



UNIVERSITÀ DEGLI STUDI DI PADOVA
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DOTTORATO DI RICERCA IN
INGEGNERIA INFORMATICA ED ELETTRONICA INDUSTRIALI
XVII CICLO

SEDE AMMINISTRATIVA: UNIVERSITÀ DEGLI STUDI DI PADOVA

**Formal Model and Conceptual
Architecture of the Annotation Service
for Dynamic Ubiquitous Knowledge
Environments**

COORDINATORE:

Ch.mo Prof. Concettina Guerra

SUPERVISORE:

Ch.mo Prof. Maristella Agosti

DOTTORANDO:

Ing. Nicola Ferro

Dicembre 2004

Sommario

Questa tesi riguarda lo studio delle tematiche e dei problemi inerenti la possibilità di apporre annotazioni su contenuti digitali, come documenti testuali, immagini e documenti multimediali in genere. Questi contenuti digitali sono gestiti in modo automatico da diversi sistemi di gestione di biblioteche digitali e, più in generale, da diversi tipi di sistemi di gestione delle informazioni.

Pur essendo una tematica in parte già affrontata, la ricerca sulle annotazioni presentava ancora molte problematiche aperte riguardanti la mancanza di chiarezza su cosa sia un'annotazione, su quali siano le sue caratteristiche e i suoi modi di impiego, e su quali siano l'architettura e le funzionalità che un servizio di annotazioni debba avere. Queste problematiche sono dovute, soprattutto, al fatto che fino ad oggi sono stati sviluppati sistemi e modelli di annotazione per obiettivi specifici, dando così origine ad una visione molto frammentaria dell'annotazione e delle sue modalità di gestione, visione legata a contesti molto particolari e priva di una valenza più generale.

Quindi, la tesi si pone come obiettivo il proporre una visione unitaria e integrata dell'annotazione, obiettivo che spazia dal definire cosa sia effettivamente un'annotazione, al fornire un modello formale per descrivere l'annotazione e al progettare l'architettura di un sistema capace di offrire funzionalità di annotazione sui documenti gestiti da diversi sistemi di gestione di biblioteche digitali e, più in generale, da diversi tipi di sistemi di gestione delle informazioni.

I principali risultati originali conseguiti nella tesi sono: studio e analisi dell'annotazione in una prospettiva sia storica che contemporanea volto a delineare e definire la reale complessità del problema, che è spesso sottostimato e solo parzialmente affrontato; definizione sia di un modello concettuale dell'annotazione che di un modello formale unitario dell'annotazione, modelli che non sono ancora presenti nell'ambito di ricerca sulle annotazioni; definizione e progetto di un'architettura flessibile e di validità generale per realizzare un servizio di annotazioni per diversi sistemi di gestione delle informazioni, architettura ancora oggi non presente nella letteratura sull'argomento.

Abstract

This thesis is a study of the themes and the issues concerning the annotation of digital contents, such as textual documents, images, and multimedia documents in general. These digital contents are automatically managed by different kinds of digital library systems and, more generally speaking, by different kinds of information management systems.

Even though this topic has already been partially studied, the previous research work on annotations has left us with many open issues. These issues concerned the lack of clarity about: what an annotation is, what the features and the way of using an annotation are, and what architecture and functionalities a system with annotation capabilities has to provide. These issues are mainly due to the fact that, up to now, models and systems for annotations have been developed for specific purposes. As a result of this, there is a fragmentary picture on the annotation and its management, which are tied to specific usages and lack a general validity.

The goal of the thesis is to provide a unified and integrated picture on the annotation. Thus, the thesis ranges from defining what an annotation is by providing a formal model of the annotation to designing the architecture of a system with annotation capabilities.

The major contributions of this thesis are the provision of the groundwork needed to properly delineate the complexity of the problem, which is often overlooked, and the definition of both a conceptual model and a unified formal model of the annotation, which is not touched upon in previous research work. Finally, the thesis designs the conceptual architecture of a system capable of providing different information management systems with annotation functionalities.

Acknowledgements

Sincere thanks are due to Massimo Melucci, Nicola Orio, and Luca Pretto for the time they spent with the author in discussing the aspects related to annotations and their automatic construction and their relationship with hypertexts. The author would also like to thank Lucio Benfante for his help on the architectural aspects of the annotation service.

The author wishes to thank Ulrich Thiel and Ingo Frommholz for the fruitful discussions about annotations and for sharing their ideas on these topics.

Finally, many thanks to Randy Carbone for his willingness in helping the author in reviewing the thesis as far as the English is concerned.

The work reported in this thesis has been partially funded by the ECD project, which is a joint program between the Italian National Research Council (CNR) and the Ministry of Education (MIUR), with regards to law 449/97-99. The work was also partially supported by the DELOS Network of Excellence on Digital Libraries, as part of the Information Society Technologies (IST) Program of the European Commission (Contract G038-507618).

Contents

Sommario	i
Abstract	iii
Acknowledgements	v
Contents	vii
List of Figures	ix
List of Acronyms	xi
1 Introduction	1
1.1 Structure of the Thesis	3
2 Annotations	5
2.1 Historical Viewpoints	5
2.1.1 The Term Annotation	6
2.1.2 Terms Related to Annotation	6
2.1.3 The Term Gloss	8
2.1.4 The Term Scholium	9
2.1.5 The Term Postil	9
2.2 Presents Perspectives	10
2.2.1 Digital Libraries	13
2.2.2 The Web	14
2.2.3 Databases	15
2.3 Key points	16
3 Methodological Approach	19
3.1 Architectural Approach	19
3.1.1 Handles and Identifiers	21
3.2 Modelling Approach	25
3.2.1 How to Model Annotations	25
3.2.2 How to Link Annotations to Digital Objects	27
3.2.3 Conceptual Model of the Annotation	28
4 Formal Model of Annotation	33
4.1 Document, Annotation, and Digital Object Sets	33
4.2 Handle	36

4.3	Author and Group of Authors	36
4.4	Stream	37
4.5	Segment	42
4.6	Sign of Annotation	43
4.7	Meaning of Annotation	43
4.8	Annotation	45
4.9	Document–Annotation Hypertext	50
5	Conceptual Architecture of the Flexible Annotation Service Tool	59
5.1	FAST Conceptual Architecture	59
5.2	Data Logic Layer	61
5.2.1	Annotation Storing Manager	61
5.2.2	Annotation Textual Indexing Manager	62
5.2.3	Annotation Abstraction Layer	64
5.3	Application Logic Layer	65
5.3.1	Automatic Annotation Manager	65
5.3.2	Information Retrieval On aNNotations	65
5.3.3	Annotation Service Integrator	66
5.3.4	Gateway	67
5.4	Interface Logic Layer	70
5.4.1	Administrative User Interface	71
5.4.2	Client User Interface	71
6	Conclusions and Future Work	73
6.1	Contributions	73
6.2	Future Work	74
	Bibliography	77

List of Figures

3.1	Overview of the architecture of FAST with respect to different IMSs. . .	20
3.2	Combination of basic signs.	26
3.3	Entity–Relationship schema for modelling annotations.	29
4.1	Example of document–annotation hypertext H_{da}	52
4.2	Not allowed annotations cycle.	53
4.3	Allowed annotations patterns.	54
4.4	Example of the H'_{da} subgraph, obtained from the document–annotation hypertext H_{da} of figure 4.1.	56
5.1	Detailed architecture of the FAST system.	60
5.2	Architecture of IRON.	62
5.3	Sequence diagram for creating annotations.	64
5.4	Sequence diagram for searching documents by exploiting annotations. .	67
5.5	Example of DoMDL document.	69
5.6	Entity–Relationship schema of figure 3.3 with respect to the mapping to DoMDL.	70
5.7	Example of DoMDL document with annotations.	71

List of Acronyms

3G	Third Generation Mobile System
AAL	Annotation Abstraction Layer
AAM	Automatic Annotation Manager
ADSL	Asymmetric Digital Subscriber Line
API	Application Program Interface
ASI	Annotation Service Integrator
ASM	Annotation Storing Manager
ATIM	Annotation Textual Indexing Manager
AUI	Administrative User Interface
CAS	Core Annotation Service
CLEF	Cross-Language Evaluation Forum
CP	Characteristic Pattern
CRUD	Create–Read–Update–Delete
CS	Characteristic Structure
CUI	Client User Interface
DAO	Data Access Object
DAS	Distributed Annotation System
DBMS	DataBase Management System
DCMI	Dublin Core Metadata Initiative
DL	Digital Library
DLSS	Digital Library Service System
DO	Digital Object
DOI	Digital Object Identifier
DoMDL	Document Model for Digital Libraries

EMMA	Extensible MultiModal Annotation
ER	Entity–Relationship
FAST	Flexible Annotation Service Tool
GUI	Graphical User Interface
GW	Gateway
HCI	Human Computer Interaction
HIR	Hypertext Information Retrieval
HTML	HypertText Markup Language
HTTP	HyperText Transfer Protocol
IDF	International DOI Foundation
IEI	Istituto della Enciclopedia Italiana fondata da Giovanni Treccani
IETF	Internet Engineering Task Force
IMS	Information Management System
IR	Information Retrieval
IRON	Information Retrieval ON
IRON²	Information Retrieval On aNNotations
IRON-SAT	IRON - Statistical Analysis Tool
IRS	Information Retrieval System
ISBN	International Standard Book Number
ISO	International Organization for Standardization
JDBC	Java DataBase Connectivity
JPEG	Joint Photographic Experts Group
LAN	Local Area Network
LS	Lexical Signature
MADCOW	Multimedia Annotation of Digital Content Over the Web
MIME	Multipurpose Internet Mail Extensions
NISO	National Information Standards Organization
OAI	Open Archives Initiative
OAI-PMH	Open Archives Initiative Protocol for Metadata Harvesting
OCLC	Online Computer Library Center

OMG	Object Management Group
OO	Object Oriented
P2P	Peer-To-Peer
PDA	Personal Digital Assistant
POI	PURL-based Object Identifier
pSQL	propagate SQL
PURL	Persistent URL
RDBMS	Relational DataBase Management System
RDF	Resource Description Framework
RFC	Request for Comments
SE	Search Engine
SQL	Structured Query Language
TO	Transfer Object
UI	User Interface
UML	Unified Modeling Language
UMTS	Universal Mobile Telecommunication System
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
W3C	World Wide Web Consortium
WLAN	Wireless LAN
WS	Web Services
XML	eXtensible Markup Language

Chapter 1

Introduction

Nowadays, the notion of isolated information resources or applications is increasingly being replaced by a distributed and networked environment, where there is almost no distinction between local and remote information resources and applications. Indeed, a wide range of new technologies allow us to envision ubiquitous and pervasive access to information resources and applications. A wide range of wired and wireless technologies make it possible to offer almost ubiquitous connectivity; examples of such technologies are *Local Area Networks (LANs)*, *Wireless LANs (WLANs)*, *Asymmetric Digital Subscriber Line (ADSL)* and other broadband connections, *Third Generation Mobile System (3G)* networks as *Universal Mobile Telecommunication System (UMTS)* networks. Moreover, a variety of devices, that range from desktop computers to *Personal Digital Assistants (PDAs)*, mobile phones, and other handheld devices (Agosti and Ferro, 2003b), and a series of emerging architectural paradigms, such as *Web Services (WS)*, *Peer-To-Peer (P2P)* and Grid architectures, are now available and allow us to design and develop services and systems that are more and more user-centered.

This continuously evolving scenario uncovers and calls for new possibilities and advanced services also in the field of *Information Management Systems (IMSS)*, which can provide a wider range of functionalities to their users.

In particular, *Digital Library (DL)* systems are currently in a state of evolution: today they are simply places where information resources can be stored and made available, whereas for tomorrow they will become an integrated part of the way the user works. For example, instead of simply downloading a paper and then working on a printed version, a user will be able to work directly with the paper by means of the tools provided by the DL system and share their work with colleagues. This way, the user's intellectual work and the information resources provided by the DL system can be merged together in order to constitute a single working context. Thus, the DL is no longer perceived as something external to the intellectual production process or as a mere consulting tool, but as an intrinsic and active part of the intellectual production process (Agosti and Ferro, 2004).

This turning point of DL systems clearly emerges also from the outcomes of the third brainstorming meeting, organized by DELOS¹, the European Network of Excellence on Digital Libraries funded by the EU's 6th Framework Programme, and held in Corvara, Italy on the 8–9 July 2004²:

¹<http://www.delos.info/>

²<http://www.delos.info/pastdelosevents.html>

the main conclusions were that digital libraries had to become more user-centred, that digital libraries should not just be passive repositories but required more active collaboration and communication tools, and that there was a need for more generic digital library management systems. . . Finally, the brainstorming meeting investigated the possibility of developing a phrase that could be used instead of “digital libraries”. Combinations of various terms were considered, including the adjectives “pervasive”, “ambient” or “collaborative” and the entities “garden”, “factory” and “architecture”, but the phrase *dynamic ubiquitous knowledge environments* was said to be the favoured choice of the meeting (DELOS, 2004)

and

terms like “Digital Ubiquitous Knowledge Environments” instead of “Digital Libraries” emphasize the development of user-centered DL systems, acting as communication and collaboration tools and leading to a generic “out-of-the-box” DL system, reducing customization efforts for individual stake-holders, be they users, information providers, or curators (Rauber, 2004).

Annotations are effective means in order to enable this paradigm of interaction between users and DLs, since they are a very well-established practice and widely used. Annotations are not only a way of explaining and enriching an information resource with personal observations, but also a means of transmitting and sharing ideas in order to improve collaborative work practices. Furthermore, annotations represent a bridge between reading and writing, that facilitates the user’s first approach when they begin dealing with an information resource; thus, a DL offering annotation capabilities can be appealing to the user’s needs. Finally, annotations allow users to naturally merge personal contents with the information resources provided by the DL, making it possible to embody the paradigm of interaction between users and DLs which has been envisaged above (Agosti and Ferro, 2004).

The research work addressed in this thesis is aimed at investigating and studying the themes and the issues concerning the annotation of information resources in the context of different DL systems and, more generally speaking, in the context of different IMSs. The scope of this research work ranges from defining what an annotation is by providing a formal model of the annotation to designing the architecture of a system able to provide annotation capabilities on information resources managed by different kinds of IMSs.

The major contributions of this thesis are to provide the groundwork needed to properly delineate the complexity of the problem, which is often underestimated and only partially addressed, and to define a formal model of the annotation, which is still missing from previous research on this topic. With respect to this last point, Buneman et al. (2002) state that:

view annotation is becoming an increasingly useful method of communicating meta-data among users of shared scientific data sets, and to our knowledge, there has been no formal study of this problem

and Bottoni et al. (2003) point out that:

strangely enough, there is not an agreement yet on the definition of digital annotation, or on how to distinguish it from other digital entities (e.g. hyperlinks, metadata, newsgroup messages). Furthermore, an analysis of the basic operations, to be enabled by a digital annotation system, seems to be lacking.

Finally, the thesis proposes also the architecture of a distributed annotation system, capable of adding annotation functionalities to existing IMSs, such as DL systems.

1.1 Structure of the Thesis

This thesis is organized as follows. Chapter 2 provides us with a thorough discussion on annotations from both an historical point of view and a present perspective in order to highlight the key points about annotations. Chapter 3 introduces our methodological approach to annotations; this approach covers both modelling aspects for the purpose of defining a model for the annotation, and architectural aspects for the purpose of designing an architecture for a system with annotation functionalities. Chapter 4 reports our proposal for a formal model of the annotation in order to mathematically define the annotation. Chapter 5 explains our proposal for a conceptual architecture for a system capable of providing different IMSs with annotation functionalities. Finally, Chapter 6 summarizes the contributions of the thesis and presents a future outlook of the research work.

Chapter 2

Annotations

This chapter is devoted to introduce and delineate the issues and the key points concerning the annotation.

Section 2.1 presents a thorough study of the annotation from an historical point of view; on the other hand, Section 2.2 analyzes the present perspectives about annotations. Finally, Section 2.3 gathers up the observations made in the previous two sections in order to highlight some key points about annotations that have to be taken into account.

2.1 Historical Viewpoints

A basic step in approaching the problem of annotations is to define the meaning of the different terms that come into play and to investigate their historical usage over the course of time. All of this is necessary in order to gather information as to delineate the contours of the problem.

Thus, in this section, we have conducted a research about annotations that has a literary approach. We firmly believe that studying the terms, their meaning, their etymology, and the way they have been used can provide us with a solid groundwork on which we can build the subsequent research. When we talk about annotations, we deal with a concept that has been stratified for a long period of time in our culture, and literary research is the most effective way to benefit from the pre-existing knowledge of our cultural heritage. The outcomes of this research are key points concerning the features of the annotation, that we should take into account when we develop a model for the annotation and when we design a system capable of providing annotation functionalities.

This kind of research is complementary to user studies that are conducted in order to gather user requirements. Furthermore, it completes user studies with a knowledge that users are not often able to express, because often they overlook what they have naturally absorbed from their cultural heritage. In some sense, we are conducting a user study, where our user is the history of the annotation and we ask what features of the annotation are relevant for us.

We have adopted the following methodology: first, we look up the meaning of the term at hand in the dictionary; then, we investigate its etymology and its historical usage; finally, we gather the information provided by the two previous steps so that we can emphasize some key points about the annotation.

2.1.1 The Term Annotation

Both Hanks (1979, p. 57) and IEI (1986, p. 198) define the word *annotation*, firstly, as the *act of annotating* and, secondly, as *a note added in explanation especially of some literary work*. IEI (1986, p. 198) further observes that the word *annotation* can also be used in sentences with a passive sense, as thing worthy of annotation, with the meaning of *noteworthy* and *worth remembering*. The word *annotation* is closely related to the verb *annotate* that, in turn, means *to supply (a written work, such as an ancient text) with critical or explanatory notes* (Hanks, 1979, p. 57) and *to note down, to write down, to record something* (IEI, 1986, p. 198).

Both Cortelazzo and Zolli (1999, p. 107) and Hanks (1979, p. 57) trace the etymology of *annotation* back to the Latin word *annōtātio*, that simply means *annotation, note* (Calonghi, 1986, p. 189). The Latin word *annōtātio*, in turn, derives from the Latin verb *annōtāre*, which means *to annotate* and *to observe in writing* (Calonghi, 1986, p. 190). Finally, the Latin verb *annōtāre* comes from the Latin word *nōta*, that means *note, mark* (Calonghi, 1986, p. 1823), plus the intensifying prefix *ād*, which in compound words means *to approach, to tend* and thus *to add* (Calonghi, 1986, pp. 37–40). Both Calonghi (1986, p. 1823) and Cortelazzo and Zolli (1999, p. 1047) agree that the Latin word *nōta* has an uncertain etymology.

This brief discussion highlights some interesting points about the *annotation*. Firstly, the *annotation* is not only an object or something that is passive, but it also contains the notion of activity, as its first meaning “act of annotating”. In this sense, the *annotation* calls for an active involvement by the subject who is engaged in the act of supplying explanatory matter or keeping record of something. Furthermore, the *annotation* covers, in its second meaning, also the purpose of this active involvement, that is to produce an intellectual work in order to add an explanation to some literary work, as an example. This idea is further supported by the meaning of the verb *to annotate*, which broadens the spectrum of the word *annotation* towards keeping record of something. Therefore, on the whole, the *annotation* requires an active involvement in order to produce an intellectual work that has to be recorded. These facets of the *annotation* are present also in the etymology of the word *annotation*: the Latin verb *annōtāre* means to make written observations, comments or remarks in the durable and recordable written form; on the other hand, the Latin word *nōta* recalls the note or the mark put to remember or highlight something. Finally, the outcomes of the act of *annotating* are also taken into account: indeed, both the annotated object, in a passive sense, and the content of the *annotation*, in an active sense, become noteworthy and worth recording.

2.1.2 Terms Related to Annotation

The range of our investigations can be widened in order to take into consideration also synonyms and terms related to the word *annotation*.

Spooner (1999, p. 14) provides the following synonyms of the term *annotation*: *comment, commentary, elucidation, explanation, footnote, gloss, interpretation, and note*. We can also add to this list the word *jotting*, which is a very brief *annotation* (Hanks, 1979, p. 789), and the word *scholium*, which is a particular kind of *annotation* (IEI, 1994, p. 158).

Table 2.1 provides the definitions for the different terms listed above. Note that these definitions often refer to printed documents or texts, since they are taken from

Word	Definition
Comment	<i>a note explaining or criticizing a passage in a text and explanatory or critical matter added to a text</i>
Commentary	<i>an explanatory series of notes or comments</i>
Elucidation	<i>making clear (something obscure or difficult)</i>
Explanation	<i>the act or process of explaining and a statement or occurrence that explains and a clarification of disputed terms or points</i>
Footnote	<i>a note printed at the bottom of a page, to which attention is drawn by means of a reference mark in the body of the text</i>
Gloss	<i>a short or expanded explanation or interpretation of a word, expression, or foreign phrase in the margin or text of a manuscript</i>
to Jot	<i>to write a brief note of</i>
Jotting	<i>something jotted down</i>
Interpretation	<i>the act or process of interpreting or explaining; elucidation and the result of interpreting; an explanation</i>
Note	<i>a brief summary or record in writing especially a jotting for future reference and a short written statement giving any kind of information and a critical comment, explanatory statement, or reference in the the text of a book, often preceded by a number</i>
Notes	<i>short descriptive or summarized jottings taken down for future reference</i>
Observation	<i>the act of observing or the state of being observed and a comment or remark and the facts learned from observing</i>
Postil	<i>a commentary or marginal note, as in a Bible</i>
Record	<i>an account in permanent form, especially in writing, preserving knowledge or information about facts or events</i>
Scholium	<i>a commentary or annotation, especially on a classical text</i>

Table 2.1: Definition of the terms related to annotation from (Hanks, 1979) .

an English dictionary (Hanks, 1979). On the other hand, we should consider that their validity is not limited only to printed documents, but it can also be applied and extended to information resources in a digital context.

As it can be noticed from table 2.1, these words are often defined by terms used to describe other words in the list, or they refer to the same notion of explaining, expounding, interpreting, clarifying, recording something. In this way, they reveal how closely related they are.

In conclusion, the terms listed in table 2.1 support, refine and enforce what has been observed above about the word annotation, introducing further kinds of annotation which cover different needs and tasks, such as the gloss, the postil, the note, the jotting, and so on.

Now, we can move a step further and investigate in more detail the terms gloss, scholium, and postil in order to understand the rich semantics of the annotation and how it has evolved with the passing of time and its current consequences.

2.1.3 The Term Gloss

As reported in (Hanks, 1979, p. 620) and (Cortelazzo and Zolli, 1999, p. 673), the word gloss derives from the ancient Greek word γλῶσσα (*glōssa*), that means *tongue, language, idiom, spoken word, foreign or obsolete word* (Rocci, 1989, p. 393).

As reported in (IEI, 1987, pp. 652–653), at the time of the ancient Greeks, the term gloss meant an obscure, archaic, dialect, or rare locution that required an additional explanation. These locutions were object of study by grammarians or object of research by scholarly poets, especially the Alexandrine poets, who embellished their compositions with these terms. Then, gloss meant the explanations themselves of such locutions, either collected in wide-ranging lexicons or as interlinear notes placed on top of the words to explain. This was a methodology of study and a lexicographical practice that dates back to very ancient times (there were glosses to Homer already in the V century B.C.) and that was fully developed by the grammarians of the Alexandrine age. During the Bizantine age and the Middle Ages, the term gloss meant an interlinear or marginal note to a biblical or juridical codex. For the biblical codices, the gloss was a very short paraphrase to explain a passage of the Bible, sometimes together with a mention to its allegorical interpretation. On the other hand, for the juridical codices, the glosses were explanatory annotations, that constituted a thorough commentary to the text.

The gloss was a practice that flourished especially in the juridical context, as reported by IEI (1951, pp. 427–429). During the Roman Empire, one of the usual literary forms of the Roman jurisprudence was the comment to the works of former jurists, so that it is often possible to distinguish the annotated text from the annotation to the text; furthermore, the glosses were sometimes physically separated from the annotated text. However, the most famous use of this kind of method of study is due to the Bolognese school: indeed, the word gloss denoted the way of studying the Justinian Code practised in Bologna, which began in the II century A.D. The Bolognese gloss passed from a simpler form to a more complex one, that is it passed from simple interlinear notes to a real theoretical treatment of the subject. The glossarist reveals the contradictions (*contrāriētātēs*¹) of the Justinian books, raises doubts (*dūbītātīōnēs* or *dūbīētātēs*), that often give rise to controversies (*dissēnsiōnēs*). The contradictions often find an explanation (*sōlūtīo*) and the doubts disappear by means of an appropriate distinction (*distinctiō* or *diffērentiā*). The glossarist teaches the Justinian books and creates cases in point and examples that originate glosses pointing out the different cases (*cāsūs*); furthermore, the glossarist fixes and defines rules derived from the texts he studies, and, accordingly, creates glosses that report such rules (*rēgūlae*) and definitions (*dēfīnītīōnēs*). In conclusion, the Bolognese gloss was a way of doing research aimed at defining and elucidating the law.

This discussion about the term gloss points out some interesting facets of the annotation, that has not fully emerged in the previous observations about the term annotation. The intellectual work entailed by the gloss is of very high quality, because it is a method both of study and of research. This kind of intellectual work gives us an idea of how strong the active involvement required by the gloss is: it does not concern only the author himself, but it is also capable of involving and stimulating a wide community

¹The italicized words in brackets are the Latin technical terms used to indicate the specific technique applied in each step of the method of study. Note that they are the translation of the word which precedes them.

of people that works, studies and does research on a subject. Thus, it turns out that an annotation may comprise a public dimension, because it becomes the vehicle for carrying and transmitting ideas and knowledge to other people, or it may comprise a shared dimension, if the recipients of the annotation are less numerous. Finally, the research or study aspects, and the public or shared dimension entailed by the gloss help us to understand how durable and recordable the annotations are. Indeed, they are not only comments and remarks to a text, but also an autonomous intellectual work, that worths recording.

2.1.4 The Term Scholium

As reported in (Hanks, 1979, p. 1305) and (Cortelazzo and Zolli, 1999, p. 1479), the word scholium derives from the ancient Greek word *σχόλιον* (*schólion*), that means *comment, explanation* (Rocci, 1989, p. 1793). The ancient Greek word *σχόλιον* (*schólion*), in turn, comes from the ancient Greek word *σχολή* (*scholé*), that means *scholar activity* and *school* (Rocci, 1989, p. 1793).

IEI (1950, p. 198–199) reports that the word scholium designates short annotations or explanations written by a reader in the margin of a manuscript. The distinguishing features of the scholium are the fact that they are anonymous and fragmentary. The scholium is anonymous because, initially, the reader writes in the margin of the manuscript his own observations or passages taken from a commentary for both personal use and scholastic needs. The next owner of the manuscript often extends the scholium or modifies it. Thus, the lack of organic unity is explained in this way. Often the scholia contain also citations by the authors from which the observations are taken; this way, they are very useful in order to reconstruct the doctrines and the works of ancient grammarians that may no longer exist.

The term scholium suggests another facet of the annotation: it may be created for personal purposes, that is the annotation may entail a private dimension, since the main recipient of the annotation is the author themselves. However, the private dimension may represent only the initial intention of the annotation, because also other people reading an annotated text can benefit from existing annotations and can modify or extend them; thus, the annotation passes from a private dimension to a shared one. Taken to the extreme this process encompasses the possibility that an annotation becomes the means to study the thought of authors that otherwise would be lost; thus, the annotation passes from a private dimension to a public one. In conclusion, private annotations are part of this spectrum of possibilities and this makes us aware of the necessity to carefully preserve private annotations, because they may become worth recording also for different reasons from the ones that motivated their creation.

2.1.5 The Term Postil

As introduced in table 2.1, a postil is a short annotation – often a marginal or interlinear note – to a text, handwritten by a scholar or by the author himself in order to express observations, explanations, or criticisms. During the Middle Ages, the postils were a scholastic practice and they sometimes represented comments that were broader than simple notes (IEI, 1991, p. 1030).

