From Knowledge Abstraction to Management

CHANDOS
INFORMATION PROFESSIONAL SERIES

Series Editor: Ruth Rikowski
(email: Rikowskigr@aol.com)

Chandos' new series of books is aimed at the busy information professional. They have been specially commissioned to provide the reader with an authoritative view of current thinking. They are designed to provide easy-to-read and (most importantly) practical coverage of topics that are of interest to librarians and other information professionals. If you would like a full listing of current and forthcoming titles, please visit www.chandospublishing.com.

New authors: we are always pleased to receive ideas for new titles; if you would like to write a book for Chandos, please contact Dr Glyn Jones on g.jones.2@elsevier.com or telephone +44 (0) 1865 843000.

From Knowledge Abstraction to Management

Using Ranganathan's Faceted Schema to develop conceptual frameworks for digital libraries

---

Aparajita Suman

ELSEVIER

AMSTERDAM • BOSTON • CAMBRIDGE • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO
Chandos Publishing is an imprint of Elsevier

CP
CHANDOS
PUBLISHING

Chandos Publishing
Elsevier Limited
The Boulevard
Langford Lane
Kidlington
Oxford OXS 16B
UK
store.elsevier.com/Chandos-Publishing-/IMP_207/

Chandos Publishing is an imprint of Elsevier Limited

Tel: +44 (0) 1865 843000
Fax: +44 (0) 1865 843010
store.elsevier.com

First published in 2014

ISBN 978-1-84334-703-3 (print)
ISBN 978-1-78063-369-5 (online)

Library of Congress Control Number: 2014931609

British Library Cataloguing-in-Publication Data.
A catalogue record for this book is available from the British Library.

Typeset by Domex e-Data Pvt. Ltd., India
Printed in the UK and USA.

List of figures and tables

Figures

Tables

List of abbreviations

AI	artificial intelligence
API	application programming interface
CASE	computer-aided software engineering
CRG	Classification Research Group
DAML	DARPA Agent Markup Language
DAML-ONT	DAML Ontology
DARPA	Defense Advanced Research Projects Agency
DDC	Dewey Decimal Classification
DIDL	Digital Item Declaration Language
DL	Description Logics
FOAF	Friend of a Friend
GUI	graphical user interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IR	information retrieval
ISKO	Society for Knowledge Organization
IST	Information Society Technologies
JSON	JavaScript Object Notation

JSP	Java Serverside Page
KIF	Knowledge Interchange Format
KL-One	Knowledge Language One
KO	knowledge organization
KR	knowledge representation
LCC	Library of Congress Classification
LIS	Library and Information Science
MDA	Model Driven Architecture
MIT	Massachusetts Institute of Technology
MPEG	Moving Picture Experts Group
MVC	model view controller
NSF	National Science Foundation
OCL	Object Constraint Language
OIL	Ontology Inference Layer
OKBC	Open Knowledge Base Connectivity
OMG	Object Management Group
OWL	Web Ontology Language
PHP	Hypertext Preprocessor
PMEST	Personality, Matter, Energy, Space, Time
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SHOE	Simple HTML Ontology Extensions
SIMILE	Semantic Interoperability of Metadata and Information in unlike Environments
SIOC	Semantically-Interlinked Online Communities

SKOS	Simple Knowledge Organization System
SPARQL	SPARQL Protocol and RDF Query Language
UDC	Universal Decimal Classification
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
XHTML	Extensible Hypertext Markup Language
XMI	XML Metadata Interchange

About the author

Aparajita Suman has around 15 years' work experience, 12 of which are exclusively in planning, setting up and managing specialized knowledge management and communication systems. She has helped organizations to use knowledge and to enable information and computer technologies in order to achieve their goals in social development in the following areas: training/capacity development workshops; strategic and operational action plans; design, implementation, monitoring and evaluation of communication strategies; developing corporate communication/information, education and communication materials; documentation; and handling conventional as well as digital media.

In her previous assignments Aparajita has filled the multiple roles of a trusted advisor and strategist across a wide range of sectors, including the private sector, national/international-level organizations, state and national government institutions (Defence Research and Development Laboratory, Hyderabad; ICICI Knowledge Park, Hyderabad; Indigene Pharmaceuticals; UNAIDS; UN Office on Drugs and Crime; Oxfam India; FHI 360). Setting new standards of excellence in communication, mass media advocacy and training, she has successfully built strong and effective knowledge management and communications systems using a combination of people, processes and technology.

Aparajita is passionate about sharing her experiences, and often dons the professorial hat in teaching courses, presenting papers and conducting training programs. She has been invited to speak at several prestigious conferences and associations across India and has also been a jury member for the Saamanjasya event at Vinod Gupta School of Management, Indian Institute of Technology, Kharagpur for the NGO Case Study Awards.

Academic background: Aparajita holds a doctorate in Semantics of Knowledge Management and has earned Master's degrees in three faculties, including one in Information Management from the Indian Statistical Institute, Bangalore. She also holds postgraduate qualifications in Intellectual Property Rights (World Intellectual Property Office, Geneva; National Law School of India University, Bangalore) and Corporate Communications, along with international certifications in computers (International Computer Driving Licence), logical frameworks, e-commerce, and has publications to her credit.

Training in online advertising (Advanced Online Advertising Program, Google and NIIT Imperia) and communication (Oxfam South Asia digital communication training) has helped her further to hone and update her skills.

1

Introduction: knowledge abstraction: problems and context-based solution

Abstract: Knowledge and information portals are not new to the present generation of professionals, and have become more or less synonymous with storage and retrieval of digital information. As the volume of information increases, and likewise the demand for it, it is essential to devise a logical way of information organization. Ontology, extensively used in philosophy to define the state of being, has now been borrowed and customized to define the existence of a digital resource. But, ontology per se cannot be the panacea for all the problems associated with the retrieval of information from web. The concept and the technology need to be redefined. Only then will there be way out of the present web of information where chaos reigns and there is little organization.

This chapter provides an overview of the context-based faceted approach, including a brief introduction, facets of the problem, and context of the concept of knowledge abstraction. It also provides hypotheses, scope, and methodology adapted to finding a way through the information overload.

Key words: Ontology, analytico-synthetic organization, Colon Classification, UML.

Background

The knowledge portal does not need any formal introduction; it has become synonymous with the storage and retrieval of digital information on the internet/intranet. Originally a scholarly venture to provide user information at the desktop, it is now one of the most popular and reliable sources of information.

The increasing volume of and demand for information have made the logical organization of information essential. The concept of ontology has recently been borrowed from philosophy and is being used as a mechanism to define the existence of a document on the World Wide Web.[1] Ontologies can be used to define the following stages in facilitating a distributed information retrieval system:

- initial construction
- assisting users to form queries
- decomposing and translating queries expressed in one or more high-level domains into a query plan for specific data sources.

However, ontology alone is not enough to express the problems associated with the retrieval of information from the web. Redefining both the concept and technology for the purpose of knowledge portals is the only way out of the present web of information, where there is chaos and little organization.

Knowledge organization: problems and context-based solution

A successful strategy allows the concept to grow from the core and represent various manifestations. The era of print documents and library classification schemes provides a ray of

hope, particularly Colon Classification and analytico-synthetic classification. The most impressive result of this classification scheme for documents in a library, designed by Dr. S. R. Ranganathan, is the way in which documents are arranged on the shelf. The system reflects every user's information needs in a spectrum from precise domain to peripheral subjects, serving the APUPA (Alien-Penumbral-Umbral-Penumbral-Alien) model of establishing formal relationships between subjects as on the shelves of a library. The idea proposed in this book builds on this key principle; why can't online resources be arranged in the same way on the web?

A web user looking for a document on democracy, for example, would first use the search engine to search for "democracy" and then select the most relevant result in an attempt to find the specific information or required document. However, this will satisfy only the information seeker's primary information need. The role of the digital librarian goes beyond implementing a simple keyword box-based search; we should be able to map and display logically the subject domain and related cross domains in the digital repository. Searching for a particular piece of information should return the specific document and the whole spectrum of associated concepts/domains, similar to the APUPA pattern on the shelf of the physical library.

This can be achieved in an interesting way if the shelf arrangement is based on the scientific mapping of different subjects used in Colon Classification and the same concept is then extrapolated to enable facet analysis.

Facetization: extrapolation of concepts for framework

According to the theory behind Colon Classification, each subject or domain consists of facets, which are defined as distinct divisions of a domain.

A domain is made of Entities (E); entities have Properties (P); and there are Actions by or on the entities. Hence categories of concepts belong to distinct divisions of the domain, such as Entity, Property, and Action. Entities are called "Personality" (P) and properties "Matter" (M), while actions such as "Energy" (E) and the concepts of Space (S) and Time (T) are associated across domains. Hence, every domain represented as a subject (BS) has facets P, M, E, S and T; the simplest possible implementation of the theory of Faceted Classification and the facet formula (BS),P;M:E'S'T serves as a generic framework to model the domain.

The more complex implementation of this model deals with compound and complex subjects with cross-domain, intra-facet and inter-facet concepts. The framework based on this concept will facilitate semantic searching using Unified Modeling Language (UML) concept maps. It will also enable domain experts without much coding expertise to create ontologies for the resources when submitting them to digital libraries.

An approach to bring hope

The problem is not lack of information, but the lack of reliable tools for its retrieval. This is the result of the Internet's size, distributed nature and rapid evolution. Today most search engines can index and search documents; however, the different techniques available (including exact word, truncation, Boolean and proximity searches, as well as other techniques such as fuzzy, soundex, ranking of query results, case sensitive/insensitive), are nowhere near the ideal approach, which would be based on the way information is logically organized in the human mind. The study of reusable knowledge components (ontologies) is a similar type of

approach. Marking up web pages semantically using terms from an explicit ontology can improve retrieval and help to integrate data many pages in length. Study of information organization in libraries worldwide is needed. The same concept can also be applied to digital libraries; however, it must be remembered that the basic anatomy of digital libraries/information portals is completely different from that of traditional libraries.

Most existing libraries are now going digital, while at the same time many new digital libraries are being established, the reason being the ever-increasing volume of literature published online. Users are now demanding information services in digital format because of the associated ease and speed of retrieval. Tools that will allow proper semantic querying to identify things that match a meaning in natural language and vice versa are needed, along with corresponding definition and storage mechanisms. This book takes up an analytical study of existing ontology tools and techniques and attempts to develop a conceptual framework from a digital library perspective. From the current perspective of global knowledge representation, there is a need to remodel a meaningful semantic internet for the modeling, representation and exchange of knowledge.

Tim Berners-Lee's[2] vision projects the Semantic Web as a gradual, stepwise tower of semantic languages. The development of knowledge resources in the form of ontological vocabularies is a significant contribution to this project. There was a need to develop a layered ontology structure for representing various domain perspectives; conceptual meanings and their interpretations were supposed to be inherent in the representation and expressed in a multi-tier ontology.

However, the choice of knowledge representation methodology depends on the purpose as well as the intended

audience. It has been found that conceptual models can be developed using UML.[3] As knowledge-base resources form an essential component of any information system, so they should use a knowledge representation language that is independent of the application domain. Moreover, the language should be clear, easy to understand and portable. UML provides a modular approach to knowledge representation and an ontologies-based modular framework in digital libraries would facilitate answering specific and specialized questions. The transformation of UML models into the language of ontology can be accomplished using different available tools, but the approaches will vary. The challenge lies in deciding on different approaches to translate a subset of UML class diagrams in general and, beyond this common core, on the different approaches for the translation of constructs. The actual choice for a specific translation always correlates directly with the purpose of the translation. A successful strategy will start from the common core and will make more specific transformation choices based on the intended use of the resulting ontology.

The advantages of UML as a conceptual modeling language have been listed by Baclawski et al.[4] as:

- It has a growing user audience in the software domain for object modeling languages and other information system design.
- The graphical notation for UML is easy to comprehend and use and is suitable for human-to-human knowledge transfer.
- UML can be extended to suit the need of ontology definitions.
- Object Constraint Language allows expression of rules and constraints.

- UML conceptual models can be easily translated into other ontology languages such as Resource Description Framework Schema (RDFS) or DARPA Agent Markup Language (DAML) or even into object-oriented database systems.

Semantics are embedded in UML in various forms: a direct mathematical model of the system described by using UML;[5] a description using the specification language Z;[6] and operational semantics that describe the evolution of a UML model on the addition of new elements.[7] Semantics for Object Constraint Language (OCL) are represented in the class diagrams, as proposed by Richters and Gogolla[8] and Hamie et al.[9]

The Semantic Web and its Extensible Markup Language (XML)-based languages[10] are the main pillars of future web development. Domain ontologies embedded in the framework form the major components of Semantic Web applications; these are formal organizations of domain knowledge and so enable knowledge sharing between different knowledge-based applications. The Semantic Web enables machine-understandable data to be shared across the net; ontologies give machine-understandable meaning to its data. UML appears to be the best of the existing technologies and tools for the formal conceptualization of particular domains. These interoperable ontologies facilitate web/knowledge repositories that understand both user and information seeker.

As the concept of the Semantic Web has grown, various representation languages have come into being and have later evolved to be standards. The majority of ontology languages have been developed on top of RDFS, in accordance with the World Wide Web Consortium (W3C) recommendation to use Resource Description Framework (RDF) languages, RDFS and XML and XML Schema, as the standard for

specifying metadata on the web. Later, W3C proposed Web Ontology Language (OWL) (2008) as an extension of XML, RDF and RDFS and a new standard for ontology development.[11]

Using UML for developing a conceptual framework vis-à-vis RDFS(FA) has several advantages:

- The meta-modeling approach is bootstrapped using a well-formed language that has well-defined semantics; i.e. the UML. RDFS(FA) does not have a bootstrap layer which would allow multi-layering over its top-level layer and hence would become a source of ambiguity.
- RDFS(FA) uses a Directed Label Graph (DLG) to describe layers.
- UML models are more expressive than DLGs and they are easier to understand and codify by software architects, designers and developers. Using stereotyping enables the type of an element to be distinguished easily. DLG does not provide this visual effect.
- RDFS(FA) was developed based on RDFS and to solve problems in the RDFS specification.
- RDFS(FA) describes a portion of the RDFS specification.

The basic idea is to make information retrieval more precise by developing a faceted approach for organizing the information in digital library repositories. This approach will help information professionals to develop a conceptual framework which will serve as a guideline for ontology formation for the purpose of digital libraries and as a basis for further skills in ontology development. To begin with, facet formulae will be studied and their key capabilities will be applied to a generic framework for knowledge organization in digital libraries or elsewhere on the web. A UML-based framework will be developed to serve as a model to build

ontologies in different domains based on facet formula, as prescribed in the theory of classification for each domain. This framework will provide a generic tool for mapping both the subject intention (depth) and extension (scope) in terms of its distinct facets.

UML can be used to generate visually configurable query/search/filter engines, while the ontologies thus generated will lead to the development of mature vocabularies, along with the mapping of different facets/aspects of the resource/background problems (based on facet analysis). The UML/RDF vocabulary will enable conceptual modeling constructs, while the ultimate meta-modeling approach will be defined by the semantics of UML.

The conceptual framework, when implemented, provides the structure for ontologies being built in different domains. The advantage of this framework lies in the subject analysis in the process of facetization and mapping it into the facet formula. Since knowledge of facetization and its process cannot be presupposed, the UML framework intrinsically incorporates knowledge structures for domains based on facetization. This model can be integrated in any Semantic Web application where the relations and correlations remain intact, irrespective of the ontology language used and without any need for the subject expert to understand this.

References

1. Gruninger, M., Bodenreider, O., Olken, F., Obrst, L., and Yim, P. (2008) "Ontology Summit 2007 – Ontology, Taxonomy, Folksonomy: Understanding the Distinctions," *Applied Ontology* 2: 191–200.
2. Berners-Lee, T. (1998) "Semantic Web Road Map," W3C. Available at: *http://www.w3.org/DesignIssues/Semantic.html* [Accessed 2 September 2013].

3. Kabilan, V. and Johannesson, P. (2003) "Semantic Representation of Contract Knowledge Using Multi Tier Ontology," in *Proceedings of the Semantic Web and Databases Workshop*, 395–414.
4. Baclawski, K., Kokar, M., Kogut, P., Hart, L., Smith, J., Holmes, W., Letkowski, J., and Aronson M. (2001) "Extending UML to Support Ontology Engineering for the Semantic Web," in *Proceedings of the Fourth International Conference on UML (UML2001), Toronto*.
5. Breu, R. (1997) "Towards a Precise Semantics for Object-oriented Modeling Techniques," in H. Kilov and B. Rumpe (eds) *Proceedings ECOOP'97 Workshop on Precise Semantics for Object-Oriented Modeling Techniques*, 53–59.
6. Evans, A. (1998) "Developing the UML as a Formal Modeling Notation," in P.-A. Muller and J. Bézivin (eds) *Proceedings of UML'98 International Workshop, Mulhouse, France, June 3–4, 1998*, 297–307.
7. O'Vergaard, G. (1998) "A Formal Approach to Relationships in the Unified Modeling Language," in M. Broy, D. Coleman, T. S. E. Maibaum, and B. Rumpe (eds) *Proceedings PSMT'98 Workshop on Precise Semantics for Modeling Techniques* (Technische Universität München TUM-I9803).
8. Richters, M. and Gogolla, M. (1998) "On Formalizing the UML Object Constraint Language OCL," in T. W. Ling, S. Ram, and M. L. Lee (eds) *Proceedings of the 17th International Conference Conceptual Modeling (ER'98) Lecture Notes in Computer Science, number 1507* (Springer-Verlag).
9. Hamie, A. (1998) "Interpreting the Object Constraint Language," in *Proceedings of the 5th Asia Pacific Software Engineering Conference (APSEC'98)* (IEEE Press).
10. Djuric, D., Gasevic, D., and Devedzic, V. (2003) "A MDA-based Approach to the Ontology Definition Metamodel" (University of Belgrade).
11. Naeve, A., Nilsson, M., and Palmér, M. (2001) "The Conceptual Web – Our Research Vision," in *Proceedings of the First Semantic Web Working Symposium* (CID, Centre for User Oriented IT Design, Royal Institute of Technology, Stockholm). Available from: *http://kmr.nada.kth.se/papers/SemanticWeb/pospaper.pdf* [Accessed 15 August 2006].

2

Non-semantic vs. Semantic Web: the architecture and tools

Abstract: The non-semantic web has grown "like a virus," acquiring gigantic proportions in a very short time. Today, people use the World Wide Web to carry out multiple tasks such as accessing and publishing information, shopping, operating businesses, socializing, and interacting with their peers.

A "Semantic Web," in which the day-to-day mechanisms of trade, bureaucracy, and our daily lives will be handled by machines talking to machines is yet to be realized. A computer can accomplish these tasks only with human direction because the main reader of any web page is the human user, not a machine. Recent research in this regard has been in the area of the use of ontologies in the information domain. This could provide the much sought-after panacea for the difficulty of finding, accessing, presenting, and maintaining the information required by a wide variety of users.

This chapter presents a detailed introduction to the Semantic Web and its features. Selected tools and technologies are examined to highlight the features/applications significant for digital libraries and electronic collection development.

Key words: Semantic Web, knowledge management, RDFS, SKOS, Semantic Web tools.

Background

The growth of the digital world could be termed as nothing but prodigious. The current information infrastructure is beyond what could have been anticipated even by the creators of the World Wide Web. The volume of information is enormous; the World Wide Web indexes at least 63.42 billion pages,[1] and search engines have become like gods, claiming to answer any information request almost instantaneously.

The web has grown "like a virus," acquiring gigantic proportions in an incredibly short time. Today we use the web to carry out tasks such as accessing and publishing information, shopping, operating businesses, socializing, and interacting with our peers. However, a computer can accomplish these tasks only if directed by a person, since the main reader of any web page is the human user, not a machine. Text documents written in Hypertext Markup Language (HTML) constitute the majority of the World Wide Web, but whereas HTML is used for text coding, the content of multimedia objects – images, interactive forms, etc. – is categorized using metadata tags.

Years ago, when no one could have imagined the growth of the World Wide Web, there was no context in which to discuss semantics. Now, however, one hears much discussion about developing a global web of information aided by a semantic approach; an approach that will lead to the meaningful interpretation of search queries. The concept of semantics is furthered by the Semantic Web, which involves publishing data in a language that is equally intelligible to the machine and to the human user. The first article on the Semantic Web appeared in *Scientific American* in 2001.[2] It described the Semantic Web as a: "Web of actionable

information – information derived from data through a semantic theory for interpreting the symbols." According to semantic theory, the term "meaning" establishes a logical connection between terms and, consequently, a state of interoperability between systems. The idea builds tangible coordinates around the concept of the web, as envisioned by Tim Berners-Lee at the first World Wide Web Conference in 1994.

Berners-Lee expressed his vision of the Semantic Web in the following terms:[3]

> I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. A "Semantic Web", which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The "intelligent agents" people have touted for ages will finally materialize.

According to the *Scientific American* article, although the proposed concept was straightforward, it was difficult to implement and the problem was compounded by the exponential growth of resources on the web. There was discussion of intelligent intermediaries, namely agents, bots, self-learning systems, artificial intelligence (AI) algorithms, and so on. However, dealing with heterogeneous data and different types of information was no easy task. Tools and standards have continued to evolve from the conception of the Semantic Web until today, in the attempt to make it a reality.

The Semantic Web

Definition

To put it simply, the Semantic Web will enable computers to "read" and use the web. The idea is to add metadata to web pages which will allow machines to "read" the concept over the World Wide Web. This will not bestow AI, but it will give machines the tools to help them to find, exchange, and interpret information. The Semantic Web will therefore be an extension of the World Wide Web, but will not replace it.

Why the Semantic Web

The Semantic Web will help users to put into their queries context. Information never exists in isolation but is always set within a context or some set of basic criteria. Using the current World Wide Web, to narrow down the conditions in one search interface is difficult, and most of the time impossible. The Semantic Web allows users to enter preferences into a sort of computerized agent which searches the web to find the best option. Its unique feature is that this is not done by looking at the titles or content of pages, but by sifting through metadata for clear identifications and definitions.

Metadata is simply machine-readable data describing other data, invisible to a normal user but clearly visible to computers in the Semantic Web. Metadata also helps in more complex, focused web searches that produce more accurate results. For example, the non-Semantic Web (Web 1.0 and Web 2.0) would search for:

<item>Horse</item>

while the Semantic Web (part of Web 3.0) would search for:

<item rdf:about="http://dbpedia.org/resource/Horse">
Horse</item>.

What the Semantic Web is and what it is not[4]

Many components

The Semantic Web is not a single component. It will not be an all-inclusive entity, but will be made up of numerous semantic components spread across the web. These components will make use of open-standard languages and, by making the data universally accessible, will become part of one Semantic Web. However, the components will not be controlled by a single company or institution, and the decentralization will be the same as in today's web.

Part of the existing web

The Semantic Web is part of the existing web. In the Semantic Web, new metadata and data are added to the existing HTML web and merge with it seamlessly, similarly to XML. The difference is that the new metadata is coded through a web ontology language (OWL).

Not only unstructured data

The Semantic Web involves not only unstructured data. Although it is commonly confused with unstructured data, it is actually about structured data, providing the means/tools for the conversion of data/content into a structured form that can be used by other software/tools/technologies.

Simple ontologies

The Semantic Web requires simple ontologies. It does not always require complex ontologies and an awareness of sophisticated languages, but is more to do with the concept of wrapping every piece of information in the relevant context and, similarly, having the necessary tools/technologies to help end-users in their semantic quest for information.

Not just about web pages

The Semantic Web is not all about web pages. Calling it a "web" has added to the misconception that it is only about the web. The Semantic Web is about *information*, whether that be data on the web, in applications, or in databases (the hidden web).

Not just about AI

The Semantic Web is not just about AI. It was actually proposed as a pragmatic solution when attempts to use AI to regulate the growth of information on the web had failed. AI cannot be regarded as the sole benefit of the Semantic Web, but the next generation of AI will certainly benefit from richer semantics. Semantic Web schema will make data more accessible, easier to integrate, and reusable.

Empowering the information creator

The Semantic Web does not involve just data mining, search engines, and spidering; the problem of non-semantic search and retrieval will not be solved just by adding to search algorithms. The Semantic Web is more about empowering the creator/publisher of the information. The providers/developers of content – and the ultimate end-users – benefit from using the

available services; even without any knowledge of mining technologies and tools, one can embed meaningful metadata in the primary content.

Content management

It becomes easy to manage the content, and so developers and content providers do not have to look to relational data models. At the same time, end-users gain by having better ways to collect and manage content.

Already in use

The Semantic Web is not simply a research project. The concept of the Semantic Web is already in use and has started to become part of the World Wide Web.

Components of the Semantic Web

Various standards and tools, namely XML, XML Schema, RDF, RDF Schema, and OWL, are organized in a stack to make up the semantic Web:

- XML: the basic syntax for content structure within documents; does not associate any semantics with the meaning of the embedded content.
- XML Schema: a language that both provides and restricts the structure and content of elements present within XML documents.
- RDF: a language that helps in the expression of data models which describe objects ("resources") and their relationships; these models can also be expressed using XML syntax.

- RDF Schema: a vocabulary that describes properties and classes of RDF-based resources, and also has semantics for general hierarchies involving classes and properties.
- OWL: a language that provides better vocabulary for description of properties and classes, relations between classes (e.g. disjointness), equality, cardinality (e.g. "exactly one"), enumerated classes, and characteristics of properties.
- SPARQL:[5] a protocol that helps in querying Semantic Web data sources.

Features and properties[6]

Globally inclusive

The reason for both the popularity and the associated problems of the web is that it allows everyone to link to all published documents and allows everyone to view all published web pages. The page owner's permission is not needed to hyperlink to their page, and similarly the page owner needs no permission from those linking to the page before changing or moving/deleting the linked page.

The Semantic Web will be fully realized only when anyone will be able to create metadata for any page and share the same with everyone over the Internet.

Collaborative

Metadata can be useful for many things. In the Semantic Web it will essentially be shared and the creation of the common pool of semantic data will be a collaborative effort.

Interoperable metadata

The Semantic Web is all about the user creating semantic data intentionally, in the same way that web pages are published. The motive is to benefit from the sharing of metadata and to derive the maximum benefit from the smallest contribution. This will motivate people to create metadata formats that are customized for every tool so as to help customers or meet specific short-term business needs. By the same token, the metadata will be made interoperable with other tools for permanence and sustainability.

Architecture of the Semantic Web: work behind the screen

Semantic metadata could be defined as that extra piece of information pertaining to a page that helps search and retrieval tools to produce meaningful results.

Metadata in web pages[7]

On-the-fly text parsing

Scanning the text of a web page helps search tools to discover meaningful information. This will not be the usual keyword parsing, but will include features for understanding the grammatical structure of sentences.

Microsoft's MindNet database and Princeton's WordNet can provide the foundations for establishing the structure of sentences. Generally, they cover words and word relationships in the complete range of commonly used words, but with increasing specificity; for example, a specific list might be used for a certain discipline. In this way, semantic tools will

extract semantic information on the fly and will be a "value add" for any software/utility.

Embedded in the page

Metadata can be directly embedded into a page by the content publisher. A useful application is in filtering software, where text is filtered based on the content provider's mark-up (e.g. adult-only content, not for children, etc.). However, this mark-up will not be a blanket statement; rather, it will accommodate finer features and sub-features of the text. By embedding such metadata, publishers can minimize the risk of legal problems, and also make their pages more useful.

User-published files

A user can create metadata files to describe resources on the web and place them on a web server. This is very similar to embedding the metadata in a specific page but, as the user does not have access to the page itself, the metadata is just put on a server and cross-linked. This is not an active method because tools/crawlers will need to be trained to pick up the linked metadata file from a server different than the one on which the web page resides. Maintaining a standard across the metadata files generated by different users of the same web page could pose a problem; the challenge would be to weed out irrelevant metadata.

