a Case Study in Supporting DIstributed, Loosely-Controlled and evolvInG Engineering of oNTologies (DILIGENT).” In *Proc. of ESWS*, pp. 16–30, Heraklion, Greece, May 10-12 2004.

[RD02] Myriam Ribière and Rose Dieng-Kuntz. “A Viewpoint Model for Cooperative Building of an Ontology.” In *Proc. of ICCS*, pp. 220–234, Borovets, Bulgaria, July 15-19 2002.

[RIS94] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. “GroupLens: An Open Architecture for Collaborative Filtering of Netnews.” In *Proc. of CSCW*, pp. 175–186, Chapel Hill, NC, USA, October 22-26 1994.

[RJR02] Govindan Ravindran, Muhammad Jaseemudin, and Abdallah Rayhan. “A Management Framework for Service Personalization.” In *Proc. of MMNS*, pp. 276–288, Santa Barbara, CA, USA, October 6-9 2002.

[RL03] Maya Rodrig and Anthony LaMarca. “Decentralized weighted voting for P2P data management.” In *Proc. of MobiDE*, pp. 85–92, San Diego, California, USA, September 19 2003.

[RMF00] Frank Ramsak, Volker Markl, Robert Fenk, Martin Zirkel, Klaus Elhardt, and Rudolf Bayer. “Integrating the UB-Tree into a Database System Kernel.” In *Proc. of VLDB*, pp. 263–272, Cairo, Egypt, Sept. 10-14 2000.

[RP02] Kanda Runapongsa and Jignesh M. Patel. “Storing and Querying XML Data in Object-Relational DBMSs.” In *Proc. of EDBT Workshops*, pp. 266–285, Prague, Czech Republic, March 24-28 2002.

[SAL96] Michael Stonebraker, Paul M. Aoki, Witold Litwin, Avi Pfeffer, Adam Sah, Jeff Sidell, Carl Staelin, and Andrew Yu. “Mariposa: A Wide-Area Distributed Database System.” *VLDB J.*, **5**(1):48–63, January 1996.

[SB88] Gerard Salton and Chris Buckley. “Term-Weighting Approaches in Automatic Text Retrieval.” *Information Processing and Management*, **24**(5):513–523, 1988.

[SCH00] Edith Schonberg, Thomas Cofino, Robert Hoch, Mark Podlaseck, and Susan L. Spraragen. “The Human Element: Measuring Success.” *Commun. ACM*, **43**(8):53–57, 2000.

[Sch01] Harald Schöning. “Tamino - A DBMS designed for XML.” In *Proc. of ICDE*, pp. 149–154, Heidelberg, Germany, April 2-6 2001.

[Sei98] Robert Seidman. “Personally, On Personalization.” *Online Insider*, **5**(24), 1998.

[SGA02] D.V. Sreenath, William Grosky, and Frédéric Andrès. *Intelligent Virtual Worlds: Technologies and Applications in Distributed Virtual Environments*, chapter Metadata-Mediated Browsing and Retrieval in a Cultural Heritage Image Collection. World Scientific Publishing Company, Singapore, 2002.

[SGM00] Yannis Stavrakas, Manolis Gergatsoulis, and Theodoros Mitakos. “Representing Context-Dependent Information Using Multidimensional XML.” In *Proc. of ECDL*, pp. 368–371, Lisbon, Portugal, Sept. 18-20 2000.

[SGR00] Yannis Stavrakas, Manolis Gergatsoulis, and Panos Rondogiannis. “Multidimensional XML.” In *Proc. of DCW*, pp. 100–109, Quebec City, Canada, June 19-21 2000.

[Shi01] Dongwook Shin. “XML Indexing and Retrieval with a Hybrid Storage Model.” *Knowledge and Information Systems*, **3**(2):252–261, May 2001.

[SHY04] Kazunari Sugiyama, Kenji Hatano, Masatoshi Yoshikawa, and Shunsuke Uemura. “User-Oriented Adaptive Web Information Retrieval Based on Implicit Observations.” In *Proc. of APWeb*, pp. 636–643, Hangzhou, China, April 14-17 2004.

[Sil00] Chuck Silvers. “UBC: An Efficient Unified I/O and Memory Caching Subsystem for NetBSD.” In *Proc. of Freenix*, pp. 285–290, San Diego, USA, June 18-23 2000.

[SJJ98] Dongwook Shin, Hyuncheol Jang, and Honglan Jin. “BUS: An Effective Indexing and Retrieval Scheme in Structured Documents.” In *Proc. of ACM DL*, pp. 235–243, Pittsburgh, PA, USA, June 23-26 1998.

[SKW00] Albrecht Schmidt, Martin Kersten, Menzo Windhouwer, and Florian Waas. “Efficient Relational Storage and Retrieval of XML Documents.” In *Proc. of WebDB*, pp. 137–150, Dallas, Texas, USA, May 18-19 2000.