Both Hanks (1979, p. 1145) and Cortelazzo and Zolli (1999, p. 1239) trace the etymology of postil back to the Latin terms *pōst illā* (*verbā textūs*) that mean *after those (words in the text)*, which often was the opening phrase of such annotation.

Thus, the word *postil* points out in its etymology itself one of the main aspects concerning the annotation: the annotation is the result of an intellectual work on an existing text and it follows an already existing text. Thus, the annotation comprises a temporal dimension that is often not explicit but that limits the creation of the annotation to the existence of another text. This temporal relationship between the annotation and the annotated text does not mean that the annotation cannot be considered as a stand-alone intellectual work – and some glosses and scholia are by right autonomous pieces of knowledge – but it imposes a temporal ordering between the existence of an annotated text and the annotation annotating it, that cannot be neglected.

2.2 Presents Perspectives

Many user studies are aimed at understanding annotation practices and discovering common annotation patterns. Marshall (1997) studied personal annotative practices of American college students in order to point out the form the annotations take on in the textbooks and the function of the annotations derived from their form. Marshall (1997, pp. 237–238) discovered that:

First, annotations are *procedural signals*, cluing in the student to where an assignment starts, what material is important (and as we will see, unimportant), and what material might require a second (or successive readings). Second, annotations are *placemarks*; they hold the quotes that are being reserved for the paper that the student will write at the end of the term, the chemical reactions and term definitions the student must memorize for the final, the theorem that is key to the proof in the homework assignment. Third, they are an *in situ way of working problems*. Fourth, annotations record *interpretive activity*, either from another reader (e.g. a professor's explanation), or as the result of careful reading (the student has interpreted it him or herself). Fifth, and most elusively, these markings act as a *visible trace of a reader's attention*, a focus on the passing words, and a marker of all that has already been read (as if these words are now possessed). Finally, the markings may just be incidental, *reflecting the material circumstance of reading*.

Marshall (1998) carries on her research work and categorizes annotations along several dimensions, that reflect the form which annotations may take on: formal versus informal annotations, explicit versus tacit annotations, annotations as writing versus annotations as reading, hyperextensive versus extensive versus intensive annotations, permanent versus transient annotations, published versus private annotations. Finally, Marshall and Brush (2002), Marshall and Brush (2004), and Shipman et al. (2003) investigate the relationship among private, shared and public annotations and how they can be exploited to find useful passages in the text.

It is worth noting how the findings of Marshall (1997) and Marshall (1998) agree with the outcomes of the study conducted in the previous section about the historical perspective on annotations. Indeed, both glosses and scholia are, to some extent, *placemarks*, an *in situ way of working problems*, they record an *interpretive activity*, and so on. Also the different dimensions of the annotation are taken into account by the historical perspective: glosses are often more formal annotations than scholia and

postils, that are usually informal; scholia can be tacit annotations due to their fragmentariness while glosses can be explicit annotations; all the kinds of annotations described in Section 2.1 act as a bridge between reading and writing; glosses may be considered intensive annotations, postils may be more extensive annotations and both glosses and scholia often contain references to other authors and quotations of other texts, that is a way of being hyperextensive annotations; the stratification of glosses and scholia in our cultural heritage is a clear sign of the passage from transient to permanent annotations; finally, the difference among postils, scholia, and glosses comprises the distinction between private and public annotations. On the other hand, neither Marshall (1997) nor Marshall (1998) explicitly points out the temporal dimension entailed by annotations and the temporal ordering between annotations and annotated objects, which has been discussed in Section 2.1.5 talking about the term postil.

As introduced in Chapter 1, we aim at designing a system capable of managing annotations in an automatic way in order to support users and their annotative practices. In this context, Phelps and Wilensky (1997) suggest a list of desirable properties for annotations: annotations should appear *in situ*, that is on the documents themselves; they should be *highly expressive*; they should be *format and platform independent*; they should be *extensible, yet composable*, that is they should allow different styles of annotation; they should be *distributed, open, and robust*, that is they may reside in a place while referring to documents in another place.

In addition, a lot of research work deals with: the employment of ad-hoc devices or handheld devices which enable reading appliances with annotation capabilities (Marshall et al., 1999, 2001b; Marshall and Ruotolo, 2002; Schilit et al., 1998); the design and development of document models and systems which support annotations (Phelps and Wilensky, 1996, 1997, 2000, 2001) in digital libraries (Agosti et al., 2003b; Agosti and Ferro, 2003a; Agosti et al., 2004; Gueye et al., 2004; Rigaux and Spyrtatos, 2004), in the Web (Bottoni et al., 2004, 2003; Fogli et al., 2004; Kahan and Koivunen, 2001; Nagao, 2003; W3C, 2004a,b), in collaborative systems and working groups (Frommholz et al., 2003, 2004), and databases (Bhagwat et al., 2004; Buneman et al., 2004, 2001, 2002).

All of this research work has led to different viewpoints about what an annotation is, some of which are described in (Agosti et al., 2004):

- *annotations are metadata*: they can be considered as additional data which concern an existing content, that is annotations are metadata, because they clarify in some way the properties and the semantics of the annotated content. For example, the Annotea² project developed by the *World Wide Web Consortium (W3C)* (Kahan and Koivunen, 2001) considers annotations as metadata and interprets them as the first step in creating an infrastructure that will handle and associate metadata with content towards the Semantic Web³. Another example is MPEG-7 ISO (2004), formally named “Multimedia Content Description Interface”, which is a standard for annotating and describing the multimedia content data. MPEG-7 supports some degree of interpretation of the information’s meaning, which can be passed onto, or accessed by, a device or a computer code. MPEG-7 is not aimed at any one application in particular; rather, the elements that MPEG-7 standardizes support as a broad range of applications as possible. As a further example,

²<http://www.w3.org/2001/Annotea/>

³<http://www.w3.org/2001/sw/>

in the context of *DataBase Management System (DBMS)* Bhagwat et al. (2004) considers annotations as “information about data such as provenance, comments, or other types of metadata”;

- *annotations are contents*: they are additional content which concern an existing content (Nagao, 2003); indeed, they increase existing content by providing an additional layer of content that elucidates and explains the existing one. This viewpoint about annotations entails an intrinsic dualism between annotation as content enrichment and annotation as stand-alone document (Agosti and Ferro, 2003a):
 - *annotation as content enrichment*: in this view annotations are considered as mere additional content regarding an existing document and so they are not autonomous entities but in fact they rely on already existing information resource in order to justify their existence;
 - *annotation as stand-alone document*: in this view annotations are considered as real documents and are autonomous entities that maintain some sort of connection with an existing document.

This twofold nature of the annotation is clear if we think about the process of studying a document: firstly, we can start annotating some interesting passages that require an in depth investigation, which is an annotation as content enrichment; then we can reconsider and collect our annotations and we can use them as a starting point for a new document, covering the points we would like to explain better which is an annotation as a stand-alone document. In this case the annotation process can be seen as an informal, unstructured elaboration that could lead to a rethinking of the annotated document and to the creation of a new one. Also Bottoni et al. (2003) agree with this viewpoint about annotations and consider them to be reliant on the annotated objects; in this way, Bottoni et al. (2003) consider annotations as content enrichment;

- *annotations constitute an hypertext*: they allow the creation of new relationships among existing contents, by means of links that connect annotations together and with existing content. In this sense we can consider that existing content and annotations constitute a hypertext, according to the definition of hypertext provided in (Agosti, 1996). This hypertext can be exploited not only for providing alternative navigation and browsing capabilities, but also for offering advanced search functionalities. Furthermore, Marshall (1998) considers annotations as a natural way of creating and growing hypertexts that connect information resources in a DL system by actively engaging users. Finally, the hypertext existing between information resources and annotations enables different annotation configurations, that are *threads of annotations*, i.e. an annotation made in response to another annotation, and *sets of annotation*, i.e. a bundle of annotations on the same passage of text (Agosti and Ferro, 2003a; Agosti et al., 2004);
- *annotations are dialog acts*: they are part of a discourse with an existing content. For example, Frommholz et al. (2003, 2004) consider annotations as the document context, intended as the context of the collaborative discourse in which the document is placed. Also Fogli et al. (2004) agree, to some extent, with this viewpoint about annotations. Indeed, they interpret annotations as a means that

allow a “two way exchange of ideas between the authors of the documents and the documents users”.

However, these viewpoints are not completely disjointed, on the contrary, they may overlap and they may be simultaneously present in some situations.

In the following sections, we will go into more detail about the current viewpoints concerning annotations and we will present some interesting cases of usage of annotations and IMSs with annotation capabilities in the context of digital libraries, the Web, and databases.

2.2.1 Digital Libraries

Digital libraries are not only the digital versions of traditional libraries and archives, but offer means which go beyond mere presentation of the content stored in digital repositories. In the following we point out this fact by discussing two definitions of digital libraries, which come from two different fields. The more computer science oriented view is expressed in the introduction in the first issue of the *International Journal on Digital Libraries*, cited by Fuhr et al. (2001):

Digital Libraries are concerned with the creation and management of information resources, the movement of information across global networks and the effective use of this information by a wide range of users.

Librarians have a different definition of DL, as proposed by the Digital Library Federation, 1998, cited by Fuhr et al. (2001):

Digital Libraries are organisations that provide the resources, including the specialised stuff, to select, structure, offer intellectual access to, interpret, distribute, preserve the integrity of, and ensure the persistence over time of collections of digital works so that they are readily and economically available for use by a defined community or set of communities.

Annotations can be exploited in order to provide users with the distinguishing features of DL systems highlighted above. Note, however, that also archives have to be taken into account in this context, although they are not explicitly mentioned in the previous definitions. The *creation* of new information resources is supported by annotations in two ways. First, when users add annotations to existing information resources, they in turn become new information resources themselves. Second, annotations can also assist in the creation of new information resources. Through annotations, new ideas and concepts can be discussed and the results of such a discussion can then be integrated into the newly created object. Annotations might increase and expand the information resources managed by the digital library. In this way, they may provide *interpretations* of information resources. User communities benefit from such interpretations in that they help the understanding of the annotated resource and contain additional information about it. As an example, in the Humanities interpretation is one of the basic tasks scholars perform: systems like COLLATE (Frommholz et al., 2003) or IPSA (Agosti et al., 2003b) support this task through annotations. Annotations support user communities in *accessing* the information resources provided by the digital library in a personalised and customized way: indeed, users can create annotations that link different documents, enabling alternative paths for browsing digital contents and thus structuring them in alternative ways.

Different layers of annotations can coexist on the same document: a private layer of annotations accessible only by the annotations author themselves, a collective layer of annotations, shared by a team of people, and finally a public layer of annotations, accessible to all the users of the digital library. In this way, user communities can benefit from different views of the information resources managed by the digital library (Marshall, 1997; Marshall and Brush, 2002, 2004). A DL can encourage cooperative work practices, enabling the sharing of documents and annotations, also with the aid of special devices, such as XLibris (Schilit et al., 1998). Finally, as suggested in (Marshall et al., 2001a; Marshall and Ruotolo, 2002), searching, reading and annotating a DL can be done together with other activities, for example working with colleagues. This may also occur in a mobile context, where merging content and wireless communication can foster ubiquitous access to DL systems, improving well established cooperative practices of work and exploiting physical and digital resources. The wireless context and the small form factor of handheld devices challenge our technical horizons for information management and access and require specialized solutions in order to overcome the constraints imposed by such kinds of devices, as reported in (Agosti and Ferro, 2003b).

In the context of a DL system it is also possible to create automatic annotations, which may facilitate the user's first approach with a document. Automatic annotations can be created by using topic detection techniques in order to associate each annotation with its related topic, which constitutes the context of the annotation. In this way, a document can be re-organized and segmented into topics, whose dimension can range in many different sizes, and annotations can present a brief description of those topics. Then, by applying automatic hypertext construction techniques, similar to those presented in (Agosti and Melucci, 2000), those pairs of topics and annotations can be linked together, proposing an alternative way of navigating the content of a digital library.

Finally, Rigaux and Spyrtos (2004) and Gueye et al. (2004) propose a data model for the composition and metadata management of documents in a distributed setting, such as a DL system. They allow the creation of *composite documents*, that are made up of either composite documents or *atomic documents*, that can be any piece of material uniquely identifiable. A set of annotations is associated to each composite document, where Rigaux and Spyrtos (2004) and Gueye et al. (2004) interpret annotations as terms taken from a controlled vocabulary or taxonomy to which all authors adhere. They provide algorithms to automatically compute the annotations of composite documents starting from the annotations of its composing atomic documents, by means of a subsumption relation defined within the taxonomy mentioned above.

2.2.2 The Web

As previously introduced, the Annotea project (Kahan and Koivunen, 2001) considers annotations as metadata. Annotea defines annotations as comments, notes, explanations, or other types of external remarks that can be attached to any Web document or a selected part of the document without modifying the document. Annotaea uses *Resource Description Framework (RDF)*⁴ and *eXtensible Markup Language (XML)*⁵ for describing annotations as metadata and *XPointer*⁶ for locating the annotations in the

⁴<http://www.w3.org/RDF/>

⁵<http://www.w3.org/XML/>

⁶<http://www.w3.org/XML/Linking>

annotated document. Annotea employs a client-server architecture, where annotations reside in dedicated servers and a specialized browser is capable of retrieving them upon request, when visiting a Web page. Koivunen and Swick (2001) and Koivunen et al. (2003) move a step further and employ annotations as an extension of the bookmarks in order to improve the collaboration among users: indeed, the additional data provided by annotations are exploited to describe, organize, categorize, share, and search for the bookmarks.

Moreover, the W3C Multimodal Interaction Working Group⁷ is developing the *Extensible MultiModal Annotation (EMMA)* markup language (W3C, 2004b). EMMA is a markup language intended for providing semantic interpretations for a variety of inputs, such as speech, natural language text, and *Graphical User Interface (GUI)* input. The language is focused on annotating the interpretation information of single and composed inputs, and it is expected that this markup will be used primarily as a standard data interchange format between the components of a multimodal system. The general purpose of EMMA is to represent information automatically extracted from a user's input by an interpretation component. EMMA provides a simple structural syntax for the organization of interpretations and instances, and an annotative syntax derived from RDF to apply the annotation to the input data at any level.

As a further example, *Multimedia Annotation of Digital Content Over the Web (MADCOW)* is based on a client-server architecture as Annotea is. Servers are repositories of annotations to which different client can connect, while the client is a plug-in for a standard Web browser (Bottoni et al., 2004). MADCOW employs *HyperText Transfer Protocol (HTTP)* in order to annotate Web resources and allows both private and public annotations. Moreover, it allows different pre-established types of annotations, such as explanation, comment, question, solution, summary, and so on; in this respect, MADCOW opts for a solution similar to the one of COLLATE, which is not Web-based but it models annotations as different types of dialog acts (Frommholz et al., 2003).

2.2.3 Databases

Annotations are used also in the context of the DBMSs and, in particular, in the case of *curated databases* and *scientific databases*. SWISS-PROT⁸ is a curated protein sequence database, which strives to provide a high level of annotation, such as the description of the function of a protein, its domains structure, and so on. In this case, the annotations are embedded in the database and merged with the annotated content. BIODAS⁹ provides a *Distributed Annotation System (DAS)*, that is a Web-based servers system for sharing lists of annotations across a certain segment of the genome. In this case, the annotation are not mixed together with the content they annotate, but they are separated from it. Annotations have types, methods and categories: the annotation type is selected from a list of types that have biological significance; the annotation method is intended to describe how the annotated feature was discovered and may include a reference to a software program; the annotation category is a broad functional category that can be used to filter, group and sort annotations. Finally, annotations may also be associated with Web *Uniform Resource Locators (URLs)* that provide additional

⁷<http://www.w3.org/2002/mmi/Group/>

⁸<http://www.expasy.org/sprot/>

⁹<http://biodas.org/>

human readable information about the annotation itself (Stein et al., 2002). Another example is SEED, a P2P system which aims to provide the biology community a suite of open source tools to enable distributed teams of researchers to rapidly annotate new genomes. In particular, the SEED enables researcher to create, collect, and maintain sets of gene annotations organized by group of related biological and biochemical functions across many organisms (Overbeek et al., 2004).

In the context of scientific databases, Buneman et al. (2004) proposes an archiving technique in order to manage and archive different versions of such kinds of databases, as time moves on. Buneman et al. (2004) exploit the hierarchical structure of scientific data in order to represent the content and the different versions of the database with a tree structure. They attach annotations to the nodes of the tree, annotations that contain time-stamp and key information about the underlying data structure. Thus, these annotations are metadata about the database itself. These annotations different from the annotations contained in the database, that are metadata about genome sequences. In conclusion, this annotated tree structure provides an additional data layer, that allows the development of efficient algorithms in order to archive and search for the different versions of the database.

Buneman et al. (2001, 2002) investigate the usage of annotations with respect to the *data provenance* problem, sometimes also referred to as *data lineage* or *data pedigree*, which is the description of the origins of a piece of data and the process by which it arrived in a database. Buneman et al. (2001) distinguishes between *why-provenance*, which explains of why a given piece of data is in the database, and *where-provenance*, which explains where a given piece of data comes from. Data provenance is a relevant issue in the field of curated and scientific databases, such as genome databases, because in this field there are few databases that are sources of data, so that we can actually that they receive the experimental data; all the other databases are in some sense views of these source databases or of other views. The distinguishing feature of these databases is the fact that they have to be curated: in fact, they provide corrections and annotations to the original source data made by experts. It is now clear that data provenance is essential to any user interested in the accuracy and timeliness of the data. In particular, where-provenance is important for understanding the source of errors in data and for carrying annotations through database queries, problems addressed in (Buneman et al., 2002). Bhagwat et al. (2004) carry on the research about where-provenance and propose and implement an extension to a relational DBMS and an extension to *Structured Query Language (SQL)*, called *propagate SQL (pSQL)*, which provides a clause for propagating annotations to tuples through queries. Bhagwat et al. (2004) intend annotations to be an information about data such as provenance, comments, or other types of metadata; they envisage the following applications of annotations in DBMS: tracing the provenance and flow of data, reporting errors or remarks about a piece of data, and describing the quality or the security level of a piece of data.

2.3 Key points

On the whole, the line of reasoning conducted in Sections 2.1 and 2.2 provides us with some distinguishing features of the annotation that we should take into account. Thus, we summarize the main findings pointed out in the two previous sections.

First-class intellectual work annotations are a valuable intellectual work, as it emerges from the discussion in Section 2.1 and from the user studies reported in Section 2.2. The spectrum of this intellectual work is very broad, because it ranges from explaining and enriching an information resource with personal observations to transmitting and sharing ideas and knowledge on a subject. In conclusion, annotations can be geared not only to the way of working of the individual and to a method of study, but also to a way of doing research.

Various facets annotations comprise different viewpoints, as discussed in Section 2.2: they may be considered as metadata, content, hypertext, or dialog acts. Moreover, the boundaries between these viewpoints are not sharp and they may coexist. All of these viewpoints have to be taken into account, especially because they are tightly coupled with and are the expression of the different kinds of intellectual work that an annotation may bear.

Different scopes annotations involve different scopes: they can be private, shared or public, according to the type of intellectual work that is carried out. Moreover, the boundaries between these scopes are not fixed but they may vary and evolve with the passing of time.

Active involvement annotations call for an active involvement, whose degree varies according to aim of the annotation: private annotations requires the involvement of the authors, although shared or public annotations involve the participation of a whole community. Thus, annotations are suitable for improving collaboration and co-operation among users.

Temporal dimension annotations implicitly entail a temporal dimension, that regulates the temporal ordering among annotations and annotated information resources.

System viewpoint annotations support a wide range of usages, as the previous discussion shows. Thus, annotations functionalities should not be embedded in any given system, but rather in a stand-alone system capable of providing annotation functionalities to other systems.

Overall Comments

The research conducted regarding the annotation, its history, its different usages, and the current perspectives about it helps us in order to understand how rich the annotation is and how complex the semantics of the annotation are. The annotation has a long history, its usage has been stratified with the passing of time, and it has been played out in our cultural heritage with great importance.

We have to be aware of all the complexity entailed in the annotation if we aim to define a model for the annotation and design a system capable of providing annotation functionalities. In particular, we have to consider that the annotation is not a simple tool, that users may exploit in order to carry out some task. On the contrary, the *annotation is a real service*, which paves the way for a whole set of methodologies, functionalities, and features that users can exploit in order to carry out their intellectual

work. Thus, we have to define a model and to design a system capable of providing users with annotations as a service and not only annotations as a tool.

Finally, the title of the thesis just stems from these last observations: we aim at defining a *formal model* and at designing a *conceptual architecture* in order to provide users with *annotation* as a *service* in the context of the *dynamic ubiquitous knowledge environments*, that are the evolution of DL systems.

Chapter 3

Methodological Approach

From the discussion conducted in Chapter 2 and, in particular, from the key points about annotations presented in Section 2.3, it comes out that both the architectural and the modelling approaches become a key factor in order to enable the design of a system capable of providing users with annotations as a service, and not only as a tool.

Section 3.1 describes our architectural approach and its consequences; Section 3.2 introduces our approach to model annotations and explains its characteristics.

3.1 Architectural Approach

Annotations have a wide range of usages in different *Information Management Systems (IMSs)*, ranging from DBMSs to DLs and corresponding to the different viewpoints about annotations, introduced in Section 2.2. Annotations are a key technology for actively involving users with an IMS and this technology should be available for each IMS employed by the user. Indeed, the user should benefit from a uniform way of interaction with annotation functionalities, without the need of changing their annotative practices only because a user works with different IMSs (Agosti and Ferro, 2004).

Furthermore, annotations create an hypertext that allows users to merge their personal content with the information resources provided by diverse IMSs, according to the scenario envisaged in Chapter 2: this hypertext can span and cross the boundaries of the single IMS, if users need to interact with diverse IMSs. The possibility of having a hypertext that spans the boundaries of different IMSs is quite innovative because such hypertext is usually confined within the boundaries of a single IMS (Agosti and Ferro, 2004). Moreover, IMSs do not usually offer hypertext management functionalities; for example, DL systems do not normally have a hypertext connecting information resources with each other. Thus, annotations can be a way of associating a hypertext to a DL in order to enable an active and dynamic usage of information resources (Agosti et al., 2004).

Finally, there are many new emerging architectural paradigms, such as P2P or WS architectures, that have to be taken into account (Agosti and Ferro, 2004). On the whole, as pointed out by Phelps and Wilensky (1997), annotations should be “distributed and robust” with respect to different IMSs and architectural paradigms.

Thus, our architectural approach is aimed at *flexibility*, because we need to adopt an architecture which is flexible enough to support both various architectural paradigms and a wide range of different IMSs. Indeed, a flexible architecture allows the design

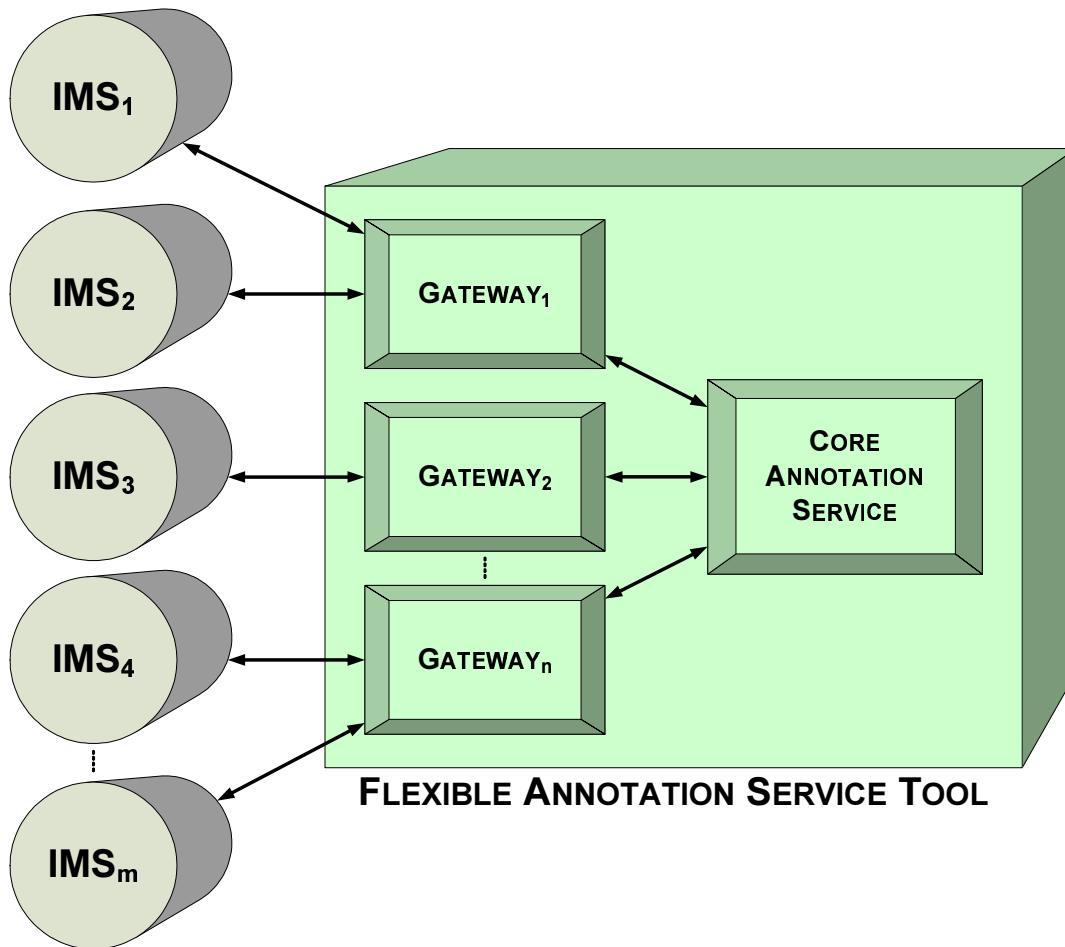


Figure 3.1: Overview of the architecture of FAST with respect to different IMSs.

of a system with a widespread usage, so that users can benefit from its functionalities without limitations due to the architecture of a particular IMS.

We named our target system *Flexible Annotation Service Tool (FAST)*, meaning that it is a flexible tool capable of providing users with annotations as a service, as explained in Section 2.3.

In order to fulfil the requirements introduced above, our architectural approach is twofold (Agosti and Ferro, 2003a, 2004):

1. to make FAST a stand-alone system, i.e. FAST is not part of any particular IMS;
2. to separate the core functionalities of the annotation service, from the functionalities needed to integrate it into different IMSs.

Figure 3.1 shows the general architecture of the FAST system and its integration with different IMSs: the *Core Annotation Service (CAS)* is able to interact with different gateways, that are specialised for integrating the CAS into different IMSs. From the standpoint of an IMS the FAST system acts like any other distributed service of the IMS, even if it is actually made up of two distinct modules, the gateway and the CAS; on the other hand, the FAST system can be made available for another IMS by simply creating a new gateway. Note that the additional layer introduced by the

gateway allows the integration of the CAS also with legacy systems, that may benefit from the availability of annotation functionalities.

The choice of making FAST a stand-alone system is coherent with the approach adopted by different systems: for example, Annotea by the W3C (Kahan and Koivunen, 2001), MADCOW (Bottoni et al., 2004), and BIODAS (Stein et al., 2002) rely on stand-alone servers, that store and manage annotations separated from the annotated objects. On the other hand, the choice of separating the core functionalities of the annotation service, from the functionalities needed to integrate it into the different IMSs is quite new. In fact, you will not be able to find an architecture like this in the literature about annotation systems, to the best of our knowledge.

As an important consequence of this architectural choice, let us look at the following:

the FAST system knows everything about annotations, however it cannot do any assumption regarding the information resources provided by the IMS, being that it needs to cooperate with different IMSs.

This situation is very different from what is commonly found today. For example, both Annotea and MADCOW are stand-alone systems but they are targeted to work with Web pages. Indeed, they assume that the annotated object has a structured compliant with *Hypertext Markup Language (HTML)* (W3C, 1999), as an example, and that they can use HTTP (Fielding et al., 1999) to transport annotations. In conclusion, they rely on both of these assumptions for adding, storing and managing annotations. On the contrary, FAST cannot assume that it is dealing with either HTML documents or the HTTP protocol, but it has to avoid any constraints concerning both the annotated information resource and the available protocols. The only assumption about information resources that FAST can make is that (Agosti et al., 2004):

each information resource is uniquely identified by a *handle*, which is a name assigned to an information resource in order to identify and facilitate the referencing to it.