Service provided by site

A better option is for the hosting site to enable users to add their own metadata directly. For example, articles on Microsoft Developer Network can be commented on by users; similarly, CNET News (news.cnet.com) allows people

to comment on news articles. These sites use a range of web-based interfaces for commenting and store metadata entered by users in databases owned by the sites.

Batch text parsing (crawler)

This service is available with current search engines. The crawler can extract information such as the approximate number of pages linking to a particular page, the frequency of occurrence of a specific word, the similarity of pages to a given page, and so on. Even the text parsing is quite close to that described above. If the crawlers become capable of metadata crawling as a specific activity, this will be a semantic value addition to existing search engines.

Specialized metadata server

Some search and retrieval tools use their own servers for storing and querying metadata about documents. Third Voice, for example, allows users to store annotations and related links for web pages visited. This metadata server works as a backend for the semantic crawling tool and thus allows users to collaborate and share annotations. The challenge is to interest users in collaborating and sharing by using innovative reward/benefit methods.

Generic metadata server

A generic metadata server would be a storage and retrieval engine for all of the tools that use shared and collaborative metadata. Such a system, which may sound utopian, would allow users to mark pages or store URLs, or to store information relating to the usefulness/popularity of users' comments regarding a web page.

Challenges of Semantic Web implementation[7]

Centralization

People do not like to store their data on centralized servers because they are concerned about privacy. Directly annotating/tagging a document can also lead to accusations of false or inappropriate marking-up. To address these concerns, it is essential for the metadata to be neither on servers that are too centralized nor on servers that host the data that is being tagged. It would be wise for there to be a standard way in which tools can query metadata servers and not to encourage proprietary standards.

Scaling metadata

Scalability factors constitute the major difference between metadata and data. Data is accessed one piece at a time, but metadata is useful only in aggregate. But again, it has to be presented in a manner that is interesting to users, and this is not possible when done by the publisher. Ideally, metadata should be stored with the web page; however, web page owners can remove metadata which they consider not to be useful.

Decentralize data, centralize metadata

Metadata can be indexed, searched, and processed by tools; fast querying is made possible by centralization, and decentralization helps in rapid retrieval. However, real efficiency will be achieved only when the storage of metadata is decentralized and indexing and querying are centralized.

Metadata explosion

The real worry for advocates of semantic tagging is the possibility that the volume of metadata that is generated could outgrow the volume of data that is being tagged. Consequently, the number of queries made against that metadata could be greater than the number of requests for the related web pages. Storing a page is very different from storing dynamically changing metadata about that page; the metadata may be greater in size than the page itself. Storage capacity sufficient to cache the entire web would be needed. Only a few centralized players would be able to afford the expense involved, and the resulting over-centralization would defeat the purpose of the Semantic Web.

Syndication and aggregation

Simply defined, syndication is the transmission of information from one source to many; aggregation is the pooling of information from many sites to one. Syndication can therefore be termed a decentralizing flow, and aggregation a centralizing one. Databases aggregate data by using functions that have been defined across groupings of the data.

This brings us back to the question of what would be the ideal and practical approach: aggregation, syndication, or both. The answer is not one standard approach, but multiple approaches through multiple channels – a series of flows involving three stages: centralization, aggregation, and subsequent decentralization. However, the decision as to how the data will be decentralized, syndicated, and aggregated will depend largely on the context and case. Human intervention at different stages of establishing filtering rules, retention policies, and so on cannot be completely ruled out.

To summarize, the architecture will be:

- decentralized and open
- centralized uses of metadata that are associated with tools
- inclusive and interoperable
- application of aggregation, syndication and filtering techniques.

Semantic Web technologies

The Semantic Web is talked about as a separate entity to or replacement of the existing web, whereas it is actually an extension and enhancement of the web by the application of Semantic Web technologies. These technologies are considered in layers, where each layer resides on top of and extends the functionality of the layer below.

As shown in Figure 2.1, Hypertext Transfer Protocol (HTTP) and Uniform Resource Identifiers (URIs) form the

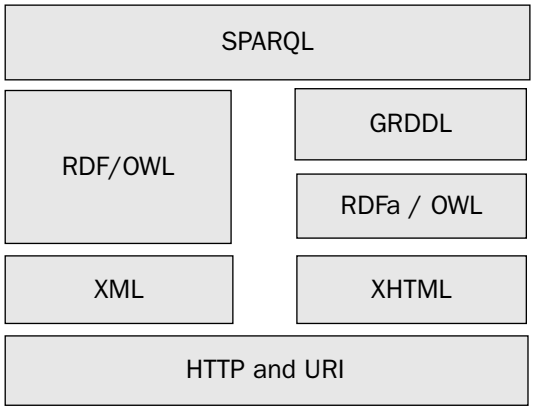

Figure 2.1 The semantic web technology stack

base layer. These are web fundamentals and every proposed Semantic Web technology is based on them.

Tools and applications

Alex Iskold's primer, "The Structured Web,"[8] explains two approaches to semantic applications (Figure 2.2):

1. *Bottom up*: involves direct embedding of semantic annotations (metadata) into the data
2. *Top down*: relies on analysis of existing information; the solution would be a natural language processor with the ability to understand text in the natural manner.

Freebase[9]

Freebase is a large collaborative database of knowledge comprised of metadata composed by community members. This data is harvested from many sources using an API (application programming interface) and aims to create a

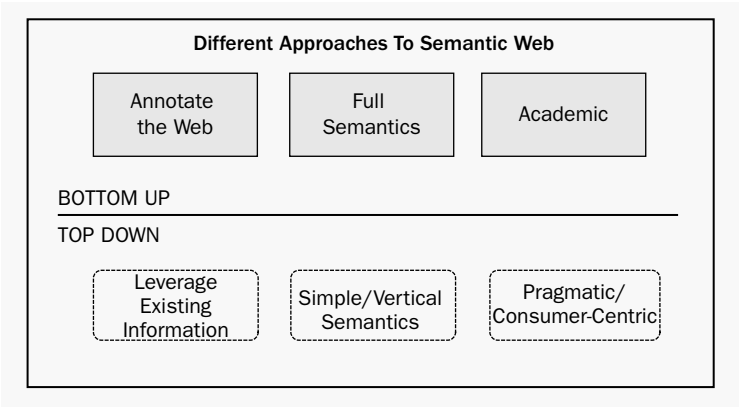

Figure 2.2 Different approaches to the Semantic Web

global resource that allows people (and machines) to access common information. The pages appear similar to a Wikipedia page and the app can make content suggestions when new data is entered. The topics are organized by type and the pages can be connected with links and semantic tagging.

Powerset[10]

This could be called a search engine with natural language abilities. It is based on semantic technologies and can make meaningful connections to help in the development of a semantic database. Both meaning and knowledge are extracted automatically (Figure 2.3).

Twine[11]

Twine is the first mainstream Semantic Web application. It learns automatically about the user and his/her interests as it is populated with content. Whenever new data is entered, the tool picks it out and tags the content with semantic tags,

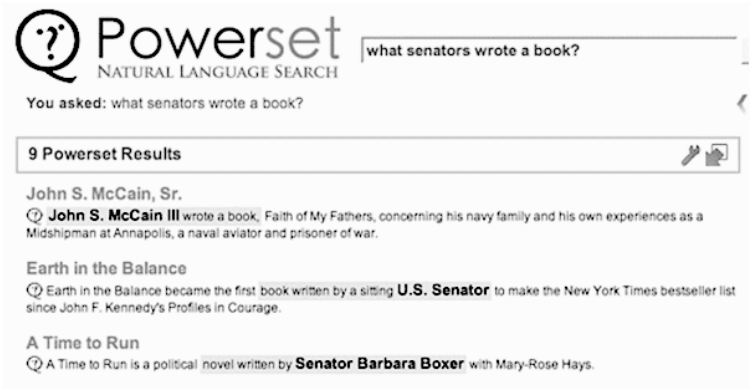

Figure 2.3 Powerset results

e.g. a person's name, thus creating new, semantically rich data that is not entirely user generated. There is also a machine-learning setup against Wikipedia to "learn" about new concepts. This demonstrates implementation of the bottom-up approach, where the user decides the crawling strategy.

AdaptiveBlue[12]

AdaptiveBlue is a product from SmartLinks which helps web publishers to add meaningful (semantically adapted) links to their sites. These come as pop-ups (in-page overlays) and provide additional contextual information pertaining to a particular link. In other words, the input is unstructured information which is turned into structured form by taking the basic item and adding semantics.

Hakia[13]

This uses natural language processing methods to provide meaningful search results. Its approach of using sentence analysis to analyze the concept of a search query differentiates it from other major search engines, which analyze keywords. Post-editing/human interaction is essential, but the rest of the analysis is computer powered by two technologies: QDEX Infrastructure (query detection and extraction technology) analyzes search queries at sentence level, and Semantic Rank Algorithm uses ontological semantics for related concepts.

Talis[14]

Talis is a platform which could be termed a hybrid of Web 2.0 and the Semantic Web. It helps developers to create apps for sharing, remixing, and reuse of data.

TrueKnowledge[15]

This is made up of multiple components, including natural language analysis, a knowledge base, and external databases. The external databases help to find immediate answers to frequently asked questions by pointing to a web page, offering a specific answer, and explaining the logic by which the answer was arrived at.

TripIt[16]

TripIt is an interesting app which manages travel planning by means of "itinerator" technology (patent pending). It allows useful information to be extracted from e-mails so as to make a well-structured and meaningful presentation of a travel plan.

Gnosis[17]

Gnosis is a Firefox extension which helps users to identify the people, companies, organizations, geographies, and products on a web page. Any web page, when viewed via Gnosis, is filled with various types of annotations; each meaningful word that Gnosis recognizes is colored according to the annotation type.

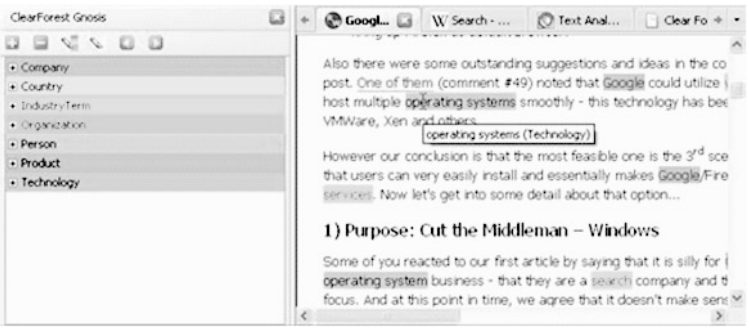

Figure 2.4 Gnosis search interface

Zabasearch[18]

This is what is termed a people search engine. In Zabasearch, queries have a person-centric perspective and the attributes used, such as geography and date of birth, mostly characterize people. Tags are used to create links or relationships between people and a user feedback loop helps in self-correction of the data.

Semantic Web-enabled knowledge management

Knowledge management is facing many problems, including information overload, inefficient keyword searching, heterogeneous information integration, and geographically distributed intranet problems, to name but a few. Semantic Web technologies can be applied to solving these problems.

On-To-Knowledge[2]

The On-To-Knowledge[2] (OTK) project[19] is a significant example that proposes content-driven knowledge-management solutions via evolving ontologies. It supports knowledge management by providing a tool environment powered by Semantic Web technology. Its main emphasis is on acquiring, maintaining, and providing access to information sources which are weakly structured.

- *Acquiring*: Text mining and extraction techniques help in the extraction of semantic information from textual information. Supported tools also include ontology extraction from text (OntoExtract and OntoWrapper).
- *Maintaining*: RDF, XML, and OIL (Ontology Inference Language) are used to describe the syntax and semantics

of semi-structured information sources; tool support includes ontology editor (OntoEdit) and ontology storage and retrieval (Sesame), so as to enable automatic maintenance and view definitions of knowledge.

- *Accessing*: Push-services and agent technology help users to access information. Tool support includes ontology-based information navigation and querying (RDFFerret), and ontology-based visualization of information (Spectacle).

RDF Vocabulary Description Language Schema (RDFS)[20]

RDFS helps to create a basic language framework by adding classes, subclasses, and properties to resources; for example, the resource Mars could be a subclass of the class *planet*, and its property could be red.

Simple Knowledge Organization System (SKOS)[21]

SKOS classifies resources in terms of broader or narrower. This allows the designation of preferred and alternate labels, thus letting people post/publish thesauri and glossaries online.

Neurocommons[22]

This is an open RDF database compiled from major life sciences databases by Science Commons. Its main focus is neuroscience and it is accessible via a web-based front end which uses SPARQL query language.

FOAF (Friend of a Friend)[23]

FOAF is a popular application of the Semantic Web used to describe relationships among people using the terms of RDF.

SIOC (Semantically-Interlinked Online Communities)[24]

SIOC is a vocabulary of terms and relationships that are available for modeling web data spaces such as discussion forums, RSS feeds, weblogs, mailing lists, image galleries, and so on.

SIMILE[25]

This is a joint project by the MIT Libraries and MIT Computer Science and Artificial Intelligence Laboratory. It seeks to enhance interoperability among digital assets, relevant vocabularies/ontologies, metadata, and useful services.

Linking Open Data[26]

Linking Open Data is a community-led effort for the creation of RDF data on the web which will be openly accessible and interlinked. These RDF datasets are drawn from a collection of data sources which focus on the linked data style of RDF publishing.

Semantic Web Ping Service

The Semantic Web Ping Service is no longer available. It was a notification service which tracked the creation and modification of RDF-based data sources, and was provided for loosely coupled monitoring of RDF data. It also provided a breakdown of RDF data sources. The tracking was done by SIOC, FOAF, DOAP (Description of a Project), RDFS and OWL vocabulary.

Piggy Bank[27]

Piggy Bank is a plug-in. Its mechanism is the extraction or translation of web scripts into RDF information which is stored on the user's computer and retrieved separately from the original context. It can also be used in other contexts. The Semantic Bank service, developed by the Simile Project, combines the tagging of information with the concept of new web languages.

Triplify[28]

Triplify is another plug-in which works for database-driven web applications. It exposes RDF, Linked Data, and JSON (JavaScript Object Notation), can be configured very simply, and is supported by an RDF data source registry. The Triplify configuration can be reused without major modifications because pages on the World Wide Web are served by a relatively small number of applications. This enables easy "semantification" of web applications.

Webpages Database[29]

This is an application which helps users to create an RDF database in the browser itself.

Conclusion

In light of the growth pattern of the Semantic Web, it is likely that we will soon be able to build applications which will explore, integrate, and exploit large amounts of heterogeneous semantic data generated from a variety of distributed sources. Recent research in this regard has focused on the use of ontologies in the e-science community. The aim is to structure data consistently so as to add value

and allow its multiple use. Granular data can be sourced from multiple online resources, combined, and then displayed in a unified manner, depending on the context and need, thus also adding value to the compilation process.

This could be the much sought-after panacea for the difficulty of finding, accessing, presenting, and maintaining the information required by a wide variety of users. Thus the ultimate dream of intelligent access to heterogeneous, distributed information could be realized, enabling software products (agents) to mediate between user needs and the available information sources.

References

1. Worldwidewebsize.com, "The size of the World Wide Web (the Internet)." Available at: *http://www.worldwidewebsize.com/* [Accessed 2 September 2013].
2. Berners-Lee, T., Hendler, J., and Lassila, O. (2001) "The Semantic Web: A New Form of Web Content that is Meaningful to Computers will Unleash a Revolution of New Possibilities," *Scientific American* (17 May).
3. Berners-Lee, T., with Fischetti, M. (1999) *Weaving the Web* (San Francisco: Harper), Ch. 12.
4. Cardoso, J. (2007) "The Semantic Web Vision: Where are We?" *IEEE Intelligent Systems* 22(5): 84–88.
5. W3C (2008) "SPARQL Query Language for RDF." Available at: *www.w3.org/TR/rdf-sparql-query* [Accessed 2 September 2013].
6. Allen, J. (2001) "Making a Semantic Web." Available at: *http://www.netcrucible.com/semantic.html* [Accessed 2 September 2013].
7. Beal, V. "What is the Semantic Web?" Available at: *http://www.webopedia.com/DidYouKnow/Internet/2007/Semantic_Web.asp* [Accessed 22 September 2013].
8. Iskold, A. (2007) "The Structured Web – A Primer." Available at: *http://www.readwriteweb.com/archives/structured_web_primer.php* [Accessed 2 September 2013].

9. Freebase: An Open, Shared Database of the World's Knowledge, *http://www.freebase.com/*.
10. Powerset has now been repackaged as Bing, *http://www.bing.com*.
11. Twine was repackaged as Evri. Evri is no longer available (see "Paul Allen-backed News Aggregator Evri Calls It Quits," October 5, 2012. Available at: *http://www.geekwire.com/2012/paul-allen-backed-evri-calls-it-quits/* [Accessed 22 September 2013]).
12. AdaptiveBlue has now been repackaged as GetGlue, *http://www.getglue.com*.
13. Hakia search engine beta, *http//www.hakia.com*.
14. *http://www.talis.com*.
15. TrueKnowledge has now been repackaged as Evi.com, *http://www.evi.com*.
16. TripIt, *http://www.tripit.com*.
17. Wikipedia, "Gnosis," *http://en.wikipedia.org/wiki/Gnosis*.
18. Zabasearch, *http://www.zabasearch.com*.
19. On-To-Knowledge, *http://www.ontotext.com/research/otk*.
20. W3C (2004) "RDF Vocabulary Description Language 1.0: RDF Schema." Available at: *www.w3.org/TR/rdf-schema/* [Accessed 2 September 2013].
21. SKOS Simple Knowledge Organization System, *www.w3.org/2004/02/skos/*.
22. Science Commons, The Neurocommons. Available at: *http://sciencecommons.org/projects/data/* [Accessed 2 September 2013].
23. The Friend of a Friend (FOAF) project, *www.foaf-project.org/*.
24. SIOC-project.org (Semantically-Interlinked Online Communities), *http://sioc-project.org/*.
25. Simile Project, *simile.mit.edu/*.
26. Linking open data. W3C Semantic Web Education and Outreach Community Project. *http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData*.
27. Piggy Bank, *http://simile.mit.edu/piggy-bank/*.
28. Triplify.org, *http://triplify.org*.
29. Webpages, *http://www.webpages.com/database/createtable.htm*.

3

Knowledge organization: its role within the unfathomed space

Abstract: Knowledge organization for the sake of knowledge organization is a myth; the basic units of knowledge organization are semantic relations between concepts. To the librarian and information professional, knowledge organization has always been, and will continue to be, a specific purpose-oriented activity. The purpose of organizing the documents/information or their surrogates in a collection has always been to assist users in browsing, to formulate search expressions, and to present information services and products to the target user. There are different approaches which are not explicit theories but which represent different views of knowledge, language, understanding, and social organization. A comprehensive understanding of the fundamental and time-tested principles of knowledge organization will help in developing new tools and technologies for the Semantic Web, and for ontologies in particular.

This chapter presents findings from a literature review to gauge the state of the art in the areas of ontology, the Semantic Web, digital libraries and web ontology techniques. The advantages and limitations of a range of tools and technologies are also identified and presented. The chapter reviews web ontology tools and technologies specifically from a digital library perspective.

Keywords: Knowledge organization, web ontology, ontology languages, DAML, OIL.

Background

Knowledge organization (KO) entails both systems and processes. Bibliographic records, Dewey Decimal Classification (DDC)/Library of Congress Classification (LCC)/Universal Decimal Classification (UDC) systems, thesauri, semantic networks etc. can be cited as systems, while classification, document description, indexing and subject analysis can be termed as KO processes. These processes are used in information intermediary institutions, namely libraries, archives, museums, online databases, back-of-the-book indexing, personal information management systems etc. The scope of KO as a concept can be universal (covering all fields of knowledge) or specific to certain domains or document types. KO for its own sake is a myth. To the librarian and information professional, it has always been and will always be a purpose-oriented activity. The purpose of organizing the documents/information or their surrogates in a collection has always been to assist users in browsing and in formulating search expressions and to present information services and products to the target user.

The literature review presented in this chapter looks at developments and significant work in the area of the Semantic Web, ontologies and their application in the development of conceptual frameworks from the digital library perspective. There are different approaches which are not explicit theories, but which represent different views of knowledge, language, understanding and social organization. In the current landscape of global knowledge representation, the web is being realigned as a meaningful Semantic Web for modeling, representation and knowledge exchange.

It is strikingly clear that the Semantic Web as such is nothing but a vision for extending the future of the web, wherein information is given explicit meaning. Pursuit of the realization of this concept has led to the development of machines for information processing and integration. The key strengths of XML are its ability to define custom-made tagging schemes and the flexible approach of RDF for data representation.

Ontologies are emerging as a key solution to the problem of information exchange across different applications, especially in a cooperative work environment. The very growth of the Internet and the World Wide Web has been guided by the aspiration for information and document exchange between people. Although the concept has been very successful, it cannot be applied to facilitate information exchange between software applications. Ontology-based approaches and information management systems have been developed to create common and shareable knowledge repositories, but only in a given application domain.

Ontology has a long history of development as a concept and practice both in philosophy and in computer science. Research efforts gathered pace from 1990 to explore the possibility of its use in knowledge representation (KR) and to give an impetus to AI. This led to the development of various frame-based KR languages like HTML, SHOE (Simple HTML Ontology Extensions), XML, XOL (Ontology Exchange Language), OIL and different knowledge acquisition approaches. Ontologies represent the terms that can be used for the description and representation of an area of knowledge. This makes them critical for those applications that entail the searching or merging of information from diverse communities. XML Document Type Definitions and XML Schemas attempt to exchange data between parties, but they don't produce the desired output because they lack semantics.

Knowledge organization and digital context

In libraries, KO includes such activities as classification, indexing and document description, where the basic associated factors can be listed as:[1]

- information retrieval function: facilitating searches in resources such as catalogues and bibliographies
- document information function: provision of information to users about documents of importance (e.g. in the form of abstracts and notes), as well as the procedure for obtaining information
- ordering function: arrangement on the shelf, or other types of linear ordering.

Empirical research: conventional KO

DDC, LCC, UDC and other similar systems are traditional approaches to KO. They are more or less based on scientific disciplines and classify knowledge according to a scientific worldview. Their basis is an intellectual aspiration to reflect a true order of reality. However, other than a few principles, they have not established a specific methodology for bibliographic classification. Classification is implicitly treated as a "neutral reading" of the true semantic relations.[1]

Semantic research: digital KO

Some approaches to classification are characterized by the use of logic rather than of empirical research. Facet analysis is the most influential approach in Library and

Information Science (LIS) that is primarily based on logic. W. C. Berwick Sayers was able to structure the methodology of library classification in terms of canons and axioms which may be viewed as belonging to this approach.[2]

Empirical research methods are not suitable in the digital context because of the volume of information/knowledge to be organized. The World Wide Web has itself evolved to become one gigantic knowledge resource.[3] In this regard, when the idea of making the web meaningful was proposed, the obvious means could have been the tools and principles used by librarians to organize the vast repository of information in a library.

Tim Berners-Lee[4] has proposed his vision of the Semantic Web, a place where all information and knowledge will be meaningfully organized. The model will be developed on the basis of a graduated tower of semantic languages.

As the concept of the Semantic Web has grown, various representation languages have come into being and later evolved into standards. A vast majority of ontology languages have been developed on top of the RDFS, in accordance with the W3C recommendation to use RDF languages as the standard for specifying metadata on the web. Later, the W3C proposed OWL (latest version, 2008) as an extension of XML, RDF and RDFS and as a new standard for ontology development.[5]

Significant contributions in the mega-project of the Semantic Web have been development of KRs in the form of digital libraries. Strategies can be devised in digital libraries for the preservation of digital content by addressing three major issues of digital collections, namely emulation, migration and the normalization of data.[6]

Digital libraries preserve the metadata in either an extended METS (Metadata and Encoding and Transmission

Standard) schema[7] or an MPEG-21 Digital Item Declaration Language schema.[8] Generally, composite mixed-media objects are stored in an XML-based structural markup format such as HTML, SMIL (SMIL, 2001), XHTML+SMIL[9] or HTML+TIME,[10] rather than proprietary formats such as Shockwave, Macromedia Director or Flash.

Knowledge organization: major landmarks

KO per se has been defined by scholars in the following different ways:

Kiel: The term "knowledge organization" literally means making knowledge an "organum" (Greek = instrument, aid) for particular purposes.[11]

Dahlberg: "Knowledge Organization … is the science of structuring and systematically arranging knowledge units (concepts) according to their inherent knowledge elements (characteristics) and the application of concepts and classes of concepts ordered by this way for the assignment of the worthwhile contents of referents (objects/subjects) of all kinds."[12]

Anderson: "The description of documents, their contents, features and purposes, and the organization of these descriptions so as to make these documents and their parts accessible to persons seeking them or the messages that they contain. Knowledge organization encompasses every type and method of indexing, abstracting, cataloguing, classification, records management, bibliography and the creation of textual or bibliographic databases for information retrieval."[13]

Conventional methods

Classification systems have been part of the discipline of LIS for a long time and the term "knowledge organization" is very closely related to library classification. Around 1900, many scholars emphasized the idea that the classification of books in libraries is based on the organization of knowledge. These included Charles A. Cutter (1837–1903), Ernest Cushing Richardson (1860–1939), Henry Bliss (1870–1955) and William Charles Berwick Sayers (1881–1960). The term "knowledge organization" is said to be related to their work, according to which "book classification" is basically knowledge organization.

Cutter wrote:[14] "I believe that the maker of a scheme for book arrangement is the most likely to produce a work of permanent value, if he keeps always before his mind a classification of knowledge."

Sayers expressed the opinion that:[15] "A book classification must hold the minuteness of the knowledge classification as an ideal to which it must approximate as nearly as possible"; and further: "It must be clearly borne in mind, however, that the classification of knowledge should be the basis of the classification of books; that the latter obeys in general the same laws, follows the same sequence."

The process of classification is the assigning of elements or units to classes according to some criteria. The criterion of classification is likeness, whereby like things are united and unlike things become separated. Things can appear alike in many different ways, including functionality or practicality. Classification involves attributes, to which values are assigned. An attribute could be weight or color; values might be 85 kilograms or red, respectively. Thus, the way in which we classify things reflects conceptions which are embedded in our language and mental conceptual structures. In

libraries, documents are classified so as to facilitate information retrieval by browsing or other means. Thus, the criteria of classification and knowledge organization are deeply associated with relevance. It is the pragmatic criteria, not the inherent qualities, of the objects which become the sole determiners of classification.

The evolution of library classification schemes is traced in Table 3.1.

Library of Congress Classification

Library of Congress Classification was developed by Herbert Putnam in 1897, just before he assumed the librarianship of Congress. He took advice from Charles Ammi Cutter, and the system, which was designed to meet the specific purposes of the Library of Congress, was influenced by Cutter Expansive Classification and the DDC.

Table 3.1 **Evolution of library classification schemes**

Mid-4th century BC	Aristotle proposed his outline of the theory of categories and differentiated between species and genus, species being the specification of the genus (*per genus proximum et differentiam specificam*). He thus established the science of classification and taxonomy.
331 BC	Alexander the Great established the Library of Alexandria, which is said to have had alphabetical catalogues.
1929	Henry Evelyn Bliss's *The Organization of Knowledge and the System of the Sciences* is published.
1933	Ranganathan's *Colon Classification* (1st edition) is published.
1937	S. R. Ranganthan's *Prolegomena to Library Classification* is published.
1940–53	Henry Evelyn Bliss's *A Bibliographic Classification*.

Dewey Decimal Classification

Melvil Dewey developed this system of library classification in 1876. It has since been modified and expanded through many revisions.

Universal Decimal Classification

UDC was developed by Paul Otlet and Henri la Fontaine at the end of the nineteenth century. It makes use of auxiliary signs to indicate the special aspects of a subject and interrelationships. The element of faceted or analytico-synthetic organization can be clearly seen in it.