[SRF87] Timos K. Sellis, Nick Roussopoulos, and Christos Faloutsos. “The R^+-tree: A Dynamic Index for Multidimensional Objects.” In *Proc. of VLDB*, pp. 3–11, Brighton, England, September 1-4 1987.

[STZ99] Jayavel Shanmugasundaram, Kristin Tufte, Chun Zhang, Gang He, David J. DeWitt, and Jeffrey F. Naughton. “Relational Databases for Querying XML Documents: Limitations and Opportunities.” In *Proc. of VLDB*, pp. 302–314, Edinburgh, Scotland, September 7-10 1999.

[SYU99] Takeyuki Shimura, Masatoshi Yoshikawa, and Shunsuke Uemura. "Storage and Retrieval of XML Documents Using Object-Relational Databases." In *Proc. of DEXA*, pp. 206–217, Florence, Italy, August 30 - September 3 1999.

[TFC03] Chrisa Tsinaraki, Eleni Fatourou, and Stavros Christodoulakis. "An Ontology-Driven Framework for the Management of Semantic Metadata Describing Audiovisual Information." In *Proc. of CAiSE*, pp. 340–356, Klagenfurt, Austria, June 16-18 2003.

[Th89] Jean-Marc Thévenin. *Architecture d'un Système de Gestion de Bases de Donnees Grande Mémoire*. PhD thesis, Paris VI University, December 20 1989.

[Th02a] David Thévenin. "Multi-Access User Interface for Papillon." In *Proc. of Papillon Workshop*, p. 8, Tokyo, Japan, July 16-18 2002.

[TH02b] Pankaj M. Tolani and Jayant R. Haritsa. "XGRIND: A Query-Friendly XML Compressor." In *Proc. of ICDE*, pp. 225–234, San Jose, California, USA, February 26 - March 01 2002.

[TR03] Dimitrios Tsoumakos and Nick Roussopoulos. "A Comparison of Peer-to-Peer Search Methods." In *Proc. of WebDB*, pp. 61–66, San Diego, California, USA, June 12-13 2003.

[TSW04] Christoph Tempich, Steffen Staab, and Adrian Wranik. "Remindin': semantic query routing in peer-to-peer networks based on social metaphors." In *Proc. of WWW*, pp. 640–649, NY, USA, May 17-20 2004.

[UKT03] Armin Ulbrich, Dolly Kandpal, and Klaus Tochtermann. "Dynamic Personalization in Knowledge-Based Systems from a Structural Viewpoint." In *Proc. of MIS*, pp. 126–142, Graz, Austria, September 17-20 2003.

[Ull88] Jeffrey D. Ullman. *Principles of Database and Knowledge-Base Systems*, volume I. Computer Science Press, 1988.

[VBH03] Richard Vdovjak, Peter Barna, and Geert-Jan Houben. "Designing a Federated Multimedia Information System on the Semantic Web." In *Proc. of CAiSE*, pp. 357–373, Klagenfurt, Austria, June 16-18 2003.

[VKC86] Patrick Valduriez, Setrag Khoshafian, and George P. Copeland. "Implementation Techniques of Complex Objects." In *Proc. of VLDB*, pp. 101–110, Kyoto, Japan, August 25-28 1986.

[VRT01] Alistair Veitch, Erik Riedel, Simon Towers, and John Wilkes. "Towards Global Storage Management and Data Placement." Technical Report HPL-SSP-2001-1, Hewlett-Packard Labs, Palo Alto, USA, March 2001.

[WL04] Kan Hung Wan and Chris Loeser. "An Overlay Network Architecture for Data Placement Strategies in a P2P Streaming Network." In *Proc. of AINA (1)*, pp. 119–125, Fukuoka, Japan, March 29-31 2004.

[WLO01] Raymond K. Wong, Franky Lam, and Mehmet A. Orgun. "Modelling and Manipulating Multidimensional Data in Semistructured Databases." In *Proc. of DASFAA*, pp. 14–21, Hong Kong, China, April 18-20 2001.

[WZW85] S. K. Michael Wong, Wojciech Ziarko, and P. C. N. Wong. "Generalized Vector Space Model in Information Retrieval." In *Proc. of SIGIR*, pp. 18–25, Montreal, Canada, 1985.

[YIN00] Yasuo Yamane, Nobuyuki Igata, and Isao Namba. "High-performance XML Storage/Retrieval System." *Fujitsu Sci. & Tech. J.*, **36**(2):185–192, December 2000.

[ZL04] Cai-Nicolas Ziegler and Georg Lausen. "Analyzing Correlation between Trust and User Similarity in Online Communities." In *Proc. of iTrust*, pp. 251–265, Oxford, UK, March 29 - April 1 2004.