This assumption is coherent with the assumption made by Rigaux and Spyrtos (2004) and Gueye et al. (2004) who refer to and compose documents only by identifiers and annotate them with metadata from a taxonomy of terms.

Over the past years, various syntaxes, mechanisms, and systems have been developed in order to provide handles or identifiers for information resources. The mechanisms and the standards discussed in the following sections are all suitable to be used as handles, according to the assumption made above.

3.1.1 Handles and Identifiers

The following sections introduce various solutions for uniquely identifying information resource, proposed by different organizations and standardization bodies.

URI, URN, and URL

The *Internet Engineering Task Force (IETF)*¹ defines:

- the *Uniform Resource Identifier (URI)* (Berners-Lee, 1994b; Berners-Lee et al., 1998; Kunze, 1995; Mealling and Denenberg, 2002);

¹<http://www.ietf.org/>

- the *Uniform Resource Name (URN)* (Berners-Lee, 1994b; Kunze, 1995; Mealling and Denenberg, 2002; Moats, 1997; Sollins and Masinter, 1994);
- the *Uniform Resource Locator (URL)* (Berners-Lee, 1994a,b; Kunze, 1995; Mealling and Denenberg, 2002; Sollins and Masinter, 1994).

An URI is a compact string of characters for identifying an abstract or physical resource. URIs are characterized by the following definitions, as specified in (Berners-Lee et al., 1998):

- *uniform*: it allows different types of resource identifiers to be used in the same context, even when the mechanisms used to access those resources may differ; it allows uniform semantic interpretation of common syntactic conventions across different types of resource identifiers; it allows introduction of new types of resource identifiers without interfering with the way that existing identifiers are used; and, it allows the identifiers to be reused in many different contexts, thus permitting new applications or protocols to leverage a pre-existing, large, and widely-used set of resource identifiers;
- *resource*: a resource can be anything that has identity. Not all resources are network “retrievable”; e.g., human beings, corporations, and library books can be considered resources as well. The resource is the conceptual mapping to an entity or set of entities. Thus, the resource does not necessarily have to correspond to the mapped entity at any given time, instead it is the conceptual mapping itself. In conclusion, a resource can remain constant even when its content—the entities to which it currently corresponds—changes over time, provided that the conceptual mapping is not changed in the process;
- *identifier*: an identifier is an object that can act as a reference to something that has an identity. In the case of URI, the object is a sequence of characters with a restricted syntax.

The term URL refers to the subset of URIs that identify resources via a representation of their primary access mechanism (e.g., their network “location”), rather than identifying the resource by name or by some other attribute(s) of that resource. The term URN refers to the subset of URI that are required to remain globally unique and persistent even when the resource ceases to exist or becomes unavailable (Berners-Lee et al., 1998).

DOI

The *International DOI Foundation (IDF)*² defines the *Digital Object Identifier (DOI)*, which is an *actionable identifier* for intellectual property on the Internet (Paskin, 2004). Firstly, the IDF defines an identifier from different viewpoints:

- (1) an identifier is *an unambiguous string or “label” that references an entity*. An example of such an identifier is the *International Standard Book Number (ISBN)*³, which is a unique number assigned to a title or edition of a book or other monographic publication (serial publications excluded) published or produced by a specific publisher or producer (ISO, 1992);

²<http://www.doi.org/>

³<http://www.isbn-international.org/>

- (2) an identifier is a *numbering scheme*, such as a formal standard, an industrial convention, or an arbitrary internal system. This numbering scheme provides a consistent syntax for generating individual labels or identifiers, as stated in (1), that denote and distinguish separate members of a class of entities; we can still use the ISBN, as an example. The intention is establishing a one-to-one correspondence between the members of a set of labels (numbers), and the members of the set counted and labelled. An important point is that the resulting number is simply a label string, but it does not create a string that is “actionable” in a digital or physical environment without further steps being taken;
- (3) an identifier is an *infrastructure specification*: a syntax by which any identifier as stated in (1) can be expressed in a suitable form for use with a specific infrastructure, without necessarily specifying a working mechanism; an example of such an identifier is the URI. This is sometimes known as creating an “actionable identifier” which means that in the context of that particular piece of infrastructure, the label can now be used to perform some action;
- (4) an identifier is a *system for implementing labels (identifiers as stated in (1)) through a numbering scheme (identifiers as stated in (2)) in an infrastructure using a specification (identifiers as stated in (3)) and management policies*. This sense of “identifier” denotes a fully implemented identification mechanism that includes the ability to incorporate labels, conforms to an infrastructure specification, and adds to these practical tools for the implementation such as registration processes, structured interoperable metadata, and an administrative mechanism.

The DOI is a system which provides a mechanism to interoperably identify and exchange intellectual property in the digital environment. It is an identifier as stated in (4) above. One of the components is a syntax specification (identifier as stated in (2)). DOI conforms to a URI (identifier as stated in (3)) specification. It provides an extensible framework for managing intellectual content based on proven standards of digital object architecture and intellectual property management, and it is an open system based on non-proprietary standards (Paskin, 2004).

OpenURL

The *National Information Standards Organization (NISO) Committee AX*⁴ defines the OpenURL framework, which aims at standardizing the construction of “packages of information” and the methods by which they may be transported over networks. The intended recipients of these packages are networked service providers that deliver context-sensitive services. To enable such services, each package describes not only the resource for which services are needed, but also the network context of a reference to the resource in question (NISO, 2004a,b). Thus, OpenURL is a standard syntax for transporting information (metadata and identifiers) about one or multiple resources within URLs, i.e. it provides a syntax for encoding metadata and identifiers, limited to the world of URLs (Paskin, 2004).

⁴http://www.niso.org/committees/committee_ax.html

PURL

The *Online Computer Library Center (OCLC)* defines the *Persistent URL (PURL)*⁵, which is an URL from a functional standpoint. However, instead of pointing directly to the location of an Internet resource, a PURL points to an intermediate resolution service, that associates the PURL with the actual URL and returns that URL to the client as a standard HTTP redirect. The client can then complete the URL transaction in the normal fashion.

The *PURL-based Object Identifier (POI)*⁶ is a simple specification for resource identifiers based on the PURL system, closely related to the use of the *Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH)* defined by the *Open Archives Initiative (OAI)*⁷ (OAI, 2004). The POI is a relatively persistent identifier for resources that are described by metadata “items” in OAI-compliant repositories. Where this is the case, POIs are not explicitly assigned to resources – a POI exists implicitly because an OAI “item” associated with the resource is made available in an OAI-compliant repository. However, POIs can be explicitly assigned to resources independently from the use of OAI repositories and the OAI-PMH, if desired.

Lexical Signatures

This section describes a proposal for identifying Web documents that is different from what has been discussed up to now. Indeed, the *Lexical Signatures (LSs)*, proposed by Park et al. (2004), aim at uniquely identifying a Web document by means of a signature extracted from its content and not by means of using some identifiers, as in the case of URLs.

LSs are a bunch of keywords extracted from a Web document that are used as query for a *Search Engine (SE)*. In this way, if a Web document cannot be found by means of its URL, then the LS lexical signature of the document can be submitted to a SE in order to search and locate the document anyway. Park et al. (2004) state that LSs should have the following features:

- LSs should easily locate the requested document and, if a SE retrieves more than one document, the requested one should be the top-ranked document;
- LSs should be useful enough to find relevant information when the precise documents that are being searched for are lost;
- LSs should be robust enough to find documents that have been slightly modified;
- new LSs should have minimal overlap with existing LSs;
- LSs should have minimal search engine dependency.

Park et al. (2004) propose several algorithms for extracting keywords from Web documents and computing effective LSs.

In conclusion, LSs represent an interesting alternative with respect to various kinds of identifiers and handles, due to the fact that LSs offer the possibility to almost uniquely identify a Web document by exploiting its own content.

⁵<http://purl.oclc.org/>

⁶<http://www.ukoln.ac.uk/distributed-systems/poi/>

⁷<http://www.openarchives.org/>

3.2 Modelling Approach

Our modelling approach is aimed at addressing the following issues:

- how to model annotations, described in Section 3.2.1;
- how to relate annotations to digital objects, explained in Section 3.2.2.

Finally, Section 3.2.3 gathers the suggestions and the choices introduced in Sections 3.2.1 and 3.2.2 in order to define a conceptual model for the annotation.

3.2.1 How to Model Annotations

As we discussed in Chapter 2, the annotation is a very complex concept with rich semantics. In order to face this complexity, we introduce the distinction between the *meaning of annotation* and the *sign of annotation* (Agosti and Ferro, 2003a; Agosti et al., 2004).

The *meaning of annotation* is a main aspect concerning the concept of annotation, which identifies conceptual differences within the semantics of the annotation or part of it. For example, looking at all of these the different points of view concerning annotations introduced in Section 2.2, we can see that they correspond to different meanings of annotation. Furthermore, given a viewpoint, we can identify different meanings of annotation within it: for example, within the viewpoint called “annotation as content” we can point out, at least, three different meanings of annotation:

- *comprehension and study*: annotating a document is a way to investigate and understand a concept better. This process principally involves a private scope, because the recipient of an annotation is the person who created it, although other people reading an annotated document could benefit from existing annotations;
- *interpretation and elucidation*: annotating a document could be a way of adding comments and explaining sentences within it. The aim is to make it more comprehensible and to exchange ideas on a topic; an example could be an expert in literature who explains and annotates the Divine Comedy. This process principally involves a public scope, because the recipients of an annotation are people who are not necessarily related to the creator of the annotation;
- *cooperation and revision*: a team of people could annotate a document for various purposes, as they are working on a common document or they are reviewing someone else’s work; annotating a text is thus a way of sharing ideas and opinions in order to improve a text. This process principally involves a shared scope, because the recipient of an annotation is a team of people working together on a given subject.

As a further example, if we consider annotations as metadata, the meaning of the annotation could be provided by some standard metadata specification, such as the ones provided by the *Dublin Core Metadata Initiative (DCMI)*⁸ which is concerned with the development of interoperable online metadata standards. Finally, it is also possible to organize the meanings of annotations according to some kind of hierarchy,

⁸<http://dublincore.org/>

such as a taxonomy or an ontology, in order to provide navigation capabilities among different meanings of annotation.

Even if the meanings of annotation are, in some way, similar to the dimensions of annotation proposed by Marshall (1998), they differ because they are in fact different conceptual and semantic facets of the annotation, while dimensions of annotation are a categorization that reflects the form which they may take on.

The *sign of annotation* is a way of representing a meaning of annotation, i.e. a way in which a meaning of annotation is materialized. For example, we can identify the following basic signs of annotations:

- *textual sign*: is a textual materialization of the semantics of an annotation and it is expressed by a piece of text added to a document or a piece of a document;
- *graphic sign*: is the graphic materialization of the semantics of an annotation and it is expressed by a graphic mark added to a document or a piece of a document;
- *reference sign*: is the hypertextual materialization of the semantics of an annotation and it is expressed by a link between two texts.

Also other signs of annotation could be added to this list, as audio or video signs.

Those basic signs can be combined together to express more complex signs of annotation. For example if we take a textual sign, which is an additional explanation of a concept, it can also be combined with some marks, which are graphic signs, in order to highlight the content which it refers to, as shown in Figure 3.2(a). As a further example, a reference sign can be used together with an arrow, known as a graphic sign. You will note that this arrow is pointing to the referred document; in addition it can be combined with a textual sign, that provides a further explanation, as shown in Figure 3.2(b).

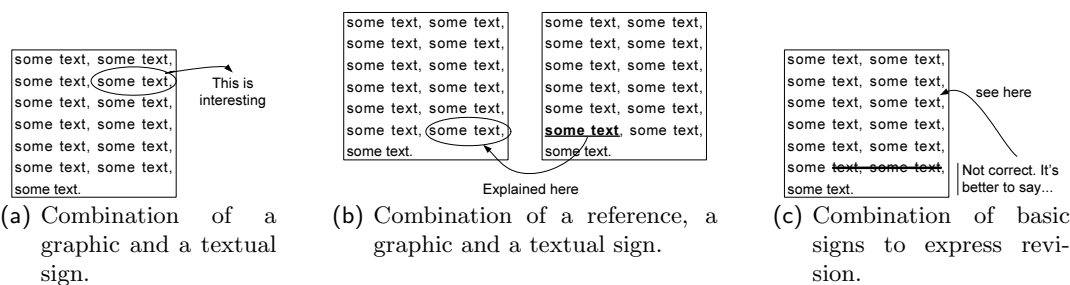


Figure 3.2: Combination of basic signs.

It is worth observing that the combination and compounding of those basic signs allows us to express the different meanings of annotation, explained above: for example, if a person is studying a document, he simply can use a graphic sign to highlight the important content, and so the “comprehension and study” meaning of annotation can be expressed. During the revision of an article, the author can use a graphic sign to delete some incorrect pieces of the text; then, a textual sign can be used to correct them; finally, a reference sign can be compounded with a graphic sign in order to indicate another piece of text that justifies the correction, as shown in Figure 3.2(c). In conclusion, the “cooperation and revision” meaning of annotation can be expressed in this way.

On the whole, an annotation is expressed by one or more signs of annotation, that in turn are characterised by one or more meanings of annotation, thus defining the overall semantics of the annotation.

The choice of explicitly distinguishing between the meaning and the sign of annotation is quite new in the field of annotations. Indeed, annotations are generally typed as a whole according to some pre-defined set of annotation types (Bottoni et al., 2004, 2003; Frommholz et al., 2003; Kahan and Koivunen, 2001; W3C, 2004a), but there is usually no means for describing the semantics of an annotation with the desired level of precision, whereas this is possible with the meanings of annotation. Furthermore, annotation types do not allow any kind of navigation among different types, while meanings of annotations can be organized in order to provide such facility of use.

Some interesting insides about the choice of distinguishing between meaning and sign of annotation can be gained from the field of *Human Computer Interaction (HCI)*. Indeed, Bottoni et al. (1999) deal with visual languages and define *Characteristic Structures (CSs)* as sets of image pixels forming functional or perceptual units whose recognition results in the association of that CS with a meaning. Then, they call *Characteristic Patterns (CPs)* the CSs along with descriptions of the CSs and a relation that associates descriptions to CSs and viceversa. The distinction between CSs and CPs resembles the distinction between sign and meaning of annotation; also Fogli et al. (2004) recognize this correspondence and say that “an annotation is a complex CS interpreted by a human as a CP”. On the other hand, Bottoni et al. (2003) adopts the CSs and CPs mechanism in the context of annotations too, but they use this mechanism in order to place annotations on information resources rather than to distinguish between the semantics and the materialization of annotations.

3.2.2 How to Link Annotations to Digital Objects

According to widely accepted terminology, we adopt the term *Digital Object (DO)* in order to refer to objects managed by an IMS.

Paskin (2004) defines the DO as “a data structure whose principal components are digital material, or data, plus a unique identifier for this material”. Gonçalves et al. (2004a) say that “information in digital libraries is manifest in terms of *digital objects*, which can contain textual or multimedia content (e.g., images, audio, video), and *metadata*” and they define a DO as a tuple constituted by a unique handle, structured contents, and metadata (Gonçalves et al., 2004a, p. 294). Finally, Bottoni et al. (2003) define the DO as a typed tuple of attribute–value pairs with, at least, two mandatory attributes: a unique identifier and the actual content of the DO; furthermore, Bottoni et al. (2003) consider annotations as DOs with specific attributes, i.e. annotations are specialised DOs.

All of these definitions of DO agree with the assumption that FAST can refer to DOs by using handles. Thus, the mechanism for linking annotations to DOs is based on the usage of unique handles to DOs.

In the following, we need terminology to distinguish between the DOs managed by the IMS, that we call *documents*, and the DOs managed by FAST, that are *annotations*; when we use the generic term DO, we mean a DO that can be either a document or an annotation.

Once we have decided to use handles as basic mechanism for linking annotations to DOs, we still have to consider what kind of links an annotation can have with a DO.

Annotations can be linked to DOs with two main types of links:

- *annotate link*: an annotation annotates a DO, which can be a document or another annotation.
The “annotate link” is intended only to allow an annotation to annotate one or more parts of a given DO. Thus, this kind of link lets the annotation express *intra-DO relationships*, meaning that the annotation creates a relationship among the different parts of the annotated DO;
- *relate-to link*: an annotation relates to a DO, which can be a document or another annotation.
The “relate-to link” is intended only to allow an annotation to relate to one or more parts of other DOs, but not the annotated one. Thus, this kind of link lets the annotation express *inter-DO relationships*, meaning that the annotation creates a relationship between the annotated DO and the other DOs related to it.

With respect to these two main types of link, we introduce the following constraint:

an annotation must annotate one and only one DO, which can be either a document or another annotation, that is an annotation must have one and only one “annotate link”.

This constraint means that an annotation can be created only for the purpose of annotating a DO and not exclusively for relating to a DO. Then, an annotation can annotate one and only one DO, because the “annotate link” expresses *intra-DO relationships* and thus it cannot be mutual to multiple DOs different from the annotated one. Finally, this constraint does not prevent the annotation from relating to more DOs, i.e. from having more than one “relate-to link”.

3.2.3 Conceptual Model of the Annotation

In order to have a first formalization of the annotation, able to describe all the involved entities and the relationships among them, we have decided to represent them using a conceptual modelling tool of general use as the *Entity-Relationship (ER)* model is. This way, we can illustrate and discuss the choices of our modelling approach with the aid of a clear and easily understandable modelling tool.

The proposed conceptual schema is shown in Figure 3.3 and, as with the previous discussion, it is centred around two main issues: how to model annotations and how to link annotations to digital objects. The next sections describe these two issues in more detail.

How to Model Annotations

The ANNOTATION entity represents the abstraction of the annotation, i.e. it expresses the existence of an object capable of annotating another object, without having to specify its characteristics any further. This is the pivotal entity, which provides the basis for modelling annotations.

The ANNOTATION entity has the following attributes: **ID** is a unique identifier for the annotation, e.g. an URI or a DOI; **Created** and **Modified** represent, respectively,

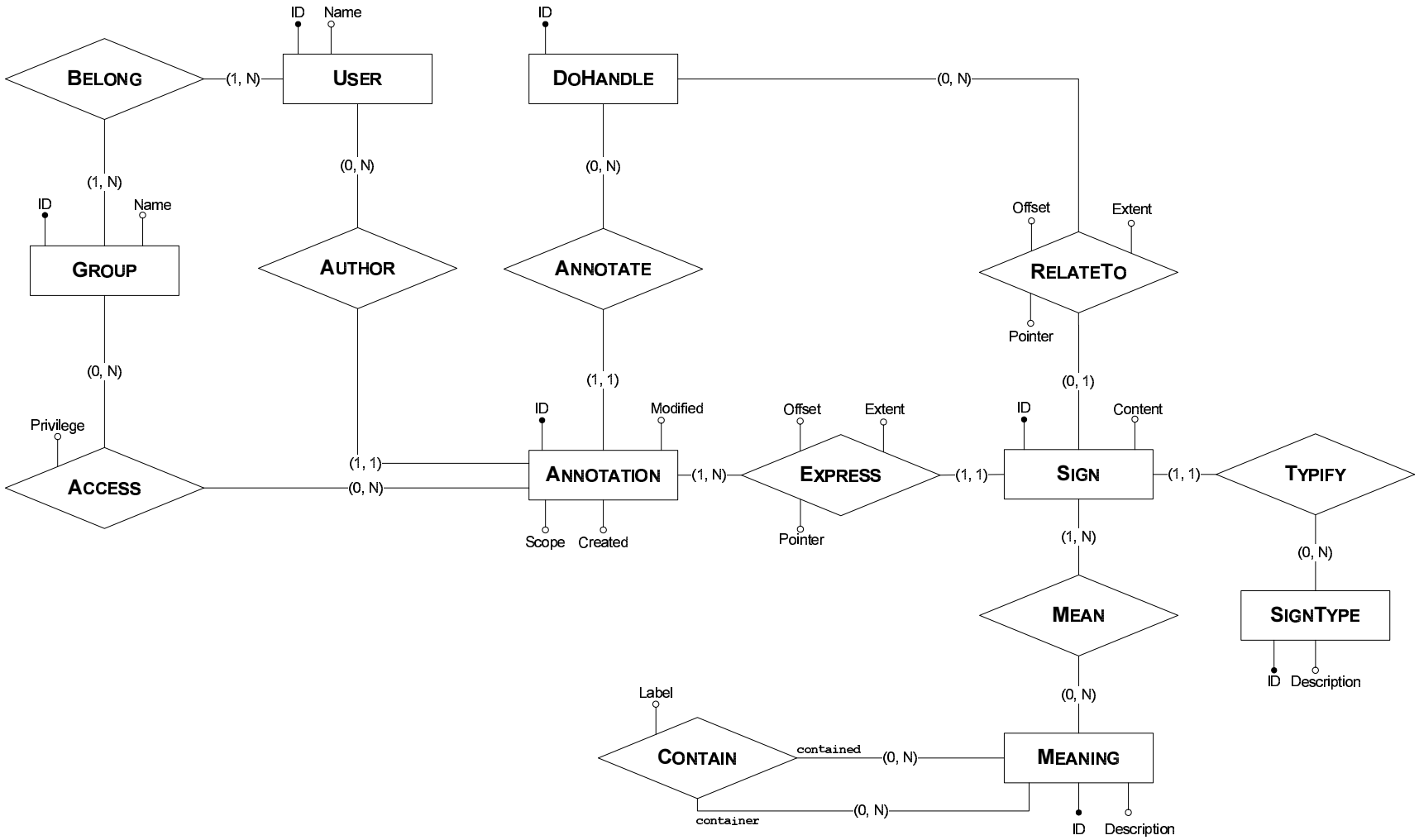


Figure 3.3: Entity-Relationship schema for modelling annotations.

the creation date and the last modified date of the annotation; and **Scope** specifies if the annotation is private, shared, or public.

The discussion carried out in the previous sections showed that the **ANNOTATION** entity alone is not sufficient for covering the semantics of the general concept of annotation, so it needs to be partnered with the two other entities **MEANING** and **SIGN**, respectively representing the meaning of annotation and the sign of annotation.

The **MEANING** entity is characterised by a unique identifier, called **ID**, and also by a **Description** attribute, which describes the meaning of annotations. On the **MEANING** entity there is a recursive relationship, called **CONTAIN**, that expresses the existence of both broader and narrower meanings. For example, **CONTAIN** allows us to model the fact that within the “annotation as content” meaning we can further distinguish a “comprehension and study” meaning or a “interpretation and elucidation” meaning, as discussed in the previous section. Thus, the meanings of annotation can be organised into some sort of hierarchy. Subsequently, some navigation facilities within this hierarchy can be provided to the user. The **CONTAIN** relationship expresses the fact that a meaning may be contained in one or more other meanings and that it may contain one or more other meanings. In conclusion, **CONTAIN** allows us to define a graph of meanings of annotation. The **Label** attribute describes the kind of relationship between two meanings of annotation, if necessary.

The **SIGN** entity has a unique identifier, called **ID**, and a **Content** attribute, which represent the actual content of the sign of annotation, e.g. a piece of text or an image. The **SIGNTYPE** entity describes the kind of a sign of annotation, e.g. a textual sign or a graphic sign, and makes it possible to correctly interpret the **Content** attribute of a **SIGN** entity. The **SIGNTYPE** entity is connected to the **SIGN** entity by means of the **TYPIFY** relationship, which expresses the fact that a **SIGN** must have exactly one **SIGNTYPE**, while a **SIGNTYPE** may specify one or more **SIGN** entities.

Two relationships, called **EXPRESS** and **MEAN**, allow the three entities **ANNOTATION**, **MEANING** and **SIGN** to work together in order to define the semantics and the materialization of an annotation. The **EXPRESS** relationship denotes that an **ANNOTATION** entity has to be expressed by one or more **SIGN** entity, and that a given **SIGN** entity has to be employed one and only one in order to express an **ANNOTATION** entity. The attributes of **EXPRESS** allow us to physically identify which part of the **DO** has to be annotated. In particular, the **Pointer** attribute identifies a portion of a **DO**, e.g. it could be an **XPath**⁹ expression when using an XML document; the **Offset** attribute selects a starting offset with respect to the portion identified by **Pointer**, e.g. the initial character within an XML element; finally, the **Extent** attribute specifies the size of the sign of annotation, e.g. the number of characters that are annotated within the portion identified by **Pointer** starting from **Offset**.

The **MEAN** relationship expresses the fact that a **SIGN** entity has to be related to one or more **MEANING** entities and that a **MEANING** entity may characterise one or more **SIGN** entities.

How to Link Annotations to Digital Objects

As explained in the previous section, the **ANNOTATION** entity represents the abstraction of an object capable of annotating another object. In order to connect annotations to **DOs** we need also an entity that represents the abstraction of an object that can be

⁹<http://www.w3.org/XML/Linking>

annotated. This entity is called DOHANDLE and represents a DO by means of using a handle, according to the assumption made in Section 3.1 concerning the possibility of using handles to identify DOs. Thus, the cornerstones for connecting annotations to DOs are the ANNOTATION and DOHANDLE entities which represents the fact that there are two kinds of related objects: DOs that can be annotated and annotations that annotate those DOs.

The relationship between annotations and annotated DOs is represented by the ANNOTATE relationship, which links an ANNOTATION entity to the DOHANDLE entity that it annotates. This relationship expresses the fact that an annotation must annotate one and only one DO and that a DO may be annotated by one or more annotations, according to the constraint introduced in Section 3.2.2 concerning the “annotation link”.

Once we have annotated a DO, the annotation itself can be considered as a DO eligible to be annotated. Thus, the conceptual schema has the following additional constraint: once the annotation has been created, an occurrence of the DOHANDLE entity corresponding to the annotation have to be added, in order to allow the newly created annotation to be annotated as well. Users can therefore create not only sets of annotations concerning a DO, but also threads of annotations, i.e. annotations which reply to one another. These threads of annotations are the basis for actively involving users with the system and for enabling collaboration.

The RELATETO relationship is used for the purpose of relating the annotation to other DOs, thus making the “relate-to link” introduced in Section 3.2.2. The RELATETO relationship associates a sign of annotation with the DO it refers to. In addition, the RELATETO relationship allows a SIGN entity to refer or not to a DO, while a DO may be referred to by one or more signs of annotation. The attributes of RELATETO have the same meaning of the attributes of EXPRESS. In conclusion, the EXPRESS relationship specifies the origin of the link and the RELATETO relationship identifies the destination of the link, in the case of inter-DO relationships.

The USER entity represents a user, granted by the system. The AUTHOR relationship relates an annotation with its author; a user may create one or more annotations, while an annotation must be created by one and only one user.

Finally, the GROUP entity represents a users’ group, related to users by means of the BELONG relationship: the USER entity has to belong at least to one GROUP of user – or more, if necessary – and a GROUP entity contains one or more USER. The ACCESS relationship allows an ANNOTATION entity to be shared by one or more groups of users, and a GROUP may share one or more ANNOTATION entities. The ACCESS relationship has the **Privilege** attribute, which specifies the privileges, e.g. read or modify, granted to a GROUP sharing the annotation.

Overall Comments

Table 3.1 provides us with an overview and an interpretation of the conceptual schema of figure 3.3. It analyzes the conceptual schema along two dimensions: the first one regards the distinction between the Information Management System and the Flexible Annotation Service Tool; the second one concerns the role played by the entities and the relationship, i.e. if they aim to interrelate objects or to model objects. Both entities and relationship are listed in table 3.1: note entities are listed in the upper part of a box and relationships are listed in the lower part of a box, separated by a line.

	Information Management System	Flexible Annotation Service Tool
Object Relating		ANNOTATION DOHANDLE GROUP USER <hr/> ACCESS ANNOTATE AUTHOR RELATETo
Object Modelling	Proprietary constructs	ANNOTATION MEANING SIGN SIGNTYPE <hr/> CONTAIN EXPRESS MEAN TYPIFY

Table 3.1: Summary of the proposed conceptual schema.