Colon Classification

Dr. S. R. Ranganathan developed this first-ever faceted (or analytico-synthetic) classification. Its name derives from the fact that it uses colons to separate facets in class numbers.

Thesauri

Thesauri are tools that are used for information retrieval, query expansion and indexing. The basic vocabulary in a domain is selected, to which is added information about homonyms, synonyms, part-to-whole terms, generic terms, "associative terms" etc.

Transition tools and technologies

The significant advancement in the twenty-first century has been in the development of transitional technologies, resulting in attempts at automatic classification and AI as a

means of retrieval. Interoperability has been the key guiding principle, though interpreted in varying ways by different people, to support the ability of systems to talk to one another or to switch from one subject retrieval system to another, etc. This emphasis on interoperability has led to an increased interest in the mapping of one information tool/technology onto another.

The search for a universally accepted information language seems to be a thing of the past, and the impetus of research has shifted to mapping one information language to another. Examples that can be cited include mapping between LCSH (Library of Congress Subject Headings) and DDC; efforts to integrate three Chinese thesauri using an automated vocabulary-switching system, with greatly improved search results, etc. Another form of interoperability is the development or adaptation of retrieval systems beyond their original purpose of indexing written materials. The timeline of the development of information retrieval (IR) systems is traced in Table 3.2.

Indexing

Indexing is the process of tracing an "information object" or a pointer in a record or in an index for the purposes of retrieval. The popular forms of indexes can be found in library catalogs, bibliographical databases etc. Wellisch[16] has explained the term "index," its history and meanings, but did not discuss the application of indexes in LIS and their significant role in information retrieval.

Typically, "indexing" is used to describe the process of assigning keywords/descriptors to pieces of information in bibliographical or other records. To the layperson this may be different from classification (the process of assigning a classification code to a bibliographical record), but in

Table 3.2 **Development of information retrieval systems**

1945	Vannevar Bush envisages the Memex, a device for creating links between related topics in different research papers. This is said to be the first envisioning of hypertext/hypermedia.
1951	"Information retrieval" founded by Calvin Mooers.
1952	Establishment of the Classification Research Group (CRG).
1957–60	Cranfield experiments.
1962	Establishment of the Documentation Research and Training Center, a division of the Indian Statistical Institute.
1963	The beginning of online information services.
1963	The Science Citation Index is started by Eugene Garfield.
1967	Publication of *Anglo American Cataloging Rules*, 1st edition (AACR1).
1989	Foundation of the International Society for Knowledge Organization (ISKO).
1990	Beginning of full text databases and the Internet.
1998	Founding of Google by Larry Page and Sergey Brin.

principle the two are not very different, a good example of the similarity being a thesaurus based on faceted classifications. The thing that makes the real difference in KO is the use of either controlled or non-controlled vocabularies.

The indexer generally analyzes different parts of the document, viz. title, abstract, summary, conclusion, etc., and the quality of indexing depends on this analysis. Different techniques may be used, namely human/intellectual analysis, computer-based statistical analyses etc. The popular subject indexing process includes subject analysis, followed by the extrapolation of the subjects according to the specific system being used; the indexing terms can be derived from the text of the document or can be assigned by the indexer.

Book indexing[17]

This is also commonly termed "BoB" or back-of-book indexing, in which indexes are constructed which are placed near the end of the book. This information tool, in which access points are by specific subject, plays a role that is complementary to the table of contents. Stauber[18] remarks that while "at first glance [it is] the driest part of the book, on closer inspection the index may provide both interest and amusement from time to time."

Facet analysis

Facet analysis is probably the most distinctive approach in the history of KO. It is more methodical in comparison to traditional approaches and is more generic when compared to computer-based approaches. The name most associated with this approach is that of S. R. Ranganathan, who influenced it greatly. Important dates in the history of its development are the publication of the first edition of *Colon Classification* in 1933 and the establishment of the British CRG in 1952.

Faceted classification is also known as analytico-synthetic classification. The name is derived from the two main processes involved in the composition of a call number: analysis involves breaking down each subject into its basic concepts, and synthesis involves the combining of the relevant units and concepts for better description of the subject matter.

Controlled vocabulary

Svenonius[19] defines controlled vocabulary as follows: "vocabulary … is said to be controlled if it consists of a restricted subset of possible terms. Such a subset, in that it

contains only those terms 'authorized' for use, is sometimes called an authority list. In addition to terminological restriction, most Controlled Vocabularies articulate semantic relationships between terms in the vocabulary, the most common of these being the inclusion of hierarchical relationship."

A controlled vocabulary is a conjoint term to represent those terms or symbols that are used in indexing. Those terms which are not part of the standard list are known as free text terms, natural language terms, keywords etc. A controlled vocabulary helps to reduce the variability of expressions used to characterize the documents which are being indexed, e.g. not using synonyms and working to remove ambiguity (homonyms). Various subject headings and thesauri are classic examples of controlled vocabularies.

Information retrieval

The two disciplines of IR and KO, while apparently different, are actually strongly related sub-disciplines within LIS. Their strengths and weaknesses can be evaluated only when we consider them together and then evaluate their corresponding strengths and weaknesses.

IR takes raw ideas and makes them part of the standards approach, for example, the concept of genre and genre-analysis. The fundamental difference between Cranfield experiments[20] and user-oriented views is that in the former recall and precision are evaluated, while the latter relies on the user's evaluation.

Search engines

Currently available search engines tend to be successful when an information request is reasonably simple and

Table 3.3 Fine-tuning search engine technology – some examples

Swoogle[21]	A customized directory and search engine that searches for all RDF resources available on the web, including ontologies.
OntoSelect Ontology Library	This offers the same kind of services for DAML, RDF/ RDFS and OWL ontologies.
Ontaria[22]	A directory of semantic web data that can be searched as well as browsed through. It focuses on RDF vocabularies with OWL ontologies.

everyday vocabulary is used. As ever, the difficulties begin when one tries to make a universally applicable classification tool. When specialized queries or technical vocabulary are used, the search engine begins to fail and either produces hundreds or thousands of hits, many of them totally irrelevant, or else produces nothing. The development and advanced application of search engine technology represent the final and concluding attempt by computer science professionals to classify knowledge. This statement can be validated by the fact that almost all subsequent research has been in the area of fine-tuning search engine technology. Some landmark examples are listed in Table 3.3.

Landmarks in semantic knowledge organization

As noted above, the other logical approach to KO is the semantic approach; many applications and technologies have been developed around this concept.

Fuzzy logic

The major task that has occupied researchers for quite some time has been the improvement of ways in which to access the World Wide Web. Work has ranged in the areas either of using "fuzzy logic" for the design of linguistic information retrieval systems so as to improve the cognitive aspects of the retrieval activity, or of applying it for retrieval purposes. It was realized that many of the techniques being employed were fundamental to KO in libraries and so a dialogue was initiated between information specialists and computer scientists. This led to the development of the techniques discussed below, namely automatic indexing, weighting and correlation of terms, etc.

Automatic indexing

Natural language processing was used for automatic indexing, where special algorithms were deployed to decide which terms within a text were important. However, the major shortcomings were the cost involved and the monolingual character of the technique.

Automatic classification

Automatic classification transcends language barriers, but is very sophisticated and expensive to incorporate into search engines. However, it has had considerable success when applied as part of a search engine. One of the most successful attempts at automatic classification is GERHARD (German Harvest Automated Retrieval and Directory), which is particularly noteworthy, being based on German rather than English-language web resources. The GERHARD system is

a fully automatic indexing and classification system for the German-language web for integrated browsing and searching, and has the added enhancement of a trilingual (German, English and French) interface.

Ontology

Closely linked to automatic classification is the use of classification structures developed in expert systems, which are nowadays viewed as ways of organizing knowledge and indexing and are known as ontologies. Ontology can be used as a retrieval device, just like the traditional classification, but its use is confined to the retrieval of digital materials.

Ontologies resemble classifications in that they define concepts and relationships in a systematic manner, but they are superior in their ability to specify the semantics and relationships explicitly and to express them in a manner that is understood by the computer. Traditional tools such as DDC or a hybrid of a bibliographical classification and a thesaurus can be used as the basis of ontology. But, as ever, as soon as the human element is removed the system has to depend on matching, and it can match only what it has been programmed to match. The success of such an enterprise is generally greater in a specific subject field, e.g. medicine, rather than across the whole of knowledge. This is where the people who are interested in AI become involved, because they are interested in building systems that will perform the task of classification and information retrieval.

The conceptualization of ontologies to wrap meaning/context around information bits/knowledge resources was a major breakthrough. Thomas Gruber has defined ontology as "a specification of a representational vocabulary for a shared domain of discourse definitions of classes, relations, functions, and other objects."[23]

The origin of ontology lies in philosophy and it can be traced back to Aristotle, who defined it as a description of the world as it exists. Its evolution can be traced in the works of Husserl, Kant, Frege and Carnap, among others. As a branch of philosophy, ontology stands for the study of basic existence, the kinds and structures of objects, their properties, associated events, processes and relations in all possible ways. Philosophers often use the term as a synonym for "metaphysics." The term "ontology" (or *ontologia*) was coined in 1613, independently, by two philosophers: by Rudolf Göckel, in his *Lexicon philosophicum* and by Jacob Lorhard, in his *Theatrum philosophicum*. The first occurrence of the term in English was in Bailey's dictionary of 1721, which defined ontology as "an Account of being in the Abstract."[24]

The tools of philosophical ontology have been applied to solving practical problems, e.g. concerning the nature of intellectual property. Later, ontology was also applied in logic. The discipline of information systems has borrowed the term from philosophy and reinterpreted its meaning to suit the information sciences. AI groups worked on ontology in the 1980s and computer science took over towards the end of the twentieth century. Ontology has a totally different role in information systems from that which it has in philosophy. In the commonly available literature, ontology in the context of philosophy is written with in an upper-case "O," and in the context of information science with a lower-case initial letter.[25]

In the information sciences ontologies are used for the visualization of topic hierarchies and their interrelationships. The ontology is displayed as a tree, with the keyword as the root node and the child concepts as leaf nodes, or nodes if they also have sub-concepts. The user can collapse or expand the tree by clicking on it and there are many ontology visualization tools available for this purpose.

Ontology languages

Ontology languages are formal languages used for the construction of ontologies. They support the encoding of information pertaining to specific domains and include reasoning rules to support information processing. The languages used to represent ontologies mainly belong to three categories: logic based (first-order logic), frame based (frame logic) and web based. Logic-based languages are specific to AI applications, two examples being Knowledge Interchange Format (KIF) and Knowledge Language One (KL-One).[26] KL-One supports frame data structures that have network notation and well-defined semantics.

Web-based ontology languages

DARPA Agent Markup Language (DAML)[27]

DAML is specifically designed as a language and tool to support the concept of the Semantic Web. DAML typically represents the standard that uses a web-compatible language and a reasoning aspect.

Ontology Inference Layer (OIL)[28]

The OIL knowledge interchange language is an extension of RDF. It was developed in the European Information Society Technologies (IST) project, On-To-Knowledge, and is both a representation and an exchange language for ontologies. Primitives from frame-based languages, formal semantics, and reasoning services from DL unify the language. The ontology description is divided into three layers: object level (concrete instances), first meta-level (ontological definitions) and second meta-level (relations), as well as a limited set of axioms.

DAML+OIL[29]

This combines components of a web language, elements of DL and the methodology of frame reasoning systems according to the OIL. This provides a rich set of constructs for the creation of ontologies and the marking up of information, thus making it machine readable and understandable. It is compatible with the RDFS and also has semantics for describing the meanings of terms.

Resource Description Framework (RDF)[30]

This is a standard XML framework to facilitate the description and interchange of metadata. RDF, with its simple format of resources, properties and statements is able to define robust metadata, e.g. ontological structures etc. RDF is more decentralized (quite unlike Topic Maps) because here the XML is stored along with the information resources. The original broad goal of RDF was to define a mechanism for describing resources that neither makes assumptions about a particular application domain nor defines (a priori) the semantics of any application domain. RDFS defines the terms that will be used in RDF statements and gives specific meanings to them.

Ontology Web Language (OWL)[31]

The OWL semantic mark-up language has been derived from DAML+OIL and helps in the creation and exchange of web ontology. It contains three 'sub-languages' characterized by their corresponding levels of complexity: OWL Lite, OWL DL and OWL Full.

Simple HTML Ontology Extension (SHOE)[32]

SHOE is an HTML-based knowledge representation language and provides various categories, namely attributes, relationships, inferences drawn, etc., which can be easily defined by ontologies. The semantics and abilities provided by SHOE are relatively rich and enable the embedding of content-specific information into documents.

Other Semantic Web applications

Topic Map (TM)[33]

TM is an ISO standard for describing knowledge structures and mapping them to relevant information resources. It was created for use as an ontology framework for information retrieval. TMs are characterized by the presence of a rich semantic model; this will support information retrieval in general, along with an unlimited range of other applications.

Semantic annotation[34]

Semantic annotation provides information about what entities (or, more generally, semantic features) appear in a text and where. Formally, semantic annotations are a specific sort of metadata which provides references to entities in the form of URIs or other types of unique identifiers. Several annotation tools have been developed to be used in the Semantic Web and provide users with the ability to annotate documents for various purposes.

Neurocommons[35]

Neurocommons is an open RDF database which was developed by Science Commons for compiling major life

sciences databases with a focus on neuroscience. It can be accessed via a web-based front end by using the SPARQL query language.

Friend of a Friend (FOAF)[36]

FOAF is a popular Semantic Web application for the description of relationships among people and other agents in RDF terms.

SIOC[37]

This project provides a vocabulary of terms along with relationships for modeling web data spaces. Examples include discussion forums, weblogs, blogrolls/feed subscriptions, mailing lists, shared bookmarks, image galleries etc.

SIMILE[38]

SIMILE was a joint project of W3C, Massachusetts Institute of Technology Libraries and MIT Computer Science and Artificial Intelligence Laboratory which sought to enhance interoperability among digital assets, schemata/vocabularies/ ontologies, metadata and services.

Linking Open Data

Linking Open Data is a community-led effort to create openly accessible and interlinked RDF data on the web. The RDF data sets are drawn from a broad collection of data sources where the focus is on the Linked Data style for publishing RDF on the web.

Visual presentation

Another aspect of retrieval which has received some attention in recent years is the way in which information is presented on the screen. The need to build upon the experience and exploit the skills of traditional librarianship in facilitating browsing in digital collections, rather than just the location of specific items, has also been emphasized.

Limitations of existing approaches of knowledge organization

Many studies have pointed out that the existing tools and techniques of ontology development are based on the concepts of AI knowledge representation, although they have not been used much in other interdisciplinary research work.[39]

Despite all of these studies, there was a definite gap which indicated the need for industry standards to enable distributed global knowledge representation. The challenge was to support the construction of user queries using domain ontologies that might be initially unfamiliar to the user, and to allow queries to span multiple information sources by representing and computing the mappings between domain ontologies and the ontologies supported by individual information sources.

Computer science approaches

Since the 1950s computers have been important on many levels, including in genuine algorithmic-based approaches to KO. Other approaches to KO are semi-automatic or machine aided, such as the technique "text categorization," which is

based on predetermined categories. Computers have also influenced KO, for example by providing citation databases which allow new forms of analysis and KO.[40]

User-oriented and cognitive approaches

User-oriented approaches were mainly based on information about or from users, for example, examination of log files, interviews and so on. The most important contribution of user-oriented approaches has been to stress that users' needs should be the focus of all LIS services and products. Libraries are not an end in themselves and should not reflect patronizing attitudes.

Domain-analytic approaches

The domain-analytic view implies that if LIS professionals learn about domains, especially about the paradigms and sociological issues of domains, then better classifications can be constructed and documents can be better indexed. Different domains may require different descriptions and classifications of objects in order to serve their specific purpose within their appropriate social context, the criteria of classification thus being domain specific.

- The cognitive sciences and psychology interpret knowledge representation and organization in the same way as in the human brain. Thus, the basic organizational unit of knowledge becomes the concept.
- Computer scientists study to determine the manner in which the knowledge may be represented and organized in computers.

- Library science focuses on KO at all locations, namely the library shelves, catalogs, databases etc. of public libraries, research libraries and national libraries. A broader view might include other locations.
- The purpose of documentation is to organize knowledge into primary, secondary and tertiary subject literatures, their genres and types of documents.

The majority of studies in domains take place in the academic environment and the majority of the literature is produced by university teachers and their research assistants. Large organizations such as OCLC (Online Computer Library Center), Elsevier or the Wilson Company have the manpower and financial backing to maintain a well thought-through and consistent research program, which is not the case with universities.

A greater problem lies in the use of the term "project." Much of the research that is undertaken is short term. The maximum period for which funding is likely to be granted is generally around two years. Often a six-months' pilot project is the first step and this may not be followed up by further support. The result is a number of small projects, often devised to solve a particular and frequently domestic problem, even though they may have a wider significance in order to attract support. An integral condition of funding is that the findings must be published and, ideally, also be presented at conferences.

Tim Berners-Lee's vision of the Semantic Web

> … a plan for achieving a set of connected applications for data on the Web in such a way as to form a consistent logical web of data …

> … an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation …[5]

Ontologies present the world as we wish to view it – abstract and simple; this is described in a language equipped with a formal semantics. In knowledge representation, ontology describes concepts and relationships in an application domain; this must be understandable by humans and/or by software agents. In information systems and databases, ontologies serve as conceptual schema, providing a formal and non-ambiguous setup, both from the syntactic and the semantic point of view. This makes ontologies useful in many fields, such as intelligent information integration, cooperative information systems, information retrieval, digital libraries, e-commerce and knowledge management.

Ontologies evolved to become the foundational technology of the Semantic Web by annotating web documents with machine-interpretable information concerning their content. Research in the Semantic Web has led to the standardization of specific OWLs.

In the fields of computer science and AI, ontology languages formalize the construction of ontologies. They help in encoding the knowledge pertaining to specific domains and thus include reasoning rules to support the processing. Ontology languages can be defined as declarative languages that have the generalizations of frame languages, along with the first-order logic or description logic.

More precisely, ontology language is a means to specify the domain of interest at the conceptual level and expresses constraints by declaring what should necessarily hold in any possible concrete instantiation of the domain.

A standard ontology language serves the following purposes:

- it provides a common syntax so that users can share semantics
- it presents a syntactic web based on standards
- it provides a wide variety of platforms for "explicit specification"
- it manages a variety of Graphical Notations, namely Semantic Networks, Topic Maps, UML, RDF etc.
- it takes care of logic-based formalisms by using formats like DL, Rules, First Order Logic, Conceptual Graphs
- many languages use an "object oriented model" with the following elements:
  - Objects/Instances/Individuals – elements of the domain of discourse
  - Types/Classes/Concepts – sets of objects sharing certain characteristics
  - Relations/Properties/Roles – sets of pairs (tuples) of objects

Making the right choice

The following concepts/works/projects have been found to be of special significance.

Tom Gruber states that "An ontology is an explicit specification of a conceptualization."[41] He clarifies this further: "That is, ontology is a description (like a formal specification of a program) of the concepts and relationships that can exist for an agent or a community of agents. This

definition is consistent with the usage of ontology as set-of-concept-definitions, but more general. It stands for a set of representational elements for modeling a particular domain of knowledge or discourse. These are typical classes (or sets), attributes and relations among class members."[42] To begin with the computational agents have some conceptualization of the meaning of terms and at times the software programs provide a specification of the inputs and outputs.

Ontology modularization

Ontology modularization is the splitting of existing ontologies into reasonable modules which can be handled separately. Work has been done in different disciplines, namely ontology engineering aspects – focusing on the modeling of ontologies and their reuse when building an ontology; knowledge representation and reasoning aspects – concerned with building reasons to take advantage of the possibility of modularizing ontologies. This can be useful for improving aspects such as scalability and tractability.[43]

Ontology in informatics

The concept of ontology has been applied in the RDF, RDFS and OWL layers in the architecture of the Semantic Web.

Barry Smith defines ontology[44] in information systems as follows: "An ontology is a software (or formal language) artifact designed with a specific set of uses and computational environments in mind. An ontology is often something that is ordered by a specific client in a specific context and in relation to specific practical needs and resources."

Ontology in information science

Semantic relations are meaningful associations between two or more concepts, entities or sets of entities. The Semantic Web describes the relationships between things and their properties by using the idea of ontology.

Ontology is used to define the common terms and concepts that can be used to describe and represent an area of knowledge. An ontology can have a wide range, starting from a taxonomy (knowledge with minimal hierarchy) and extending to a thesaurus[45] (words, synonyms and some relationships). It can also represent a conceptual model with complex knowledge or a logical theory with very rich, complex, consistent and meaningful knowledge.

Borst and Top[46] have elaborated Gruber's definition: "Ontologies are defined as formal specification of a shared conceptualization." Studer, Benjamins and Fensel[47] have combined both Gruber's and Borst's definitions as: "Ontologies are explicit formal specification of a shared conceptualization," while Sure and Studer[48] have explained the term as follows:

- explicit: because the type of concepts used and associated constraints are explicitly defined
- formal: machine-readable with elements of natural language
- shared: here the knowledge captured is consensual and is not private to some individual but accepted by a group
- conceptualization: abstract modeling of a phenomenon by identifying the relevant concepts.

The following definition by John F. Sowa,[49] in particular, depicts ontologies as being useful for faceted knowledge representation. He explains: "The subject of *ontology* is the

study of the *categories* of things that exist or may exist in some domain. The product of such a study, called *an ontology*, is a catalog of the types of things that are assumed to exist in a domain of interest *D* from the perspective of a person who uses a language *L* for the purpose of talking about *D*. The types in the ontology represent the *predicates*, *word senses*, or *concept and relation types* of the language *L* when used to discuss topics in the domain *D*."

Noy and McGuinness[50] take a pragmatic approach: "An ontology is a formal explicit description of concepts in a domain of discourse – classes (sometimes called concepts); properties of each concept describing features and attributes of the concept – slots (sometimes called as roles and properties as well); and restrictions on slots – facets (sometimes also called as role restrictions)."

Ontology evaluation

Ontology evaluation is an important part of the engineering and selection of suitable ontologies, whereas ontology validation covers technologies that try to validate ontological specifications.

Ontology interoperability[51]

Interoperability is one of the most important objectives of the Semantic Web. Ontology interoperability deals with the interoperability of ontologies specified in the same language or in different languages. It facilitates knowledge exchange between different languages and finds compatible fragments in incompatible languages. Ontology interoperability is closely related to ontology integration.

Ontology integration or merging

Ontology integration is the combining of two or more ontologies to obtain a single ontology, and thus merging them. The result is a single ontology having/including all the knowledge of the merged ontologies.

Why ontology languages

Formal semantics in ontologies describe the meaning of knowledge. Here, semantics neither refers to subjective intuitions nor is it open to different interpretations by different persons (or machines). In the context of ontology languages, semantics enforce the meaning of the expressed knowledge as a set of constraints over the domain. Semantics devise the models of a statement precisely, thus studying all the possible instantiations of the domain that are compatible with a statement. Ontology devises a set of models which is the intersection among all the models of each statement in the ontology. The models of ontology represent possible realizable situations only.

1. Reasoning: the notion of deduction can be defined by using ontologies. Thus, with a defined ontology an additional statement can be deduced from the ontology if it is true in all the models of the ontology. This means that if a statement is not true in all the models of an ontology, then it is not a valid deduction from it. This process of deriving valid deductions from an ontology is called reasoning.
2. Class membership: allows one to deduce whether an object is an instance of a class.
3. Classification: allows one to deduce all the subclass relationships between the existing classes in the ontology.

4. Equivalence of classes: allows one to deduce whether two classes are equivalent, i.e. they have the same extension.
5. Consistency of a class: allows one to check that some class does not of necessity have an empty extension.
6. Consistency of the ontology: allows one to check that the ontology admits at least a model, i.e. there is at least a possibility to have an instantiation of the domain compatible with the ontology.

Ontology languages: history and evolution[52]

In the last few years several ontology languages have been developed, in the context of the Semantic Web. These have been classified into different categories.

1. CycL
2. DOGMA (Developing Ontology-Grounded Methods and Applications)
3. F-Logic (Frame Logic)
4. KL-One
5. KIF
6. KM programming language
7. LOOM (ontology)
8. OKBC (Open Knowledge Base Connectivity)
9. OCML (Operational Conceptual Modeling Language)
10. Ontolingua based on KIF
11. RACER
12. PLIB (Parts LIBrary)

Markup ontology languages

In these languages, a markup scheme, most commonly XML, is used for the encoding of knowledge.

DAML+OIL (DARPA Agent Markup Language+OIL)

- Has been developed by a joint committee from the USA and the European Union (IST) and is built on RDFS.
- DAML – a DARPA project for allowing semantic interoperability in XML. Hence, DAML+OIL shares the same objective as OIL.
- Replaces the initial specification, which was called DAML-ONT (DAML Ontology), and was also based on OIL.
- OILEd, OntoEdit, Protégé2000 and WebODE can author DAML+OIL ontologies.

Ontology Inference Layer (OIL)

- Developed in the On-To-Knowledge project.
- Syntax and semantics are based on existing proposals (OKBC, XOL and RDFS).
- Provides modeling primitives commonly used in frame-based approaches to ontological engineering (concepts, taxonomies of concepts, relations and so on), and formal semantics and reasoning support.
- Has the following layers: Core OIL groups, the OIL primitives, Standard OIL, the complete OIL model, using more primitives than those defined in RDFS.
- Instance OIL adds instances of concepts and roles to the previous model; and Heavy OIL.

SHOE (Simple HTML Ontology Extension)

- Developed at the University of Maryland and used to develop Ontology Markup Language (OML).
- Created as an extension of HTML, incorporating machine-readable semantic knowledge in HTML documents or other web documents.
- Makes it possible for agents to gather meaningful information about web pages and documents, thus improving search mechanisms and knowledge gathering.

OML (Ontology Markup Language)

- Developed at the University of Washington.
- Partially based on SHOE, was first considered an XML serialization of SHOE.
- Shares many features with SHOE.
- Four different levels of OML exist.
- Abbreviated OML includes conceptual graphics features; Standard OML is the most expressive version of OML.
- There are no other tools for authoring OML ontologies other than the existing general-purpose XML edition tools.

Resource Description Framework (RDF) and RDF Schema (RDFS)

- Developed by the W3C for describing web resources.
- Allows the specification of the semantics of data based on XML in a standardized, interoperable manner.
- Provides mechanisms to explicitly represent services, processes and business models, while allowing recognition of non-explicit information.

- An RDF data model is equivalent to the semantic networks formalism.
- However, the RDF data model does not provide mechanisms for defining the relationships between properties (attributes) and resources – this is the role of RDFS.
- Is a standard framework for description of resources on the web.
- The building block of this data model is called a statement, an object-attribute-value triple.
- Concrete syntax is needed by abstract data models for representation and transmission. As RDF has the syntax of XML, it inherits the benefits associated with it.
- It is also possible to have those syntactic representations of RDF which are not based on XML.
- RDF is domain independent. Users define their own terminology in RDFS because assumptions about a particular domain are not made.
- In XML Schema, the structure of XML documents is constrained, whereas in RDFS the vocabulary used in RDF data models is used.
- RDFS has templates for defining knowledge models that are similar to frame-based approaches. RDFS is widely used as a representation format in many tools and projects, such as Amaya, Protégé, Mozilla, SilRI and so on.

Strengths

- Simple, basic scheme, relatively easy to learn.
- Built-in extensibility mechanism (metaclass notion).
- Description Logic (DL) is a descendant of early concept languages such as KL-One.

- Classes are defined in a distributed manner, not one class definition.
- Classes do not need to have a name.
- Expressivity is limited by decidability of subsumption reasoning.
- Non-intuitive modeling for non-DL users.