Table 3.1 highlights that the ANNOTATION, DOHANDLE, GROUP, and USER entities, together with the ACCESS, ANNOTATE, AUTHOR and RELATETo relationships, represent the bridge between the IMS and the FAST. Then, table 3.1 clearly visualizes that the ANNOTATION entity plays a double role: it works both as a bridge between the IMS and the FAST and as a cornerstone for modelling the annotation. Finally, table 3.1 stresses that we have a bunch of entities and relationship for modelling the annotation, but the modelling of the documents is completely remitted to the proprietary constructs adopted by the IMS into which the FAST is integrated.

The proposed conceptual schema is quite innovative, because it describes the annotation with a degree of detail not present in other similar proposals. Furthermore, it provides us with great flexibility, due to the fact that we can express the different aspects of an annotation, couple them together and, at the same time, it does not constrain us to fixed types of annotations. Thus, our proposal represents an enhancement and a generalization with respect to the models proposed by (Kahan and Koivunen, 2001; Sannomiya et al., 2001). Finally, being a conceptual schema, our model can be easily mapped to different models, such as a relational schema, a RDF schema or a XML schema; this way it provides us with great flexibility when dealing with different architectural choices.

Chapter 4

Formal Model of Annotation

This chapter introduces and explains the proposed formal model for describing the annotation, which relies on the concepts discussed in Chapter 3. In particular, the modelling approach presented in Section 3.2 provides the basic notions, as for example the meaning and the sign of annotation, needed to understand this formal model; the ER schema of annotation depicted in figure 3.3 on page 29 can be used as a useful map of the concepts formalized in the following sections.

4.1 Document, Annotation, and Digital Object Sets

As introduced in Section 3.2.2, we will need to deal with two kinds of DOs, that are *documents* and *annotations*: the term *document* means a generic DO managed by the IMS, while the term *annotation* means the specific kind of DO that is managed by the FAST. The following definition introduces the different sets of DOs we will need to deal with.

Definition 4.1: Let us define the following sets:

- D is the **set of documents** and $d \in D$ is a generic **document**. U_D is the **universe set of documents**, which is the set of all the possible documents, so that $D \subseteq U_D$.
- A is the **set of annotations** and $a \in A$ is a generic **annotation**. U_A is the **universe set of annotations**, which is the set of all the possible annotations, so that $A \subseteq U_A$.
- $DO = D \cup A$ is the **set of digital objects** and $do \in DO$ is either a document or an annotation. $U_{DO} = U_A \cup U_D$ is the **universe set of digital objects**, so that $DO \subseteq U_{DO}$.

Note that DO (capital italic letters) is the set of digital objects just defined, DO (capital letters) is the acronym for Digital Object and do (lowercase italic letters) is a digital object $do \in DO$.

The universe sets U_D , U_A , and U_{DO} are abstract sets, since they contain all the possible needed objects, whether they actually exists or not in any given moment; on the other hand, the sets D , A , and DO are tangible sets that contain the objects that already exists in a given moment: if we pick out an element from D , A , or DO we are dealing with a DO that has been created even before we start working with it; in other words, the element already exists. The D , A , and DO sets are, in certain sense,

time-variant sets, since we can add, delete or modify elements of these sets with the passing of time. On the other hand, the U_D , U_A , and U_{DO} sets are, in certain sense, *time-invariant sets*, since they already contain every possible object we may need to deal with.

As discussed in Sections 2.1.5 and 2.3, the time dimension is very important when we deal with annotations, because annotations must annotate a DO that already exists. Thus, we need some mechanism for rendering the time dimension explicit, if necessary. Consider the following examples, that make use of the set DO but have a more general validity:

- creation of a new DO:
 1. we start with the set of digital objects at time k : $DO(k)$;
 2. we create a new digital object, that is we pick out an element from the universe set of digital objects that does not belong to $DO(k)$ ¹: $do \in \overline{DO}(k) \subseteq U_{DO}$;
 3. we end up with a new set of digital objects at time $k+1$, which contains the newly created digital object²: $DO(k+1) = (DO(k) \cup \{do\}) \in 2^{U_{DO}}$.

Thus, we have the following temporal ordering:

$$\underbrace{\underbrace{DO(k)}_{\text{step 1}} \dashrightarrow \underbrace{do \in \overline{DO}(k)}_{\text{step 2}}}_{\text{time } k} \dashrightarrow \underbrace{DO(k+1) = DO(k) \cup \{do\}}_{\text{step 3}}_{\text{time } k+1}$$

both step 1 and 2 happen at time k , but at that time the newly created digital object does not yet belong to the set $DO(k)$ of digital objects at time k ; step 3 happens at time $k+1$ and represents the new set of digital objects that now also contains the newly created and existing digital object;

- deletion of an existing DO:
 1. we start with the set of digital objects at time k : $DO(k)$;
 2. we choose an existing digital object in the set of digital objects at time k : $do \in DO(k)$;
 3. we end up with a new set of digital objects at time $k+1$, which does not contain the previously chosen digital object³: $DO(k+1) = (DO(k) \setminus \{do\}) \in 2^{U_{DO}}$.

Thus, we have the following temporal ordering:

$$\underbrace{\underbrace{DO(k)}_{\text{step 1}} \dashrightarrow \underbrace{do \in DO(k)}_{\text{step 2}}}_{\text{time } k} \dashrightarrow \underbrace{DO(k+1) = DO(k) \setminus \{do\}}_{\text{step 3}}_{\text{time } k+1}$$

¹Given a set X and a set U_X such that $X \subseteq U_X$, the *complement set* of X with respect to U_X is the set $\overline{X} = \{x \in U_X \mid x \notin X\}$.

²Given a set X , 2^X is the *power set* of X , that is the set consisting of all subsets of X , inclusive of the empty set \emptyset and X itself. The cardinality of 2^X is given by $|2^X| = 2^{|X|}$.

³Given two sets X and Y , the *difference of X with respect to Y* is the set $X \setminus Y = \{x \in X \mid x \notin Y\}$.

both step 1 and 2 happen at time k ; step 3 happens at time $k + 1$ and represents the new set of digital objects which does not contain the previously existing digital object;

- modification of an existing DO:
 1. we start with the set of digital objects at time k : $DO(k)$;
 2. we choose an existing digital object in the set of digital objects at time k : $do \in DO(k)$;
 3. we choose a new digital object, which is the modified version of the previously chosen digital object and does not belong to $DO(k)$: $do' \in \overline{DO}(k) \subseteq U_{DO}$;
 4. we end up with a new set of digital objects at time $k + 1$, which contain the modified version of the digital object: $DO(k + 1) = (DO(k) \setminus \{do\} \cup \{do'\}) \in 2^{U_{DO}}$.

Thus, we have the following temporal ordering:

$$\begin{array}{c}
 \text{time } k \qquad \qquad \qquad \text{time } k+1 \\
 \overbrace{\underbrace{DO(k)}_{\text{step 1}} \dashrightarrow \underbrace{do \in DO(k)}_{\text{step 2}} \dashrightarrow \underbrace{do' \in \overline{DO}(k)}_{\text{step 3}}} \dashrightarrow \underbrace{DO(k + 1) = DO(k) \setminus \{do\} \cup \{do'\}}_{\text{step 4}}
 \end{array}$$

both step 1, 2 and 3 happen at time k , but at that time the modified digital object does not yet belong to the set $DO(k)$ of digital objects at time k ; step 4 happens at time $k + 1$ and represents the new set of digital objects that now contains the modified digital object.

These three basic examples reveal our strategy for addressing the time dimension:

- time k : identify an initial set $DO(k)$ to work with;
- time k : identify the digital objects to work with, that can belong to $DO(k)$ or not. If the identified digital objects belong to $DO(k)$, then they already exist; on the other hand, if the identified digital objects do not belong to $DO(k)$, then they do not exist yet and this step represent their creation;
- time $k + 1$: identify the new set $DO(k + 1)$ that results from performing the appropriate operations on the set and the digital objects previously identified.

In all of the cases, both $DO(k)$ and $DO(k + 1)$ contain only digital objects that already exist: this mechanism allows us to state without ambiguity which objects we are dealing with in any instant and when they come into play.

$DO(k)$ and $DO(k + 1)$ univocally identify the digital objects we are dealing with, which are given by $DO(k) \triangle DO(k + 1)$ ⁴. In particular, the deleted digital objects are given by $DO(k) \setminus DO(k + 1)$ and the newly created digital objects are given by $DO(k + 1) \setminus DO(k)$. Thus, we can talk about the digital objects identified by the transition from $DO(k)$ to $DO(k + 1)$. We assume that the operations previously shown are atomic, i.e. no operation can occur during the execution of another operation, as not to have concurrency issues, race conditions, or deadlocks.

⁴Given two sets X and Y , the *symmetric difference between X and Y* is the set $X \triangle Y = (X \setminus Y) \cup (Y \setminus X) = (X \cup Y) \setminus (X \cap Y)$.

In conclusion, this mechanism provides us with a means to clearly identify what objects are involved in a given operation, when they come into play, and the ordering among the different steps of an operation.

Note that it is very difficult to find literature on the topic that takes into account an explicit formalization of the time dimension: to the best of our knowledge, our proposal agrees with the idea proposed by Rigaux and Spyratos (2004). Indeed, our mechanism extends, formalizes, and makes explicit what is implicit in the approach adopted by Rigaux and Spyratos (2004) when they say: “in order to define a document formally, we assume the existence of a countably infinite set \mathcal{D} whose elements are used by all authors for identifying the created documents. . . in fact, we assume that the creation of a document is tantamount to choosing a (new) element from \mathcal{D} ”. Indeed, the set \mathcal{D} used by Rigaux and Spyratos (2004) corresponds to the set U_{DO} of definition 4.1 and the creation of a document corresponds to the transition from $DO(k)$ to $DO(k+1)$, where an object from U_{DO} is chosen and it is added to $DO(k)$ thus originating $DO(k+1)$, as explained above. Furthermore, we provide a formal mechanism for describing also the deletion and the modification of a DO.

In the following, we will use the notation $DO(k)$ that explicitly points out the time dimension only when needed; otherwise we will use the simpler notation DO , without explicitly pointing out the time dimension. We will also use a similar notation for the other sets we will define in the following.

4.2 Handle

As discussed in Section 3.1, each DO is uniquely identified by means of an *handle*, whose examples are presented in Section 3.1.1.

Definition 4.2: H is the **set of handles** such that $|H| = |DO|$ and $h \in H$ is a generic **handle**. U_H is the **universe set of handles**, which is the set of all the possible handles, such that $|U_H| = |U_{DO}|$; it follows that $H \subseteq U_H$. We define a bijective function $h : U_H \rightarrow U_{DO}$ which maps an handle to the DO identified by it⁵:

$$\forall do \in U_{DO}, \exists! h \in U_H \mid h(h) = do \Rightarrow h^{-1}(do) = h$$

The relationship between the sets H and U_H is the same as the one between the sets DO and U_{DO} , described in section 4.1.

4.3 Author and Group of Authors

DOs – both documents and annotations – always have an author who created them. The author, in turn, belongs to one or more groups of authors.

Definition 4.3: Let us define the following sets:

- AU is the **set of authors** and $au \in AU$ is a generic author; U_{AU} is the **universe set of authors**, which is the set of all the possible authors, so that $AU \subseteq U_{AU}$. We

⁵ $\exists!$ is the *unique existential quantifier*, and it is read “there exists a unique . . . such that . . .”.

define a function $au : AU \rightarrow 2^H$ which maps an author to the handles to the DOs authored by him. The following constraint must be adhered to:

$$\forall au \in AU, au(au) \neq \emptyset$$

that is each author in AU must author, at least, one DO;

- $GR \subseteq 2^{AU}$ is the **set of groups of authors** and $G \in GR$ is a generic group of authors; $U_{GR} = 2^{U_{AU}}$ is the **universe set of groups of authors**, which is the set of all the possible groups of authors, so that that $GR \subseteq U_{GR}$. We define a function $gr : AU \rightarrow 2^{GR}$ which maps an author to groups of authors he belongs to. The following constraint must be adhered to:

$$\forall au \in AU, gr(au) \neq \emptyset$$

that is each author in AU must belong to, at least, one group of authors.

The relationship between the sets AU , GR and U_{AU} , U_{GR} is the same as the one between the sets DO and U_{DO} , described in section 4.1.

Note that:

- a DO may have, in general, more authors, that is there may exist $au_1, au_2 \in AU \mid au(au_1) \cap au(au_2) \neq \emptyset$. Obviously, an author may own more DOs, since $au(au)$ is an element of the power set of H ;
- the constraint on the gr function can be expressed also as $\forall au \in AU, \exists G \in GR \mid au \in G$. Obviously, an author may belong to more groups of authors, since GR is a subset of the power set of AU or, equivalently, since $gr(au)$ is an element of the power set of GR .

4.4 Stream

DOs can be very different – texts, images, audio, videos, hypertexts, multimedia objects, and so on – and also the way in which their structure and content is modelled and expressed can widely vary across different conceptual and logical models of DL and DO. Nevertheless, many of such models share the idea that beyond representing the structure of the DO the model has to take into account also a mechanism for representing the actual content of the DO.

For example, in the *Document Model for Digital Libraries (DoMDL)* (Castelli and Pagano, 2002a,b) each DO can be associated with many different *manifestation* entities, that represent files containing different parts of the DO itself. Both Navarro and Baeza-Yates (1997) and Gonçalves et al. (2004a) use the notion of *stream*, which is an ordered sequence of symbols representing the actual content of a DO or part of it. Finally, Bottoni et al. (2003) defines the content of a DO as a function from a set of indices to a set representing the vocabulary of the symbols.

The following definition introduces the concept of stream in order to represent the actual content of a DO or of part of it. The definition of stream is inspired by (Gonçalves et al., 2004a; Navarro and Baeza-Yates, 1997) but with some differences which will be discussed below.

Definition 4.4: A **stream** sm is a finite sequence⁶:

$$sm : D = \{1, 2, \dots, n\} \rightarrow \Sigma, \quad n \in \mathbb{N}$$

where Σ is the *alphabet of symbols*. In particular, we can distinguish two main kinds of stream:

- **logical stream** lsm : is a stream in which each element $\sigma \in \Sigma$ represents a *logical symbol* within the stream;
- **physical stream** psm : is a stream in which each element $\sigma \in \Sigma$ represents a *physical symbol* within the stream.

We also allow the existence of an **empty stream** $esm = \emptyset$.

SM is the **set of streams** and $sm \in SM$ is a stream. U_{SM} is the **universe set of streams**, that is the set of all the possible streams. It follows that $SM \subseteq U_{SM}$.

The relationship between the sets SM and U_{SM} is the same as the one between the sets DO and U_{DO} , described in section 4.1.

The stream is required to be neither a surjective nor an injective function. We can exploit the non surjectivity of the stream in order to use standard sets – characters, numbers, and so on – as a codomain for a stream; otherwise, if the function was constrained to be surjective, we would have been forced to use an “ad hoc” codomain for each different stream. On the other hand, since the stream is a not injective function, it is not invertible: unfortunately, given a symbol, we cannot trace this symbol back to its position within the stream.

For example, if we consider the following piece of text

1	2	3	4	5	6	7	8	9	10
T	i	t	l	e	␣	T	e	x	t

then we can define the stream

$$sm : D = \{1, 2, \dots, 10\} \rightarrow \Sigma = \{A, B, \dots, Z, a, b, \dots, z, \sqcup\}$$

such that $sm(1) = T, sm(2) = i, \dots, sm(10) = t$. Note that if the stream was constrained to be surjective, we should use a codomain Σ' constituted only by the letters of the piece of text shown above, i.e. $\Sigma' = \{T, i, t, l, e, \sqcup, x\}$. In any case, from a given symbol, for example “t”, we cannot unambiguously determine its position in the stream, because the stream is not injective – “t” is given by both $sm(3)$ and $sm(10)$.

Now we will discuss the distinction between logical and physical streams by means of an example. Although the map shown above between natural numbers and letters is quite intuitive, it should be pointed out that the elements of the set Σ are symbols that abstract the underlying encoding of the text. For example, if we consider an ASCII text, each element of the set Σ corresponds to exactly one byte in the physical text stream; thus, we should use the codomain $\Sigma_{ASCII} = \{\$41, \$42, \dots, \$5A, \$61, \$62, \dots, \$7A, \$20\}$ instead of Σ in order to represent the actual stream. On the other hand, in the case of a UNICODE text, each element of the set Σ corresponds to two bytes in the physical text

⁶Note that Gonçalves et al. (2004a) admits also the possibility of infinite sequences (definition 1 plus definition A3), which are not feasible in a software system.

stream; thus, we should use the codomain $\Sigma_{\text{UNICODE}} = \{\$00\$41, \$00\$42, \dots, \$00\$5A, \$00\$61, \$00\$62, \dots, \$00\$7A, \$00\$20\}$ instead of Σ . In this latter case we are forced to define the elements of Σ_{UNICODE} as two-bytes pairs in order to map the indices of D into the symbols of Σ_{UNICODE} ; thus, we would not be able to access each byte individually. If we would like to access each byte of the UNICODE stream, we should define the following domain $D' = \{1, 2, \dots, 20\}$ and codomain $\Sigma'_{\text{UNICODE}} = \{\$00, \$41, \$42, \dots, \$5A, \$61, \$62, \dots, \$7A, \$20\}$ for the stream, but in this case we would lose the correspondence with the ten letters of the piece of text, because two indices in D' would correspond to each letter of the piece of text. This example points out the fact that on one hand we have a logical stream, which represents the piece of text, and on the other hand there are one or more physical streams that represent the physical encoding of the piece of text shown above. Similar, and even more complex, considerations can be made in the case of audio, images, and video streams, where the complexity of such streams increases the choices available for representing them both in logical and in physical terms. Another example is the compression of streams, where more symbols in one stream correspond to less symbols in the other stream.

This observation points out the necessity of carefully defining the level of abstraction of a stream and the degree of detail that have to be adopted in defining streams. In other words, should we model the physical encoding of a stream or some more abstract representation of that stream? Depending on the case, both levels of abstraction can be needed: for example, when we do a macro-comparison of two digital libraries, we can use more abstract streams; on the other hand, if we want to precisely describe the functioning of some component of a digital library, as a repository, we need to use streams that better represent the physical encoding of the objects in the repository. However, past experience in the field of DBMSs teaches us that is better to keep distinct the logical level and the physical level. This is the motivation why in definition 4.4 we distinguish between logical streams and physical streams.

Note that Navarro and Baeza-Yates (1997) make use of logical streams but they do not specify much about physical streams, leaving them to the implementation of the system. Neither Bottoni et al. (2003) nor Gonçalves et al. (2004a) have addressed this problem at all, but in (Gonçalves and Fox, 2002) it turns out that streams are essentially identified by *Multipurpose Internet Mail Extensions (MIME)* types (Freed and Borenstein, 1996a,b,c; Freed et al., 1996; Moore, 1996) and thus, they are substantially physical streams. Finally, since the notion of *manifestation* used by Castelli and Pagano (2002a,b) refers to a physical file holding part of the content of a DO, they essentially use physical streams too and in a way that resembles the implementation of streams by Gonçalves and Fox (2002).

We believe that the research field of digital libraries also needs to clearly distinguish between the logical and physical level and that this distinction is a prerequisite for each formal model of DL which aims to be sufficiently clear, expressive and flexible. Moreover, an explicit and formal mechanism for modelling the relationship between logical and physical streams and the properties of such relationship is needed; this will be the subject of the next definition.

Definition 4.5: Given two streams $sm_1, sm_2 \in SM$, let us define the **stream mapping relation**:

$$SMR(sm_1, sm_2) = \{(i, j) \in D_{sm_1} \times D_{sm_2} \mid sm_1(i) \text{ is mapped to } sm_2(j)\}$$

Let us define the **stream mappings set** $SMS = \{\mathcal{SMR}(sm_1, sm_2)_i\}$ such that:

1. $\forall sm \in SM, \exists \mathcal{SMR} \in SMS \mid \mathcal{SMR}(sm, sm) = \{(i, j) \in D_{sm} \times D_{sm} \mid i = j\}$
2. $\forall \mathcal{SMR}(sm_1, sm_2) \in SMS, \exists \mathcal{SMR}(sm_2, sm_1) \in SMS \mid$
 $\mathcal{SMR}(sm_2, sm_1) = \mathcal{SMR}^{-1}(sm_1, sm_2) =$
 $\{(j, i) \in D_{sm_2} \times D_{sm_1} \mid (i, j) \in \mathcal{SMR}(sm_1, sm_2)\}$
3. $\forall \mathcal{SMR}(sm_1, sm_2), \mathcal{SMR}(sm_2, sm_3) \in SMS, \exists \mathcal{SMR}(sm_1, sm_3) \in SMS \mid$
 $\mathcal{SMR}(sm_1, sm_3) = \mathcal{SMR}(sm_1, sm_2) \circ \mathcal{SMR}(sm_2, sm_3) =$
 $\{(i, k) \in D_{sm_1} \times D_{sm_3} \mid (i, j) \in \mathcal{SMR}(sm_1, sm_2) \wedge (j, k) \in \mathcal{SMR}(sm_2, sm_3)\}$

Let us define the **stream mappings set indicator function**:

$$\chi_{SMS}(sm_1, sm_2) = \begin{cases} 1 & \text{if } \mathcal{SMR}(sm_1, sm_2) \in SMS \\ 0 & \text{if } \mathcal{SMR}(sm_1, sm_2) \notin SMS \end{cases}$$

Each element $(i, j) \in \mathcal{SMR}(sm_1, sm_2)$ represents the fact that the i -th symbol in the first stream is related to the j -th symbol in the second stream. The \mathcal{SMR} relation represents and embeds the algorithm that allows us to map symbols of one stream into those of the other one. In particular, in the case of the relationship between logical and physical streams the \mathcal{SMR} relation represents the fact that given logical symbols are encoded with given physical symbols. In this way, it clearly model the distinction and the passing from the logical level to the physical one. For example, the \mathcal{SMR} relation could represent the mapping between the pixels of an image and its *Joint Photographic Experts Group (JPEG)* encoding.

In general, the stream mapping relation allows us to express many-to-many relationships among symbols of two streams. In particular, we are interested in expressing, at least, the following relationships:

- a one-to-one relationship among the symbols of the two streams, as in the previous example of the piece of text and its ASCII encoding;
- a one-to-many relationship among the symbols of the two streams, as in the previous example of the piece of text and its UNICODE encoding;
- a many-to-one relationship among the symbols of the two streams, as in the case of compression of a stream into another.

Moreover, the stream mapping relation provides us with a further degree of freedom, since we can have symbols in a stream that do not have a correspondence in the other stream; this way, we can model some kind of loss of information due to different encodings.

Finally, the stream mapping relation allows the same logical symbol to be encoded in different ways according to its position in the stream. For example, the same letter “t”, which in the previous example appears in the third and tenth position of the stream, could be encoded in two different ways, if we apply some compression algorithm to that stream.

Moreover, definition 4.5 allows us to associate a set of physical streams to the same logical stream, providing us with a mechanism to enable different encodings of the same

physical stream. We could also create a *chain of streams*, i.e. we can specify that a logical stream is encoded with a given physical stream, and that this latter physical stream is mapped to another physical stream, and so on. These latter observations led us to introduce the stream mappings set, which contain the stream mapping relations and holds the intuitive and expected properties for this kind of set:

- *reflexive*: for each stream, the obvious mapping of the stream to itself exists;
- *symmetric*: if we know how to map one stream to another, we can also map the second stream back to the first one;
- *transitive*: if we know the mapping between a stream and another and we also know the mapping between the second stream and a third one, as a consequence we now know how to map the first one to the third one.

Now we can study how definition 4.5 impacts the set of streams SM and how it contributes to enforce the distinction between the logical level and the physical one.

Proposition 4.1: The following relation:

$$\mathcal{SMS} = \{(sm_1, sm_2) \in SM \times SM \mid \chi_{SMS}(sm_1, sm_2) = 1\}$$

is an equivalence relation over the set of streams SM . The sets:

$$\mathcal{SM} = \{sm_1, sm_2 \in SM \mid (sm_1, sm_2) \in \mathcal{SMS}\}$$

are the equivalence classes of all the streams of SM which are mapped one into another and SM/\mathcal{SM} is the quotient set.

Proof:

The relation is:

- **reflexive**: $\forall sm \in SM, \exists \mathcal{SMR}(sm, sm) \in SMS \Rightarrow \chi_{SMS}(sm, sm) = 1$;
- **symmetric**: $\forall sm_1, sm_2 \in SM \mid \chi_{SMS}(sm_1, sm_2) = 1, \exists \mathcal{SMR}(sm_2, sm_1) \in SMS \Rightarrow \chi_{SMS}(sm_2, sm_1) = 1$;
- **transitive**: $\forall sm_1, sm_2, sm_3 \in SM \mid \chi_{SMS}(sm_1, sm_2) = \chi_{SMS}(sm_2, sm_3) = 1, \exists \mathcal{SMR}(sm_1, sm_3) \in SMS \Rightarrow \chi_{SMS}(sm_1, sm_3) = 1$.

Thus, it is an equivalence relation. □

We can choose the logical streams as representatives of the equivalence classes; this way, the \mathcal{SMS} equivalence relation allows us to deal only with logical streams, abstracting us from the physical level and hiding the details of the representation and the encoding of logical streams into physical ones. Thus, the \mathcal{SMS} equivalence relation enforces the distinction between the logical and the physical level and provides us with a means for working and reasoning at a logical level, clearly separating it from the physical one.

Furthermore, we can iterate this line of reasoning and use this equivalence relation as a basic mechanism in order to introduce further levels of abstraction and to create a kind of hierarchy among streams. Indeed, we could introduce over the quotient set

SM/SM an equivalence relation similar to SMS in order to express the fact that two or more logical streams can be mapped one into another. This way, we can maintain more abstract classes of equivalent logical streams over the quotient set SM/SM , keeping them distinct from the different ways in which they can be encoded; this different encoding are, in turn, represented by the less abstract equivalence classes over the set SM . This procedure can be repeated as many times as many levels of abstraction are needed. For example, suppose that we have a piece of text that can be represented either as a sequence of characters or as a scanned image. These are two different logical streams, which can be, respectively, encoded with many different physical streams. In this case, a first level of abstraction is to put all of the physical streams that encode the characters stream into one equivalence class, created over SM , and all of the physical streams that encode the image stream into another equivalence class, created over SM . However, a higher level of abstraction is to put both the equivalence class of the text stream and the one of the image stream into a new and more abstract equivalence class, created over SM/SM , in order to express the fact the both of them are representations of the same piece of text.

For all these reasons, definition 4.5 and proposition 4.1 constitute a step forward with respect to previous models (Bottoni et al., 2003; Castelli and Pagano, 2002a,b; Gonçalves et al., 2004a,b; Navarro and Baeza-Yates, 1997), which partially address this issue or do not address it at all. On the other hand, definitions 4.4, 4.5, and proposition 4.1 are fully compatible with the definition of stream provided by both Navarro and Baeza-Yates (1997) and Gonçalves et al. (2004a); thus, we can utilize the proposed distinction between logical and physical streams into both the models provided by Navarro and Baeza-Yates (1997) and Gonçalves et al. (2004a) in order to extend such models, if necessary.

4.5 Segment

The handles discussed in Section 3.1.1 may be capable not only of identifying uniquely a DO, but also of pointing to a part of the identified DO. For example, an URL can point to any desired anchor within an HTML document, or we can use an XPath expression to point to a specific element within an XML document. On the other hand, it is not always possible to identify parts of a DO with an arbitrary degree of detail; for example, an URL cannot point to a given word of an HTML document, if this word is not marked with an anchor. Thus, we need some further mechanism for identifying parts of a DO with the desired degree of detail.

The following definition introduces the notion of *segment*, which is a mechanism for selecting parts of a stream; this mechanism can be partnered with the handle of a DO in order to provide access to a DO with the desired degree of detail.

Definition 4.6: Given a stream $sm : D = \{1, 2, \dots, n\} \rightarrow \Sigma$, $n \in \mathbb{N}$, $sm \in SM$, a **segment** is a pair:

$$st_{sm} = (a, b) \mid 1 \leq a \leq b \leq n, \quad a, b \in \mathbb{N}$$

A **stream segment** is the restriction $sm|_{[a,b]}$ of the stream sm to interval $[a, b]$ associated with the segment st_{sm} . ST is the **set of segments** and $st_{sm} \in ST$ is a generic segment; U_{ST} is the **universe set of segments**, which is the set of all the possible segments, so that $ST \subseteq U_{ST}$.

The relationship between the sets ST and U_{ST} is the same as the one between the sets DO and U_{DO} , described in section 4.1. Definition 4.6 resembles the definition of segment provided in (Gonçalves et al., 2004a; Navarro and Baeza-Yates, 1997).

We can assume that logically related symbols of logical streams are contiguous and have an ascending ordering. This assumption fits well with definition 4.6, that selects a series of contiguous symbols. On the other hand, definition 4.5 allows us to disregard this constraint for the mapping to physical streams, since the stream mapping relation SMR allows us to map the contiguous symbols of the logical stream into non-contiguous symbols of the physical stream. For example, the indices $D_{lsm} = \{1, 2, 3, 4, 5\}$ of a logical stream could be mapped to the indices $D_{psm} = \{13, 7, 19, 9, 15\}$ of a physical stream. If we choose the segment $st_{lsm} = (2, 4)$ which is associated with the interval $[2, 4]$ for the logical stream, we are not forced to map it to the interval $[7, 9]$, obtained by mapping the segment st_{lsm} to a corresponding segment $st_{psm} = (7, 9)$ of the physical stream. On the contrary, we can map each index in the interval $[2, 4]$ to its corresponding index in the physical stream, obtaining the set of indices $\{7, 19, 9\}$, which do not fit in the interval $[7, 9]$. See (Navarro and Baeza-Yates, 1997) for further explanation about the ordering in multimedia streams.

This possibility is important because symbols that are contiguous in a logical stream can correspond to non-contiguous symbols in a physical streams, due to some kind of compression, as an example. Moreover, proposition 4.1 allows us to reason only in terms of logical streams, which comply with the assumption made above, without worrying about the physical streams that are in the same equivalence class of the logical stream. This observation further highlights the benefits that may arise by clearly distinguishing between the logical and the physical level.

4.6 Sign of Annotation

As discussed in Sections 3.2.1 and 3.2.3, the sign of annotation is a way of materializing the semantics of an annotation. The following definition formalizes the notion of sign of annotation.

Definition 4.7: A **sign of annotation** is a stream. SN is the **set of signs of annotation** and $sn \in SN$ is a sign. U_{SN} is the **universe set of signs of annotation**, which is the set of all the possible signs of annotation, so that $SN \subseteq U_{SN}$.

The relationship between the sets SN and U_{SN} is the same as the one between the sets DO and U_{DO} , described in section 4.1.

In the following we will use the term *sign of annotation*, or briefly stated as *sign*, to indicate a stream that belongs to an annotation. On the other hand, we will use the term *stream* to indicate a stream that belongs to a DO without the need of specifying if the DO is a document or an annotation.

4.7 Meaning of Annotation

As discussed in Sections 3.2.1 and 3.2.3, the meaning of annotation represents part of the semantics of an annotation. The following definition formalizes the notion of meaning of annotation.

Definition 4.8: M is the **set of meanings of annotations**, and $m \in M$ is a generic **meaning of annotation**.

The **meanings graph** is a labeled directed graph⁷ (G_M, l_M) , where $G_M = (M, E_M \subseteq M \times M)$ and $l_M : E_M \rightarrow L_M$ with L_M set of labels.

The **meanings function** $m : SN \rightarrow 2^M$ associates each sign of annotation with its corresponding meanings of annotation. The following constraint must be satisfied:

$$\forall sn \in SN, m(sn) \neq \emptyset$$

that is each sign of annotation has, at least, one meaning of annotation.

Differently from the set of the previous definitions, the set of meanings M is a time-invariant set, because we assume that meanings represent a pre-existing knowledge that does not change with the passing of time. So, all the needed meanings of annotation are already elements of the set M .

The goal of the meanings graph is to provide some sort of structure and hierarchy among the meanings of annotation in order to navigate and browse through them. See Sections 3.2.1 and 3.2.3 for a description of how to navigate through meanings of annotation. The relation E_M can be constrained in many ways in order to obtain the desired structure of meanings. The labelling function l_M can be further exploited to distinguish different kinds of arcs in the set E_M in order to better explain the kind of relationship between two different meanings.

Gonçalves et al. (2004a) introduces the general notion of *structure* in DLs, represented with a labeled directed graph, as a means of expressing different kinds of structure that could be needed in DLs, such as taxonomies, metadata, and so on. Thus, the meanings graph adheres to this definition of structure and it is a structure aimed at allowing the navigation through the different meanings of annotation.

The meanings function allows us to associate each sign of annotation with its corresponding meanings in order to clarify the semantics of the sign, as explained in Sections 3.2.1 and 3.2.3. Note that the meaning function is neither injective nor surjective.

We are interested in studying the sharing of common meanings among different signs, meant as basic mechanism for relating and gathering up different signs that express common semantics. This is very helpful, for example, in the case of annotations of two different users: they may use different signs to indicate the importance of a passage – an asterisk and an exclamation mark – and knowing that these two different signs have the same meaning allows these two users to communicate and collaborate together.

The most immediate way of approaching this issue is to introduce the following relation:

$$\mathcal{M}_1 = \{(sn_1, sn_2) \in SN \times SN \mid m(sn_1) \cap m(sn_2) \neq \emptyset\}$$

This relation allows us to point out the signs that directly share some common meanings. However, this relation is not able to relate two signs that do not directly share a common meaning.

⁷For a reference about graphs see (Bollobás, 1998; Diestel, 2000).

Thus, a step forward is to take into account also the meanings graph $G_M = (M, E_M)$ and its reflexive transitive closure⁸ $G_M^* = (M, E_M^*)$ in order to introduce the following relation:

$$\mathcal{M}_2 = \left\{ (sn_1, sn_2) \in SN \times SN \mid \exists m \in M, m_1 \in m(sn_1), m_2 \in m(sn_2), \right. \\ \left. (m_1, m_2) \in E_M^* \vee (m_2, m_1) \in E_M^* \vee \right. \\ \left. ((m, m_1) \in E_M^* \wedge (m, m_2) \in E_M^*) \vee ((m_1, m) \in E_M^* \wedge (m_2, m) \in E_M^*) \right\}$$

The \mathcal{M}_2 relation means that two signs s_1 and s_2 are in relation if among their meanings, given by $m(sn_1)$ and $m(sn_2)$, at least:

- one is the ancestor of the other $\left((m_1, m_2) \in E_M^* \vee (m_2, m_1) \in E_M^* \right)$, or
- they both have a common ancestor $\left((m, m_1) \in E_M^* \wedge (m, m_2) \in E_M^* \right)$, or
- they both are the ancestors of a common meaning $\left((m_1, m) \in E_M^* \wedge (m_2, m) \in E_M^* \right)$, or
- two meanings are equal – as in the case of the \mathcal{M}_1 relation.
Indeed, $\mathcal{M}_1 \subseteq \mathcal{M}_2$ because $\forall s_1, s_2 \in SN \mid (sn_1, sn_2) \in \mathcal{M}_1 \Leftrightarrow \exists m \in m(sn_1) \cap m(sn_2) \Rightarrow (m, m) \in E_M^* \Leftrightarrow (sn_1, sn_2) \in \mathcal{M}_2$.

\mathcal{M}_2 is a very broad relation that allows us to relate different signs according to the four strategies outlined above. As needed, we could use limited versions of \mathcal{M}_2 that adopt only some of the strategies introduced above – for example, \mathcal{M}_1 uses only the last strategy.

Further strategies can be envisaged in order to group signs on the basis of their meanings – for example, we could take into consideration the predecessor of a meaning instead of its ancestor, as it is in \mathcal{M}_2 . Thus, the \mathcal{M}_1 and \mathcal{M}_2 relations are examples of utilization of the meanings graph, but they do not intend to be exhaustive. For example, Rigaux and Spyrtos (2004) propose a subsumption relation over the terms of a taxonomy and a way of navigating through them, that can also be very useful in the context of the meanings of annotation.

4.8 Annotation

As discussed in Section 3.2.2, annotations can be linked to DOs by means of two main types of links, i.e. the *annotate link* and the *relate-to link*.

Definition 4.9: Let LT be the **set of link types**; an element $lt \in LT$ corresponds to one of the allowed link types. The set LT contains the following link types: $LT = \{\text{Annotate}, \text{RelateTo}\}$.

⁸The *transitive closure* of a graph $G = (V, E)$ is a graph $G^+ = (V, E^+)$ such that for all $v, w \in V$ there is an edge $(v, w) \in E^+$ if and only if there is a path from v to w in G that has at least one edge. The *reflexive transitive closure* of a graph $G = (V, E)$ is a graph $G^* = (V, E^*)$ such that for all $v, w \in V$ there is an edge $(v, w) \in E^*$ if and only if $(v, w) \in E^+ \vee v = w$.

As in the case of the set of meanings of annotation M , also the set of link types LT is a time-invariant set, because we assume that link types represent a pre-existing knowledge that does not change with the passing of time. So, all the needed link types are already elements of the set LT .

As discussed in Sections 2.3 and 3.2, an annotation can have different scopes, i.e. it can be *private*, *shared*, or *public*.

Definition 4.10: Let $SP = \{\text{Private}, \text{Shared}, \text{Public}\}$ be the **set of scopes** and $sp \in SP$ is a scope. Let us define the following relations:

- **equality relation** =

$$\{(sp, sp) \in SP \times SP \mid sp \in SP\} = \\ \{(\text{Private}, \text{Private}), (\text{Shared}, \text{Shared}), (\text{Public}, \text{Public})\}$$

- **strict ordering relation** \prec

$$\{(\text{Private}, \text{Shared}), (\text{Private}, \text{Public}), (\text{Shared}, \text{Public})\}$$

- **ordering relation** \preceq

$$\{(sp_1, sp_2) \in SP \times SP \mid sp_1 = sp_2 \vee sp_1 \prec sp_2\}$$

As in the case of the set of types LT , also the set of scopes SP is a time-invariant set, because we assume that an annotation can have only one of the three scopes listed above. Note that (SP, \preceq) is a *totally ordered set*.

Now we can introduce the definition of annotation. The mechanism, introduced in section 4.1, on how to address the time dimension is now fundamental in order to properly define the annotation.

Definition 4.11: An **annotation** $a \in A(k)$ is a tuple:

$$a = \left(h_a \in H(k), au_a \in AU(k-1), G_a \in 2^{GR(k-1)}, sp_a \in SP, \right. \\ \left. \mathcal{A}_a \subseteq SN(k) \times LT \times ST(k) \times SM(k-1) \times H(k-1) \right)$$

where:

- h_a is the unique handle to the annotation a , i.e. $h(h_a) = a$;
- au_a is the author of the annotation a , i.e. $h_a \in au(au_a)$;
- G_a are the groups of authors which can access the annotation, such that $G_a \subseteq gr(au_a)$;
- sp_a is the scope of the annotation a – Private, Shared, or Public;
- the \mathcal{A}_a relation means that the annotation a by means of a sign in $SN(k)$ is annotating or relating to a segment in $ST(k)$ of a stream in $SM(k-1)$ of a DO identified by its handle in $H(k-1)$.

We introduce the following auxiliary sets, which will make the following discussion easier:

- the set of the signs of annotation that belong to the annotation a :
 $SN_a = \{sn \in SN(k) \mid \exists \alpha \in \mathcal{A}_a, \alpha = (sn, t, st_{sm}, sm, h)\}$
- the set of the handles to DOs that are subject to the tasks of the annotation a :
 $H_a = \{h \in H(k-1) \mid \exists \alpha \in \mathcal{A}_a, \alpha = (sn, t, st_{sm}, sm, h)\}$

The following constraints must be adhered to:

1. the annotation a must annotate one and only one DO, that cannot also be related to, that is:

$$\begin{aligned} & \exists! h \in H_a \mid \\ & \left(\forall sn \in SN_a, \exists! \alpha \in \mathcal{A}_a, \alpha = (sn, \text{Annotate}, st_{sm}, sm, h) \right) \wedge \\ & \left(\nexists \alpha_1 \in \mathcal{A}_a, \alpha_1 = (sn_1, \text{RelateTo}, st_{sm_1}, sm_1, h) \right) \end{aligned}$$

2. a sign in SN_a cannot relate to more than one DO, that is:

$$\begin{aligned} & \forall sn \in SN_a \mid \exists \alpha_1, \alpha_2 \in \mathcal{A}_a, \\ & \alpha_1 = (sn, \text{RelateTo}, st_{sm_1}, sm_1, h_1), \alpha_2 = (sn, \text{RelateTo}, st_{sm_2}, sm_2, h_2) \\ & \Rightarrow \alpha_1 = \alpha_2 \end{aligned}$$

3. there no exists another annotation $a_1 \in A(k-1)$ that shares signs of annotation with a , that is:

$$\nexists a_1 \in A(k-1) \mid SN_a \cap SN_{a_1} \neq \emptyset$$

4. if the annotation $a \in A(k)$ annotates another annotation $a_1 \in A(k-1)$ then the scope sp_a of a must be less to or equal to the scope sp_{a_1} of a_1 and the two annotations must have the same groups of authors, if the scope of a_1 is Shared, or they must have the same author, if the scope of a_1 is Private:

$$\begin{aligned} & \forall h \in H_a \mid \exists \alpha \in \mathcal{A}_a, \alpha = (sn, \text{Annotate}, st, sm, h) \wedge h(h) = a_1 \in A(k-1) \Rightarrow \\ & sp_a \preceq sp_{a_1} \wedge \left(sp_{a_1} = \text{Public} \vee \begin{cases} G_a = G_{a_1} & \text{if } sp_{a_1} = \text{Shared} \\ au_a = au_{a_1} & \text{if } sp_{a_1} = \text{Private} \end{cases} \right) \end{aligned}$$

In conclusion, the first part of the annotation tuple is devoted to provide information about the annotation itself, because it specifies the handle to the annotation, its author and groups of authors, its scope, the signs of the annotation, and the link types. On the other hand, the second part of the annotation tuple provides information about the annotated or related DOs, specifying which segment of which stream of which DO is being annotated or related to, as shown below (we do not use the time dimension notation for space reasons, as it is not needed for this observation):

$$a = \left(\underbrace{h_a \in H, au_a \in AU, G_a \in 2^{GR}, sp_a \in SP, \mathcal{A}_a \subseteq SN \times T}_{\text{information about the annotation}} \times \underbrace{ST \times SM \times H}_{\text{information about the DO}} \right)$$

Note that, differently from the case of generic DO introduced in section 4.3, the annotation is constrained to be authored by one and only one author.

Discussion about the \mathcal{A}_a Relation

The \mathcal{A}_a relation makes an extensive use of the mechanism, introduced in section 4.1, for addressing the time dimension. In particular, the \mathcal{A}_a relation aims to point out the fact that an annotation must annotate or relate to DOs that already exists. For this reason, in definition 4.11, the annotation a belongs to $A(k)$, while the annotated or related DOs belong to $DO(k-1)$ and they are identified by their handles in $H(k-1)$. This notation underlines the fact that the annotation belongs to the set of digital objects at time k , but it works with the previously existing DOs that belong to the set of digital object at time $k-1$. Thus, an annotation can annotate or relate to only already existing DOs, which is quite intuitive but it needs to be properly formalized.

A very important consequence of this choice is that:

$$h_a \notin H_a$$

Indeed, $\{h_a\} = H(k) \setminus H(k-1)$ while $H_a \subseteq H(k-1)$: h_a is just the handle identified by the transition from $H(k-1)$ to $H(k)$, as explained in section 4.1. Thus, an annotation cannot be self-referential, i.e. it cannot annotate or relate to itself, since a self-referential annotation would be useless.

Note that the notation of definition 4.11 provides a greater expressive power with respect to the ER schema of figure 3.3, because the ER schema is not capable of fully capturing the temporal dimension entailed by annotations. For this reason, we have added to the ER schema the following constraint: once the annotation has been created, an occurrence of the DOHANDLE entity corresponding to the annotation have to be added, in order to allow the newly created annotation to be annotated as well. On the contrary, according to definition 4.11 once an annotation has been created and it belongs to $DO(k)$, then at time $k+1$ it is automatically ready to be annotated, because annotations at time $k+1$ can freely work with the DOs existing at time k , that include the annotation $a \in A(k)$. Thus, definition 4.11 allows us to overcome a lack of expressive power of the ER schema introduced in Section 3.2.3.

The \mathcal{A}_a makes use of the set of signs $SN(k)$ at time k to indicate that those are the signs created just for the annotation a . Furthermore, \mathcal{A}_a uses the set of segments $ST(k)$ at time k to indicate that those segments are created solely to allow the annotation a to point to the requested part of the streams contained in $SM(k-1)$. If we consider the mechanism introduced in Section 4.1 for formalizing the temporal dimension, when at time $k-1$ we pick out a new segment $st_{sm} \in \overline{ST}(k-1) \subseteq U_{ST}$, it can refer to a stream $sm \in SM(k-1)$: indeed, that stream already exists at time $k-1$, even if the new segment belongs to the set of segment $ST(k)$ only at time k . Note that the \mathcal{A}_a relation uses the set of streams $SM(k-1)$ at time $k-1$ because those are the streams that belong to the DOs identified by their handles in $H(k-1)$. In conclusion, we deal with DOs and their corresponding streams that already exists at time $k-1$ and that are annotated or related to by using signs and segments just created for the annotation a at time k .

In the \mathcal{A}_a relation both segments and streams play a very important role in allowing an annotation to annotate or relate to the requested part of a DO. In this context, the distinction between logical and physical streams and the possibility of using the logical streams as representatives of their equivalence classes become fundamental. We can always suppose that an annotation deals with logical streams, being sure that the mapping to different physical streams is correctly managed by the notion of stream

mapping relation, as introduced in Section 4.4. In this way, an annotation can annotate a logical stream and it is implicitly annotating also all of the physical streams that are in the same equivalence class of the logical stream. Furthermore, as discussed in section 4.4, an annotation could annotate abstract streams belonging to equivalence classes created over the quotient set SM/SM , this way obtaining the access to a whole hierarchy of different representations of the content of a DO. Finally, logical streams simplify the usage of segments, because we can always refer to contiguous indices in the logical streams even if they are not contiguous in the physical streams, as observed in section 4.5. This possibility makes it easier to determine which part of the DO is being annotated or related, because we can always suppose that we deal with contiguous indices in the stream of the DO.

Discussion about the Constraints of the Annotation

The first two constraints are *intra-annotation* constraints, because they limit the \mathcal{A}_a relation, which is the core of the annotation; on the other hand, the second two constraints are *inter-annotation* constraints, because they regulate the relationships of the annotation with respect to other annotations.

The first constraint imposes the existence and uniqueness of the annotated DO and prevents the annotated DO from being related as well. In this way, the constraint introduced in Section 3.2.2:

an annotation must annotate one and only one DO, either a document or another annotation, that is an annotation must have one and only one “annotate link”

is complied with. Furthermore, it enforces the distinction between the “annotate link” and the “relate-to link”, because the annotated DO cannot be also related. Thus, it underlines that the role of the “annotate link” is to express intra-DO relationships, while the “relate-to link” is requested to express only inter-DO relationships. Furthermore, each sign must cooperate – one and only one time – in expressing such intra-DO relationships, i.e. there is not any sign whose only link is the “relate-to link”. A consequence of this constraint is:

$$\begin{aligned} \forall h_1, h_2 \in H_a \mid \exists \alpha_1, \alpha_2 \in \mathcal{A}_a, \\ \alpha_1 = (\text{sn}_1, \text{Annotate}, st_{\text{sm}_1}, \text{sm}_1, h_1), \alpha_2 = (\text{sn}_2, \text{Annotate}, st_{\text{sm}_2}, \text{sm}_2, h_2) \\ \Rightarrow h_1 = h_2 \end{aligned}$$

The second constraint aims to keep the semantics of a sign as clear as possible: if a sign sn was related to more DOs, it would be not clear which of its meanings – given by $m(\text{sn})$ – should be applied to each related DO. In conclusion, this constraint together with the first one states that a sign of annotation must annotate one and only one segment of a DO and it may relate to one and only one segment of another DO.

The third constraint ensures that the signs of an annotation are not shared with any other annotation in order to preserve the mechanism of sharing a common semantics among annotations. As explained in section 4.7, the sharing of meanings of annotation by means of the \mathcal{M}_1 and \mathcal{M}_2 relations is the mechanism to point out a common semantics among annotations; on the other hand, the direct sharing of signs of annotation could be misleading. Indeed, a sign is a materialization of a meaning and the same sign used by two different users can have completely different meanings or two different

signs can have the same meaning. Consider, for example, two users that use a star, the first to indicate an important passage while the other to indicate a wrong passage; or otherwise two users that use a star and an exclamation mark, both to indicate an important passage. Note that this constraint does not prevent the existence of two signs that looks exactly the same, but it means that these two signs are different elements in the set SN .

The fourth constraint prevents pathologic situations as, for example, the one of a public annotation that annotates a private annotation. In such cases there is a *scope conflict* – in the example, the author of the private annotation can see both the public and the private annotation, but another user can see only the public annotation which is annotating something hidden to this user. The general rule to address this issue is:

- a public annotation can be freely annotated or related, without further restrictions;
- a shared annotation can be annotated or related only by a shared or private annotation, provided that they have the same groups of authors, if they are shared. It follows that a shared annotation cannot be annotated or related to by a public annotation;
- a private annotation can be annotated or related only by a private annotation, provided that they have the same author. It follows that a private annotation can be annotated or related by neither a public annotation nor a shared annotation.

4.9 Document–Annotation Hypertext

As explained in Sections 2.2 and 3.1, we consider that existing DOs and annotations constitute a hypertext, according to the definition of hypertext provided in (Agosti, 1996).

Definition 4.12: The *document–annotation hypertext* is a labeled directed graph:

$$(H_{da} = (DO, E_{da} \subseteq A \times DO), l_{da})$$

where:

- $DO = A \cup D$ is the set of vertices;
- $E_{da} = \{(a, do) \in A \times DO \mid \exists \alpha \in \mathcal{A}_a, \alpha = (sn, t, st_{sm}, sm, h^{-1}(do))\}$ is the set of edges;
- $l_{da} : E_{da} \rightarrow LT$ is the labelling function, such that for each $e = (a, do) \in E_{da}$ there is a lt -labeled edge from the annotation a to the generic digital object do :

$$l_{da}(a, do) = \begin{cases} \text{Annotate} & \text{if } \exists \alpha \in \mathcal{A}_a \mid \alpha = (sn, \text{Annotate}, st_{sm}, sm, h^{-1}(do)) \\ \text{RelateTo} & \text{if } \exists \alpha \in \mathcal{A}_a \mid \alpha = (sn, \text{RelateTo}, st_{sm}, sm, h^{-1}(do)) \end{cases}$$

The document–annotation hypertext is constructed by putting an edge between an annotation vertex and a DO vertex, if the annotation is annotating or relating to that DO. Note that we used $h^{-1}(do)$ in E_{da} in order to track the DO back to its handle;

the edge is then labeled with the corresponding link type. Each edge $e = (a, do) \in E_{da}$ always starts from an annotation $a \in A$, while it does not exist $e \in E_{da}$ that starts from a document $d \in D$.

Note that we deal with a graph H_{da} and not with a multigraph, i.e. a graph where multiple edges between the same vertices are allowed – as it may happen in the case of an annotation relating to a different part of the same DO. Thus, we consider that multiple edges with the same direction between the same vertices are collapsed into a single edge.

It would have been possible to use H instead of DO as set of vertices, thus, defining the hypertext by means of handles to DOs instead of real DOs. This choice would require an intensive use of the $h(h)$ function in order to obtain the DO corresponding to the handle and work with it. Furthermore, using handles instead of DOs would make the semantics of E_{da} less clear, because we should use a $E'_{da} \subseteq H \times H$ relation that does not clearly point out that edges start only from annotations to DOs. Thus, it is not clear what advantages this further level of indirection would provide at a conceptual level, a part from expressing that we are not forced to directly use DOs but we could use handles instead. On the other hand, in the FAST system, it is convenient to implement the document–annotation hypertext by handles, in order to keep FAST separated from the different IMSs; however, this is only an implementation issue.

Table 4.1 summarizes the graphical conventions, adopted in the following figures.



	Annotate Link	RelateTo Link
Document 	continuous line labeled A	dotted line labeled R
Annotation 	continuous line labeled A	dotted line labeled R

Table 4.1: Graphical conventions.

Figure 4.1 shows an example of document–annotation hypertext H_{da} :

- $D = \{d_1, d_2, d_3, d_4, d_5\}$, we can assume that the subscript of each document indicates the time in which the document became an element of the set D ;
- $A = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}\}$, we can assume that the subscript of each annotation indicates the time in which the annotation became an element of the set A ;
- we can express, for example:
 - *annotations sets concerning a document*: $\{a_1, a_2\}$ is an annotation set concerning the document d_1 ;
 - *annotations sets concerning an annotation*: $\{a_8, a_9\}$ is an annotation set concerning the annotation a_7 ;
 - *annotations threads concerning a document*: $\{a_1, a_3, a_4\}$ is an annotation thread concerning the document d_1 ;
 - *annotations threads concerning an annotation*: $\{a_8, a_{10}\}$ is an annotation thread concerning the annotation a_7 ;

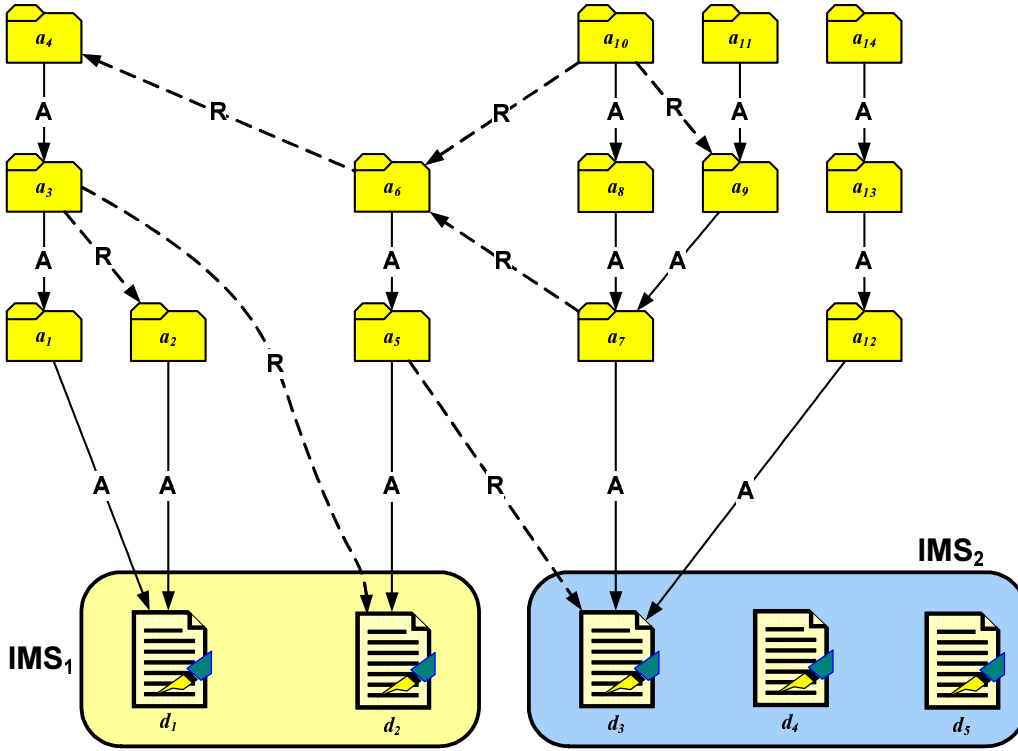


Figure 4.1: Example of document-annotation hypertext H_{da} .