Limitations

RDF and RDFS have limited expressivity. In a nutshell, RDF is limited to binary ground predicates, while RDFS is represented by a subclass hierarchy and a property hierarchy, along with domain and range definitions.

- No cardinality specification.
- No formal features of subclass relation.
- No formal features of properties.
- Inverse, transitive, symmetric.
- Local scope of properties: rdfs:domain and fs:range define a unique domain/range of a property for all classes. Thus RDFS cannot declare class-specific domain/range/restrictions.
- Disjointness of classes: one can state only subclass relationships.
- Boolean combinations of classes: does not allow building of new classes by a combination of other classes using the methods of union, intersection and complement.
- Cardinality restrictions: it is difficult to define restrictions regarding the number of distinct values that a property may take.
- Special characteristics of properties: it is difficult to define special characteristics of properties, such as unique, inverse of, transitive.

OWL Lite/Core

Basic features are:

- cardinality restrictions
- local range constraints
- unique properties
- disjointness and completeness
- equality of resources
- inverse and transitive properties
- datatypes (reference to XML Schema)
- DL extensions for expert language users
- Boolean combinations
- nameless classes.

Frame-based languages

Ontolingua[53] *and Frame Logic*[54]

- Representatives of frame-based approaches.
- Both use frame-based modeling primitives for a logical framework, but apply different strategies.
- Attributes constitute the central modeling primitive but these do not have a global scope. They can be applied to the predefined classes only, while the "same" attribute can be associated with range and value restrictions.
- The intention behind designing Ontolingua was to support the design and specification of ontologies in a clear and meaningful manner based on KIF.
- KIF has been extended by Ontolingua through the use of additional syntax for the inclusion of abstract axioms into defined forms with ontological significance.

- Frame ontology represents the set of KIF expressions that are allowed by Ontolingua to define an ontology. It specifies those representation elements of object-centered representation systems which are supported with special-purpose syntax and code.
- An Ontolingua ontology consists of class definition, interrelations, functions and related axioms.
- Frame Logic specifies frame systems, object-oriented databases and logical programs.
- It also works on integration of constructs for conceptual modeling into a logical framework that would be coherent.

Fundamentally, Ontolingua and Frame Logic integrate frames (i.e. classes) into a logical framework, the only difference between them being in the manner in which they realize frame-based modeling primitives in a logical language.

Description logic-based

1. KL-One
2. RACER
3. OWL

First-order logic-based

CycL and KIF[55] are first-order-predicate logic languages. They have many features in common, such as both are oriented on predicate logics and appear to be extensions of first-order logic. CycL provides richer modeling primitives than KIF (e.g. various quantifiers and micro-theories).

CycL

- Developed in the Cyc project[56] for the purpose of developing that type of ontology which would impart AI to computers.
- Presents the largest worldwide formalized ontology.
- A formal language with syntax originating from first-order predicate calculus; however, it extends first-order logic through the use of second-order concepts.
- Vocabulary consists of terms: semantic constants, non-atomic terms, variables, numbers, strings, etc.
- A set of CycL sentences forms a knowledge base.

Knowledge Interchange Format (KIF)[57]

- Used for knowledge exchange between different computer systems.
- Can also be used as a language for expressing and exchanging ontologies.
- The following are essential to the design of KIF:
  - The language has declarative semantics.
  - The language is logically comprehensive.

XML vis-à-vis RDF

- No agreement on structure and vocabulary.
- RDF provides a formalism for metadata annotation and a way to write it down in XML, but it does not give any special meaning to vocabulary such as subClassOf or type.
- RDFS extends RDF with a schema vocabulary that allows definition of basic vocabulary terms and the relations between those terms.

OWL: an introduction

In the genesis of the Semantic Web, the need was felt for a language that could clearly explain the hidden semantics of classes and associated properties in web documents. Further, a language was required that goes beyond the basic semantics of RDFS, thus allowing machines to perform useful reasoning tasks.

A family of knowledge representation languages was developed for authoring ontologies and was named the Web Ontology Language. The concept was built on OWL DL and OWL Lite semantics (built on DL), while OWL Full used a semantic model, compatible with RDFS. This can also be described as a revision of the DAML+OIL web ontology language. The ontology was developed by the US/UK ad hoc Joint Working Group on Agent Markup Languages.

OWL supports applications that process information available on the web, rather than presenting it to humans only. It thus facilitates greater machine understanding of web content than does any other available ontology language by providing additional vocabulary and a formal semantics. Today, OWL is considered to be one of the fundamental technologies of the Semantic Web.

OWL in Tim Berners-Lee's architecture

In the *schema layer*, simple web ontology languages define a hierarchical description of concepts ("is-a" hierarchy) and properties. These use the general model in the metadata layer to define the basic metamodeling architecture of web ontology languages.

In the *logical layer*, more powerful web ontology languages, based on the basic metamodeling architecture, define a much richer set of modeling primitives.

While URI is used by RDF and OWL to link, use and extend distributed resources, the URI technology allows a web user to refer to a page or other web resource in the most unique manner. Because of this, it is possible to display a page or download a file, or to distinctly name every resource on the web.

Definition

The ideal acronym for Web Ontology Language should be WOL, not OWL. OWL is different from other XML flavors because it is used to support reasoning of information even outside its own context, quite unlike variants and XML schema, which help to define the structure of information within a document only.

OWL can also be termed the core component of the Semantic Web because it has been recommended by the W3C for the expression of ontologies. OWL forms an environment of metadata, ontologies expressed in machine-readable format and the software tools; together, these allow the expression of the semantic relations among heterogeneous and distributed resources.

OWL has evolved from the combination of various technologies recommended by the W3C, namely XML, RDF and URI. This helps in the description of terms and their interrelationships in a particular document.

Predecessors of OWL and their influence

As discussed above, OWL was not the first web-enabled ontology language; its design was influenced by several pre-existing languages, DAML+OIL playing the major role. DAML+OIL was, in turn, heavily influenced by OIL.

SHOE

SHOE was one of the first attempts at defining an ontology language for deployment on the web. It placed emphasis on the fact that ontologies should be tightly interlinked and subject to change and this is why it included a number of directives to allow the importing of other ontologies, local renaming of imported constants, stating versioning and compatibility information between ontologies. The extra-logical vocabulary of OWL was developed to deal with this, though it did not influence the syntactic and semantic design.

DAML-ONT

In 1999, the DAML program was initiated to provide the foundations of a next-generation "Semantic" Web. RDFS was a good starting-point, but it did not meet DAML's requirements. Thus, DAML-ONT was developed on the basis of extended RDF with language constructors from object-oriented and frame-based knowledge representation languages. However, the DAML-ONT project did not have an adequate semantic specification and could not have supported both humans and machines. Moreover, the property restrictions had global rather than local scope.

OIL

Parallel to the development of DAML-ONT, another web-oriented ontology language called OIL was developed by another group of researchers. It combined elements from DL, frame languages and web standards and placed a strong emphasis on formal rigor. It was able to maintain compatibility with RDFS, but did not deal with the precise details of RDF semantics.

DAML+OIL

More recently, the DAML-ONT and OIL groups combined their efforts and this led to the development of DAML+OIL. The merged language had a formal semantics provided by its own DL-style model theory. The DL-derived language constructors were retained while the frame structure was discarded in favor of DL-style axioms.

DAML+OIL was more tightly integrated with RDF but it provided a meaning for those parts of RDF which were consistent with its own syntax and DL-style model theory. The multiple influences on OWL led to a number of problems, some because of conflicting requirements, some syntactical and some even arising from a need to extend previous solutions.

Along with those discussed above, there were also influences from established formalisms and knowledge representation paradigms. A few significant ones are listed below.

Description Logics

This is a group of knowledge representation formalisms that are class based (concept based). Its unique feature is the use of different constructors for the building of complex classes from simpler ones. It strongly influenced the design of OWL, especially the features such as the formalization of semantics, the choice of language constructors and the final integration of data types and data values. As is apparent from its name, logics are the key feature of DL, i.e. formal languages with well-defined semantics.

Frames paradigm

In the context of the Semantic Web, readability and general ease of use were very important for an ontology language.

These were addressed in OWL by providing a surface syntax based on the frames paradigm. Frames group together information about each class and thus make ontologies easier to read and understand, even for users who are not familiar with DL.

In frame-based languages each class is described by a frame which includes the name of the class, identifies the more general class and lists a set of "slots." Property frames describe properties such as range and domain constraints, transitivity, inverse property relationships, etc. Class frames are said to be semantically equivalent to a DL axiom.

The frame style of the abstract syntax makes it easier to read (compared to the RDF/XML syntax), understand and use.

OWL components

The fundamental concepts of OWL are vocabularies (representing the meanings of terms) and the relationships between those terms. Thus, organizing information using OWL becomes more efficient because one does not have to embed software-specific logic in applications.

Vocabularies stand for "individuals" within which are a set of "property assertions." An ontology includes a set of axioms for placing constraints on sets of individuals (called "classes") and their permitted relationships. These impart semantics by allowing systems to interpret additional information coming from the provided data.

OWL, along with its components, provides the ability to indicate if two classes or their properties are identical. Additional information is provided by OWL declarations to allow RDF data to be worked on by rule-checking and theorem-proving systems.

OWL: variants

OWL Lite

- Originally intended to provide a solution to the problem of classification hierarchy and simple constraints.
- Supports cardinality constraints, and permits cardinality values of 0 or 1.
- Evolved with the concept of providing tool support with expressive relatives, and allowing a quick migration path using thesauri and other taxonomies.
- In practice, development of OWL Lite tools has become difficult, quite like the development of tools for OWL DL and OWL Full.
- Supports users who basically need simple constraint features along with classification hierarchy.

OWL-DL

- Was developed to allow maximum possible expressiveness but not at the cost of computational completeness.
- All OWL language constructs along with their restrictions constitute OWL-DL.
- The idea was to help OWL to support the current DL business segment.
- Helps users to achieve maximum expressiveness along with computational completeness and deciding the features of reasoning systems.

OWL Full

- Is based on an altogether different semantics from that of OWL Lite or OWL DL.
- Designed to impart compatibility with RDFS.

- Helps users to achieve maximum expressiveness, along with the syntactic freedom of RDF.
- Allows an ontology to add to the predefined meaning derived using RDF or OWL vocabulary.
- Is an extension of a restricted use of RDF and RDFS, but does not allow classes to be used as individuals and the language constructors cannot be applied to the language itself.
- All RDF and RDFS combinations are allowed.
- Contains OWL DL, but goes well outside the standard DL framework.
- Reasoning in OWL Full is undecidable because the restrictions required in order to maintain the decidability of OWL DL do not apply to OWL Full.[58]

Semantics for OWL Full

OWL Full has a model-theoretic semantics that is a vocabulary extension of the RDF model theory.[59] A correspondence between this semantics and the semantics of OWL DL has been established because the model theory for OWL DL has the same consequences as this RDF-style model theory. The OWL Full model theory has been given "non-normative" status for OWL ontologies that can be written in the abstract syntax.

Features and advantages:

- Easy to understand and use.
- Extends existing web standards such as XML, RDF, RDFS.
- It is possible to provide automated reasoning support.
- Takes the basic fact-stating ability of RDF[60] and the class- and property-structuring capabilities of RDFS Schema and extends them.

- Declares classes and organizes them in a subsumption ("subclass") hierarchy.
- Classes can be termed either as intersections/unions/complements of various classes or as a list of specific objects.
- Can declare properties, organize these properties into a "subproperty" hierarchy, and provide domains and ranges for these properties, again as in RDFS. The domains of OWL properties are called OWL classes; ranges can be either OWL classes or externally defined data types.
- Can state that a property is transitive, symmetric, functional or is the inverse of another property.
- Can express which objects (also called "individuals") belong to which classes, and what the property values are of specific individuals.
- Major extension over RDFS is the ability to provide restrictions on how properties local to a class behave.
- Has evolved as a solution to the syntactic and semantic problems of existing languages.

What makes OWL a Semantic Web language is not its semantics, as it uses the same standard as that followed by a DL, but the use of URI references for names, the use of XML schema data types for data values and the ability to connect to documents in the World Wide Web.

Conclusion

The chapter has outlined the fundamental aspects of KO as a field of study. The basic assumptions and attitudes have

been that the basic units of KO are semantic relations between concepts. Such relations cannot be established by universalistic assumptions, but must be primarily understood as being domain specific and not covered by scientific disciplines. KO in LIS cannot ignore concepts, theories and findings in specific disciplines. The methods of KO in LIS are, on a fundamental level, based on the same philosophical assumptions as the methods of science and scholarship. This ultimately implies that, through the gradual evolution of the universe of knowledge and the field of digital documents, the principles which are practiced in libraries to organize knowledge have remained relevant or, rather, have become more relevant.

The growth and evolution of the Semantic Web has also been reviewed and it was found that the new tools and technologies of the Semantic Web, ontologies in particular, can be developed of the basis of the same fundamental and time-tested principles of KO in libraries.

The growth of OWL is marked by ambitious design goals, multiple influences and the constraints imposed by structural requirements. Because it has not been possible to satisfy all of the constraints on OWL simultaneously, multiple styles of using OWLs have developed, along with the multiple flavors, each appropriate to different circumstances. OWL DL is fairly close to DL and includes features that tend to satisfy most of the requirements of an OWL in the Semantic Web. It uses the data-typing mechanisms of RDF and built-in XML schema data types. Entailment in OWL DL is compatible with entailment in RDF and RDFS.

Another flavor of OWL, i.e. OWL Lite, provides a simpler alternative for development in OWLs. Although it lacks some of the features OWL DL it has effectively tractable inference, closely related to the inference already implemented in description logic systems, namely FaCT[61] and RACER.[62]

The growing use of OWL and the assessment of how the language is being used and how modeling trends have emerged is an interesting area of study.

References

1. Kabilan, V. and Johannesson, P. (2003) "Semantic Representation of Contract Knowledge using Multi-Tier Contract Ontology," *Proceedings of Semantic Web and Databases Workshop (SWDB 2003)*.
2. Hjørland, B. (2013) "Facet Analysis: The Logical Approach to Knowledge Organization," *Information Processing & Management* 49(2): 545–557.
3. Hjørland, B. (2007) "Approaches to Knowledge Organization (KO)," lecture given at the University of Rome. Available at: dlist.sir.arizona.edu/2130/01/Approaches_to_Knowledge_Organization.ppt.
4. Berners-Lee, T. (1999) "Semantic Web Roadmap." Available at: *http://www.w3.org/DesignIssues/Semantic.html*.
5. Naeve, A., Nilsson, M. and Palmér, M. (2001) "The Conceptual Web – Our Research Vision," in *Proceedings of the First Semantic Web Working Symposium* (CID, Centre for User Oriented IT Design, Royal Institute of Technology, Stockholm).
6. Hunter, J. and Choudhury, S. (2006) "PANIC: An Integrated Approach to the Preservation of Composite Digital Objects Using Semantic Web Services," *International Journal on Digital Libraries* 6(2): 174–183.
7. Metadata Encoding and Transmission standard. *http://www.loc.gov/standards/mets/*.
8. MPEG. MPEG Home, *www.mpeg.org/*.
9. Synchronized Multimedia Integration Language (SMIL) 1.0 Specification. Available at: *http://www.w3.org/TR/REC-smil/*.
10. Timed Interactive Multimedia Extension for HTML (HTML+TIME). *www.w3.org/TR/NOTE-HTMLplusTIME*.

11. Kiel, E. (1994). "Knowledge Organization Needs Epistemological Openness: A Reply," *Knowledge Organization* 21(3): 148–152.
12. Dahlberg, I. (2006) "Definitionen aus dem Begriffsfeld 'Wissensorganisation'" (Bad König, 12 September 2006). [In German and in English].
13. Anderson, J. D. (1996) "Organization of Knowledge," in Feather, J. and Sturges, P. (eds.) *International Encyclopedia of Information and Library Science* (London & New York: Routledge), pp. 336–353.
14. In Grauballe, H., Kaae, S., Lykke, M. and Mai, J.-E. (1998) *Klassifikationsteori*, 2nd edn (Copenhagen and Alborg: Danmarks Biblioteksskole).
15. Sayers, W. C. (1915) *Canons for Classification* (London: Grafton).
16. Wellisch, H. H. (1995). *Indexing from A to Z*, 2nd edn (New York: H. W. Wilson).
17. Collison, R. L. (1957) *Book Collecting* (London), p. 121.
18. Stauber, D. M. (2004) *Facing the Text: Content and Structure in Book Indexing* (Cedar Row Press).
19. Svenonius, E. (1990) "Design of Controlled Vocabularies," in Kent, A. (ed.) *Encyclopedia of Library and Information Science* 45 (Supp. 10): 82–109 (New York: Marcel Dekker).
20. Hjørland, B. (2007) "What is Knowledge Organization (KO)?" Available at: *http://www.iva.dk/bh/lifeboat_ko/concepts/knowledge_organization.htm* [Accessed 2 September 2013].
21. SWOOGLE Semantic Web Search. *http://swoogle.umbc.edu/*.
22. *ONTARIA* – Easy Access To The Semantic Web. *www.w3.org/2004/ontaria/*.
23. Gruber, T. R. (1993) "A Translation Approach to Portable Ontology Specifications," *Knowledge Acquisition* 5(2): 199–220.
24. Smith, B. and Welty, C. (2001) "FOIS Introduction: Ontology – Towards a New Synthesis," *Proceedings of the International Conference on Formal Ontology in Information Systems 2001*, pp. 3–9.
25. Fonseca, F. (2007) "The Double Role of Ontologies in Information Science Research," *Journal of the American Society for Information Science and Technology* 58(6): 786–793.

26. McGuinness, D. L., Fikes, R., Hendler, J. and Stein, L. A. (2002) "DAML+OIL: An Ontology Language for the Semantic Web," *Intelligent Systems* 17(5): 72–80 (IEEE).
27. DAML.org. *www.daml.org/*.
28. Fensel, D. et al. (2001) "OIL: An Ontology Infrastructure for the Semantic Web," IEEE Intelligent Systems (March–April): 38–45. Available at: *http://www.cs.man.ac.uk/~horrocks/Publications/download/2001/IEEE-IS01.pdf* [Accessed 22 September 2013].
29. DARPA Markup Language (DAML+OIL). *www.daml.org/language/*.
30. Resource Description Framework (RDF), W3C Semantic Web. *www.w3c.org/RDF/*.
31. OWL Web Ontology Language Overview. *www.w3.org/TR/owl-features/*.
32. SHOE. *www.cs.umd.edu/projects/plus/SHOE/*.
33. topicmap.com. *www.topicmap.com/*.
34. KIM, "Semantic Annotation." *www.ontotext.com/kim/semanticannotation.html*.
35. The NeuroCommons Project. *http://neurocommons.org/*.
36. The Friend of a Friend (FOAF) project. *www.foaf-project.org/*.
37. Semantically-Interlinked Online Communities. *sioc-project.org*.
38. SIMILE Project. *simile.mit.edu/*.
39. Cranefield, S. and Purvis, M. (1999) "UML as an Ontology Modelling Language," in *Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*. Available at: *http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.10.5116&rep=rep1&type=pdf* [Accessed 2 September 2013].
40. Hjørland, B. (2007) "Approaches to Knowledge Organization," lecture given at the University of Rome, 20 April.
41. Gruber, T. R. (1993) "Toward Principles for the Design of Ontologies Used for Knowledge Sharing," *International Journal of Human–Computer Studies* 43: 907–928. Substantial revision of paper presented at the International Workshop on Formal Ontology, March 1993, Padova, Italy. Available as Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University. Revision: 23 August 1993.

42. Gruber, T. R. (2009) "Ontology," in Liu, L. and Tamer Özsu, M. (eds.) *Encyclopedia of Database Systems* (Springer-Verlag). Available at: *http://tomgruber.org/writing/ontology-definition-2007.htm* [Accessed 2 September 2013].
43. Hoffmann, J., Weber, I., Kaczmarek, T., Scicluan, J. and Ankolkar, A. (2008) "Combining Scalability and Expressivity in the Automatic Composition of Semantic Web Services." Available at: *http://icwe2008.webengineering.org/Program/Proceedings/ISBN978-0-7695-3261-5/3261a098.pdf* [Accessed 2 September 2013].
44. Smith, B. (2003) "Ontology," in Floridi (ed.) *Blackwell Guide to the Philosophy of Computing and Information* (Oxford: Blackwell), pp. 155–166.
45. Trinkunas, J. and Vasilecas, O. (2007) "Building Ontologies from Relational Databases Using Reverse Engineering Methods," in *International Conference on Computer Systems and Technologies – CompSysTech'07*.
46. Borst, P., Akkermans, H. and Top, J. (1997) "Engineering Ontologies," *International Journal of Human–Computer Studies* 46(2–3): 365–406.
47. Studer, R., Benjamins, V. and Fensel, D. (1998) "Knowledge Engineering: Principles and Methods," *IEEE Transactions on Data and Knowledge Engineering* 25: 161–197.
48. Sure, Y. and Studer, R. (2005) "Semantic Web Technologies for Digital Libraries," *Library Management* 26(4/5): 190–195.
49. Sowa, J. F. (2000) *Knowledge Representation: Logical, Philosophical, and Computational Foundations* (Pacific Grove, CA: Brooks Cole Publishing Co.).
50. Noy, N. F. and McGuinness, D. L. (2001) "Ontology Development 101: A Guide to Creating Your First Ontology," Stanford Medical Informatics (SMI-2001-0880). Available at: *http://www.ksl.stanford.edu/people/dlm/papers/ontology101/ontology101-noy-mcguinness.html* [Accessed 2 September 2013].
51. Berners-Lee, T. (1999) "Semantic Web Roadmap." Available at: *http://www.w3.org/DesignIssues/Semantic.html*.
52. "Ontology Language," Wikipedia. *http://en.wikipedia.org/wiki/Ontology_language*.

53. Farquhar A., Fikes, R. and Rice, J. (1997) "The Ontolingua Server: A Tool for Collaborative Ontology Construction," *International Journal of Human–Computer Studies* 46: 707–728.
54. Kifer, M., Lausen, G. and Wu, J. (1995) "Logical Foundations of Object-Oriented and Frame-Based Languages," *Journal of the ACM*, 42.
55. Genesereth, M. R. (1991) "Knowledge Interchange Format," in Allenet, J. et al. (eds.) *Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (KR-91)* (Morgan Kaufman Publishers), pp. 238–249. See also *http://logic.stanford.edu/kif/kif.html*.
56. Lenat, D. B. and Guha, R. V. (1990) *Building Large Knowledge-based Systems. Representation and Inference in the Cyc Project* (Reading, MA: Addison-Wesley).
57. Pan, J. Z. and Horrocks, I. (2001) "Metamodelling Architecture of Web Ontology Languages," Information Management Group. Available at: *www.cs.man.ac.uk/~horrocks/Publications/download/2001/rdfsfa.pdf*.
58. Horrocks, I., Sattler, U. and Tobies, S. (1999) "Practical Reasoning for Expressive Description Logics," in Ganzinger, H., McAllester, D. and Voronkov, A. (eds.) *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, pp. 161–180 (Springer).
59. W3C Recommendations: Resource Description Framework (RDF) and Web Ontology Language (OWL) (2004). Available at: *http://xml.coverpages.org/ni2004-02-10-a.html*
60. Gao, M. and Liu, C. (2005) "Extending OWL by Fuzzy Description Logic," *17th IEEE International Conference on Tools with Artificial Intelligence, 2001 (ICTAI 05)*.
61. Horrocks, I., Sattler, U. and Tobies, S. (1999) "Practical Reasoning for Expressive Description Logics," in Ganzinger, H., McAllester, D. and Voronkov, A. (eds.) *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, pp. 161–180 (Springer).
62. Haarslev, V. and Moller, R. (2001) "Description of the RACER System and its Applications," in *DL2001 Workshop on Description Logics, Stanford, CA*.

4

Developing conceptual frameworks: evolution and architecture

Abstract: The development of object-oriented frameworks and product line architectures was one of the most important technological leaps forward in the field of computation. This led to the implementation of standardized software architecture which can be specialized by application-specific code, and also considerably decreased the length of the source code. However, one must take account of the extra effort required for framework development, plus the higher cost input. Framework development seems still to be in its infancy, despite significant developments in the tools and technologies, because of a uniformity of efforts and absence of standards. The recent efforts towards standardization of the UML have led to the hope of harnessing it as a notational basis for projects dealing with framework development, but it is not without lacunae in its constructs.

This chapter introduces the concept of frameworks and its significance to the current context. It also presents important concepts associated with frameworks and reviews the available tools and technologies of framework generation.

Key words: Framework, framework design, web application frameworks.

Background

Object-oriented frameworks and product line architectures became popular during the 1990s. During this period a vast number of frameworks were developed for various domains: graphical user interfaces (GUIs), graph-based editors, business applications, e-commerce, network servers, etc. These were one of the most important technological leaps forward in the field of computation with reusable components. An additional advantage was their scalability on a large scale.

Simply defined, a framework represents a collection of fully or partially implemented components along with their predefined cooperation patterns. This leads to the implementation of software architecture for a set of applications which can be specialized by application-specific code, and therefore some of the components are designed as points of variation. To make application-specific use, these points of variation are built on top of the framework by reusing its source code and architecture design. The use of frameworks also leads to standardization of the application structure because of the reuse of the architecture design. In addition to these advantages, the considerable shortening of the source code by means of the adaptation of a framework can also not be discounted.

However, one must take into account the extra effort that goes into the development of a framework and the higher cost input, as compared to the cost of developing a specific application. Although there have been significant developments in the tools and technologies for framework development, the uniformity of efforts and the absence of standards suggest that the area is still in its infancy.

The recent efforts for the standardization of UML have led to hopes of harnessing it as a notational basis for projects

dealing with framework development. However, there are a few lacunae in the constructs for the modeling of frameworks, and indications of variation points and so on are also missing. Tools have been developed to model framework variation points, along with extensions to assist with framework development and adaptation.

Evolution of frameworks

Historically, building architects encoded knowledge about the design and construction of buildings in the form of design patterns.[1] These design patterns captured recurring architectural arrangements and the basic design relationships, as well as the rules for how and when to apply the encoded knowledge.

This methodology was picked up by researchers working within the object-oriented field and led to the exploration of how software frameworks can be documented using (software) design patterns.[2]

The idea of frameworks was propagated[3] because it was felt that:

- Designing reusable software is difficult and it is not easy to design objects so as to build flexible, modular, reliable code for general reuse.
- The existing software components supported reuse of code, but not the reuse of knowledge.
- Frameworks can support reuse of design and code.
- Experienced programmers should not start from first principles every time and it would be useful to have successful, reusable conceptual designs.

- A particular program instance fails to convey constraints, trade-offs and other non-functional forces applied to the "architecture."

Frameworks are not codes, but reusable designs. Because they are not concrete software but abstract ideas, it is difficult to document them. Documentation for a framework has multiple purposes, and patterns can help to fulfill each of these.

Purpose of frameworks

Frameworks are templates that are intended to aid the fast and efficient development of software by avoiding the duplication of effort at low levels of coding. A good framework thus increases productivity tremendously in big and complex systems, by limiting/defining the choices during development.

Frameworks are useful for:[4]

- developing some applications
- noticing that there are common parts to the applications
- factoring the applications and making the common parts common
- using the resulting framework.

Components of a framework

The components of an ideal framework include:

- categories of functional elements
- principles of accessibility and correlation
- units of content analysis

- defined aspects of analysis
- stages of analysis.

Architecture

Frameworks are composed of frozen spots and hot spots. The frozen spots define the basic architecture of a software system, which remains unchanged (frozen) for different instances of the application framework. Hot spots represent those parts where, to achieve specific functionality, programmers add their own code. Abstract and concrete classes constitute a framework in an object-oriented environment, where its instantiation includes composing and subclassing of the existing classes.

Various user interface elements and functionalities are used to control and manipulate the following elements:[5]

- Container – these contain other elements, e.g. a notebook and its several panels.
- Base – these can be placed in a container element or used at different locations in any HTML page.