- *multiple annotations threads concerning a document*:
 $\{a_7, a_8, a_{10}\}$ and $\{a_{12}, a_{13}, a_{14}\}$ are two different annotations threads both concerning the document d_3 ;
- *multiple annotations threads concerning an annotation*: $\{a_8, a_{10}\}$ and $\{a_9, a_{11}\}$ are two annotations threads both concerning the annotation a_7 ;
- *nested annotations threads concerning a document*:
 $\{a_8, a_{10}\}$ and $\{a_9, a_{11}\}$ are two different and nested annotations threads both concerning the document d_3 .

Figure 4.1 also points out another important feature of the document-annotation hypertext: it can span and cross the boundaries of the single IMS, as discussed in Section 3.1. The IMS_1 manages d_1 and d_2 , while the IMS_2 manages d_3 , d_4 , and d_5 . There are annotations that act as a bridge between two IMSs: for example, a_5 annotates d_2 , which is managed by IMS_1 , and refers to d_3 , which is managed by IMS_2 . This is a quite innovative characteristic of the document-annotation hypertext, which is a direct consequence of making FAST a stand-alone system not tailored to a specific IMS.

Proposition 4.2: The *document-annotation hypertext* holds the following properties:

1. the graph does not contain loops:

$$\forall a \in A, \nexists e = (a, do) \in E_{da} \mid a = do$$

2. each annotation a is incident with one and only one edge labeled Annotate:

$$\forall a \in A, \exists! e = (a, do) \in E_{da} \mid l_{da}(e) = \text{Annotate}$$

3. the graph does not contain cycles:

$$\begin{aligned} \nexists C = a_1 a_k a_{k-1} \cdots a_2 a_1 \mid \\ e_1 = (a_1, a_k), e_k = (a_k, a_{k-1}), \dots, e_2 = (a_2, a_1) \in E_{da}, \quad k > 1 \end{aligned}$$

4. given a set $A' \subset A$ there are, at least, $|A'|$ edges in H_{da} incident on elements of A' . Thus, the following relationship holds for the size⁹ of H_{da} :

$$\varepsilon(H_{da}) \geq |A|$$

Proof:

We can show that:

1. from definition 4.11, it follows that $h_a \notin H_a$ and, as explained in section 4.8, $\nexists \alpha \in \mathcal{A}_a \mid \alpha = (\text{sn}, t, st_{\text{sm}}, \text{sm}, h_a)$; thus, $\nexists e = (a, a) \in E_{da}$;
2. from definition 4.11, we have the following constraint: $\exists! h \in H_a \mid \forall \text{sn} \in SN_a, \exists! \alpha \in \mathcal{A}_a, \alpha = (\text{sn}, \text{Annotate}, st_{\text{sm}}, \text{sm}, h)$; it follows that $\forall a \in A, \exists! do \in DO \mid \exists \alpha \in \mathcal{A}_a, \alpha = (\text{sn}, \text{Annotate}, st_{\text{sm}}, \text{sm}, h^{-1}(do))$; thus, there exists a unique edge such that $l_{da}(a, do) = \text{Annotate}$;

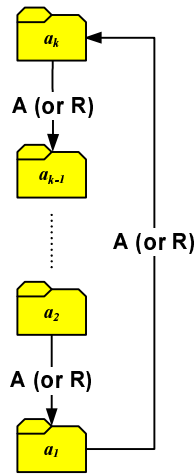


Figure 4.2: Not allowed annotations cycle.

3. annotations entail a temporal dimension, since each annotation must annotate or relate to an already existing DO, as explained in Sections 4.1 and 4.8. From definition 4.11, the \mathcal{A}_a relation involves the set $H(k-1)$ of handles to DOs that already belong to the set of digital objects at time $k-1$, while the annotation belongs to the set $A(k)$. Thus, by means of the \mathcal{A}_a relation an annotation $a \in A(k)$ can annotate or relate to only DOs that already exist at time k , i.e. an annotation cannot annotate or relate to another annotation that does not already exist at time k . It follows that cycles, as the one shown in figure 4.2, where the oldest annotation $a_1 \in A(1)$ annotates or relates to the newest annotation $a_k \in A(k)$ with $k > 1$, are not possible. Indeed, when

⁹The size $\varepsilon(G)$ of a graph $G = (V, E)$ is the number of edges in the graph, that is $\varepsilon(G) = |E|$.

the oldest annotation $a_1 \in A(1)$ was created at time 1, the newest annotation $a_k \in A(k)$, $a_k \notin A(0)$ did not exist yet and so it could not have been involved in \mathcal{A}_{a_1} , which makes use of DOs belonging to the set of digital objects at time 0.

Note that this issue does not exist for document $d \in D$ vertices, since edges can start only from annotation vertices.

4. since for the item number 2 each annotation a must be incident with one and only one Annotate edge, then for $|A'|$ annotations there are, at least, $|A'|$ edges; it may be more if there are also RelateTo edges. In H_{da} there are $|A|$ annotations and therefore $\varepsilon(H_{da}) \geq |A|$. \square

Proposition 4.2 expresses the constraints imposed on the annotation in definition 4.11 in terms of a graph: firstly, the graph does not contain loops corresponding to self-referential annotations that are useless for our purposes; secondly, each annotation is incident with one and only one edge of kind “Annotate link”, thus, formalizing the constraint on the link types introduced in section 4.8; thirdly, since each annotation can annotate or relate to an already existing DO, the third property ensures that there are not any cycles where the oldest annotation a_1 annotates or relates to the newest annotation a_k , as shown in figure 4.2; finally, the fourth property sets a lower bound to the size of H_{da} .

Figure 4.3 shows the patterns, which can be obtained by combining the allowed link types: note that each pattern is characterized by only one edge of type “Annotate link”; furthermore an annotation is not allowed to exclusively have “RelateTo link” edges.

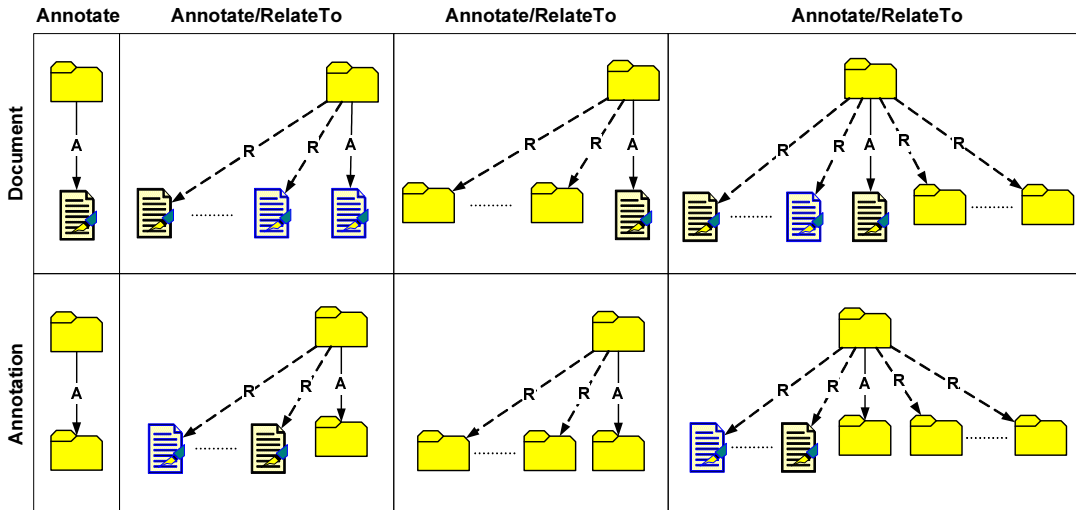


Figure 4.3: Allowed annotations patterns.

Note that the example of document–annotation hypertext shown in figure 4.1 is the results of the combination of these basic allowed annotations patterns.

Proposition 4.3: Let $H'_{da} = (DO', E'_{da})$ be the subgraph of H_{da} , such that:

- $E'_{da} = \{e \in E_{da} \mid l_{da}(e) = \text{Annotate}\}$

- $DO' = \{do \in DO \mid \exists e' \in E'_{da}, e' = (a, do)\}$

H'_{da} is the subgraph whose edges are of kind Annotate and whose vertices are incident with at least one of such edges. Let $H''_{da} = (DO'', E''_{da})$ be the underlying graph of H'_{da} , that is the undirected version of H'_{da} .

The following properties hold:

1. $\varepsilon(H'_{da}) = \varepsilon(H''_{da}) = |A|$;
2. H''_{da} is a forest;
3. every tree in H''_{da} contains a unique document vertex d .

Proof:

We can show that:

1. according to proposition 4.2, $\varepsilon(H'_{da}) = \varepsilon(H''_{da}) \geq |A|$ but since in H'_{da} and H''_{da} there are only Annotate edges, we have that $\varepsilon(H'_{da}) = \varepsilon(H''_{da}) = |A|$;
2. ab absurdo: if H''_{da} was not a forest, then it would be a cyclic graph. The only way of obtaining a cycle in H''_{da} is that in H_{da} :

$$\exists a \in A, \exists e_1 = (a, do_1), e_2 = (a, do_2) \in E_{da}, do_1 \neq do_2 \mid \\ l_{da}(e_1) = l_{da}(e_2) = \text{Annotate}$$

i.e. an annotation exists in H_{da} from which two edges of kind Annotate start from, but this contradicts the definition 4.12 given for the graph H_{da} and thus, H''_{da} is a forest;

3. since H''_{da} is a forest, its components are trees. Ab absurdo suppose that there is a tree T whose vertices are only annotations. A tree T with n vertices has $n - 1$ edges but, for proposition 4.2, in H_{da} (and also in H''_{da}) n annotations are incident with n edges; so T can not be a tree. Therefore, every tree in H''_{da} contains, at least, a document vertex d . Suppose now that there is a tree T which contains two document vertices d_1 and d_2 , $d_1 \neq d_2$. Since that for every two vertices in a tree there is a unique path connecting them, in the path $P = d_1 a_1 \dots a_i \dots a_k d_2$ there must be an annotation a_i from which in H_{da} two edges of kind Annotate start, since by definition of H_{da} there are no edges of the type $e = (d_m, d_n) \in E_{da}$. But the annotation a_i contradicts the definition of H_{da} and thus, there is a unique document vertex d in T . \square

Note that if we had not removed the “RelateTo link” edges from the graph H''_{da} , it could have contained cycles; consider figure 4.1: for example, a cycle would be $C = a_7 a_6 a_{10} a_8 a_7$, because in H''_{da} we do not consider the direction of the edges.

Figure 4.4 shows an example of the H'_{da} subgraph, obtained from the document–annotation hypertext H_{da} of figure 4.1.

Users do not usually access the whole document–annotation hypertext but, on the contrary, they can access only a subset of the document–annotation hypertext derived from their access rights. Thus, the following definition introduces an operator suitable for choosing the subset of the document–annotation hypertext that can be accessed by a user.

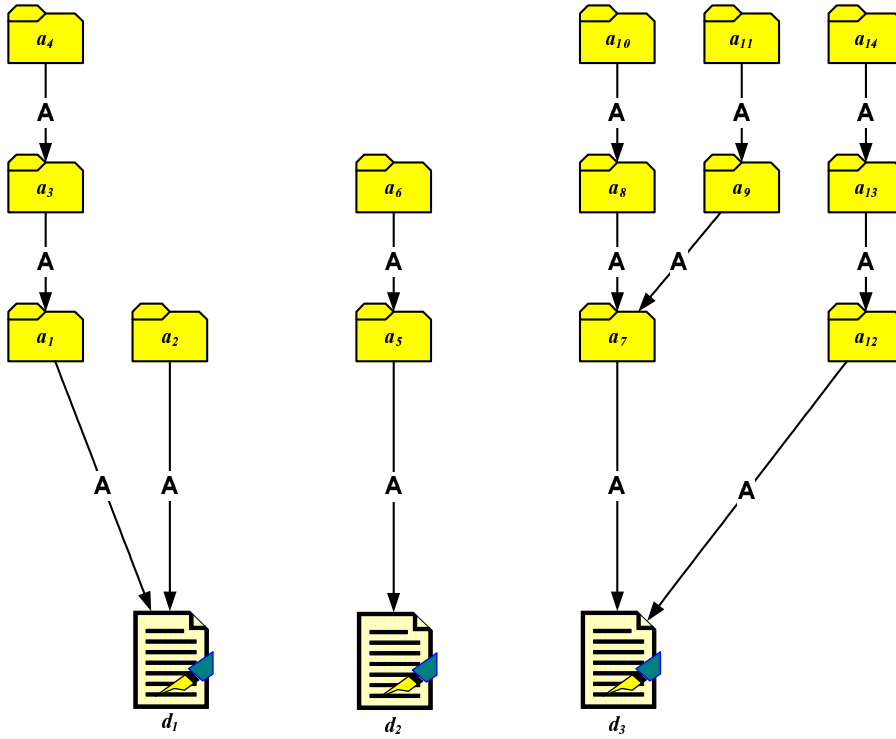


Figure 4.4: Example of the H_{da}' subgraph, obtained from the document–annotation hypertext H_{da} of figure 4.1.

Definition 4.13: Given a document–annotation hypertext H_{da} , we introduce the **document–annotation hypertext projection operator** that can have two forms:

- $\pi(H_{da}, au, G)$, with $au \in G$, constructs a new document–annotation hypertext H_{da}^π such that:

$$\begin{cases} E_{da}^\pi = \{(a, do) \in E_{da} \mid sp_a = \text{Public} \vee (sp_a = \text{Shared} \wedge G \in G_a) \vee \\ \quad (sp_a = \text{Private} \wedge au_a = au)\} \\ DO^\pi = \{do \in DO \mid \exists (a, do) \in E_{da}^\pi\} \end{cases}$$

- $\pi(H_{da}, au)$ constructs a new document–annotation hypertext H_{da}^π such that:

$$\begin{cases} E_{da}^\pi = \{(a, do) \in E_{da} \mid sp_a = \text{Public} \vee (sp_a = \text{Shared} \wedge G_a \subseteq \text{gr}(au)) \vee \\ \quad (sp_a = \text{Private} \wedge au_a = au)\} \\ DO^\pi = \{do \in DO \mid \exists (a, do) \in E_{da}^\pi\} \end{cases}$$

This operator provides us with a personalized view for the user of the document–annotation hypertext H_{da} . The first form $\pi(H_{da}, au, G)$ allows us to select vertices and edges on the basis of a given author and of a given group of authors; the second form $\pi(H_{da}, au)$ allows us to select vertices on the basis of a given author and all of the groups of authors the author belongs to.

Thus, the projection operator represents the standard way for a user to perceive the document–annotation hypertext, since a user is not allowed to access all the vertices of

the hypertext but he can access only the public ones, those belonging to him, and the ones shared with groups of authors the user belongs to.

Note that the fourth constraint of definition 4.11 ensure us that the result of the projection operator is a well–formed document–annotation hypertext: indeed, this constraint avoids the *scope conflicts*, as explained in section 4.8. Thus, the projection operator does not removes vertices and edges creating “holes” in the document–annotation hypertext due to existence, for example, of a public annotation that annotates a private annotation.

Chapter 5

Conceptual Architecture of the Flexible Annotation Service Tool

As it has been discussed in Section 3.1, the main goal of our architecture is the flexibility. In order to achieve this goal, the general architecture, depicted in figure 3.1 on page 20 allows the *Flexible Annotation Service Tool (FAST)* system:

1. to be a stand-alone system, i.e. FAST is not part of any particular IMS;
2. to separate functionalities of the *Core Annotation Service (CAS)*, from the functionalities needed to integrate it into different IMSs.

A third requirement has to be added to the previous two in order to achieve the desired level of flexibility:

3. the architecture has to model the behaviour FAST in a modular way, so that new functionalities can be added to FAST without the need of redesigning its architecture.

This chapter discusses in detail the conceptual architecture of the FAST system and how this architecture makes it possible to satisfy the requirements introduced above.

5.1 FAST Conceptual Architecture

As already introduced in Section 3.1 and shown in figure 3.1, we have decided to adopt a general architecture for FAST, which exploits a gateway in order to mediate between the CAS and the different IMSs.

In addition, we have mapped this choice into a three layer architecture, in order to meet the third requirement introduced above and to accomplish a better modularity in the design of the functionalities of the CAS.

Figure 5.1 demonstrates the complete conceptual architecture of FAST, where FAST is depicted on the right, and the generic IMS is represented on the left (Agosti and Ferro, 2004). On the whole, the architecture is organized along two dimensions:

- *horizontal decomposition* (from left to right): consists of the IMS, the gateway and the CAS. It separates the core functionalities of FAST from the problem of integrating FAST into a specific IMS.

The horizontal decomposition allows us to accomplish the first two requirements

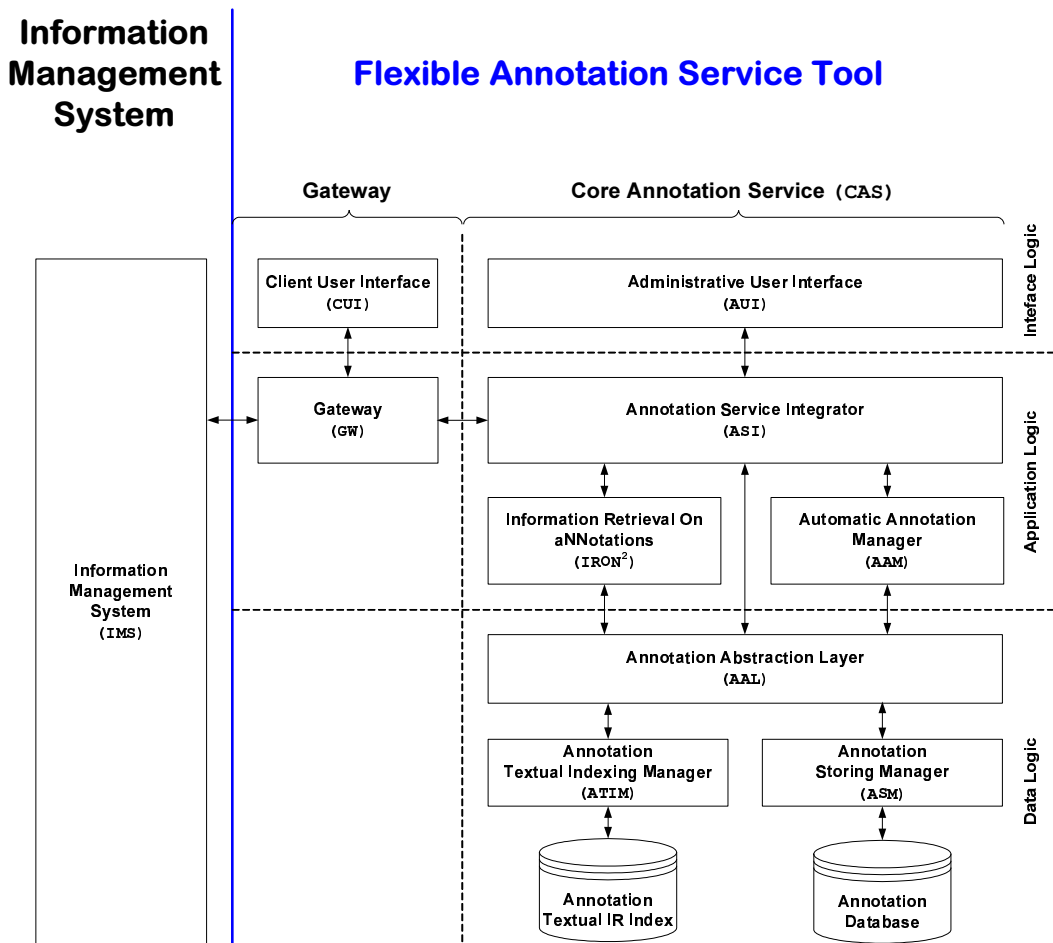


Figure 5.1: Detailed architecture of the FAST system.

of our architecture, since FAST is a stand-alone system that can be integrated with different IMSs by changing the gateway;

- *vertical decomposition* (from bottom to top): consists of three layers – the data, application and interface logic layers – and it is concerned with the organization of the CAS.

This decomposition allows us to achieve a better modularity within FAST and to properly describe the behaviour of FAST by means of isolating specific functionalities at the proper layer. Moreover, this decomposition makes it possible to clearly define the functioning of FAST by means of communication paths that connect the different components of FAST itself. In this way, we can meet the third requirement introduced above, i.e. the behaviour of the FAST system is designed in a modular and extensible way.

The conceptual architecture of FAST is designed at a high level of abstraction in terms of abstract *Application Program Interfaces (APIs)* using an *Object Oriented (OO)* approach. In this way, we can model the behaviour and the functioning of FAST without worrying about the actual implementation of each component. Different alternative implementations of each component could be provided, still keeping a coherent view of the whole architecture of the FAST system.

We achieve the abstraction level described above by means of a set of interfaces, which define the behaviour of each component of FAST in abstract terms. Then, a set of abstract classes partially implement the interfaces in order to define the actual behaviour common to all of the implementations of each component. Finally, the actual implementation is left to the concrete classes, inherited from the abstract ones, that fit FAST into a given architecture, such as a WS or a P2P architecture. Furthermore, we apply the *abstract factory* design pattern (Gamma et al., 1995), which uses a factory class that provides concrete implementations of a component, compliant with its interface, in order to guarantee a consistent way of managing the different implementations of each component.

FAST is developed using the Java¹ programming language, which ensures us great portability across different hardware and software platforms, thus providing us with a further level of flexibility.

In the following sections we describe each component of FAST, according to figure 5.1 from bottom to top.

5.2 Data Logic Layer

The data logic layer deals with the actual storage and indexing of the annotations. It consists of:

- *Annotation Storing Manager (ASM)*, described in Section 5.2.1;
- *Annotation Textual Indexing Manager (ATIM)*, described in Section 5.2.2;
- *Annotation Abstraction Layer (AAL)*, described in Section 5.2.3.

5.2.1 Annotation Storing Manager

The ASM manages the actual storage of the annotations and provides a persistence layer for storing the objects which represent the annotation and which are used by the upper layers of the architecture.

The ASM relies on a *Relational DataBase Management System (RDBMS)* in order to store annotations. The database schema is given by the mapping to the relational model of the ER schema of figure 3.3. Thus, the ASM provides a set of basic operations for storing, retrieving, updating, deleting and searching annotations in a SQL-like fashion. Furthermore, it takes care of mapping the objects which represent the annotations into their equivalent representation in the relational model, according to the *Data Access Object (DAO)*² and the *Transfer Object (TO)*³ design patterns.

The DAO implements the access mechanism required to work with the underlying data source, i.e. it offers access to the RDBMS using the *Java DataBase Connectivity (JDBC)*⁴ technology. The components that rely on the DAO are called *clients* and they use the interface exposed by the DAO, which completely hides the data source implementation details from its clients. Because the interface exposed by the DAO to clients does not change when the underlying data source implementation changes,

¹<http://java.sun.com/>

²<http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html>

³<http://java.sun.com/blueprints/corej2eepatterns/Patterns/TransferObject.html>

⁴<http://java.sun.com/products/jdbc/>

this pattern allows the DAO to adapt to different storage schemes without affecting its clients. Essentially, the DAO acts as an adapter between the clients and the data source. The DAO makes use of TOs as data carriers in order to return data to the client. The DAO may also receive data from the client in a TO in order to update the data in the underlying data source.

In conclusion, all of the other components of FAST deal only with objects representing annotations, which are the TOs of our system, without worrying about the details related to the persistence of such objects.

5.2.2 Annotation Textual Indexing Manager

The ATIM provides a set of basic operations for indexing and searching annotations for *Information Retrieval (IR)* purposes.

The ATIM is a full-text *Information Retrieval System (IRS)* and deals with the textual content of an annotation. It is based on the experience acquired in developing *Information Retrieval ON (IRON)*, the prototype IRS which has been used for participating in the *Cross-Language Evaluation Forum (CLEF)*⁵ evaluation campaigns since 2002 (Agosti et al., 2003a; Bacchin et al., 2002, 2004, 2005; Di Nunzio et al., 2004, 2005). CLEF is an international evaluation initiative aimed at providing an infrastructure for evaluating IRSs in a multilingual context.

Figure 5.2 shows the architecture of the last version of IRON, which has been used during the CLEF 2004 evaluation campaign (Di Nunzio et al., 2005). Please note that in figure 5.2 the Logging component has been duplicated to make the figure more easily legible, but there actually is only one Logging component in the system.

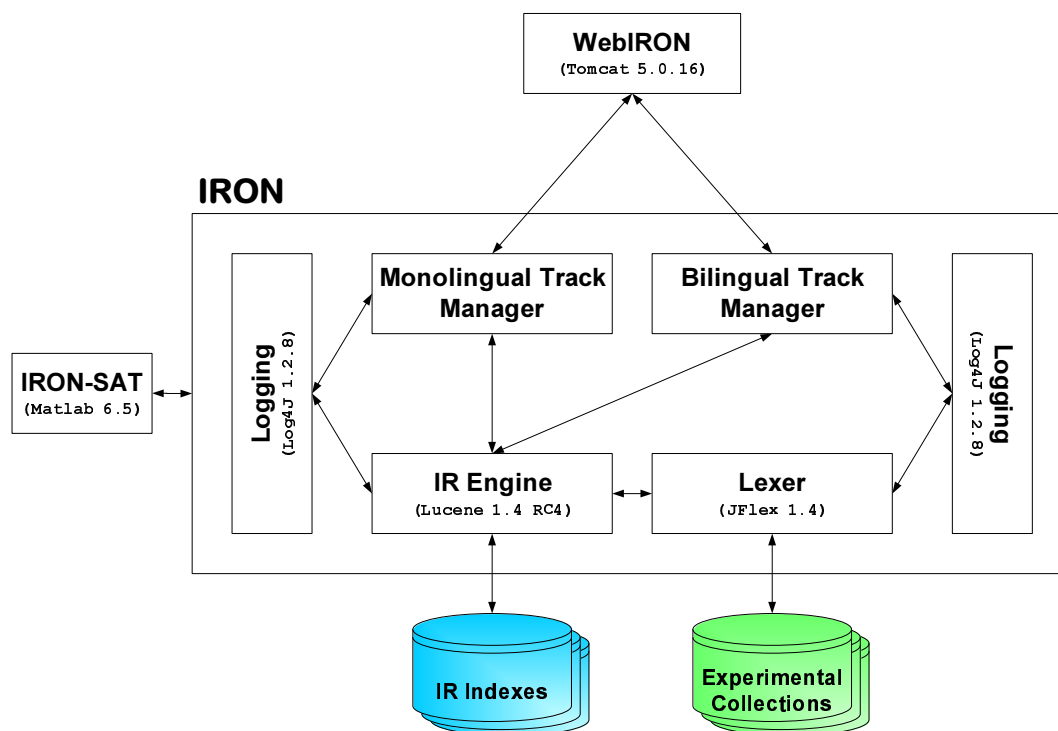


Figure 5.2: Architecture of IRON.

⁵<http://clef.isti.cnr.it/>

IRON is a Java multi-threaded program, which provides textual IR functionalities and enables concurrent indexing and searching of document collections for both monolingual and bilingual tasks. IRON is made up of the following components:

- **Lexer:** implements an efficient lexer using JFlex 1.4⁶, a lexer generator for Java. The current lexer is able to process any multilingual CLEF collection in a transparent way with respect to the document structure and to different character encodings, such as ISO 8859-1 or UNICODE⁷.
- **IR engine:** is built on top of the Lucene 1.4 RC4⁸ library, which is a high-performance text search engine library written entirely in Java. Lucene implements the vector space model, and a $(tf \times idf)$ -based weighting scheme (Baeza-Yaetes and Ribeiro-Neto, 1999; Salton and McGill, 1983; van Rijsbergen, 1979). Some parts of the Lucene library were completely rewritten, i.e. a set of parallel classes has been written without modifying the original source code of Lucene, so that IRON remain compatible with the official Jakarta distribution of Lucene. In particular, those parts of Lucene concerning the text processing, such as tokenization, elimination of stop words, stemming, and the query construction, has been modified. Furthermore Lucene has been adapted to the logging infrastructure of IRON;
- **Monolingual Track Manager:** drives the underlying IR engine and provides high-level indexing and searching functionalities in order to carry out monolingual tasks. It provides a high-level API that allows us to easily plug together the different components of an IRS. This API can be further used to create a front-end application to IRON: for example we can develop a command-line application, a GUI for a stand-alone application, or a Web based *User Interface (UI)* to IRON;
- **Bilingual Track Manager:** drives the underlying IR engine and provides high-level indexing and searching functionalities in order to carry out the bilingual tasks. As the Monolingual Track Manager, also the Bilingual Track Manager provides a high-level API that can be used to develop different kinds of UIs for IRON;
- **Logging:** provides a full-fledged logging infrastructure, based on the Log4J 1.2.8⁹ Java library. Each other component of IRON sends information about its status to the logging infrastructure, thus allowing us to track each step of the experiment.

IRON is partnered with two other tools:

- **WebIRON:** is a Java servlet¹⁰ based Web interface. WebIRON is based on the Tomcat 5.0.16¹¹ Web server, making IRON a Web application. It provides a set of wizards which help the user to set all the parameters and choose the IR components, which are needed in order to conduct a run or, more generally, an IR experiment.

⁶<http://www.jflex.de/>

⁷The lexer has been designed and developed by G. M. Di Nunzio (Di Nunzio et al., 2004, 2005).

⁸<http://jakarta.apache.org/lucene/docs/index.html>

⁹<http://logging.apache.org/log4j/docs/>

¹⁰<http://java.sun.com/products/servlet/>

¹¹<http://jakarta.apache.org/tomcat/index.html>

- **IRON - Statistical Analysis Tool (IRON-SAT)**: is a Matlab program that interacts with IRON in order to carry out the statistical analysis of the experimental results. IRON-SAT parses the experimental data and stores the parsed information into a data structure suitable for the following processing. It is designed in a modular way, so that new statistical tests can be easily added to the existing code. The statistical analysis is performed using the Statistics Toolbox 4.0 provided by Matlab 6.5¹².

5.2.3 Annotation Abstraction Layer

The AAL abstracts the upper layers from the details of the actual storage and indexing of annotations, providing uniform access to the functionalities of the ASM and the ATIM.

The AAL provides the typical *Create-Read-Update-Delete (CRUD)* data management operations, coordinating the work of the ASM and the ATIM together. For example, when we create a new annotation, we need to put it into both the ASM and the ATIM, as shown in the *Unified Modeling Language (UML)* sequence diagram (Booch et al., 1999; OMG, 2003; Rumbaugh et al., 1999) of figure 5.3.

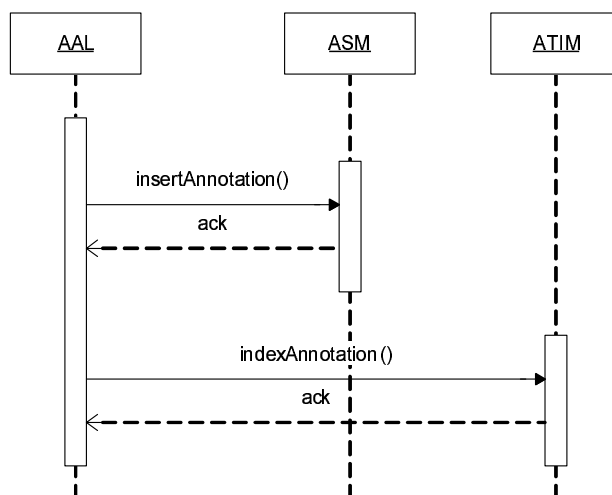


Figure 5.3: Sequence diagram for creating annotations.

Furthermore, the AAL provides search capabilities by properly forwarding the queries to the ASM or to the ATIM. Our modular architecture allows us to partner the ATIM, which is specialised for providing full text search capabilities, with other IRSs, which are specialised for indexing and searching other kinds of media. In any case, the addition of other specialised IRSs is transparent for the upper layers, due to the fact that the AAL provides the upper layers with an uniform access to those IRSs.

Note that both the ASM and the ATIM are focused on each single annotation in order to properly store and index it. On the other hand, both the ASM and the ATIM do not have a comprehensive view of the relationships that exist between documents and annotations, that is they do not take into consideration the document-annotation hypertext introduced in Section 4.9. On the contrary, the AAL has a global knowledge of the annotations and their relationships by using the document-annotation hypertext.

¹²<http://www.mathworks.com/>

For example, if we delete an annotation that is part of a thread of annotations, what policy do we need to apply? Do we delete all the annotations that refer to the deleted one or do we try to reposition those annotations? The ASM and the ATIM alone would not be able to answer this question but, on the other hand, the AAL can drive the ASM and the ATIM to perform the correct operations by exploiting the document–annotation hypertext.

In conclusion, the AAL, the ASM and the ATIM constitute an IMS specialised in managing annotations, as a DBMS is specialised in managing structured data.

5.3 Application Logic Layer

The application logic layer deals with the flow of operations within FAST in order to provide advanced annotation functionalities. It consists of:

- *Automatic Annotation Manager (AAM)*, described in Section 5.3.1;
- *Information Retrieval On aNNotations (IRON²)*, described in Section 5.3.2;
- *Annotation Service Integrator (ASI)*, described in Section 5.3.3;
- *Gateway (GW)*, described in Section 5.3.4.

5.3.1 Automatic Annotation Manager

The AAM automatically creates annotations for a given document. As discussed in Section 2.2.1, automatic annotations can be created using topic detection techniques to associate each annotation with its related topic, which constitutes the context of the annotation. In this way, a document can be re-organized and segmented into topics and annotations.

5.3.2 Information Retrieval On aNNotations

Annotations introduce a new content layer aimed at elucidating the meaning of underlying documents, so that annotations can make hidden facets of the annotated documents more explicit. Thus, we can imagine to exploit annotations for retrieval purposes in order to satisfy better the user's information needs.

Despite all of the research in modelling annotations and providing annotation–enabled systems, there is much less study regarding the usage of annotations for retrieving documents. Golovchinsky et al. (1999) compares queries based on annotations with relevance feedback, and considers annotation–based queries as an automatic technique for query construction, since queries are automatically generated from annotated text, e.g. from highlighted text. Frommholz et al. (2003) consider annotations – specifically annotations threads – as an extension of the document they belong to, creating a discourse context, in which not only the annotation itself but also its position in the discourse and its type, are exploited for searching and retrieving documents; this approach is taken up and extended in (Frommholz et al., 2004) to probabilistic datalog.

Searching documents by exploiting annotations may have different meanings depending on the different scopes of the annotation, introduced in section 4.8. For example, private annotations allow the author to greatly benefit from his personal jottings when searching for a document, while another person can consider such jottings less

useful; shared annotations may support teams in finding hot discussion topics and the annotated documents; finally, public annotations may help many people in approaching a subject by means of scholarly comments that provide more information about the annotated documents.

In any case, the common point is that we need to develop a search strategy which is able to effectively take into account the multiple sources of evidence coming from both documents and annotations. Indeed, the combining of these multiple sources of evidence can be exploited in order to improve the performances of an information management system. We aim to retrieve more relevant documents and to rank them better than the case of a query without using annotations.

IRON² exploits the formal model of annotation and the document–annotation hypertext, defined in Chapter 4, in order to provide advanced search capabilities based on annotations. At this point, we can discuss some architectural implications of a search strategy that exploits annotations, as to provide some insights about the functioning of IRON² and of the control flow within FAST. On the other hand, the exploitation of the annotations in order to search for documents is an open research problem that has not been fully addressed yet. Thus, this problem is also part of the future work, which will be presented in Chapter 6.

The UML sequence diagram of figure 5.4 shows how searching for documents by exploiting annotations involves many components of FAST. Remember that we aim at combining the source of evidence which comes from annotations, managed by FAST, with the source of evidence which comes from documents, managed by the IMS. Thus, the search strategy requires the cooperation of both FAST and the IMS in order to acquire these two sources of evidence. Firstly, FAST receives a query from the end-user, which is dispatched from the user interface to IRON². Secondly, the query is used to select all the relevant annotations, that is IRON² asks the ASI to find all the relevant annotations. Then, the document–annotation hypertext can be built and used to identify the documents that are related to the found annotations. Now we aim to combine the source of evidence which comes from the documents identified by the annotations with the one which comes from the documents managed by the IMS, as previously explained. Since the source of evidence concerning the documents is completely managed by the IMS, the FAST system has to query the IMS, which gives us back a list of relevant documents. Finally, once the FAST system has acquired this information from the IMS, it can combine this information with the source of evidence which comes from the documents identified by annotations in order to create a list of fused result documents that are presented to the users.

5.3.3 Annotation Service Integrator

The ASI integrates the underlying components and provides uniform access to them. It represents the entry point to the CAS for both the gateway and the user interface, dispatching their requests to underlying layers and then collecting the responses from the underlying layers.

The UML sequence diagram of figure 5.4 shows how the ASI plays a central role in coordinating the different components of FAST. In the example of figure 5.4, the ASI forwards the user query to IRON²; it dispatches the request for relevant documents of IRON² to the GW in order to submit this query to the IMS; then, it passes the results provided by the IMS back to IRON²; finally, it gives the fused result list produced by

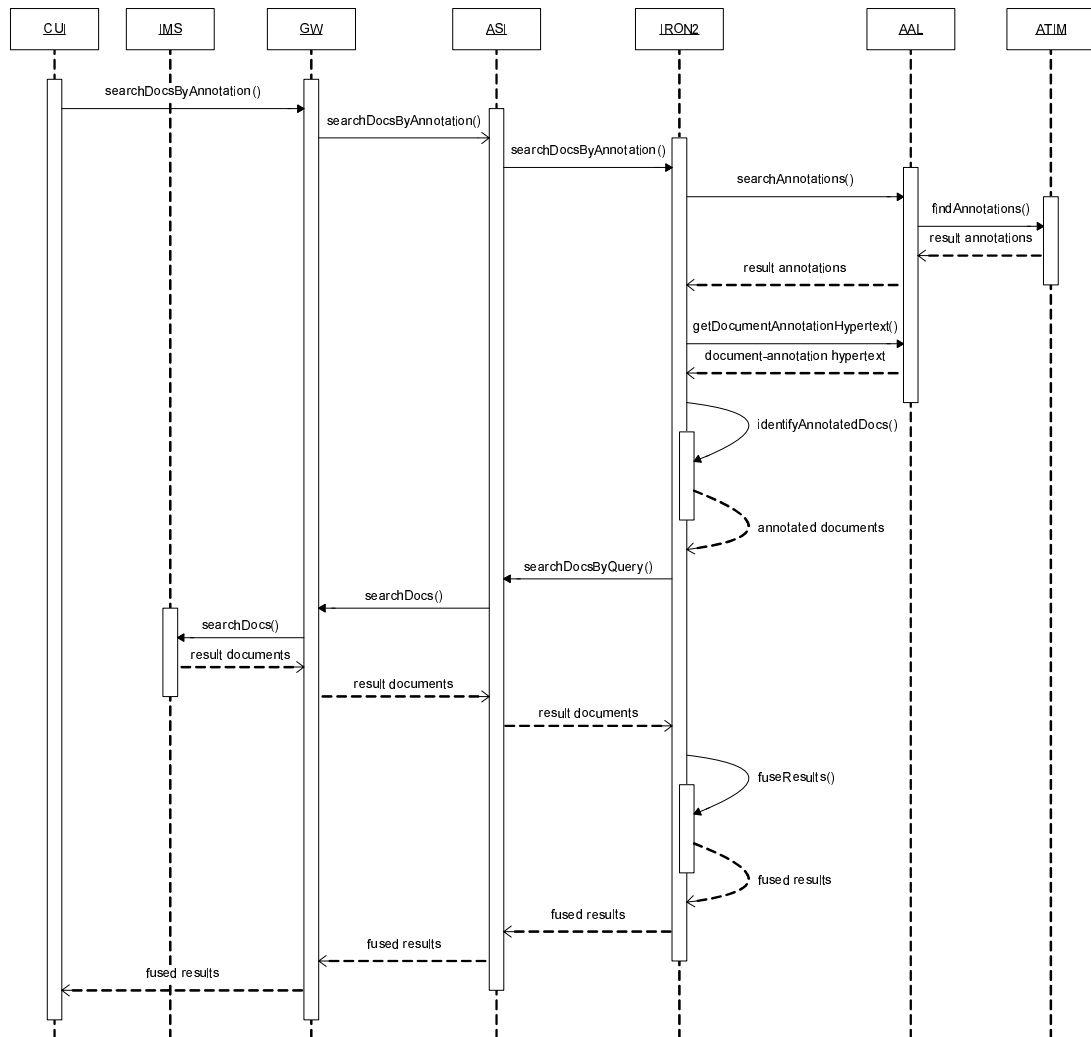


Figure 5.4: Sequence diagram for searching documents by exploiting annotations.

IRON² back to the GW in order to return this list to the user interface.

5.3.4 Gateway

As already discussed in Section 3.1, the GW provides functionalities of mediator between the CAS and the IMS. By changing the gateway, we can share FAST with different IMSs. In this way, we can provide a wide range of different architectural choices: firstly, the CAS could be connected to a IMS which uses proprietary protocols and data structures and, in this case, the gateway can implement them; secondly, we could employ WS to carry out the gateway, so that FAST is accessible in a more standardized way; finally, the gateway could be used to adapt FAST to a P2P network of IMSs.

In particular, we designed a GW that makes FAST cooperating with the OpenDLib¹³ digital library (Agosti and Ferro, 2003a). This gateway is described in the next section.

¹³<http://www.opendlib.com/>

The OpenDLib Gateway

OpenDLib is a *Digital Library Service System (DLSS)* (DELOS, 2001), that can be used to create a DL according to the requirements of a given user community, by instantiating the software appropriately and then either by loading or harvesting the content to be managed. OpenDLib provides a number of interoperating services, developed in Perl, that implement the basic functions of a DL system. The set of services is not fixed, but it can be extended to provide additional functionalities (Castelli and Pagano, 2002a,b, 2003).

The GW towards OpenDLib has a twofold aim: first, it adapts FAST to the XML-based communication protocol of OpenDLib; second, it translates the objects that represent the annotation used by FAST into the proprietary constructs of the *Document Model for Digital Libraries (DoMDL)*, which is the document model adopted by OpenDLib.

DoMDL is based on the following entities, which have been defined in (Castelli and Pagano, 2002a):

- *document*: represents the more general aspect of a document, i.e. the document as a distinct intellectual creation;
- *version*: represents a specific edition of the distinct intellectual creation, i.e. an instance along the time dimension;
- *view*: models a specific intellectual expression of an edition of a document. A view excludes physical aspects that are not integral to the intellectual perception of a document. The entity View is specialized in two sub-entities: Metadata and Content. The former view perceives a version through the conceptualizations given by its metadata representations; the latter is the view of the document content. Content has two sub-entities: Body, and Reference. The Body entity represents a view of the document content that can be either perceived as a whole or as the aggregation of other views. A Reference entity represents a view which is equal to the one that has already been registered and does not need to be explicitly stored;
- *manifestation*: models the physical formats under which a document is disseminated.

Figure 5.5 shows an example of DoMDL for a document of a thesis. We can observe that:

- we have a document, called “Thesis”;
- this document has two versions, one is called “Draft”, the other is called “Final”. The final version has some bibliographic metadata, represented in the view named “Bibdata”. This view has a manifestation, which refers to the file `dcq_final.xml`, and this is actually where the bibliographic records for the thesis are stored;
- the draft version of the thesis is organized into parts and chapters, represented respectively by the views “Parts” and “Chapters”. There are two parts, “Part 1” and “Part 2”. We can see that “Part 1” contains two chapters “Chapter 1” and “Chapter 2”, that are the same chapters contained in the view “Chapters”. So we do not have to duplicate these views, but we put a *reference* in the view

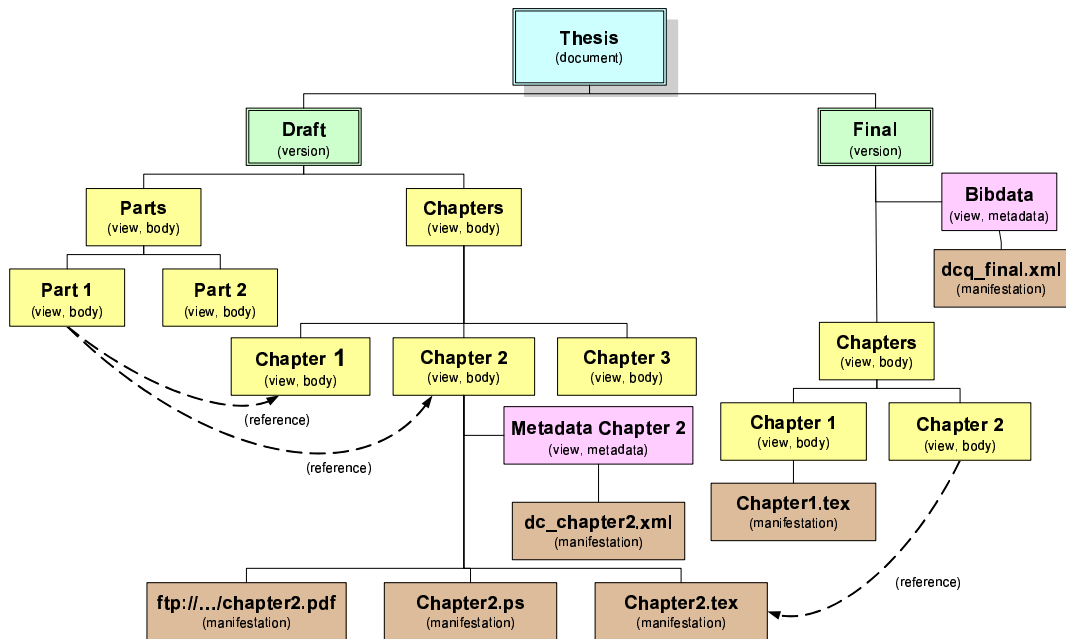


Figure 5.5: Example of DoMDL document.

“Parts” to the views “Chapter 1” and “Chapter 2” contained in the view “Chapters”. The view “Chapter 2” has many different manifestations, i.e. `chapter2.ps`, `chapter2.tex` and `ftp://.../chapter2.pdf`, and some metadata, represented by the view “Metadata Chapter 2” and stored in the manifestation `dc_chapter2.xml`;

- the final version of the thesis is organized into chapters, view “Chapters”, and we can notice that the second chapter, view “Chapter 2”, has not changed from the draft version, so we can make a *reference* to its manifestation `chapter2.tex`.

According to the constructs of DoMDL, annotations are modelled in two levels. The first level is the representation of the annotation as another kind of view, so that the DoMDL is extended, containing not only a *metadata view* and a *body view* but also an *annotation view*. The second level is the representation of the annotation as a *manifestation*.

This extension of the model is consistent with the approach taken in defining the DoMDL, preserving the way of representing the structure of a document and of its metadata. Thus, the DoMDL comes out seamlessly extended and this way of extending the DoMDL could be used as an example of transparent addition of new features to the DoMDL.

This way of representing annotations maintains the distinction between the meaning of annotation and the sign of annotation, introduced in Section 3.2. Indeed, the meaning of annotation is represented by the *annotation view*, while the sign of annotation is represented by the *annotation manifestation*. Figure 5.6 shows this distinction and highlights the entities of the ER schema of the annotation represented by the annotation view, the entities represented by the annotation manifestation, and the entities modelled by other DoMDL constructs.

In addition, this representation of the annotation allows us to easily manage both threads and set of annotations. A thread of annotation is obtained simply by nesting

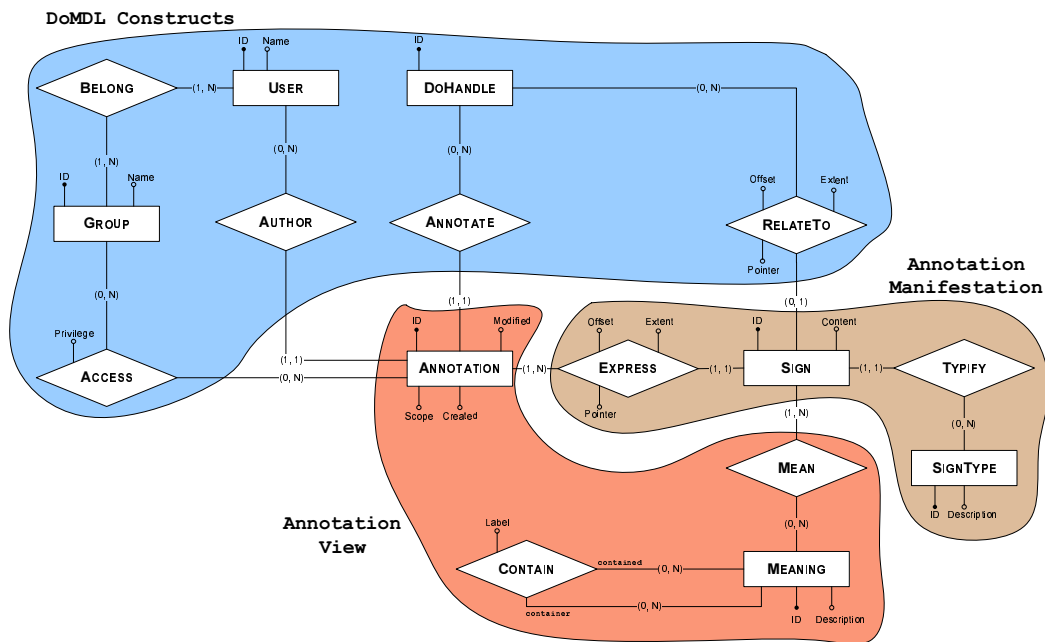


Figure 5.6: Entity-Relationship schema of figure 3.3 with respect to the mapping to DoMDL.

an annotation view into another, as it is naturally done with DoMDL; on the other hand, a set of annotations is obtained simply by putting all the annotation views in the view of the object they belong to.

Figure 5.7 reports an example of DoMDL document extended with annotations. We can observe that:

- we have added an annotation to the whole final version, represented by the view named “Annotation 1”, whose content is stored in the manifestation called `ann1_final.xml`;
- also “Chapter 2” as an annotation, represented by the view named “Annotation 2” and stored in the manifestation called `annotation2.xml`, but this annotation is shared with “Chapter 1”, so we can put a *reference* to it;
- on “Annotation 2” there is a “discussion thread”, because there is another nested annotation, represented by the view named “Annotation 3” and stored in the manifestation `annotation2.xml`. Notice that by means of the *reference* also “Chapter 1” is linked to the “discussion thread” on “Annotation 2”;
- from “Annotation 3” starts a *reference* which points to the “Chapters” view.

5.4 Interface Logic Layer

The interface logic layer deals with the interaction with the end-user. It consists of:

- *Administrative User Interface (AUI)*, described in Section 5.4.1;
- *Client User Interface (CUI)*, described in Section 5.4.2;

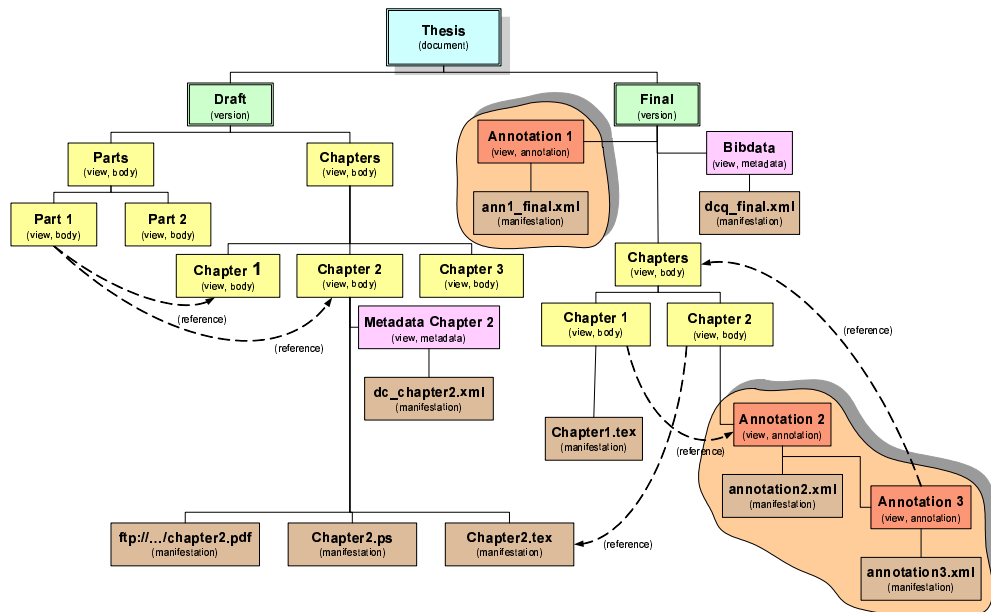


Figure 5.7: Example of DoMDL document with annotations.

5.4.1 Administrative User Interface

The AUI is a Web-based UI for the administration of FAST. It provides the different functionalities needed to configure and run FAST, such as the choice of the gateway to be used, the creation and management of the users granted by the system, and so on.

5.4.2 Client User Interface

The CUI provides end-users with an interface for creating, modifying, deleting and searching annotations.

The CUI is connected to, or even directly integrated into, the gateway, so that it represents a user interface tailored to the specific IMS for which the gateway is developed. In this way, the gateway forwards the requests from the CUI to the ASI, as it is shown in the example of figure 5.4.

Chapter 6

Conclusions and Future Work

This chapter discusses the conclusions of the research work concerning the annotations and outlines some directions for future research. In particular, Section 6.1 summarizes the major contributions of the thesis; Section 6.2 provides a future outlook for the research work that could be conducted starting from the results achieved in this thesis.

6.1 Contributions

In this thesis, we have discussed the problem of providing users with a system capable of adding annotations to different kinds of digital objects managed by IMSs that can also adhere to different architectural paradigms. In particular, we have addressed this problem in the context of DL systems and their evolution towards the *Dynamic Ubiquitous Knowledge Environments*, which aim at being “more user-centered” and “more active collaboration and communication tools” (DELOS, 2004). To this end, annotations have been studied and formalized as a service suitable for enabling and carrying out the evolution from DL systems to Dynamic Ubiquitous Knowledge Environments.

The major contributions of this thesis are as follows:

- groundwork study of the annotation, which clearly defined the contours and the complexity of the problem and obtained a set of key features of the annotation that have to be taken into account. This research added to the usual case and user studies a new complementary approach, which investigated the history of the annotation in order to capitalize on the knowledge about annotations, which comes from our cultural heritage. In particular, this study pointed out the temporal dimension involved in annotations, which by the way has never been discussed before until now. Moreover, it pointed out that documents and annotations constitute a hypertext;
- distinction between the meaning of annotation and the sign of annotation. This distinction directly comes out of our groundwork study. The notions of meaning and sign of annotation allow us to describe better both the semantics and the materialization of the annotation and to adopt a flexible approach in modelling annotations;
- conceptual model of the annotation, which provides us with a first formalization of the notions of meaning and sign of annotation. The conceptual model is an enhancement with respect to the existing models, because it allows us to describe

the annotation with the requested level of detail. In addition, it describes the semantics of an annotation in a flexible way, avoiding pre-defined types of annotations, making it possible to exploit annotations as an effective collaboration tool for users;

- formal model of the annotation, which was completely absent from the literature concerning annotations. This formal model not only captures all the aspects described by the conceptual model of annotation, but it also properly formalizes the time dimension entailed by annotations. Furthermore, it formally introduces the notion of document–annotation hypertext and explores some of its properties. Finally, it provides us with a sound theoretical basis for future research work;
- conceptual architecture of the *Flexible Annotation Service Tool (FAST)* system, which separates the core functionalities of FAST from its integration in any particular IMS. In this way, FAST acts as a bridge between different IMSs and allows the document–annotation hypertext to cross the boundaries of the single IMS, in order to exploit annotations as an active and effective collaboration tool for users.