The following are examples of some methods for combining elements:

- Element chaining – linking together of elements to perform one task, namely linking of Wizards, notebooks and dialogs.
- Wizard branching – Wizard panel flow can be changed on the basis of the previously entered data.

The framework development process

Any framework starts as a white-box framework and gradually evolves into a black-box one. The difficulty in

developing a framework increases when one tries to make design changes that the user may be able to understand.

Framework development involves three basic phases:[6]

1. Analysis of the problem area, developing an understanding of the abstract ideas and collecting instances of the programs that are to be built.
2. Segregating and defining the abstractions that can be further developed so as to include all options. Finally a theory can be derived to explain these applications.
3. Testing of the framework by executing it to check if it can solve the options.

The above method is an ideal phase analysis, which is difficult and takes time. But there may be other practical approaches:

- Two applications are identified in the same domain whose problem can be solved by using the framework.
- The project is then divided into three groups, namely a framework group and two application groups. The framework both contributes and uses software, and analyzes how other applications can reuse the framework. The application groups try to make maximum reuse of the software in the framework.

Using this approach, the main phases of development are:

1. The framework development phase: this requires the major proportion of the effort and works to produce a design that can be reused for a particular domain.
2. The phase in which the framework is used or instantiated. It is in this phase that the applications are developed.
3. The final phase, where a framework evolves and is maintained.

The following important guidelines can be followed for development of frameworks:[7]

- Borrow frameworks from existing problems and solutions.
- Develop small and focused frameworks.
- Build the frameworks using an iterative process. The process will be driven by client participation and prototyping.
- Treat frameworks as products. Planning for distribution and maintenance should include provision of documentation and support.

Steps in developing a framework[8]

- Identification of the domain area's key abstractions.
- Definition of the flexibility requirements (variation points).
- Definition of the framework design.
- Refinement of the design and its transformation into an architecture.
- Instantiation of the framework.

Difference from contemporary technologies

Patterns and frameworks[9]

Patterns are not frameworks, though they appear to be very similar and are used for documenting them. Frameworks

present an instance of a design for solving a family of problems in a specific domain. To apply frameworks to a particular problem, one needs to customize them by providing user-defined subclasses. Simply stated, the framework is developed as a semi-complete application which contains one or more instances of multiple patterns; at the same time, the same pattern can be used in many different frameworks for similar purposes.

Libraries and frameworks

Libraries are collections of utility classes/methods which the code calls so as to produce a particular functionality, while frameworks contain some functionality or flow and this functionality calls the code either to extend the flow or to make the flow a specific one.

In a library one has to understand the functionality of each method, and the involvement of many methods makes it comparatively hard to create complex interactions. Frameworks contain the basic flow and one has only to plug in the behavior. Further, frameworks are flexible, extensible and provide the necessary means to extend behavior.

The temptation to add more and more functionality, thus creating immobile and complex, bloated frameworks is a disadvantage.

APIs and frameworks

A set of classes constitute a framework that is used to handle the flow that is necessary for performing complex tasks, especially those ones which require application-specific plug-in classes. An API is just a set of method signatures and other information that is necessary to use a class or set of classes. The API doesn't have any implementation by itself and is totally distinct from the implementation.

Types of frameworks

Modeling frameworks

Modeling frameworks provide models with support components for handling tasks such as visualization, data management and model integration.

Conceptual frameworks

Conceptual frameworks are used in research for outlining possible options or for presenting the preferred approach, namely defining the problem and purpose, conducting a literature review, devising a methodology, data collection and final analysis. These are very close to empirical inquiry and take different forms according to the research question or problem at hand, e.g. exploration, description, understanding, deciding and consequential explanation.[10]

Software framework

A software framework is a reusable design for a software system (or subsystem) that may include support programs, code libraries, a scripting language or even other software to help develop and glue together the different components of a software project. Like software libraries, the software framework helps the software developer by containing source code that solves problems for a given domain and provides a simple API.

Examples of significant frameworks

Tarsier

The Tarsier modeling framework supports the development of models in high-level languages, for example C++. It

allows model developers to craft object-oriented approaches for complex modeling problems.[11]

ICMS (Interactive Component Modeling System)

ICMS supports the development of models in a tailor-made modeling language. It allows modelers with even basic programming experience to develop, integrate and visualize advanced models.[12]

BFC (Base One Foundation Component Library)

BFC is a RAD (rapid application development) framework that can be used for the development of database-centric distributed computing applications; this is done in a .NET environment.[13] It is a toolkit for building database applications on Windows and ASP.NET that are secure and fault tolerant and provides a general-purpose web application framework in conjunction with Microsoft's Visual Studio integrated development environment. It works with databases from Microsoft, Oracle, IBM, Sybase and MySQL, which could be running under either Windows or Unix/Linux.

Projects done for Marsh & McLennan and Deutsche Bank in the mid-1990s formed the foundation for the development of BFC. The securities custody system built for the BFC project was one of the earliest successful implementations of commercial grid computing. This was extended by BFC through class libraries for facilitating the development of client/server database applications on a large scale.

CNI (Compiled Native Interface, previously Cygnus Native Interface)[14]

CNI is a framework for the GNU Compiler for Java and allows Java code to source and be sourced by programs specific to a hardware/operating system. The GNU Compiler Collection consists of a free software compiler which compiles Java source code to either machine code or Java Virtual Machine bytecode. It resembles the Java Native Interface framework, a standard available with various Java virtual machines.

CSLA (Component-based Scalable Logical Architecture)[15]

The CSLA framework provides a standard way for creating robust object-oriented programs by using business objects. Objects are implemented using .NET. CSLA.NET is CSLA that has been rewritten for Microsoft .NET.

Significant features include:

- Smart data: stands for a business object that encapsulates all the data and behavior associated with the object which it represents.
- Typed collections: defines a standard way for creating a collection of objects representing a collection of objects.
- Object persistence: here, data is created, retrieved, updated and deleted by well-defined methods of business objects (those associated with data testing only).
- Persistence state maintenance: a standard way whereby a business object is allowed to maintain information about its "persistence state."

- N-Level undo: the object is allowed to easily revert back to previous states; this is useful when a user wants to undo multiple previous edits.
- Business rule tracking: the objects can maintain collections of "broken rule" objects; these rules exist for an object until it is ready to be persisted to the database.

JNI (Java Native Interface)

The code running in the Java virtual machine (JVM) is allowed to source and be sourced by applications and libraries written in other languages.

JNI is useful for writing native methods in situations where an application cannot be written entirely in the Java programming language. It is also used to modify applications written in another programming language and make them accessible to Java applications.[16]

Java developers term JNI an "escape valve" because it helps them to add functionalities to their Java applications that the Java API cannot provide. JNI is also useful for interfacing with code written in other languages and for time-critical calculations or operations.

However, it has some significant pitfalls, such as:

- It is not trivial and some effort is required to learn it. It is recommended that only advanced programmers should use it.
- If even small errors are made in JNI, this will lead to destabilization of the entire JVM. Errors are difficult to reproduce and debug.
- Applications that rely on JNI do not have the platform portability of Java.

- The JNI side does not have any garbage collection (explicit deallocation must be done by the JNI code).
- Error checking is essential so as to prevent crashing of the JNI side.

Leonardi

Leonardi is an open source model-driven application framework for GUI applications. The inbuilt model describes the information system and is enriched with presentation-specific information as well as data connectors.

Business-specific rules and behaviors can be added using the Java language. The concepts of model-driven engineering are applied by the framework to the specific field of GUIs. The execution engine allows dynamic generation of the screens based on requests from the end-user. Leonardi contains a connector-specific configuration that includes flat files, different database management system, CORBA (Common Object Request Broker Architecture), etc.

Leonardi has a studio tool which is useful for the creation of business models or their discovery from an existing source. It also fits nicely into various computer-based information systems and related contexts.

Spring (Spring Framework)

The Spring open source application framework works on the Java platform. It does not push any specific programming model and is used by the Java community as an alternative/replacement/addition to the Enterprise JavaBean (J2EE) model.

While any Java application can use the core features of the Spring Framework, there are many extensions and

improvements for developing web-based applications on a Java Enterprise platform. The Spring Framework is popular because it provides features which help in the creation of complex business applications. These fall outside the purview of the existing programming models.[17]

The Spring Framework can be applied to most application types originating from the Java platform. It is said to be a collection of smaller frameworks which are designed to work independently of each other but which provide better functionalities when used together. The frameworks are divided into the following significant building blocks:

- Inversion of control container: is responsible for lifecycle management of Java objects and also configures the application components.
- Aspect-oriented programming framework: works with functionalities which Java's object-oriented programming skills cannot implement and, if implemented, would lead to sacrifice of some of the significant features.
- Data access framework: using JDBC (Java Database Connectivity) and object-relational mapping tools it allows working with Java-based RDMSs (relational database management systems).
- Transaction management framework: harmonizes various APIs for transaction management and configurative transaction management for Java objects.
- Model-view-controller framework: HTTP and servlet-based framework provides hooks for extension and customization.
- Remote access framework: includes export and import of Java objects in configurative RPC- (Remote Procedure Call) manner. This supports RMI (Remote Method Invocation), CORBA and HTTP-based protocols.

- Authentication and authorization framework: represents those authentication and authorization processes that support industry-standard standards, protocols, tools and practices through the Spring Security sub-project.
- Remote management framework: stands for the exposure and management of Java objects. It is used for both local and remote configuration using JMX (Java Management Extensions).
- Messaging framework: stands for configurative registration of message listener objects. This is for transparent message consumption that comes from message queues via JMS (Java Message Service).
- Testing framework: this supports classes for two types of tests, namely writing unit and integration tests.

Symfony[18]

The Symfony web application framework follows the model-view-controller (MVC) design. It is a free software released under the MIT license. It expedites the creation and maintenance of web applications and is used for replacement of repetitive coding tasks. It has few prerequisites for installation and is compatible with RDMS.

Rails (Ruby on Rails)

Rails is an open source software application framework for the Ruby programming language. It is intended to be used with the Agile development methodology.[19] It uses the MVC architecture for organizing application programming. The framework is separated into various packages, namely ActiveRecord, ActiveResource, ActionPack, ActiveSupport

and ActionMailer. In addition to the standard packages, developers can make plug-ins to extend existing packages.

Swing[20]

The Swing toolkit for Java is a part of Java Foundation Classes. It was developed with the idea of providing a more sophisticated set of GUI components than the earlier Windows Toolkit. It is in sync with the look and feel of several platforms, but it supports a pluggable look and feel to allow applications to appear unrelated to the underlying platform. This means that the look and feel of components in an application can be changed without making significant changes to the application code. Using this feature, one can make an application look very different from native programs.

MFC (Microsoft Foundation Class Library)

MFC wraps those portions of the Windows API that have C++ classes. This includes functionalities that enable them to use a default application framework.[21]

HotDraw[22]

HotDraw is a two-dimensional graphics framework that is used for structured drawing editors written in VisualWorks, Smalltalk. Various editors, from computer-aided software engineering (CASE) tools to a HyperCard clone, have been created from it.

It is basically a reusable design for a drawing tool expressed as a set of classes. However, it also possesses the complete structure of a drawing tool, which needs only to be

parameterized to create a new drawing tool. It provides all the elements of a drawing editor, including a basic working editor which can be customized by the developer as required.

Objective Views

The Objective Views object-oriented framework is useful for creating applications that allow users to work with visual data. It is fully MFC compatible and works as an abstract top layer for the Windows GDI (Graphics Device Interface). Users are allowed to select objects from a palette and add to a layout. It is also useful for creating hierarchies, containers and connections.

Using Objective Views, one can also incorporate custom graphical objects and enhanced drawing capabilities. The basic features include:[23]

- a comprehensive component, property and event model for graphical objects on a canvas
- translation, rotation and scaling of graphical components using transformation matrix
- extensible command architecture with unlimited levels of undo and redo
- scrolling, panning, zooming, and drag-scrolling on the canvas
- snap to grid, ruler guides and page settings
- support for text components, single and multiple lines, word-wrap support
- text labels for symbols, with a variety of orientation options
- the import and export of Windows-enhanced metafiles (.wmf)

- connecting symbols via ports
- OLE (Object Linking and Embedding) drag-and-drop
- Symbol Designer, a utility for drawing symbols to use in applications
- migration of views-based applications to .NET using the Platform Invocation feature (PInvoke) for data exchange between MFC and C# parts of the project.

IBM's e-Components (previously San Francisco Framework)

This business-level component framework provides processes and services that have been implemented in Java. IBM's Value Added Reseller market is the principal target for this solution.

The developer's kit is a three-layer product:[24]

1. The lowermost level, termed the Base Enabling Layer or Base Layer, acts as a reusable substrate or functional lattice for other San Francisco components. It consists of both a set of Base Object Model Classes and a variety of utilities.
2. The Base Layer, some of which was derived from work on the now-defunct Taligent alliance, provides object management (naming, persistence, locking, etc.), security and other low-level functionality, such as error and event handling.
3. The Common Business Objects layer, consisting of independent, generalized business objects and frameworks common to a variety of application types that can be modified to suit particular needs.

Table 4.1 Web application frameworks

Programming language	Web application frameworks
ASP.NET	ASP.NET MVC Framework · BFC · DotNetNuke · MonoRail · Umbraco
Client-side	AJILE· ASP.NET AJAX · Clean AJAX · Dojo Toolkit · Echo · Ext · jQuery · MochiKit · MooTools · OpenLink AJAX Toolkit · Prototype JavaScript Framework · qooxdoo · Rialto Toolkit · Rico · script.aculo.us · SmartClient · Spry framework · Yahoo! UI Library
ColdFusion	ColdSpring · Fusebox · Mach-II · Model-Glue · onTap
Common Lisp	ABCL-web · BKNR · blow · UnCommon Web
Flex	Cairngorm
Java	Apache Cocoon · Apache Struts · AppFuse · Aranea framework · Click Framework · Cooee framework · framework.fleXive · Google Web Toolkit · Grails · Hamlets · IT Mill Toolkit · ItsNat · JavaServer Faces· JBoss Seam · Makumba · Mentawai · OpenLaszlo · OpenXava · Oracle ADF · Reasonable Server Faces (RSF) · Restlet · RIFE · Shale Framework · SmartClient · Spring Framework · Stripes · Tapestry · ThinWire · WebObjects · WebWork · Wicket · XTT Framework · ZK Framework
Perl	Catalyst · Interchange · Mason · Maypole
PHP	Akelos PHP Framework · CakePHP · Chisimba · CodeIgniter · Drupal · eZ Publish · FUSE · Horde · Joomla!· KohanaPHP · MODx · PHP For Applications · PHPOpenbiz · PRADO · Qcodo · Seagull PHP Framework · SilverStripe · Simplicity PHP framework · Symfony · Zend Framework · Zoop Framework
Python	CherryPy · Django · Karrigell · Nevow · Porcupine · Pylons · Spyce · TurboGears · TwistedWeb · web.py · web2py · Webware · Zope
Ruby	Camping · Cerise · Nitro · IOWA · Merb · Ramaze · Ruby on Rails
Server-side JavaScript	AppJet · firecat · Helma Object Publisher
Other/multiple languages	Alpha Five · Fusebox (ColdFusion and PHP) · HAppS (Haskell)· OpenACS (Tcl) · Seaside (Smalltalk) · Yaws (Erlang)

Source: "Comparison of Web Application Frameworks," Wikipedia. Available at: *http://en.wikipedia.org/wiki/List_of_web_application_frameworks* [Accessed 2 September 2013].

The third layer is composed of Core Business Processes, e.g. domain-specific applications, with well-defined extension points through which developers can add their specific business logic. These run on top of the Base and Common Business Objects layers and are designed to provide all the base functionalities that developers might require within a specific application domain.

Jeeves[25]

This Java engine simplifies the development of web applications by using XML for internal data representation and XSL (Extensible Stylesheet Language) for HTML output. It comprises a Java engine, Java web server and Java IDE (Integrated Development Environment) for building applications.

References

1. Alexander, C. (1979) *The Timeless Way of Building* (New York: Oxford University Press).
2. Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., and Angel, S. (1977) *A Pattern Language: Towns, Buildings, Construction* (New York: Oxford University Press).
3. Andreas, B. and Eggenschwiler T. (1993) "Frameworks in the Financial Engineering Domain: An Experience Report," *ECOOP '93 Proceedings of the 7th European Conference on Object-Oriented Programming*. Lecture Notes in Computer Science 707 (Springer), 21–35.
4. Hunt, J. and McManus, A. (1998) *Key Java: Advanced Tips and Techniques* (Springer) 120.
5. Brant, J. and Roberts, D. (1998) "The Refactoring Browser," in *Object-oriented Technology: ECOOP'98 Workshop Reader*, Lecture Notes in Computer Science 1543 (Springer), 549.

6. IBM (n.d.) "Tools Framework." Available at: *http://publib.boulder.ibm.com/infocenter/wchelp/v5r6m1/index.jsp?topic=/com.ibm.commerce.customizetools.doc/concepts/ctfcustom.htm* [Accessed 2 September 2013].
7. Johnson, R. E. (1993) "How to Design Frameworks." Tutorial notes for *Conference on Object Oriented Programming, Systems, Languages and Systems (OOPSLA '93)*.
8. Adair, D. (1995) "Building Object-Oriented Frameworks (Part 1)," *AIXpert*.
9. Bertrand, M. (1990) *Introduction to the Theory of Programming Languages* (Prentice-Hall).
10. Ifacethoughts (2007, 6 June) "Discussion Forum. Difference between a Library and a Framework." Available at: *http://ifacethoughts.net/2007/06/04/difference-between-a-library-and-a-framework/* [Accessed 2 September 2013].
11. Shields, P. and Tajalli, H. (2006) "Intermediate Theory: The Missing Link in Successful Student Scholarship," *Journal of Public Affairs Education* 12(3): 313–334. Available at: *http://ecommons.txstate.edu/polsfacp/39/* [Accessed 2 September 2013].
12. Rahman, J. M., Cuddy S. M., and Watson, F. G. R. (n.d.) "Tarsier and ICMS: Two Approaches to Framework Development." Available at: *http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.87.5203* [Accessed 2 September 2013].
13. Base One International Corp. (n.d.) "Introduction to BFC. Rapid Development of Database Applications." Available at *http://www.boic.com/bfcintro.htm* [Accessed 2 September 2013].
14. About CNI. *http://gcc.gnu.org/onlinedocs/gcj/About-CNI.html*.
15. "Component-based Scalable Logical Architecture," Wikipedia. Available at: *http://en.wikipedia.org/wiki/Component-based_Scalable_Logical_Architecture* [Accessed 2 September 2013].
16. Java Native Interface. *http://java.sun.com/j2se/1.3/docs/guide/jni/*.
17. Spring Framework. *http://www.springframework.org/*.
18. Symfony. Open Source Web PHP Framework. *http://www.symfony-project.org/*.

19. Rails. Web Development that Doesn't Hurt. *http://www.rubyonrails.org/*.
20. O'Conner, J. (2007) "Using the Swing Application Framework." Available at: *http://www.oracle.com/technetwork/articles/javase/swingappfr-136951.html* [Accessed 2 September 2013].
21. MFC Library Reference. MFC Reference. *http://msdn.microsoft.com/en-us/library/d06h2x6e(VS.80).aspx*.
22. Hotdraw Home Page. *http://st-www.cs.uiuc.edu/users/brant/HotDraw/HotDraw.html*.
23. Stingray® Studio Components. *http://www.roguewave.com/products/stingray/overview/architecture.aspx*.
24. Bohrer, K. A. (1998) "Architecture of the San Francisco Frameworks," *IBM Systems Journal* 37(2): 156. Available at: *http://domino.research.ibm.com/tchjr/journalindex.nsf/2733206779564b3d85256bd500483abf/2d424e33ebc23f5085256bfa00685dea!OpenDocument*. [Accessed 2 September 2013]
25. Jeeves. *http://sourceforge.net/projects/jeeves*.

5

Unified Modeling Language: untying the tangles

Abstract: Over time, the Internet has become a corpus of interrelated content, viz. flat documents, dynamically generated documents and data, metadata, and multiple web services based on them. The hyper-linked structure of the web creates a dynamic relationship between context and content, making it difficult to get a complete picture of the conceptual mapping of the information. It can be difficult for a human reader to extract usable meaning. This human-understandable semantics can be achieved only when the conceptual structure of the content becomes an essential part of the posted material.

Research studies have proved that UML-based ontologies can bring semantic coherence; their visual modeling can facilitate conceptual browsing and would be understandable to both machine and users. In this chapter, UML – the selected technology for framework development – is presented in detail. The chapter covers its evolution, development, features and components. It also summarizes the functioning of argoUML, a tool that can be used for the development of UML models.

Key words: Semantic Web, ontologies, UML, argoUML, meta-modeling approach.

Background

The Internet started with the vision of making information omnipresent, with the World Wide Web aspiring to facilitate easy retrieval of information. Although HTML text evolved to be pretty expressive for the human reader, but then something was needed to provide these descriptions in machine-readable format. The Internet became a corpus of interrelated content, viz. flat documents (HTML, XML etc.), dynamically generated documents and data (via JSP, PHP etc.), metadata (RDF, OWL etc.), and multiple web services based on these. The challenge was to overcome the overload by facilitating the reuse of data/information/knowledge.

The idea of the Semantic Web followed. Initiatives were mostly based on formal logic and web-based knowledge representation. However, it was felt that the semantics should be not only machine-understandable but also understandable by the end-user. Even the Semantic Web was failing to achieve the desired semantics, and this led to the vision of concept-based representation. It is difficult not only for the machine but also for a human reader to extract meaningful terms from a web page. Initiatives such as topic navigation, etc. attempted to solve the problem but were not able to provide an overview with conceptual context.

It was felt that UML-based ontologies could bring semantic coherence; the visual modeling would facilitate conceptual browsing and would be understandable to both machine and users. This standard modeling language has multiple advantages, such as a standardized graphical notation, support for XML-based formats, a wide user base, and a number of supporting CASE tools, etc.

Conceptual (un)clarity on the web[1]

The hyper-linked structure of the web leads to a dynamic relationship between context and content, thus making it difficult to get a complete picture of the conceptual mapping of information. At times, it is also difficult to extract usable meaning for a human reader. This human-understandable semantics can be achieved only when the conceptual structure of the content becomes an essential part of the posted material.

Today's Semantic Web initiative does not provide meanings, only descriptions of web resources, which is not a contextually clear way to present them.

The Conceptual Web

There have been attempts to provide both semantic information for the machine and concept-related semantics for the human user. The Conceptual Web is an extension of this form of the Semantic Web. It provides both the machine and human users an advanced understanding of the meaning of web resources, their associated conceptual context, and their inherent meaning/sense from the current context. It is not dependent on the application being used.

Conceptual modeling

Conceptual modeling is the basic building block that derives meaning for human users for both abstract ideas and concrete resources. UML is used for conceptual modeling by providing a well-proven and standardized vocabulary.

Concept browsers

Another tool that is frequently discussed is the concept browser. This allows the user to burrow through the conceptual hierarchy in the form of concept maps (typically UML diagrams) with rich annotations.

The Semantic Web

Semantic Web was conceived primarily to enable machine understanding of web resources. However, the fact that most information on the web is designed for human consumption and that the structure of the data is not evident to a robot/machine/algorithm browsing the web has been a major obstacle. It has been very difficult to derive the meaning from HTML or any other web resource because there is no common metadata framework. Semantics have been incorporated to a great extent, but are meaningful only to the human reader.

UML for ontologies in the Semantic Web

Semantic Web aimed to enrich the World Wide Web with semantic information so as to enable systems to access and use information more efficiently. A number of annotation languages were developed for this purpose. While one could generate metadata from the content of an information source, the development of ontologies became a major bottleneck. Ontologies were being built by a small number of people, and in most cases using prototype tools that provided some basic functionality for editing and storing ontological models. The Object Management Group (OMG)[2] proposed UML as a standard; OMG is a consortium of

around 800 member companies and institutions that are involved in software engineering. Many tools are available for the creation and editing of models in UML. These use direct manipulation of the models' graphical presentation.

The earlier design of UML was for human-to-human communication of models; this was proposed for use in building systems in object-oriented programming languages. It was then used for designing more declarative artifacts, database schemas[3] and knowledge models.[3] The Model Driven Architecture (MDA) from OMG is evolving fast to support the generation of application and middleware code. The heterogeneous environment for development will be based on UML and related standards, namely Meta-Object Facility (MOF), XML Metadata Interchange (XMI), etc.[2]

The use of UML has now become widespread in industry because it provides an effective and scalable approach to conceptual modeling. UML is defined in terms of a graphical syntax as well as a metamodel. The metamodel plays the role of a context-free grammar by defining the legal combinations of terminal symbols.

Further, ontologies are becoming increasingly important as they make available the basic semantic foundation for rapidly expanding technologies.[4] Current ontology research and practice are said to have originated from declarative AI knowledge representations, namely semantic networks and frames. Their initial design was to support runtime reasoning and inference. The MDA works to add more formalism to UML so as to make it useful at compile time and runtime.

When choosing an ontology representation language, one needs to consider various issues, namely the ease with which the language can be used to describe the domain, the types of automated reasoning about ontologies that may be needed, etc. We all know that whenever the representational power increases, the tractability of reasoning comes down.

Thus, the final decision has to be made on the basis of the desired inference and the possibility of supporting that inference from existing ontology representation languages; even the type of the target system plays an important role in the final decision.

In digital libraries, UML can be more useful for knowledge representation, because here the inbuilt reasoning in ontologies will entail the answering of specific and specialized questions. UML will be useful for knowledge representation by designing ontologies; here, class diagrams help in the definition of classes, associated attributes, and their inherent relationships.

UML

UML allows object-oriented analysis and design, along with associated graphical notation. The object-oriented modeling paradigm has made it all the more useful.

A unique feature of UML is that the main notation is defined in terms of a graphical model rather than a formal language. Further, it is an integration of different pre-existing models of knowledge representation in graphical mode and has emerged as a union of these approaches rather than an intersection. UML is able to provide maximal expressivity, and thus covers all aspects of a system by using different interacting diagram types, where each covers a specific aspect of the overall system.

Why UML

- The importance of knowledge representation (ontologies) is increasing, and UML can be extended to suit the needs of ontology definitions.
- There are limited commercial tools available for ontology development.

- The pool of experienced ontologists needs to make ontology modeling accessible to domain experts.
- The population of UML-experienced engineers is growing.
- Interest is growing in the development by W3C of a UML-based presentation syntax for OWL by W3C.
- A working group within OMG is developing a UML profile for OWL.
- UML provides meaningful abstracts that humans can understand, and this works for both ideas and concrete resources.
- UML has a standardized and understandable vocabulary.
- The graphical notation is easy to comprehend and use and is suitable for human-to-human knowledge transfer.
- The OCL is powerful and allows the expression of constraints that cannot be described using description logic. Of course, there is a trade-off between the expressive power of a language and the computational complexity of reasoning about it.
- UML conceptual models can be translated into other ontology languages such RDFS or DAML, or even into object-oriented database systems.
- It allows one to browse through a conceptual hierarchy in the form of concept maps (typically UML diagrams) with rich annotations. Here, visual modeling, when combined with the distribution and universal annotation property of RDF, leads to a hyper-linked web of resources that are conceptually clear.
- It allows mapping of source ontologies to destination ontologies. This is done by creating a third ontology – a collection of concepts and relations depicting semantically similar concepts.

- UML can work as a conceptual modeling language for the Semantic Web because of its wide acceptance and sophisticated tool support.
- The knowledge expressed in UML can be easily understood by the human mind via its standard graphical presentation. It is also easily available for machine processing via the XMI model interchange format and associated software libraries
- The modular nature of object-oriented modeling makes it possible to change the knowledge in a UML model.
- If one feature changes in the model, it does not essentially affect other features.
- Because UML is an abstract modeling language it can even be used for novel purposes that were not thought of during its design stage. The advantage of this is that the models do not become tied to any particular application.
- UML models are useful for deriving new knowledge; however, this can be done only after reasoning them. OCL, the associated constraint language of UML, can be useful for defining derived model elements.