6.2 Future Work

This thesis has laid the groundwork for formally modelling annotations and for designing a system with annotation capabilities. Additional research is necessary to cover the following topics:

- *usage of annotations in order to search for documents*: annotations provide us with an additional source of evidence, that is complementary with the one contained in the set of documents. Thus, we can exploit annotations with the ultimate goal of retrieving more relevant documents and of ranking them better. Furthermore, the paths that connect annotations to documents become the vehicle for moving this further source of evidence towards the documents and the document–annotation hypertext is the basic infrastructure for combining the sources of evidence which come from documents and annotations. In Section 5.3.2 we discussed this problem from an architectural point of view. As far as the methodological point of view is concerned, we plan to face this research problem in the context of *data fusion* (Croft, 2000), because we need to combine the source of evidence which comes from annotations with the one which comes from documents. Moreover, both *Hypertext Information Retrieval (HIR)* techniques (Agosti and Smeaton, 1996) and *link fusion* techniques (Xi et al., 2004) are suitable for extending our formal model in order to support a search strategy that involves annotations;
- *evaluation of retrieval performances*: once we have developed a search strategy that exploits annotations, we need to evaluate the retrieval performances of this strategy by using standard IR methodologies. We plan to adopt the Cranfield methodology (Cleverdon, 1997), which makes use of experimental collections in order to measure the performances of an IRS. The performances are measured using the standard precision and recall figures (Salton and McGill, 1983; van Rijsbergen, 1979), but according to Hull (1993) we also need a statistical methodology

for judging whether the measured performances can be considered statistically significant. In this context, the experience gained during the participation in the CLEF evaluation campaigns and the development of both IRON and IRON-SAT, will provide us with a strong basis for evaluating the retrieval performances of FAST. The next step will be to investigate the possibility of using measures that differ from precision and recall and are more tailored to the features of annotations. Finally, there is a lack of experimental test collections with annotated documents. Thus, the future research work will also concern the design and development of this kind of test collection, if necessary;

- *evolution of FAST towards a network of P2P FAST systems*: we plan to enhance our conceptual architecture in order to support a network of P2P FAST systems. To this end, we plan to exploit the ASI in the following way: when the ASI receives a request from the gateway – such as a search request – it forwards the request to the other ASI peers and then collects their answers in order to provide access to the whole network of P2P FAST systems. In this way, we will be able to implement FAST not only as a stand-alone system, that can be integrated into different IMSs, but also as a P2P network of FASTs that cooperate in order to provide advanced annotation functionalities to different IMSs.

Bibliography

Agosti, M. (1996). An Overview of Hypertext. In Agosti, M. and Smeaton, A., editors, *Information Retrieval and Hypertext*, pages 27–47. Kluwer Academic Publishers, Norwell (MA), USA.

Agosti, M., Bacchin, M., Ferro, N., and Melucci, M. (2003a). Improving the Automatic Retrieval of Text Documents. In Peters, C., Braschler, M., Gonzalo, J., and Kluck, M., editors, *Advances in Cross-Language Information Retrieval, Third Workshop of the Cross-Language Evaluation Forum (CLEF 2002) Revised Papers*, pages 279–290. Lecture Notes in Computer Science (LNCS) 2785, Springer, Heidelberg, Germany.

Agosti, M., Benfante, L., and Orio, N. (2003b). IPSA: A Digital Archive of Herbals to Support Scientific Research. In Sembok, T. M. T., Zaman, H. B., Chen, H., Urs, S. R., and Myaeng, S. H., editors, *Proc. 6th International Conference on Asian Digital Libraries. Digital Libraries – Digital Libraries: Technology and Management of Indigenous Knowledge (ICADL 2003)*, pages 253–264. Lecture Notes in Computer Science (LNCS) 2911, Springer, Heidelberg, Germany.

Agosti, M. and Ferro, N. (2003a). Annotations: Enriching a Digital Library. In Koch, T. and Sølvberg, I. T., editors, *Proc. 7th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2003)*, pages 88–100. Lecture Notes in Computer Science (LNCS) 2769, Springer, Heidelberg, Germany.

Agosti, M. and Ferro, N. (2003b). Chapter X: Managing the Interactions between Handheld Devices, Mobile Applications, and Users. In Lim, E. P. and Siau, K., editors, *Advances in Mobile Commerce Technologies*, pages 204–233. Idea Group, Hershey, USA.

Agosti, M. and Ferro, N. (2004). An Information Service Architecture for Annotations. In Agosti, M., Schek, H.-J., and Türker, C., editors, *Digital Library Architectures: Peer-to-Peer, Grid, and Service-Oriented, Pre-proceedings of the 6th Thematic Workshop of the EU Network of Excellence DELOS*, pages 115–126. Edizioni Libreria Progetto, Padova, Italy.

Agosti, M., Ferro, N., Frommholz, I., and Thiel, U. (2004). Annotations in Digital Libraries and Collaboratories – Facets, Models and Usage. In Heery, R. and Lyon, L., editors, *Proc. 8th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2004)*, pages 244–255. Lecture Notes in Computer Science (LNCS) 3232, Springer, Heidelberg, Germany.

Agosti, M. and Melucci, M. (2000). Information Retrieval Techniques for the Automatic Construction of Hypertext. In Kent, A. and Hall, C., editors, *Encyclopedia*

of Library and Information Science, volume 66, pages 139–172. Marcel Dekker, New York, USA.

Agosti, M. and Smeaton, A., editors (1996). *Information Retrieval and Hypertext*. Kluwer Academic Publishers, Norwell (MA), USA.

Bacchin, M., Ferro, N., and Melucci, M. (2002). The Effectiveness of a Graph-based Algorithm for Stemming. In Lim, E. P., Foo, S., Khoo, C. S. G., Chen, H., Fox, E. A., Urs, S. R., and Thanos, C., editors, *Proc. 5th International Conference on Asian Digital Libraries. Digital Libraries: People, Knowledge, and Technology (ICADL 2002)*, pages 117–128. Lecture Notes in Computer Science (LNCS) 2555, Springer, Heidelberg, Germany.

Bacchin, M., Ferro, N., and Melucci, M. (2004). A Probabilistic Model for Stemmer Generation. In Agosti, M., Dessì, N., and Schreiber, F. A., editors, *Proc. 12th Italian Symposium on Advanced Database Systems (SEBD 2004)*, pages 230–237. LITHOS-grafiche, Cagliari, Italy.

Bacchin, M., Ferro, N., and Melucci, M. (2005). A Probabilistic Model for Stemmer Generation. *Information Processing & Management*, 41(1):121–137.

Baeza-Yaetes, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison-Wesley, Harlow, England.

Berners-Lee, T. (1994a). Uniform Resource Locators (URL). RFC 1738.

Berners-Lee, T. (1994b). Universal Resource Identifiers in WWW. RFC 1630.

Berners-Lee, T., Fielding, R., Irvine, U. C., and Masinter, L. (1998). Uniform Resource Identifiers (URI): Generic Syntax. RFC 2396.

Bhagwat, D., Chiticariu, L., Tan, W.-C., and Vijayvargiya, G. (2004). An Annotation Management System for Relational Databases. In Nascimento, M. A., Özsu, M. T., Kossmann, D., Miller, R. J., Blakeley, J. A., and Schiefer, K. B., editors, *Proc. 30th International Conference on Very Large Data Bases (VLDB 2004)*, pages 900–911. Morgan Kaufmann.

Bollobás, B. (1998). *Modern Graph Theory*. Springer-Verlag, New York, USA.

Booch, G., Rumbaugh, J., and Jacobson, I. (1999). *The Unified Modeling Language User Guide*. Addison-Wesley, Reading (MA), USA.

Bottoni, P., Civica, R., Levialdi, S., Orso, L., Panizzi, E., and Trinchese, R. (2004). MADCOW: a Multimedia Digital Annotation System. In Costabile, M. F., editor, *Proc. Working Conference on Advanced Visual Interfaces (AVI 2004)*, pages 55–62. ACM Press, New York, USA.

Bottoni, P., Costabile, M. F., and Mussio, P. (1999). Specification and Dialogue Control of Visual Interaction through Visual Rewriting Systems. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 21(6):1077–1136.

- Bottoni, P., Levialedi, S., and Rizzo, P. (2003). An Analysis and Case Study of Digital Annotation. In Bianchi-Berthouze, N., editor, *Proc. 3rd International Workshop on Databases in Networked Information Systems (DNIS 2003)*, pages 216–230. Lecture Notes in Computer Science (LNCS) 2822, Springer, Heidelberg, Germany.
- Buneman, P., Khanna, S., Tajima, K., and Tan, W.-C. (2004). Archiving Scientific Data. *ACM Transactions on Database Systems (TODS)*, 29(1):2–42.
- Buneman, P., Khanna, S., and Tan, W.-C. (2001). Why and Where: A Characterization of Data Provenance. In Van den Bussche, J. and Vianu, V., editors, *Proc. 8th International Conference on Database Theory (ICDT 2001)*, pages 316–330. Lecture Notes in Computer Science (LNCS) 1973, Springer, Heidelberg, Germany.
- Buneman, P., Khanna, S., and Tan, W.-C. (2002). On Propagation of Deletions and Annotations Through Views. In Abiteboul, S., Kolaitis, P. G., and Popa, L., editors, *Proc. 21st ACM SIGMOD–SIGACT–SIGART Symposium on Principles of Database Systems (PODS 2002)*, pages 150–158. ACM Press, New York, USA.
- Calonghi, F. (1986). *Dizionario latino-italiano*, 3 edizione (interamente rifusa ed aggiornata del dizionario Georges – Calonghi) – 15 tiratura. Rosenberg & Sellier, Nuova Offlito, Mappano, Torino, Italia.
- Castelli, D. and Pagano, P. (2002a). A Flexible Repository Service: the OpenDLib Solution. In Carvalho, J. Á., Hübler, A., and Baptista, A. A., editors, *Proc. 6th International ICC/IFIP Conference on Electronic Publishing (ELPUB 2002)*, pages 194–202. Verlag für Wissenschaft und Forschung, Berlin, Germany.
- Castelli, D. and Pagano, P. (2002b). OpenDLib: a Digital Library Service System. In Agosti, M. and Thanos, C., editors, *Proc. 6th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2002)*, pages 292–308. Lecture Notes in Computer Science (LNCS) 2458, Springer, Heidelberg, Germany.
- Castelli, D. and Pagano, P. (2003). A System for Building Expandable Digital Libraries. In Henry, G., Marshall, C. C., and Delcambre, L., editors, *Proc. 3rd ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 2003)*, pages 335–345. IEEE Computer Society.
- Cleverdon, C. W. (1997). The Cranfield Tests on Index Languages Devices. In Spack Jones, K. and Willett, P., editors, *Readings in Information Retrieval*, pages 47–60. Morgan Kaufmann Publisher, Inc., San Francisco, California, USA.
- Cortelazzo, M. and Zolli, P. (1999). *DELI: Dizionario Etimologico della Lingua Italiana*, 2 edizione a cura di M. Cortelazzo e M. A. Cortelazzo. Zanichelli, Bologna, Italia.
- Croft, W. B. (2000). Combining Approaches to Information Retrieval. In Croft, W. B., editor, *Advances in Information Retrieval: Recent Research from the Center for Intelligent Information Retrieval*, pages 1–36. Kluwer Academic Publishers, Norwell (MA), USA.
- DELOS (2001). First DELOS Network of Excellence Brainstorming Workshop – Digital Libraries: Future Research Directions for a European Research Programme.

- <http://delos-noe.iei.pi.cnr.it/activities/researchforum/Brainstorming/>
[last visited 2004, November 22].
- DELOS (2004). Newsletter – Issue 2. <http://www.delos.info/newsletter/issue2/>
[last visited 2004, November 22].
- Di Nunzio, G. M., Ferro, N., Melucci, M., and Orio, N. (2004). Experiments to Evaluate Probabilistic Models for Automatic Stemmer Generation and Query Word Translation. In Peters, C., Braschler, M., Gonzalo, J., and Kluck, M., editors, *Comparative Evaluation of Multilingual Information Access Systems: Fourth Workshop of the Cross-Language Evaluation Forum (CLEF 2003) Revised Selected Papers*, pages 220–235. Lecture Notes in Computer Science (LNCS) 3237, Springer, Heidelberg, Germany.
- Di Nunzio, G. M., Ferro, N., and Orio, N. (2005). Experiments on Statistical Approaches to Compensate for Limited Linguistic Resources. In Peters, C., Clough, P., Gonzalo, J., Jones, G., Kluck, M., and Magnini, B., editors, *Fifth Workshop of the Cross-Language Evaluation Forum (CLEF 2004) Revised Selected Papers*. Lecture Notes in Computer Science (LNCS), Springer, Heidelberg, Germany (in print).
- Diestel, R. (2000). *Graph Theory*. Springer-Verlag, New York, USA.
- Fielding, R., Gettys, Y., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T. (1999). Hypertext Transfer Protocol – HTTP/1.1. RFC 2616.
- Fogli, D., Fresta, G., and Mussio, P. (2004). On Electronic Annotation and Its Implementation. In Costabile, M. F., editor, *Proc. Working Conference on Advanced Visual Interfaces (AVI 2004)*, pages 98–102. ACM Press, New York, USA.
- Freed, N. and Borenstein, N. (1996a). Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples. RFC 2049.
- Freed, N. and Borenstein, N. (1996b). Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. RFC 2045.
- Freed, N. and Borenstein, N. (1996c). Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. RFC 2046.
- Freed, N., Klensin, J., and Postel, J. (1996). Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures. RFC 2048.
- Frommholz, I., Brocks, H., Thiel, U., Neuhold, E., Iannone, L., Semeraro, G., Berardi, M., and Ceci, M. (2003). Document-Centered Collaboration for Scholars in the Humanities – The COLLATE System. In Koch, T. and Sølvberg, I. T., editors, *Proc. 7th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2003)*, pages 434–445. Lecture Notes in Computer Science (LNCS) 2769, Springer, Heidelberg, Germany.
- Frommholz, I., Thiel, U., and Kamps, T. (2004). Annotation-based Document Retrieval with Four-Valued Probabilistic Datalog. In Baeza-Yates, R., Maarek, Y., Roelleke, T., and de Vries, A. P., editors, *Proc. 3rd XML and Information Retrieval Workshop and the 1st Workshop on the Integration of Information Retrieval and Databases (WIRD2004)*, pages 31–38.

<http://homepages.cwi.nl/~arjen/wird04/wird04-proceedings.pdf> [last visited 2004, November 22].

Fuhr, N., Hansen, P., Micsik, A., and Sølvsberg, I. (2001). Digital Libraries: A Generic Classification Scheme. In Constantopoulos, P. and Sølvsberg, I. T., editors, *Proc. 5th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2001)*, pages 187–199. Lecture Notes in Computer Science (LNCS) 2163, Springer, Heidelberg, Germany.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading (MA), USA.

Golovchinsky, G., Price, M. N., and Schilit, B. N. (1999). From Reading to Retrieval: Freeform Ink Annotations as Queries. In Gey, F., Hearst, M., and Tong, R., editors, *Proc. 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1999)*, pages 19–25. ACM Press, New York, USA.

Gonçalves, M. A. and Fox, E. A. (2002). 5SL - A Language for Declarative Specification and Generation of Digital Libraries. In Hersh, W. and Marchionini, G., editors, *Proc. 2nd ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 2002)*, pages 263–272. ACM Press, New York, USA.

Gonçalves, M. A., Fox, E. A., Watson, L. T., and Kipp, N. A. (2004a). Streams, Structures, Spaces, Scenarios, Societies (5S): A Formal Model for Digital Libraries. *ACM Transactions on Information Systems (TOIS)*, 22(2):270–312.

Gonçalves, M. A., Watson, L. T., and Fox, E. A. (2004b). Towards a Digital Library Theory: A Formal Digital Library Ontology. In Dominich, S. and van Rijsbergen, C. J., editors, *ACM SIGIR Mathematical/Formal Methods in Information Retrieval Workshop (MF/IR 2004)*. http://www.dcs.vein.hu/CIR/mfir_2004.html [last visited 2004, November 22].

Gueye, B., Rigaux, P., and Spyrtatos, N. (2004). Taxonomy-Based Annotation of XML Documents: Application to eLearning Resources. In Vouros, G. A. and Panayiotopoulos, T., editors, *Proc. 3rd Hellenic Conference on AI – Methods and Applications of Artificial Intelligence (SETN 2004)*, pages 33–42. Lecture Notes in Computer Science (LNCS) 3025, Springer, Heidelberg, Germany.

Hanks, P., editor (1979). *Collins Dictionary of the English Language*. William Collins Sons & Co.Ltd., Glasgow, UK.

Hull, D. (1993). Using Statistical Testing in the Evaluation of Retrieval Experiments. In Korfhage, R., Rasmussen, E., and Willett, P., editors, *Proc. of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1993*, pages 329–338. ACM Press, New York, USA.

IEI (1950). *Enciclopedia Italiana di scienze, lettere ed arti*, volume XXIV (SCAR–SOC). Istituto della Enciclopedia Italiana, Istituto Poligrafico dello Stato, Roma, Italia.

IEI (1951). *Enciclopedia Italiana di scienze, lettere ed arti*, volume XVII (GIAP–GS). Istituto della Enciclopedia Italiana, Istituto Poligrafico dello Stato, Roma, Italia.

- IEI (1986). *Vocabolario della lingua italiana*, volume I (A–C). Istituto della Enciclopedia Italiana, Arti Grafiche Ricordi, Monotipia Olivieri, Milano, Italia.
- IEI (1987). *Vocabolario della lingua italiana*, volume II (D–L). Istituto della Enciclopedia Italiana, Arti Grafiche Ricordi, Monotipia Olivieri, Milano, Italia.
- IEI (1991). *Vocabolario della lingua italiana*, volume III** (Pe–R). Istituto della Enciclopedia Italiana, Arti Grafiche Ricordi, Monotipia Olivieri, Milano, Italia.
- IEI (1994). *Vocabolario della lingua italiana*, volume IV (S–Z). Istituto della Enciclopedia Italiana, Arti Grafiche Ricordi, Monotipia Olivieri, Milano, Italia.
- ISO (1992). Information and documentation – International standard book numbering (ISBN). Recommendation 2108:1992.
- ISO (2004). MPEG Standards - Coded representation of video and audio. <http://www.iso.ch/iso/en/prods-services/popstds/mpeg.html> [last visited 2004, November 22].
- Kahan, J. and Koivunen, M.-R. (2001). Annotea: an open RDF infrastructure for shared Web annotations. In Shen, V. Y., Saito, N., Lyu, M. R., and Zurko, M. E., editors, *Proc. 10th International Conference on World Wide Web (WWW 2001)*, pages 623–632. ACM Press, New York, USA.
- Koivunen, M.-R. and Swick, R. (2001). Metadata Based Annotation Infrastructure offers Flexibility and Extensibility for Collaborative Applications and Beyond. <http://www.w3.org/2001/Annotea/Papers/KCAP01/annotea.html> [last visited 2004, November 22].
- Koivunen, M.-R., Swick, R., and Prud'hommeaux, E. (2003). Annotea Shared Bookmarks. <http://www.w3.org/2001/Annotea/Papers/KCAP03/annotaabm.html> [last visited 2004, November 22].
- Kunze, J. (1995). Functional Recommendations for Internet Resource Locators. RFC 1736.
- Marshall, C. C. (1997). Annotation: from Paper Books to the Digital Library. In Allen, R. B. and Rasmussen, E., editors, *Proc. 2nd ACM International Conference on Digital Libraries (DL 1997)*, pages 233–240. ACM Press, New York, USA.
- Marshall, C. C. (1998). Toward an Ecology of Hypertext Annotation. In Akscyn, R., editor, *Proc. 9th ACM Conference on Hypertext and Hypermedia (HT 1998): links, objects, time and space-structure in hypermedia systems*, pages 40–49. ACM Press, New York, USA.
- Marshall, C. C. and Brush, A. J. B. (2002). From Personal to Shared Annotations. In Terveen, L. and Wixon, D., editors, *Proc. Conference on Human Factors and Computing Systems (CHI 2002) – Extended Abstracts on Human Factors in Computer Systems*, pages 812–813. ACM Press, New York, USA.
- Marshall, C. C. and Brush, A. J. B. (2004). Exploring the Relationship between Personal and Public Annotations. In Chen, H., Wactlar, H., Chen, C.-C., Lim, E.-P., and Christel, M., editors, *Proc. 4th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 2004)*, pages 349–357. ACM Press, New York, USA.

- Marshall, C. C., Golovchinsky, G., and Price, M. N. (2001a). Digital Libraries and Mobility. *Communications of the ACM*, 44:55–56.
- Marshall, C. C., Price, M. N., Golovchinsky, G., and Schilit, B. (1999). Introducing a Digital Library Reading Appliance into a Reading Group. In Rowe, N. and Fox, E. A., editors, *Proc. 4th ACM International Conference on Digital Libraries (DL 1999)*, pages 77–84. ACM Press, New York, USA.
- Marshall, C. C., Price, M. N., Golovchinsky, G., and Schilit, B. (2001b). Designing e-Books for Legal Research. In Fox, E. A. and Borgman, C. L., editors, *Proc. 1st ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 2001)*, pages 41–48. ACM Press, New York, USA.
- Marshall, C. C. and Ruotolo, C. (2002). Reading-in-the-Small: A Study of Reading on Small Form Factor Devices. In Hersh, W. and Marchionini, G., editors, *Proc. 2nd ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 2002)*, pages 56–64. ACM Press, New York, USA.
- Mealling, M. and Denenberg, R. (2002). Report from the Joint W3C/IETF URI Planning Interest Group: Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations. RFC 3305.
- Moats, R. (1997). URN Syntax. RFC 2141.
- Moore, K. (1996). Multipurpose Internet Mail Extensions (MIME) Part Three: Message Header Extensions for Non-ASCII Text. RFC 2047.
- Nagao, K. (2003). *Digital Content Annotation and Transcoding*. Artech House, Norwood (MA), USA.
- Navarro, G. and Baeza-Yates, R. (1997). Proximal Nodes: A Model to Query Document Databases by Content and Structure. *ACM Transactions on Information Systems (TOIS)*, 15(4):400–435.
- NISO (2004a). *The OpenURL Framework for Context-Sensitive Services – Part 1: ContextObject and Transport Mechanisms*. National Information Standards Organization (NISO). <http://library.caltech.edu/openurl/Standard.htm> [last visited 2004, November 22].
- NISO (2004b). *The OpenURL Framework for Context-Sensitive Services – Part 2: Initial Registry Content*. National Information Standards Organization (NISO). <http://library.caltech.edu/openurl/Standard.htm> [last visited 2004, November 22].
- OAI (2004). The Open Archives Initiative Protocol for Metadata Harvesting – Version 2.0. <http://www.openarchives.org/OAI/openarchivesprotocol.html> [last visited 2004, November 22].
- OMG (2003). OMG Unified Modeling Language Specification – March 2003, Version 1.5, formal/03-03-01. <http://www.omg.org/technology/documents/formal/uml.htm> [last visited 2004, November 22].

- Overbeek, R., Disz, T., and Stevens, R. (2004). The SEED: A Peer-To-Peer Environment for Genome Annotation. *Communication of the ACM*, 47(11):46–51.
- Park, S.-T., Pennock, D., Giles, C. L., and Krovetz, R. (2004). Analysis of Lexical Signatures for Improving Information Persistence on the World Wide Web. *ACM Transactions on Information Systems (TOIS)*, 22(4):540–572.
- Paskin, N., editor (2004). *The DOI Handbook – Edition 4.1.0*. International DOI Foundation (IDF). <http://www.doi.org/hb.html> [last visited 2004, November 22].
- Phelps, T. A. and Wilensky, R. (1996). Towards Active, Extensible, Networked Documents: Multivalent Architecture and Applications. In Fox, E. A. and Marchionini, G., editors, *Proc. 1st ACM International Conference on Digital Libraries (DL 1996)*, pages 100–108. ACM Press, New York, USA.
- Phelps, T. A. and Wilensky, R. (1997). Multivalent Annotations. In Peters, C. and Thanos, C., editors, *Proc. 1st European Conference on Research and Advanced Technology for Digital Libraries (ECDL 1997)*, pages 287–303. Lecture Notes in Computer Science (LNCS) 1324, Springer, Heidelberg, Germany.
- Phelps, T. A. and Wilensky, R. (2000). Multivalent Documents. *Communications of the ACM*, 43(6):83–90.
- Phelps, T. A. and Wilensky, R. (2001). The Multivalent Browser: A Platform for New Ideas. In Munson, E. V., editor, *Proc. 2001 ACM Symposium on Document Engineering*, pages 58–67. ACM Press, New York, USA.
- Rauber, A. (2004). DELOS and the Future of Digital Libraries. *D-Lib Magazine*, 10(10). <http://www.dlib.org/dlib/october04/10contents.html> [last visited 2004, November 22].
- Rigaux, P. and Spyrtatos, N. (2004). Metadata Inference for Document Retrieval in a Distributed Repository. In Maher, M. J., editor, *Proc. 9th Asian Computing Science Conference – Advances in Computer Science (ASIAN 2004) – Higher Decision Making. Dedicated to Jean-Louis Lassez on the Occasion of His 5th Cycle Birthday*, pages 418–436. Lecture Notes in Computer Science (LNCS) 3321, Springer, Heidelberg, Germany.
- Rocci, L. (1989). *Vocabolario greco italiano*, 34 edizione. Società Editrice Dante Alighieri, Italia.
- Rumbaugh, J., Jacobson, I., and Booch, G. (1999). *The Unified Modeling Language Reference Manual*. Addison-Wesley, Reading (MA), USA.
- Salton, G. and McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, USA.
- Sannomiya, T., Amagasa, T., Yoshikawa, M., and Uemura, S. (2001). A framework for sharing personal annotations on web resources using XML. In Orłowska, M. E. and Yoshikawa, M., editors, *Proc. Workshop on Information Technology for Virtual Enterprises*, pages 40–48. IEEE Computer Society Press.

- Schilit, B. N., Price, M. N., and Golovchinsky, G. (1998). Digital Library Information Appliances. In Witten, I., Akseyn, R., and Shipman, F. M., editors, *Proc. 3rd ACM International Conference on Digital Libraries (DL 1998)*, pages 217–226. ACM Press, New York, USA.
- Shipman, F., Price, M. N., Marshall, C. C., and Golovchinsky, G. (2003). Identifying Useful Passages in Documents based on Annotation Patterns. In Koch, T. and Sølvberg, I. T., editors, *Proc. 7th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2003)*, pages 101–112. Lecture Notes in Computer Science (LNCS) 2769, Springer, Heidelberg, Germany.
- Sollins, K. and Masinter, L. (1994). Functional Requirements for Uniform Resource Names. RFC 1737.
- Spooner, A. (1999). *A Dictionary of Synonyms and Antonyms*. Oxford University Press, New York, USA.
- Stein, L. D., Eddy, S., and Dowell, R. (2002). Distributed Sequence Annotation System (DAS) – Version 1.53. <http://www.biodas.org/documents/spec.html> [last visited 2004, November 22].
- van Rijsbergen, C. J. (1979). *Information Retrieval*. Butterworths, London, England, 2nd edition.
- W3C (1999). HTML 4.01 Specification – W3C Recommendation 24 December 1999. <http://www.w3.org/TR/html4/> [last visited 2004, November 22].
- W3C (2004a). Annotea Project. <http://www.w3.org/2001/Annotea/> [last visited 2004, November 22].
- W3C (2004b). EMMA: Extensible MultiModal Annotation markup language – W3C Working Draft 1 September 2004. <http://www.w3.org/TR/emma/> [last visited 2004, November 22].
- Xi, W., Zhang, B., Chen, Z., Lu, Y., Yan, S., Ma, W.-Y., and Fox, E. A. (2004). Link Fusion: A Unified Link Analysis Framework for Multi-Type Interrelated Data Objects. In Feldman, S., Uretsky, M., Najork, M., and Wills, C., editors, *Proc. 13th International Conference on World Wide Web (WWW 2004)*, pages 319–327. ACM Press, New York, USA.