The metamodeling approach

UML architecture

UML is a language initially proposed as a unification of several different visual notations and modeling techniques used for systems design.[1]

UML is now a *de facto* standard for modeling computational systems, and has recently been proposed that the language should be also used as an Ontology Representation Language.[6]

> While modeling languages such as UML are evaluated on the basis of how useful they could be for information systems development. Similarly, ontology specification languages and upper level ontologies should be constructed on principled philosophical theories about what kinds of things exist and what their basic relationships with each other are. Defining UML constructs only in terms of its mathematical meaning is not enough for a suitable ontology representation language. We claim that, in order to model reality, a modeling language should be founded on formal upper-level ontologies, i.e. it should have both formal and ontological semantics.[7]

Components of UML models[6]

- Class: an abstraction of instances that represent the same concept and carry the same features. A class has attributes and methods.
- Inheritance relationship: shows that a child class is a subclass of the parent. It has all the properties of the parent in addition to some other child-specific features.
- Association relationship: defines a semantic relationship between two classes. Associations are directed relationships. The type of relationship is captured in the names and roles of the association. Each of the two classes plays a role in the association.
- Object: stands for a particular instance of a class; has both identity and attribute values.
- Model: mainly used to describe the intention of the class – the defining rules. Here, its execution at run-time provides its extension and instances.

Further elements can be defined at the abstraction level, e.g.

- MMClass: is a top-level element that is used to derive any modeling constructs in the layers below. An instance of the MMClass defines a construct in the meta-modeling layer, but does not define a relationship.
- MMRelationship: is an abstraction for all permissible relationships between the instances of the MMClass. It defines a relationship between instances of the MMClass in the layer below.
- MMSubClassOf: is a concrete implementation of the MMRelationship class. It defines the relationship between two MMClass instances in which one instance is a subclass of the other and also specifies a subset/superset relation between MMClass instances.
- MMDomain: is a concrete implementation of the MMRelationship class. It defines a relationship between two MMClass instances in which the source requires a domain on input value, restricted to constructs of the destination type.
- MMRange: a concrete implementation of the MMRelationship class. It defines a relationship between two MMClass instances in which the *source* produces *output* values.

Diagrams

Diagrams consists of more than 12 different diagram types where each describes the system from a specific point of view. Of these, four diagrams describe the static application structure, five describe different aspects of the dynamic behaviors, and three represent ways to organize and manage the application modules.

Of these, class diagrams express a direct relation between their elements and the parts of an ontology (classes, hierarchies, class attributes and axioms).

In the context of knowledge representation and exchange, the following are some of the important tools:

- *Class diagrams* provide means for defining classes, attributes, and associations between them.
- *Object diagrams* can be used to represent sets of instances of ontologies as objects and their relations at a certain moment in time.

UML provides the concept of *profiles* to allow refinement of the basic meta-model with domain-specific modeling concepts. These modeling concepts are used for building configuration models. UML *stereotypes* are then used to further classify UML meta-model elements (e.g. classes, associations, dependencies), stereotypes being the basic means to define domain-specific modeling concepts for profiles.

Basic profile constructs

OWL ontology item	UML construct	UML stereotype
Ontology	package	ontology
Class Frame	package	classFrame
Class	class	ontologyClass
Relation Frame	package	relationFrame
Relation (property restriction)	class	Relation
Individual Frame	package	individualFrame
Individual	class	individual
	association	individualOf
	association	typeOf
Import Ontology	dependency	<none used>
Axiom (rule)	operation	Axiom
	External file	

UML has an inbuilt visual modeling mechanism to specify, visualize, construct, and document. It has a four layer meta-modeling architecture for defining the structure of complex models:

1. The *Meta-metamodel Layer.* This forms the foundation for the meta-modeling architecture. The primary responsibility is to define the language for specifying a metamodel.
2. The *Metamodel Layer.* The primary function is to define a language for specifying models.
3. The *Model Layer.* A Model is an instance of a Metamodel. The primary function is to define a language that describes an information domain.
4. *User Objects Layer.* User Objects are an instance of a Model. Their primary function is to describe a specific information domain.

Conceptual differences between KR and UML domains[7]

- UML Associations are not first class citizens; Association Classes must have defined endpoints.
- Need to keep additional information (slots, facets, rules) with classes.
- UML Attributes are not first class citizens.
- OCL is not sufficiently expressive for rule implementation.
- Limitations in tool support impact the implementation of ontology modeler add-in, and therefore impact the profile itself.
- Representation of objects (individuals), classes on one drawing.

UML: tools[8]

- AllFusion Component Modeler – *http://www.cai.com/products/alm/paradigm_plus.htm*
- Ameos – *http://www.aonix.com/ameos.html*
- AndroMDA – *http://www.andromda.org/index.html*
- ArgoUML – *http://www.argouml.org/*
- Aris UML Designer – *http://www.ids-scheer.com/aris-uml-designer*
- BridgePoint – *http://www.projtech.com/prods/bp/info.html*
- Codagen Architect – *http://www.codagen.com/products/architect/default.htm*
- Delphia Object Modeller (D.OM) – *http://www.si.fr.atosorigin.com/dom/english/index.html*
- Describe – *http://www.embarcadero.com/products/describe*
- Dia – *http://www.lysator.liu.se/~alla/dia; http://hans.breuer.org/dia; http://dia-installer.sourceforge.net*
- DOME – *http://www.htc.honeywell.com/dome*
- ESS-Model – *http://essmodel.sourceforge.net*
- FL – *http://www.novosoft-us.com/solutions/tools.shtml*
- Flux – *http://flux4eclipse.sourceforge.net*
- Fujaba – *http://www.fujaba.de*
- Innovator – *http://www.mid.de/en/innovator*
- jDes Community Edition – *http://www.javaportal.co.uk/links.uml*
- JDeveloper – *http://otn.oracle.com/products/jdev/content.html*

- Jude Bamboo – *http://objectclub.esm.co.jp/Jude/jude-e.html*
- Konesa Modeller – *http://www.canyonblue.com/products.htm*
- MasterCraft Component Modeller – *http://www.tata-mastercraft.com/overview.asp*
- MDE – *http://www.m1global.com/*
- Mega Suite – *http://www.mega.com/us/product/overview*
- MetaBase Modeler – *http://www.metamatrix.com/l3_metabase.html*
- Napkin – *http://cube42.com/tiki-index.php?page=Napkin*
- Objecteering Personal Edition – *http://www.objecteering.com/products.php*
- ObjectGeode – *http://www.telelogic.com/products/additional/objectgeode/index.cfm*
- Omondo – *www.omondo.com*
- OpenMDX – *http://www.openmdx.com*
- OpenTool – *http://www.tni-valiosys.com/?p=industry&s=aerospace&ss=opentool&type=overview*
- OptimalJ – *http://www.compuware.com/products/optimalj*
- Posseidon – *http://www.gentleware.com/products/download.php3*
- ProxyDesigner – *http://www.proxysource.com/home.asp?href=Products/ProxyDesigner.html*
- Quick UML for Java – *http://sourceforge.net/projects/quj/*
- Rapid RMA – *http://www.tripac.com/html/prod-toc.html*
- Real-time Studio Pro – *http://www.artisansw.com/products/professional_overview.asp*

- Rhapsody Developer Edition – *http://www.ilogix.com*
- Slime UML – *http://www.mvmsoft.de/content/plugins/slime/*
- Software through Pictures UML – *http://www.aonix.com/stp_uml.html*
- System Architect – *http://www.popkin.com/products/system_architect.htm*
- Tau UML Suite – *http://www.telelogic.com/products/tau/uml/index.cfm*
- UML Library for Java – *http://sourceforge.net/projects/nsuml/*
- UML Sculptor – *http://sourceforge.net/projects/umlsculptor/*
- UMLGraph – *http://www.spinellis.gr/sw/umlgraph*
- UMT-QVT – *https://sourceforge.net/projects/umt-qvt*
- UniMod – *http://unimod.sf.net/*
- Unimodeller – *http://www.unimodeler.com*
- Visible Analyst – *http://www.visible.com/Products/Analyst/vaooedition.html*
- Visual Paradigm for UML Community – *http://www.visual-paradigm.com/productinfovpumlce.php*
- Way Pointer – *http://www.jaczone.com/product/overview*

Crosswalk[9]

Table 5.1 shows mappings for translating between DAML and UML concepts and from UML to DAML.

Advantages of the UML meta-modeling approach[9]

- The layered approach provides explicit separation between different levels of abstractions. The only relationship

between elements in one layer and the previous layer is that the lower-level layer can only be an instance of elements from the upper layer (i.e. their types are defined by those at the upper layer).

- The UML modeling language is useful in bootstrapping the meta-modeling layer.
- The meta-modeling approach solves the problem of self-referencing and instantiation.
- The UML-based meta-modeling approach solves the layer-mistake problem for classes.
- The meta-modeling approach solves the layer-mistake problem for properties.
- One can construct valuators for a specific Semantic Web language.

Table 5.1 Translation between DAML and UML

DAML concept	Similar UML concepts
Ontology	Package
Class	Class
As Sets (disjoint, union)	Difficult to represent
Hierarchy	Class Generalization Relations
Property	Aspects of Attributes, Associations and Classes
Hierarchy	None for Attributes; limited Generalization for Associations, Class Generalization Relations
Restriction	Constrain Association ends, including multiplicity and roles. Implicitly as class containing the attribute
Data Types	Data Types
Instances and Values	Object Instances and Attribute Values

The meta-modeling approach for digital libraries: extrapolation

The biggest problem with digital libraries is that of *information access* and *query management*. A solution is difficult because it involves the understanding and modeling of the *content* of complex documents. The problem is magnified by the large number of documents and their heterogeneity in structure and origin. The result is that the excess of information is almost equivalent to absence of information. Tools are needed to "put the data in order" and organize them intelligibly. Only this will facilitate easy access and return answers at various levels of detail.

KR research has made significant progress in query processing and information access, whereas recent work has been more in the area of advanced reasoning techniques, the more specific work being query evaluation and rewriting using views under the constraints given by the conceptual schema, also called view-based query processing.

Conclusion

Transformation of UML models into ontology language can be taken care of by different available tools but the approaches to do so will be different. The challenge lies in deciding on different approaches to translate a subset of UML class diagrams in general. And then beyond this common core, one needs to decide on the different approaches for translation of constructs as well. The actual choice for a specific translation is always a result of the purpose of the translation. A successful strategy will start from the common core and will make more specific transformation choices based on the intended use of the resulting ontology.

Though UML has outgrown its initial purpose as a standard visual notation for constructing intuitive, high-level blueprint models of object-oriented software but still there is a need to extend/refine UML profile and partitioning of the service model into separate diagrams to simplify application. These extensions might transform UML to the pivotal element of semantic web technologies.[1]

References

1. Naeve, A., Nilsson, M., and Palmér, M. (2001) "The Conceptual Web – Our Research Vision," in *Proceedings of the First Semantic Web Working Symposium* (CID, Centre for User Oriented IT Design, Royal Institute of Technology, Stockholm). Available at: *http://kmr.nada.kth.se/papers/SemanticWeb/pospaper.pdf* [Accessed 15 August 2006].
2. Object Management Group. *http://www.omg.org/technology/cwm/*.
3. Naiburg, E. J. and Maksimchuk, R. A. (2001) *UML for Database Design* (Reading, MA: Addison-Wesley).
4. Schreiber, G. (1999) *Knowledge Engineering and Management* (Cambridge, MA: MIT Press).
5. McGuinness, D. L. and Pinheiro da Silva, P. (2003) "Infrastructure for Web Explanations," in D. Fensel, K. Sycara, and J. Mylopoulos (eds), *Proceedings of 2nd International Semantic Web Conference (ISWC2003)*.
6. Unified Modeling Language. Wikipedia. Available from: *http://en.wikipedia.org/wiki/Unified_Modeling_Language* [Accessed 3 August 2006].
7. Object Management Group – UML. *www.uml.org/*.
8. Michael Flanakin's Web Log. "Comparison: UML Tools." Available at: *http://geekswithblogs.net/flanakin/articles/CompareUmlTools.aspx* [Accessed 29 August 2013].
9. Baclawski, K., Kokar, M., Kogut, P., Hart, L., Smith, J., Holmes, W., Letkowski, J., and Aronson M. (2001) "Extending UML to Support Ontology Engineering for the Semantic Web," in *Proceedings of the Fourth International Conference on UML (UML2001), Toronto*.

6

Faceted knowledge representation: putting the leash around unleashed knowledge

Abstract: Semantic Web technologies have not replicated the rapid growth of the World Wide Web. The underlying problems are the inability of keywords to express the semantics of a query, and the functional possibilities and limitations of web applications. Searching for information is difficult and time intensive because of the lower efficiency of keyword-based search; results are often very different from the information seeker's expectations/requirements. The human mind understands and processes information in multiple dimensions, thus the search method should also involve multiple facets.

Facet analysis has been applied to knowledge organization in library and information science and also in artificial intelligence for the design of information retrieval systems. A simplified model to interface between users, information depositors, data repositories, and search engines could lead to the development of a tool which is understandable to both machine and human. The model would be based on the concept of facets and facetization so as to handle the conceptual aspect, and would also be translated to Semantic Web languages for comprehensibility by robots and crawlers.

Keywords: Facets, facetization, analytical theory, Ranganathan's Colon Classification.

Background

The World Wide Web today contains a tremendous amount of information and a variety of web applications making possible the existence of this information. Searching across them for any piece of information becomes very difficult and time intensive because of the low efficiency of keyword-based search; and most of the time the results are very different from the expectations/requirements of the information seeker. The key underlying problems are that keywords are unable to express the semantics of the user's query, and also the functional possibilities and limitations of web applications. As the human mind understands and processes information in multiple dimensions, so the searching method should also involve multiple facets. When this approach was applied in test systems, the precision of results was 93 per cent in some instances, and the precision of recall was 98 per cent.[1]

The concept of facet and facet analysis evolved within the discipline of LIS, mainly from the CRG and Dr. S. R. Ranganathan's Colon Classification theory. Ranganathan felt that the traditional enumerative bibliographic classification systems were not sufficient for knowledge organization, and so he developed the theory of facet analysis.[2,3] Existing systems were incapable of handling the combination of notations from different parts of the classification schedule in compound subjects. The theory of facet analysis[3] was proposed in 1933 as a solution to this problem and was revised in 1967. The CRG was established in 1952 in the United Kingdom to study the nature of existing bibliographic classification systems.[4] The CRG agreed that the inability to express compound subjects was a major limitation of traditional enumerative classification

systems. It started its work on the basis of Ranganathan's theory of facet analysis, but modifying those aspects of classification which it felt to be too restrictive.[5,6,7]

Since then, the theory of facet analysis has been applied to knowledge organization not only in the area of LIS but also in AI for the design of information retrieval systems. To name a few, these include *Thesaurofacet, BSI Root Thesaurus,* and *DHSS-DATA Thesaurus*, various indexing systems such as *GREMAS* and *BTI*, and knowledge-based indexing systems such as *SIMPR* and *MedIndex*.[8,9,10,11,12,13,14,15]

The Semantic Web has developed as a contemporary technology for sharing data, just as the hypertext web developed for sharing documents. The growth of the World Wide Web was very rapid because of the ease of coding in HTML and the option of hyperlinking to other resources, but Semantic Web technologies have not been able to develop at the same pace, for several reasons. To cite a few:[16] (i) the emphasis, during the development of OWL,[17] on inference over fixed datasets; (ii) the chicken-and-egg problem with all network effects, that a lack of linked data is self-sustaining in that, in the absence of things to link to, there is no incentive to put one's own data on the web; and (iii) the lack of a straightforward generic data browser which would give immediate feedback and gratification to the creator of online linked data.

To solve the problem it was proposed that a simplified model could be developed which would interface between users, information depositors, data repositories, and search engines; a tool for all that would be understandable by both computers and humans. The model had to be based on the concepts of facets and facetization in order to handle the conceptual aspect, while it should be translated into Semantic Web languages so as to be understandable by robots and crawlers. Figure 6.1 explains the interplay of various technologies in the Semantic Web context.

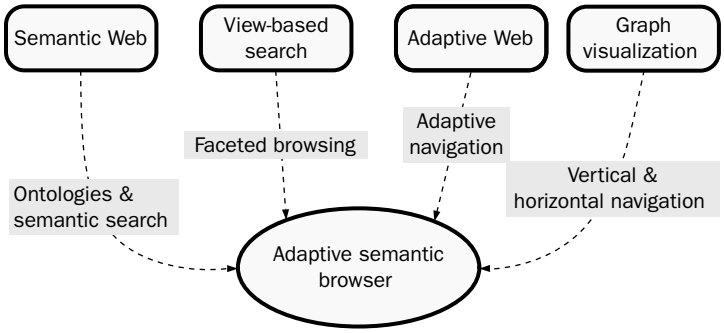

Figure 6.1 **Interplay of technologies in the Semantic Web[18]**

In this chapter, the concepts of facets and facetization are examined from the librarian's perspective and from that of the computer scientist. As the conceptualization makes the basic difference in application, so the concepts of facets and facetization have been extrapolated to computer science applications. Some significant facet-based projects/search tools have also been studied to establish the capabilities and limitations of the two schools of thought, viz. library science and computer science.

Facets: computer science vis-à-vis library science

Facets

Definition:

- A small plane surface (as on a cut gem).[19]
- Adjacent flat surfaces, normally where two adjoining body parts contact and can articulate (i.e., where they are hinged).[20]

According to its etymology, a "facet" is a "little face." The word has been used differently in different contexts in computer science, psychology,[21] and LIS, viz.:

- small components of complex entities/units
- attributes
- properties
- relations
- characteristics
- functions
- concepts.

Facets in computer science

Generally, a facet stands for a single defining aspect; this helps in the determination of a set of values for a simple type, but in computer science the facets for a data type define the set of legal values. A facet can also be defined as an element, wherein each element has a fixed attribute.

When applied to the classification of documents on the web, facets are used as synonyms for a local category, a criteria filter, or useful clusters. Facets are also used to support software reuse.

Facets in LIS

History of facets (a chronological map):[22]

- S. R. Ranganathan – 1960s (Taxonomies)
  - Issue of compound subjects
  - The universe consists of PMEST: Personality, Matter, Energy, Space, Time

- CRG – 1950s, 1970s
  - Facet analysis as a basis for all bibliographic classifications
  - Based on Ranganathan, simplified
  - Principles:
    - Division: a facet must represent only one characteristic
    - Mutual exclusivity
  - More flexible, less doctrinaire
- From classification theory to web implementation
  - An idea waiting for a technology – multiple filters/ dimensions

In classification theory Ranganathan writes: "A facet may be defined as the total set of subclasses produced when a class is divided by a single broad principle …," and he defines arrays as sub-facets: "An array may be defined as the total set of subclasses produced when a class is divided by a single specific principle, so that the resulting subclasses are mutually exclusive ..."[23]

Facets have been found to be useful both in electronic retrieval and in classification theory. S. R. Ranganathan not only introduced the term "facet" into LIS but also worked to develop the facet analysis theory. His Colon system pioneered the application of the principles of facet analysis. Unlike enumerative systems, the faceted classification scheme consists of building blocks. Facets have been defined as "clearly defined, mutually exclusive, and collectively exhaustive aspects, properties, or characteristics of a class or specific subject."[24]

While in the online scenario[25] facets have been explained as a group of concepts that a searcher might consider to be equivalents in a given search, these terms will be considered equivalent only for the purposes of a given information need.

Faceted classification becomes more relevant in the online world because it provides multiple navigational paths to single items of information. Many people consider it similar to folksonomy but, in contrast to folksonomy, the information depicted by facets can be organized in a hierarchy. Moreover, folksonomies are said to be components of the social tagging system in which individuals can "tag" web resources according to their own choice and understanding, without any control or coordination. Faceted systems are regulated by standard guidelines and a defined control vocabulary.

Faceted classification

In a contribution for the CRG, B. C. Vickery wrote that "a faceted classification is a schedule of standard terms to be used in the subject description of documents," and he has demonstrated the relation of facet analysis to the general analysis of information in his article "Analysis of Information."[26]

Mills writes that he does not see faceted classification as "a particular kind of library classification but as the only viable form enabling the locating and relating of information to be optimally predictable. … The continued existence of the library as a highly organized information store is assumed."[27]

Hjørland[28] wrote that the facets constructed in faceted classification systems such as Bliss can be used to define facets while searching online (as a heuristic tool). Actually, when this system is applied, it will allow multiple classifications to be assigned to a single object. Thus the classifications can be arranged in multiple ways, and not in a single, predetermined order.

The Colon Classification designed by S. R. Ranganathan is considered to be one of the best examples of faceted

classification. It is different from traditional library classification schemas like the DDC and Library of Congress classification, in which every document has a "correct/fixed" (or at least, agreed-upon) slot in a single, large, hierarchically organized classification system.

In today's context, faceted classification schemas can be used in faceted navigation systems, helping the user to navigate to information in a hierarchical manner, moving from a category to its sub-categories. Its major advantage over existing traditional taxonomies is that the hierarchy of categories is not fixed and can be changed as determined by the user. The faceted navigation guides the user by displaying the available categories (or facets), but will not require a user to browse through a set hierarchy (which may or may not precisely suit or meet a user's needs or way of thinking).

Facetization and facet analysis

Faceted classification can be considered as an appropriate model where the concept of facetization has been used to create structured classmarks, to provide thesaurus descriptors, and even in search interfaces. It uses

- "simple" concepts, organized into a logical structure
- a standard set of categories to analyze the concepts
- a standard "syntax" for building compound descriptions.

As discussed above, facet analysis can form a basis for classificatory structures in both traditional and online setups, because it

- is a well-established methodology
- organizes concepts in a domain into facets, and then into sub-facets (or arrays)
- identifies and visually displays within a facet the relationships of hierarchy
- collocates synonyms (or near synonyms) and controls them by means of the notation.

B. C. Vickery wrote that: "The essence of facet analysis is the sorting of terms in a given field of knowledge into homogeneous, mutually exclusive facets, each derived from the parent universe by a single characteristic of division. We may look upon these facets as groups of terms derived by taking each term and defining it, *per genus et differentiam*, with respect to its parent class … Facet analysis is therefore partly analogous to the traditional rules of logical division, on which classification has always been based …"[29] Vickery also felt that extending the list of fundamental categories would be more beneficial for the disciplines in science and technology.

Facet analytical theory

S. R. Ranganathan conceived the facet analysis theory in the 1930s. Although the application was novel, the idea was not. Earlier, Henry Bliss had proposed analytico-synthetic approaches for indexing and subject classification. Similar attempts were made by Paul Otlet in the domain of classification, and by Kaiser in indexing.

S. R. Ranganathan defines facet analysis as "the mental process by which the possible trains of characteristics which can form the basis of classification of a subject are enumerated and the exact measure in which the attributes

concerned are incident in the subject are determined. Facets are inherent in the subject."[23]

He identified the way in which knowledge is embodied in books, periodical articles, and other distributed material vis-à-vis the information seeker's approach. This analysis of conceptual aspects is relevant to be applied to any information storage and retrieval system, and can help in:

1. providing better access to the content of documents
2. narrowing down and broadening searches
3. providing alternative search patterns
4. distinguishing between no matches due to search error, and no matches between items not in the information database.

However, this level of semantics can be achieved only if the system has the ability to analyze and synthesize texts in a variety of ways. S. R. Ranganathan has given various postulates for the incorporation of a facet analytic approach in an automated or a manual system. He has further explained that the formula for the facet analysis of a well-designed analytico-synthetic scheme, with each basic subject, several entities, their properties, and actions, should always be developed in relation to space and time.

Time: The pervasive temporal aspects of the world such as day, night, year, century, etc.

Space: Geo and other spatial concepts in the relational system such as India, USA, etc.

Energy: Actions of various varieties such as managing, modeling, etc.

Matter: The properties and other attributes of the subject matter, such as name, valency, etc.

Personality: The objects and their instantial variations.

Basic subject: Chunk of a field of knowledge, subject domain; it can even be convenient as a field of activity.

This core model can be used to derive any number of representations of knowledge embodied in documents and other kinds of materials. The theory contrasts with conventional thinking in which knowledge is viewed as an integral whole. Using this model, the whole is broken into smaller units; related terms or concepts are put into relevant categories to map knowledge in a "bottom-up" fashion. These can be widely applied to a range of subject fields, both functional and linguistic in nature. The robustness of the theory accommodates compound and complex subjects because it allows the combination of individual concepts. The knowledge structure is made up of individual terms which are later analyzed into categories and a defined system syntax is applied to order them further. The resultant structure is highly effective in storage and retrieval because it is logical and predictable. This methodology can be used to create deeper and more complex knowledge structures. The formulae developed for combining terms and concepts can generate n-dimensional structures. These are appropriate to a hypertext environment in which the generated structures become useful for handling digital objects.

Facet analytical methodology

If one wants to devise a methodology for facet analysis, then the internal logic of the system must be based on the analysis of vocabulary and the categorization of terms into a standard set of functional categories, in which one must define a range of semantic relations together with the standards of vocabulary control.

In the beginning, the terms are sorted into categories, and then further analysis defines each category and brings together those terms with a common property or characteristic into arrays or groups of terms. Various principles of ordering/rules can be used to further define the order of arrays in a given category and the clustering of terms within the array. Thus, the categories or facets will be organized into a single sequence based on the standard citation order.

The classificatory structure developed in this manner will be made up of simple terms arranged systematically according to the faceted logic of S. R. Ranganathan.

Limitations

However, this methodology is not without its limitations.

- The terminology does not always reduce or simplify knowledge organization.
- The classification does not only consist of concepts in mutually exclusive sets, and can also turn into another enumerative list.
- The knowledge structure becomes n-dimensional and of great logical regularity.

Distinct advantages

- The implicit principles of categorization can be used to organize a set of properties of objects, in any given domain, and also for the domain itself.
- The theory allows for variation in the classical form.
- The compiler of a faceted structure is not restricted to the categories and the rules for combining them in the standard citation order.

- Facet analysis is a powerful tool, whereby those attributes of objects that may be of significance are incorporated into a standard formula.
- It can accommodate concepts that describe the digital environment and embedded objects.
- It can handle those objects that have complex combinations of attributes and which permit the creation of multidimensional structures.

Faceted knowledge representation

Faceted knowledge representation provides a formal mechanism for the implementation of knowledge systems. Faceted knowledge representation is also referred to as "basic unit," "inter-relations," "facet," and "final interpretation." Basically, facets occur as relational structures that combine units, and relations in which each facet stands for an aspect of a knowledge system. When these facets or relationships are interpreted, the resultant mappings can be used for translation/cross-mapping between different representations.

Faceted knowledge representation originates from the vision of designing a knowledge representation system that is applicable to a variety of domains and suits a variety of users. Here the system will provide a flexible means of coding and displaying knowledge structures depending on adjustable internal or user-defined facets. In the context of conventional knowledge representation, these appear close to formalisms, such as object-oriented design, DL, relational databases, formal concept analysis,[30] and conceptual graphs.[31]

The architecture of a faceted knowledge representation is based on a defined set of primitive notions, such as unit,

relation, and facet, and an open set of logical and relational operators. Another important aspect is the combined extensional, set-oriented, and intentional, relation-oriented approach.

Some examples of facet-based tools and technologies

Application of facet analysis in automatic indexing

Researchers in Documentation Research and Training Centre of the Indian Statistical Institute have worked on several projects of automatic indexing and automatic classification. These were based on the conceptual principles of S. R. Ranganathan and Dr. Bhattacharyya's theory of "deep structure of subject indexing languages." Prominent examples include POPSI (a knowledge representation model to support inference rules for syntax synthesis), PROMETHEUS (a system for parsing expressive titles and extracting noun phrases from documents; these are processed via a knowledge representation model for the generation of meaningful strings) and VYASA (this knowledge representation system is useful for the automatic maintenance of analytico-synthetic schemes).[32]

Web pages designed using faceted techniques

A simple XML format for metadata exchange, namely, XFML (eXchangeable Faceted Metadata Language) has been developed – this will be useful for the exchange of metadata in the form of faceted hierarchies.[33]

Faceted taxonomies for web catalogs

A set of taxonomies called facets constitutes a faceted taxonomy. This is a set of terms that are structured by a specialization/generalization relation. In this, the objects are indexed by associating each object with a combination of terms from different facets.

Conceptual clarity, compactness, and scalability are among the many advantages of a faceted taxonomy over a single hierarchical taxonomy. However, a large number of invalid compound terms can also be formed, to prevent their deployment and use (Figure 6.2).

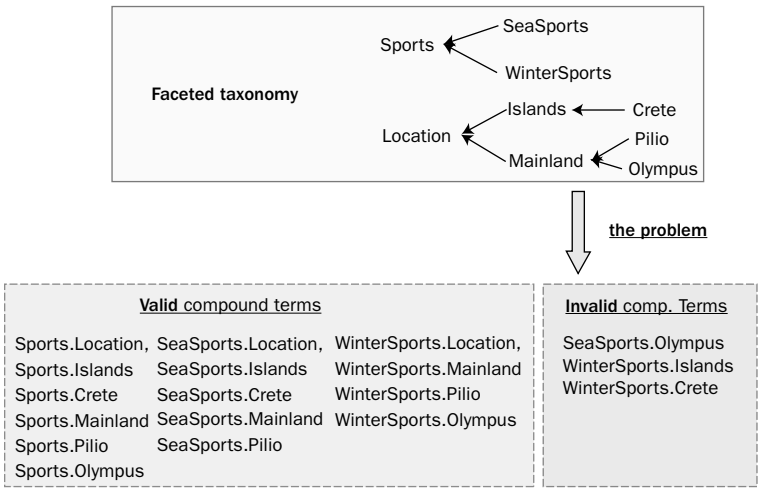

Figure 6.2 **A faceted taxonomy with two facets, Sports and Location, and further partition of the set of compound terms into sets of valid and invalid terms**

Source: Tzitzikad, Y., Spyratos, N., Constantopoulos, P., and Analyti, A. (2002) "Extended Faceted Taxonomies for Web Catalogs," *Ercim News online edition*. Available at: *www.ercim.org/publication/Ercim_News/enw51/tzitzikas.html* [Accessed 2 September 2013].

Software for facet classification and information management

This is based on a semantic infrastructure system for classifying an information asset, comprising a facet storage module storing a plurality of facets in which each facet has a set of classification terms, a facet selection, and a display engine having facet operators. This includes predefined relationships between the classification terms in one facet and the classification terms in another facet, based on established valid term combinations. For each facet, the facet selection and display engine receives term selections and determines valid terms for display from the facet storage module, based on the term selections received.

Visual navigation

Visual navigation is based on the concepts of facetization. Here, the information is organized in visual maps so that the user can select the relationship between different blocks of information. Figure 6.3 depicts the visual map for the Semantic Web.[34]

OntoViews[35]

OntoViews is a search engine based on the semantics of the content, its main feature being concept-based multi-facet and keyword search. Browsing is based on the semantic relations in the underlying knowledge base. Another major feature is the classification tree view, with explicit semantic links in item view. Other significant features include:

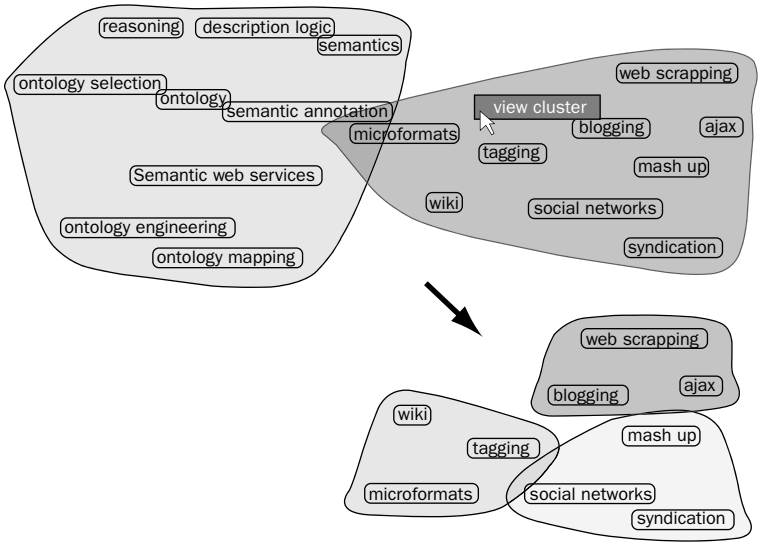

Figure 6.3 Visual map for the Semantic Web

- Developed interfaces based on user interface research make it usable for an end-user.
- It can be easily integrated and extended with additional functionality, and seamless integration of keyword and other searches in the user interface.[36]
- The search architecture allows for extensions, and the Cocoon control architecture forces a modular, reusable, and extendable design.
- All components operate independently, consuming and/or producing RDF/XML.
- It is usable with a variety of different devices, different user interfaces, and functionality for different devices.
- Is adaptable to a wide variety of semantic data where facet hierarchy projection and semantic link generation are based on extendable logic rules.

- It is scalable to accommodate large amounts of data.
- It is capable of providing its functionality to other programs such as Semantic Web services; all subparts of the system are available for use via web services.

Semantic Web data browsers

These include both generic and customized solutions. Palm-DAML, RDF Author and IsaViz are examples of generic browsers, but they have limitations on the display of data in tables and graphs, a format that a typical scientific or enterprise user expects. Others, such as CS-Aktive and mSpace provide a powerful browsing environment, but are tailored to a specific application area. However, these are unable to automatically de-reference web information by looking up URIs, and thus do not support the browsing of linked data.

Faceted browser

This uses facet-based navigation to help users browse information and to increase their ability to find items based on various dimensions. It thus helps them to develop insights pertaining to the data being searched. Of these, "context dependent" is the most useful[37] because in this the available facets, their values, and number are based on the results being browsed by the user. However, it is difficult to compute this, as the number of possible "contexts" in the browsing space multiplies with the number of items, facets, and their values.

Description

The faceted browser uses classification ontology. This describes important aspects of instances from a domain

ontology, which are later used for defining restrictions on the instances.

This allows the user to:

- place complex restrictions on instances
- sort instances
- compare instances
- search within a subset of instances.

Many softwares have included elements of faceted metadata, such as Adiuri, Siderean, and Endeca. The Flamenco project has full deployment of faceted metadata.[38] Images of England has been arranged according to faceted metadata on the website.[39]

Faceted classification software

Today many software products are available to perform the primary task of faceted classification. A few significant examples are listed below.

- Dieselpoint, Inc. has developed software for searching large datasets, along with structured and unstructured elements. Here, the emphasis is on scalable search and faceted navigation to enable ordering and classification of search results. The result sets can be navigated through using the dynamically generated menus from the underlying document properties, thus giving the user context-dependent browsing capability.[40]
- The Endeca Information Access Platform guides a user to a goal, regardless of the fact that the user had or did not have knowledge of that goal beforehand. The technology, called Guided Navigation, is covered by several patents.[41]

- iSeek.[42]
- ISYS.
- Nstein Technologies provides a structured framework built on linguistic technology and semantic search that relates and links content to drive value.[43]
- Siderean.[44]
- SpeedTrack.[45]
- Solr is an open source enterprise search server that is based on the Lucene Java search library.[46] It has features like XML/HTTP, JSON APIs, hit highlighting, caching, replication, faceted search, along with a web administration interface. It runs in a Java servlet container.
- XSEARCH.[47]

Faceted thesaurus

Faceted thesaurus is a faceted, hierarchically structured set of terms and/or concepts that can be used in information retrieval. The primary purpose of any thesaurus is to provide a controlled vocabulary for assigning terms to documents. However, the faceted hierarchical display leads to maximum efficiency and control in the design and use of the thesaurus. The notion of "facet" in the faceted thesaurus refers to S. R. Ranganathan's use of "facet," where the modeling can be directly translated into an object-oriented implementation. This provides a graphical interface for thesaurus construction, and the faceted modeling ensures that the thesaurus can be designed in a modular fashion. It also eases future updating and maintenance of the system. The graphical interface based on the faceted structure provides user-friendly access to the database and retrieves documents that users are interested in searching and browsing.

Faceted navigation

In faceted navigation, facets act as multidimensional filters, and users can also browse within a facet. This is not a Yahoo!-style pre-coordinated browse; rather, it is a tool where facets are applied at search time, in a post-coordinated manner. Faceted navigation is an active interface that allows dynamic combination of searching and browsing.[22]

Key features

1. The metadata conditions can be selected by clicking on the suggested values.
2. The user never gets an empty results list because only those metadata values that are linked to documents are suggested.
3. The metadata is organized into several independent categories, called *facets*. The values taken by these can be organized hierarchically in a taxonomy.

Advantages

- More intuitive, with simple internal organization.
- Dynamic selection of categories allows multiple perspectives and enables handling of compound subjects.
- Gently guides users into the use of Advanced Search.
- Needs fewer elements.

Disadvantages

- Lack of standards for faceted classifications makes it project specific.

- Difficulty in expressing complex relationships, due to the absence of well-defined rules.
- Inherits the limitations of faceted navigation; at times leads to limited domain applicability.
- One cannot ignore the cost involved in tagging, etc.
- Trade-off between simplicity (power and ease of understanding) and complexity (real world).

Examples of faceted navigation include the Croton search interface. Here the user can enter a query in the normal search box, but has the facility to "Refine by" the facets, such as "Incident Date," "Geography," etc. Each of these represents the top node of a different hierarchy of values, and this hierarchy can further be expanded in the user interface.

Another good example is the faceted navigation application from IBM (Figure 6.4).[47] Here, the items are described both by metadata and by content, the significant advantages being the familiarity of faceted navigation and the ease of use by untrained searchers. Because of the faceted navigation

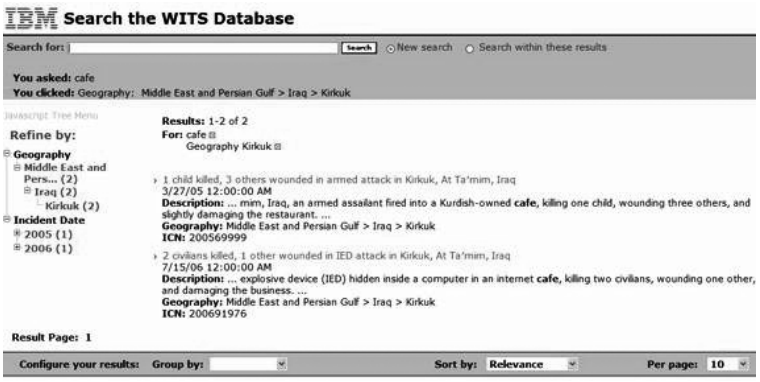

Figure 6.4 Faceted navigation application from IBM

approach, one is able to keep the query language simple and compatible with Internet search engines. A further advantage is that of *successive refinement*,[48] where the user adds more precise conditions when the initial broad query gives access to some basic documents in the area of interest.

Spiteri's simplified model for facet analysis[49]

Louise Spiteri analyzed the rules laid down by S.R. Ranganathan and the CRG for designing faceted classification systems, and proposed her own, simpler set.

In this also, classification is divided to three parts: "the Idea Plane – involving the process of analysis of a subject field to its components; the Verbal Plane – which involves the process of choosing appropriate terminology for expressing those components; and Notational Plane – which involves the process of expressing these component parts by means of a notational device."

Tagging application[50]

This consists of a web interface program (WIP) and a tagging program. The facet tree and the activation method in the WIP allow the user to select a facet element and to assign a value to the selected facet or facet element. Here, either each facet or facet element can be manually tagged by selecting a value, or the user can choose a tagging scenario and then apply it. In faceted search the engine presents the next set of facets on selection of a facet (a word or phrase) until a final facet is reached and the facet elements are displayed. When the user selects one of the facet elements, the documents from the associated posting list are displayed.

Attributes of faceted interfaces

1. The interface displays several facets.
2. It provides the information about the number of "information elements" still left when the user makes his/her choice.
3. Each choice changes the entire view:[51]
   - The search result is updated.
   - The interface displays all the facets and choices that still can be made. The values behind the choices are also updated.
   - All visible choices contain information elements.
4. A facet can also have a hierarchy.
5. When facets have too many values, only a summary is displayed.

Conclusion

Most of the traditional classification systems are hierarchical systems, in which subjects are arranged in a hierarchy of classes, divisions, and subdivisions. Unless users have a good understanding of the structure of the hierarchy, and know exactly under which one of the main classes (and divisions, subdivisions, and so on) the particular subject is classed, it takes a series of trial-and-error attempts to locate the subject.[52] A primary problem of hierarchical classification systems is that people do not always want to choose the same sameness first: "we do not always want what the citation order gives us." So, it is necessary to consider the diversity of users and to apply a range of optional, partial, and local solutions.

Unlike hierarchical systems, faceted classification systems assign to subjects only those attributes, properties, or characteristics of a specific domain that are clearly defined, mutually exclusive, and collectively exhaustive. Faceted classification provides more than one path to locate a subject. By using different aspects/facets of a subject, users can narrow down the search and locate the subject more easily. Although hierarchical and faceted systems each have their strengths and weaknesses,[53] the traditional hierarchical structure is increasingly seen to be ineffective for organizing resources on the web.[53,54,55]

Faceted classification is more flexible and hospitable in accommodating new categories and is also more suitable for representing complex ideas/objects. It is particularly useful for organizing complex items and materials in a multi-disciplinary environment.

But faceted browsing/navigation is not without its limitations, and the tools based on the concept of facet and facetization inherit these limitations. Its problems are managing information overload, along with too many facets (complex information spaces), too many restrictions (large information spaces), and, most of the time, hard-to-understand instance details.

References

1. Usanavasin, S., Nakamori, T., Takada, S., and Doi, N. (2003) "A Multi-faceted Approach for Searching Web Applications," in *Proceeding (407) Information and Knowledge Sharing* (Japan).
2. Ranganathan, S. R. (1962) *Elements of Library Classification* (Bombay: Asia Publishing House).
3. Ranganathan, S. R. (1967) *Prolegomena to Library Classification* (New York: Asia Publishing House).

4. Foskett, D. J., Mikhailov, A. I., and Gilyarevskij, R. S. (1971) "Guide for an Introductory Course on Informatics/ Documentation," *Journal of Documentation* 27(1): 51–52.
5. Austin, D. (1968) *Fields, Categories and General Systems Theory: Report to the Classification Research Group* (London: BNB).
6. Classification Research Group (1985) "The Need for a Faceted Classification as the Basis of All Methods of Information Retrieval," in L. M. Chan et al. (eds) *Theory of Subject Analysis* (Littleton, CO: Libraries Unlimited).
7. Wilson, T. D. (1972) "The Work of the British Classification Research Group," in H. Wellisch and T. D. Wilson (eds) *Subject Retrieval in the Seventies* (Westport, CN: Greenwood).
8. Aitchison, J. (1969) *Thesaurofacet* (Whetstone, Leicester: English Electric Co.).
9. British Standards Institution (1988) *BSI Root Thesaurus* (Milton Keynes: British Standards Institution).
10. Burton, P. (1985) "Expert Systems in Libraries," in F. Gibb (ed.) *Proceedings of a Conference of the Library Association Information Technology Group and the Library and Information Research Group* (London: Taylor Graham).
11. Gibb, F. and Fleming, P. (1991) "Knowledge-based Indexing: The View from SIMPR," in C. McDonald and J. Weckert (eds) *Libraries and Expert Systems. Proceedings of a Conference and Workshop Held at Charles Sturt University – Riverina, Australia, July 1990* (London: Taylor Graham).
12. Revie, C. W. and Smart, G. (1991) "The Construction and Use of Faceted Classification Schema in Technical Domains," in N. J. Williamson and M. Hudon (eds) *Classification Research for Knowledge Representation and Organization. Proceedings of the 5th International Study Conference on, Toronto, Canada, June 24–28, 1991* (Amsterdam: Elsevier).
13. Stiles, W. G. (1985) "Ranganathan, Cognition and Expert Systems," *Canadian Journal of Information Science* 10: 16–24.
14. Travis, I. L. (1989) "Applications of Artificial Intelligence to Bibliographic Classification," in *Classification Theory in the Computer Age: Conversations across the Disciplines.*

Proceedings from the Conference. November 18–19, 1988, Albany, New York (Albany, NY: Nelson A. Rockefeller College of Public Affairs and Policy).
15. Travis, L. (1990) "Applications of Knowledge-based Systems to Classifications in Libraries," in R. Aluri and D. E. Riggs (eds) *Expert Systems in Libraries* (Norwood, NJ: Ablex).
16. Berners-Lee, T. et al. "Tabulator: Exploring and Analyzing Linked Data on the Semantic Web." Available at: *http://swui.semanticweb.org/swui06/papers/Berners-Lee/Berners-Lee.pdf* [Accessed 2 September 2013].
17. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., and Stein, L. A. (2004) "OWL Web Ontology Language Reference. Technical Report" (W3C). Available at: *http://www.w3.org/TR/2004/REC-owl-ref-20040210/* [Accessed 2 September 2013].
18. Sofsem (2008) (High Tatras, Slovakia).
19. Merriam-Webster online dictionary. Available at: *http://www.merriam-webster.com/dictionary/facet* [Accessed 2 September 2013].
20. Virtual Fossils Museum, "Trilobites Glossary." Available at: *www.fossilmuseum.net/glossaries/trilobitesglossary.htm* [Accessed 2 September 2013].
21. Krauth, J. (1981) "Techniques of Classification in Psychology. I: Factor Analysis, Facet Analyses, Multidimensional Scaling, Latent Structure Analysis," *International Classification* 8(3): 126–132.
22. Reamy, T. (2006) "Taxonomies and Faceted Navigation, Getting the Best of Both" (KAPS Group, Knowledge Architecture Professional Services). Available at: *http://www.kapsgroup.com/presentations.shtml* [Accessed 2 September 2013].
23. Mills, J. and Broughton, V. (1977). *Bliss Bibliographic Classification. Second Edition. Introduction and Auxiliary Schedules* (London: Butterworth).
24. Taylor, A. G. (1992). *Introduction to Cataloging and Classification*, 8th edn (Englewood, CO: Libraries Unlimited).
25. Harter, S. P. (1986). *Online Information Retrieval. Concepts, Principles and Techniques* (New York: Academic Press, Inc.).

26. Foskett, D. J. (2003) "Facet Analysis," in M. A. Drake (ed.) *Encyclopedia of Library and Information Science*, 2nd edn (New York: Marcel Dekke), 1063–1067.
27. Mills, J. (2004) "Faceted Classification and Logical Division in Information Retrieval," *Library Trends* 52(3): 541–570.
28. Hjørland, B. (1988) "Information Retrieval in Psychology," *Behavioral and Social Sciences Librarian* 6(3/4): 39–64.
29. Vickery, B. C. (1960) *Faceted Classification. A Guide to the Construction and Use of Special Schemes* (London: ASLIB).
30. Ganter, B. and Rudolph, S. (2001) "Formal Concept Analysis Methods for Dynamic Conceptual Graphs," *Proceedings of the 9th International Conference on Conceptual Structures: Broadening the Base*, Lecture Notes in Computer Science (Springer), 143–156.
31. Sowa, J. F. (1984) *Conceptual Structures: Information Processing in Mind and Machine* (Reading, MA: Addison-Wesley).
32. Slavic, S. (2006, 3 September) Message posted on isko-l@lists.gseis.ucla.edu.
33. Van Dijck, P. (2003) "Introduction to XFML," O'Reilly xml.com. Available at: *www.xml.com/pub/a/2003/01/22/xfml.html* [Accessed 2 September 2013].
34. Idrissi, K. et al. (2003) "Object of Interest Based Visual Navigation, Retrieval and Semantic Content Identification System," *Computer Vision and Image Understanding Archive* 94(1–3) Special issue on color for image indexing and retrieval, 271–294.
35. Mäkelä, E. et al. (2004) "Ontoviews – A Tool for Creating Semantic Web Portals," in *The Semantic Web – ISWC 2004, Proceedings of Third International Semantic Web Conference, Hiroshima, Japan November 7–11*, Lecture Notes in Computer Science 3298, 797–811.
36. Mäkelä, E. (2005) "OntoViews – Semantic Portal Creation Tool." Available at: *http://www.seco.tkk.fi/events/2005/2005-11-16-finnonto/2005-11-16-makela-ontoviews-finnonto-symposio.pdf* [Accessed 2 September 2013].
37. "Faceted Browser." *http://simile.mit.edu/wiki/Faceted_Browser*.

38. The Flamenco Search Interface Project. *http://flamenco.berkeley.edu/*.
39. Images of England. *http://www.imagesofengland.org.uk/*.
40. Dieselpoint. *http://www.dieselpoint.com/?src=wikip*.
41. Oracle Endeca Guided Search. *http://www.oracle.com/us/products/applications/commerce/endeca/endeca-guided-search/overview/index.html*.
42. Iseek. Targeted Discovery. *http://www.iseek.com/iseek/home.page*.
43. Nstein Support Network. *https://support.nstein.com/wiki/*.
44. Siderean. *http://www.siderean.com/*.
45. Speedtrack, "Technology." *http://www.speedtrack.com/technology*.
46. Apache Solr. *http://lucene.apache.org/solr/*.
47. Marwick, A. (2008) "Faceted Navigation for Document Discovery." Available at: *http://www.ibm.com/developerworks/db2/library/techarticle/dm-0802marwick/* [Accessed 2 September 2013].
48. Spink, A., Wilson, T. D., Ford, N., Foster, A., and Ellis, D. (2002) "Information Seeking and Mediated Searching Study: Part 3. Successive Searching," *Journal of the American Society for Information Science and Technology* 53: 716–727.
49. Spiteri, L. (1998) "A Simplified Model for Facet Analysis: Ranganathan 101," *Canadian Journal of Information and Library Science* 23(1–2): 1–30.
50. Kauppinen, T., Henriksson, R., Sinkkila, R., Lindroos, R., Vaatainen, J., and Hyvonen, E. (Semantic Computing Research Group) (n.d.) "Ontology-based Disambiguation of Spatiotemporal Locations." Available at: *www.seco.tkk.fi/publications/2008/kauppinen-et-al-spatiotemporal.pdf* [Accessed 2 September 2013].
51. Olson, H. A. (2001) "Sameness and Difference: A Cultural Foundation of Classification," *Library Resources & Technical Services* 45(3): 115–122.
52. Kwasnik, B. H. (1999) "The Role of Classification in Knowledge Representation and Discovery," *Library Trends* 48(1): 22–47.

53. Ellis, D. and Vasconcelos, A. (2000) "The Relevance of Facet Analysis for World Wide Web Subject Organization and Searching," *Journal of Internet Cataloging* 2(3/4): 97–114.
54. Bates, M. J. (2002) "After the Dot-bomb: Getting Web Information Retrieval Right this Time," *First Monday* 7(7). Available at: *http://firstmonday.org/issues/issue7_7/bates/index.html* [Accessed 2 September 2013].
55. Broughton, V. (2002). "Facet Analytical Theory as a Basis for Knowledge Organization Tools in a Subject Portal," in *Proceeding of the Seventh International ISKO Conference, Granada, Spain*, 135–141.

7

Designing the framework: ready ... get set ... go!!

Abstract: With the increasing volume of information and demand for it, many research efforts are directed towards more refined tools and techniques for finding information. Today's web provides some descriptions of web resources, but does not represent them to users in a contextually clear way. Initiatives like topic navigation and conceptual maps have tried to solve the problem but have been unable to present concepts in context. UML-based ontologies could be a viable candidate for knowledge modeling where browsing is conceptual and understandable to both users and machines.

This chapter presents the faceted model, which can facilitate visual modeling of concepts and conceptual relations in domains. The semantic framework for knowledge organization is based on Dr. S. R. Ranganathan's Faceted Classification Schema. The model can also be applied in digital libraries to facilitate semantic browsing and search.

Keywords: Semantic search, facet formula, UML, argoUML, concept maps.

Background

With the increasing volume of information and the demand for it, many research efforts are towards the creation of more refined tools and techniques for *finding* information.

Today's web provides some descriptions of web resources, but does not represent them to the user in a contextually clear way. Tim Berners-Lee envisioned the World Wide Web as a tool of interlinked and thus more "connected" information. It was thought that it would facilitate easy retrieval of information, but somewhere the objective has been missed. HTML evolved as a structured markup language but its formatting tags provided little meaning for machines. While it has enabled large scale e-publishing possibilities, there are no pointers to tag "what" information goes in.[1] The very purpose of the web in terms of its retrieval capacity is questioned and the World Wide Web has been reduced to a maze of information.

Initiatives such as topic navigation and conceptual maps[2] have tried to solve the problem[3] but have not been able to present concepts in context, though there have been attempts to relate concepts to a degree. UML-based ontologies could be a viable candidate for knowledge modeling, where the browsing can be conceptual and understandable to both machines and users at the same time.[4]

The faceted model developed in the present research facilitates the visual modeling of concepts and conceptual relations in domains. In this research, a semantic framework has been formulated for knowledge organization, based on S. R. Ranganathan's Faceted Classification Schema. This investigates the adaptation of the classical theory of classification to a computational framework using UML, a generic model of knowledge organization. The model can be applied in digital libraries to facilitate semantic browsing and searching mainly because digital libraries host domain-specific collections and the domain knowledge is to be modeled in a way that forms underlying knowledge organization layers.

Semantic search and retrieval

As has already been discussed, the Semantic Web started as a virtual infrastructure in which applications can be developed to provide access to, to process and to handle information in reusable components. While efficiency of information retrieval is the objective, the crux of the problem can be traced back to ways and means for the logical organization of information.

Ontologies were used to model the semantic structure of the information presented, and also to describe the models of a domain that might be independent of any particular information source. Ontologies have typically been built-in models where concepts are defined and presented along with conceptual relationships expressed in formal languages.[5] Fundamental concepts, relationships and rules can be incorporated to model digital domains using ontologies. These essential concepts act as a frame of reference for further design and construction, whether in the domain per se or for applications such as digital library content representation.[6] Ontologies are being used to classify, compare, and differentiate the features of various semantic domains.[7] This demonstrates that concepts connote particular ideas and are distinct only in the context(s) in which they are perceived. However, the question is, how many different representations may suffice to represent all the different contexts distinctively, and how to enumerate them all while designing the retrieval system.

The concept of the Semantic Web emerged following the era of information overload and explosion. New research projects based on formal logic and web-based knowledge representation emerged, in which the emphasis shifted totally to machine understanding, leading to the evolution of complex languages and technologies. However, early

solutions[8] were unable to provide the desired semantics, which in turn led to the development of a different plane of thought, the Conceptual Web, where resources are put into semantic context for both machine and end-user. Clearly, ontologies per se do not suffice to solve problems associated with the retrieval of information from the web, but can be envisaged to substantially facilitate retrieval through semantically mapped bits of information.

Conceptual framework

Semantics has been defined as "the study of meaning in language, i.e., the study of the relationship between linguistic expressions and reality" in the domain of conceptual modeling.[9] "Thus the Semantic Web vision of building an enriched network of information and knowledge requires existing knowledge bases to be translated into semantic knowledge resources. This migration from existing knowledge bases to semantic knowledge base, viz., ontology, has few obstacles to overcome. Primarily, knowledge base users and domain experts cannot be expected to learn new knowledge representation formalisms or ontology languages like RDFS, DAML+OIL and OWL."[10] Moreover, "the translation from existing knowledge bases to ontology needs to be simple, reusable, and interoperable. That is semantic knowledge needs to be modeled in a form that is machine understandable yet supports human understanding and to use modeling approaches that are widely accepted, and the knowledge representation should be flexible, reusable and it should be able to interoperate with other knowledge bases."[10] This framework uses "an accepted standard like UML for ontology modelling … facilitating easy understanding and interoperability … [by developing the model and knowledge]

in a language that already has a vast global acceptance [...] we propose the use of UML conceptual models as an intermediate ontology to aid easy interoperability, sharing and reuse."[10]

Digital libraries: the need for a generic conceptual framework

Digital libraries are extremely complex information systems. Different historical and technological views, along with varied perspectives (from LIS, information retrieval, or human–computer interaction communities) have created differing definitions. Levy and Marshall felt that digital libraries are a polygamy of documents, technology, and work.[11] Lesk analyzed the relative importance of the words *digital* and *library* and concluded that many of those efforts were not associated with users' needs and the use of provided resources.[12]

The President's Information Technology Advisory Committee Panel on Digital Libraries discusses "digital libraries – the networked collections of digital text, documents, images, sounds, scientific data, and software that are the core of today's Internet and tomorrow's universally accessible digital repositories of all human knowledge. Underlying all of these is the consensus agreement that digital libraries are fundamentally complex. Such complexity is due to the inherently interdisciplinary nature of this kind of system. Digital libraries integrate findings from disciplines such as hypertext, information retrieval, multimedia services, database management, and human–computer interaction. The need to put all these together complicates the understanding of the underlying concepts and functionalities of digital libraries, thus making it difficult to construct new digital library systems."[13]

The unique requirements of digital libraries need new frameworks and theories, these can help to understand better the complex interactions among its various elements.[14] The Joint NSF-European Union Working Groups on Future Directions of Digital Libraries Research supported this claim by recommending that "new frameworks and theories be developed in order to understand the complex interactions between the various components in a globally distributed digital library."[15] Though the necessity for such an underlying theory has long been advocated, little progress has been made towards a formal framework or theory for digital libraries. A formal framework should be able to include both the general characteristics and the common features of a set of systems and explain their structures and processes.

In a library, different users approach the same collection from different perspectives, and the same holds good when they approach information on the web. Typically, their interactions are through a web browser or a familiar search engine, with a few key terms used to represent the user's need. *The problem is the sheer lack of context in the underlying web of published information.* Because search engines have little knowledge of the "context" in which the key terms are sought and there is also no contextual information attached to the web resources, so the problem becomes magnified on both sides: in user query representations and in resource descriptions. If the same facetization approach were to be applied here that is used in library classification schema, then probably one could find a suitable solution. Digital libraries could be the first test-bed for the framework, and this could then be extrapolated to the whole range of web resources.

Semantic search and retrieval using the framework

The existing efforts in faceted knowledge organization (faceted search engine, etc.) are based on computer scientists' understanding of the feature-based division of classes into sub-classes. A logical approach adopted by the present work is a model based on well-defined and tested principles of LIS whereby the complete domain of knowledge is organized in the most comprehensive manner using the principles of classification.

Differences in approach begin with the different rendering of "facets" by different groups that are trying to use facets in modeling knowledge. One example is the online directories in which faceted browse/search techniques allow some measure of classmark building, akin to topical labeling. The faceted scheme has to be composed of logical conceptual blocks that relate to the user communities of that domain.[16] A scheme in which most of the blocks are already put together in unchangeable structures does not essentially constitute a faceted scheme.

The analytico-synthetic faceted scheme as enunciated by S. R. Ranganathan is a system in which all of the concepts have been analyzed and sorted and in which they can all be combined into subject representations at generic or specific levels, as required. Here, the application of S. R. Ranganathan's work leads to the expansion of fundamental categories, as listed in some of the semantic categorization attempts, to enable analysis for all the concepts in a discipline, rather than just the commonly occurring ones. Thus a generic faceted framework can be used for knowledge organization in the universe of knowledge, irrespective of the subject domain.

One of the important applications in our classificatory approach is the incorporation of time and space division as common categories/facets. This becomes very significant in applications across domains such as medicine, space research, literature, and others. The idea of common isolates is further extensible to include many common facets that are cross domain, such as forms of resources, languages, etc.

Facetization is adopted by different projects as a basis for the division of knowledge into domains so as to break down the total into bits for easy navigation and assimilation.[17] But the perception of facets is different by different groups. While facets widely stand for shared features among concepts, sometimes a facet corresponds to specific correlates, such as geographical concepts, time concepts (years, days, and time). In general there is an absence of tangible logic at the level of categorization/conceptualization *(Why something is a category)*. Classificatory principles/schema serve as the standard for subject categorization. Categories are derived by applying characteristics or features successively to divide groups of concepts into smaller groups. The members at each level thus have more features in common as the classification/division progresses. This activity translates into building more complex inter-concept relations as the process intensifies or the depth increases. It is akin to the fact that the titles of articles dealing with micro concepts use more words as compared to the titles of monographs; that is, the length of title of micro documents is longer than that of generic topics.

Further, in general systems that claim a faceted approach, relationships are defined on a case-by-case basis, differ in representation, and become linear most of the time (commonly is-a, part-of relations). In faceted classificatory ontologies, however, relationships are defined by rules of facetization, and hence multi-dimensionality of domains is maintained.

End-users are forced to use "facets" in set/rigid paths as the system presents them, in their approach to information. These facets restrict the path that the user can take and do not offer alternatives that users may wish to form. Facets for domains are envisaged as per each domain's extension (breadth) and intension (depth), and hence present all possible pathways at different levels the user may want to pursue.

In the general approach, synthesis is based on the rules of the ontology language attached through axioms, whereas in the classical classificatory model, synthesis is guided by the rules (principles and postulates) of facetization (here the representative UML model) and is standard across all the languages of representation.

S. R. Ranganathan's faceted approach for knowledge organization

Knowledge organization is said to be intrinsically built into the classical approach to classification and organization of resources in libraries. The most impressive result of applying Ranganathan's Colon Classification for documents in a library is the way that documents are arranged on the library shelves to reflect every user's information need in a spectrum from precise domain to peripheral subjects. The role of library classification goes beyond just "finding"; it also maps the domains and displays them logically on the shelves. Colon Classification serves as a working example of the scientific mapping of different subjects and their associated concepts and is a solution to the problem of arranging documents to meet the information needs of every user.

Faceted classification itself is not a particular scheme or system; rather, it is an approach for designing a classification.[18] The methodology of faceted classification provides clear principles for the organization of concepts – first into categories and then into a linear sequence, thus producing predictable and robust structures. While these structures are good at accommodating compound subjects at any level of complexity, their internal logic makes them suitable for use in automated systems.

The philosophy of knowledge organization as envisaged by Ranganathan starts from the premise that each subject in the universe of subjects can be analyzed into facets. A facet is a distinct division of a given domain that contains interrelated concepts which are presented in hierarchies. Facets could be basic facets, isolate facets, and speciators that modify and refine isolates. Facets are postulated to be the manifestation of Five Fundamental Categories, i.e. Personality, Matter, Energy, Space and Time. The interrelationship of facets can be defined using associated concepts, physical attributes, styles and periods, agents, activities, materials, objects, and other such attributes.

According to the theory behind Colon Classification, each subject or domain is made up of facets. A facet is defined as a distinct division of a domain. For example: a domain is made up of Entities (E), entities have Properties (P), and there are Actions by or on the entities. Hence, categories of concepts belong to distinct divisions such as Entity, Property, and Action of the domain. The entities are named as "Personality," Properties as "Matter," actions as "Energy," and also Space (S) and Time (T) concepts are associated across domains. Hence every domain represented as a subject (BS) has facets P, M, E, S, and T in the simplest possible implementation of the theory of faceted classification, and the facet formula (BS),P;M:E'S'T serves as a generic

framework to model the domain. This model becomes more complex as we deal with compound and complex subjects that have cross-domain, intra-facet and inter-facet concepts.[19]

Activity chart for facet analysis of a new concept based on S. R. Ranganathan's facet formula

Example title: "Research in the cure of cancer of the ovary by laser therapy conducted in India in the 1990s":

- Main class is Medicine
  - (Medicine)
- In Medicine, the ovaries are the main concern
  - (Medicine, Ovary)
- The property of the Ovary is that it is afflicted with Cancer
  - (Medicine, Ovary; Cancer)
- The Cancer is being performed (:) on, the intent is curing the cancer (Treatment)
  - (Medicine, Ovary; Cancer: Treatment)
- Cancer is being treated with the matter – Laser Rays
  - (Medicine, Ovary; Cancer: Treatment; Laser Rays)
- And this discussion of treatment is regarding the Research phase
  - Medicine, Ovary; Cancer: Treatment; Laser Rays: Research)
- The Research has been performed within a geographical space (.), India

  - (Medicine, Ovary; Cancer: Treatment; Laser Rays: Research. India)
- During the time (') of 1990
  - (Medicine, Ovary; Cancer: Treatment; Laser Rays: Research. India ' 1990)

After translating into the codes listed for each subject and associated facets, the final output could be translated either into an artificial classification number or into a concept map or a context graph.

Knowledge framework and instances

Architecture of the conceptual framework

Why UML

UML was deemed to be a better choice because it allows descriptions using different design views.[19] Further, it was felt that by using ontologies it is easy to define hierarchies, interrelationships between classes, class attributes, and axioms to specify constraints. And the most useful feature of UML is that it can also be used as a direct ontology representation or a graphical model for most of the ontology representation languages.

UML solves the following problems:

- *No standard for ontology representation language*:[10] Use of UML[20] (class diagrams for static, structural concepts; activity diagrams for behavior) for ontology representation solves this issue, as UML is widely accepted and has a growing audience.
- *Formal ontology specification not human understandable*: Ontology languages like DAML, RDFS, and KIF may be

machine readable, but they are not human understandable outside the domain of ontology experts. UML is easy to understand, due to its graphical nature.

- *Heterogeneous knowledge bases need to interoperate, integrate, reuse*: Existing knowledge bases, as well as ontologies, use different specification languages and thus are not able to reuse each other. UML models solve this problem to a great extent.

Underlying concept of the framework

The present work adopts the facet formula from the classical facet theory of library classification to develop a generic framework for knowledge organization. The framework, being UML based, is useful for building ontologies in different domains based on the facet formula as prescribed in the theory of classification for each domain. The generic model can be used for mapping the subject extension (scope) in terms of its distinct facets. This five-facet model is used only to provide the proof of concept of effectiveness of using facetization. However, different domain experts, as well as classificationists, argue the need for 7, 11 or 13 facets for adequate representation of domain extensions. These arguments are deemed to be outside the focus of discussion of this research.

UML generates visually configurable query/search/filter engines, while the ontologies thus generated lead to the development of mature vocabularies, along with mapping of different facets based on the facet analysis.[21] UML/RDF vocabulary in the backend enables the conceptual modeling constructs, while the ultimate meta-model approach is defined by the semantics of UML.

The advantage of the framework lies in the subject analysis in the process of facetization and its mapping into the facet formula. Since knowledge of facetization and its process cannot be presupposed, the UML framework developed incorporates the knowledge structures for domains based on facetization.

UML mapping tool: ArgoUML[22]

This UML diagramming application was written in Java and released under the open source license; it is available on all the platforms supported by Java. The *Software Development Magazine*'s annual Readers' Choice Award in the "Design and Analysis Tools" category was awarded to ArgoUML in 2003.[23]

The UML standard has not yet been completely implemented by ArgoUML and it also lacks full support for some diagram types.

Previous stable releases

- First published: April 1998
- 0.7.0 (April 1999)
- 0.8.1a (October 2000)
- 0.10.1 (July 2002)
- 0.12 (October 2002)
- 0.14 (August 2003)
- 0.16.1 (August 2004)
- 0.18.1 (April 2005)
- 0.20 (February 2006)
- 0.22 (August 2006)
- 0.24 (February 2007, current).

Main features[20]

- Open source, allows extension or customization
- XMI, SVG (Scalable Vector Graphics) and PGML (Precision Graphics Markup Language) are the supported open standards
- 100 per cent platform independent
- Data Types, Stereotypes and Enumerations: supported
- Supports all 9 UML 1.4 diagrams
- Standard UML 1.4 Metamodel
- Can export diagram types viz. GIF, PNG, PS, EPS, PGML and SVG
- Available in multiple languages
- Diagram editing facility available
- Review of design and improvements possible with built-in design critics
- Extensible modules interface
- Reverse Engineering/JAR (Java ARchive)/class file import
- Diagram types supported
  - Class
  - State
  - Use case
  - Activity
  - Collaboration
  - Deployment
  - Sequence
- Support for Call States, Object Flow States
- AndroMDA compatibility
- Functions support multiple selection of model elements

- Features for support in cognition
- Reflection-in-action
  - "To Do" list
  - User model
- Comprehension and problem solving
  - Multiple explorer perspectives
  - Varied overlapping views

System requirements[20]

- Operating system with Java support
- 10MB disk space
- Pointing device
- Keyboard
- Java 2 JRE or JDK version 1.4 or higher

Installation of ArgoUML

Installation of ArgoUML is fairly simple. One has to decide on the suitability of the available options, depending on the need and system configuration.

Java Web Start

This is useful for occasional users and tests. It requires connection to the ArgoUML home page (*http://www.argouml.org*).

1. One needs to install Java Web Start (*http://java.sun.com/products/javawebstart*).
2. The ArgoUML link from the ArgoUML homepage downloads and chaches locally after the launch.

3. It is automatically started after that and one can also access it from the Java Web Start console on subsequent starts.
4. Whenever the system is connected to the Internet, it checks for the latest version and automatically downloads the updates.

Binary distribution

This is useful for those who use it regularly, and here the version of ArgoUML does not change.

1. The system should be preinstalled with Java 2 JRE.
2. The ArgoUML binary distribution should be downloaded from the ArgoUML homepage (*http://www.argouml.org*).
3. This is followed by creation of an ArgoUML installation directory.
4. The current directory should be changed to the ArgoUML installation directory and ArgoUML should be extracted there.
5. One can start the application by running the argouml.jar file, either through double click on the file, or by executing it from the command line, or also via a batch-file.

ArgoUML: principles and components[20]

ArgoUML works on the following principles, along with various components.

Project, model and diagram

One project is handled at a time by the file operations "save and open"; one project stands for a model plus diagram information; this might in turn contain many model elements

for the complete UML description. The model stored in ArgoUML is not dependent on the contents of the diagrams because all model elements might/might not be present here.

The diagram information, viz. the shapes to represent the various UML models, elements, their location, color, etc. are also contained in the ArgoUML project.

Objects

For the selected objects, right-clicking above an object will activate the functionality of ArgoUML in the menu toolbars/ pop-up menus. Here, the objects are created by the toolbars at the top of all diagrams and these can be added or removed without deleting them from the model.

Overview of the ArgoUML interface: snapshots

ArgoUML's user interface can be categorized into four panels (Figure 7.1):

Top left: Gives a hierarchical view of the current file.

Top right: Serves as editor for class diagram or any other selected part of the project.

Bottom left: Displays the "to do" list of the designer.

Bottom right: Displays the details of the object that has been selected in the diagram or the selected "to do" item.

A menu bar with commands is present at the top of the window. The File menu is used to either store the project or open another project. The upper-left part of the ArgoUML window displays a tree model with diagrams and objects. One can also adapt this view to personal user needs by filtering the displayed objects and their structures.

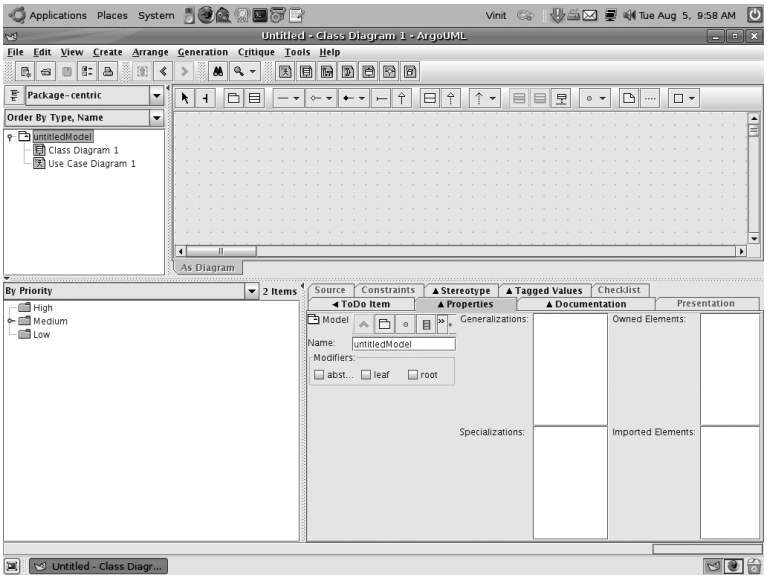

Figure 7.1 ArgoUML interface: main panel

The current diagram is displayed in the upper-right part of ArgoUML, where one can drag and drop the objects in the diagrams and can use the quick-links to create new objects that could be connected to the current objects.

The details of the currently selected object are present in the lower-right part; here, one can select the object and choose the desired details using the tabs.

The lower-left part has the list of "to do" items for the selected model.

ArgoUML allows multiple explorer perspectives where the choice menu at the top sets the current perspective (Figure 7.2). Here each perspective displays a hierarchical view of the design. Using the available explorer perspectives, one can customize and define new perspectives.

The tool opens with a blank design document (Figure 7.3).

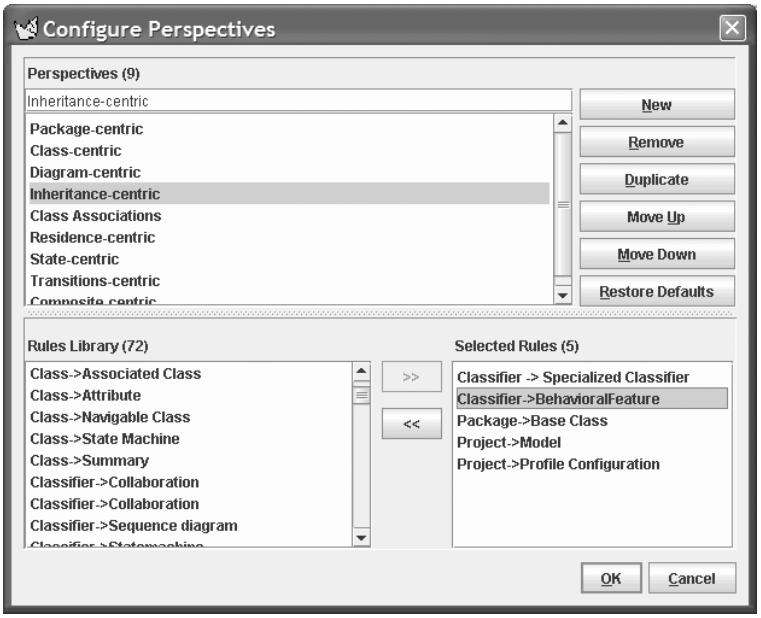

Figure 7.2 ArgoUML interface: explorer

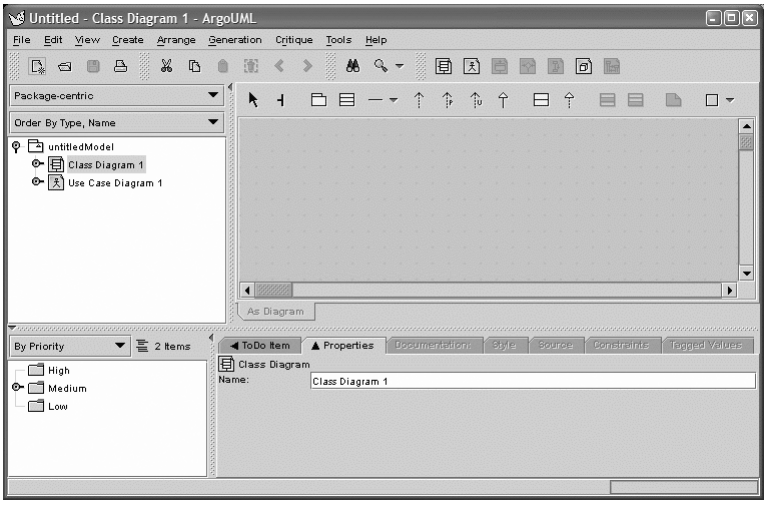

Figure 7.3 ArgoUML interface: design page

One can add classes to the untitled diagram by clicking on the class icon in the toolbar (Figure 7.4). If the toolbar button is double clicked, it "locks" that tool until the arrow tool is selected. Incomplete or problematic parts of the design are indicated by yellow sticky notes and wavy, red underlines. If one places the mouse over a sticky note or wavy line, it displays a blue tool-tip briefly describing the problem.

The name of a class or association can be set by simply selecting it and typing (Figure 7.5).

It can be later edited by double clicking on a name. One can also double click on the other compartments of the class to edit them (Figure 7.6).

The "Properties" tab gives details of the selected model element (Figure 7.7). Associations also have properties (Figure 7.8).

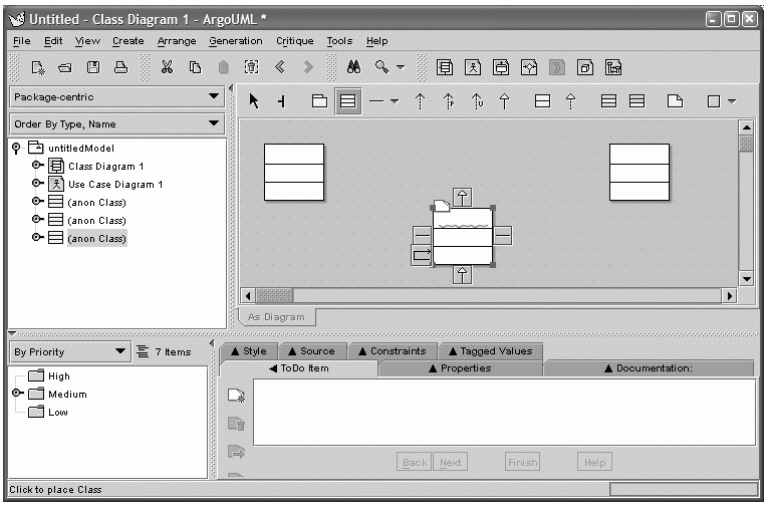

Figure 7.4 ArgoUML interface: adding the classes

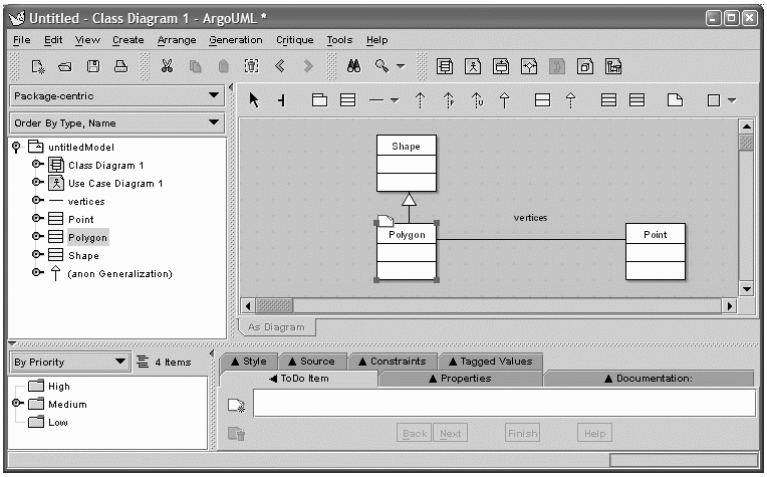

Figure 7.5 ArgoUML interface: selecting classes

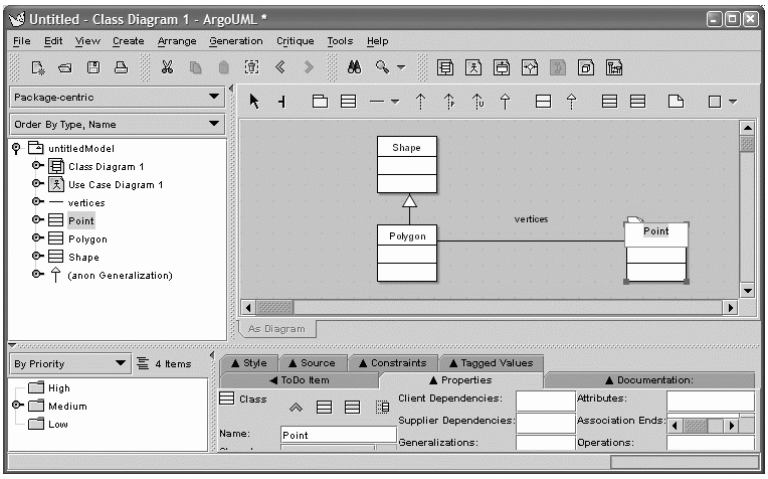

Figure 7.6 ArgoUML interface: direct text editing

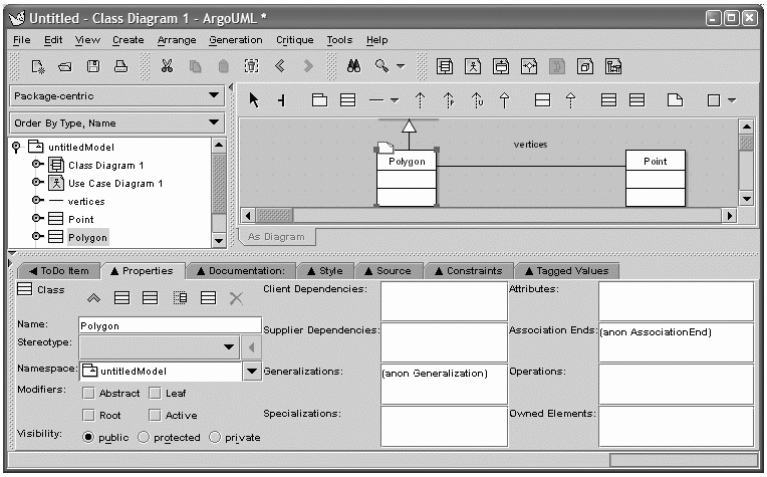

Figure 7.7 ArgoUML interface: defining class properties

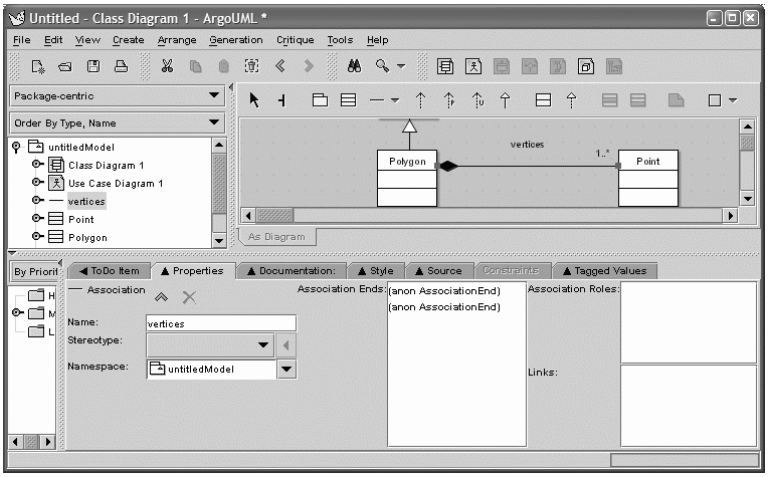

Figure 7.8 ArgoUML interface: defining association properties

Modeled structure for knowledge domains

The typical application of facetization is in knowledge organization that creates a modeled structure for domains. In

classificatory terms, this is akin to the building of classification schedules by a classificationist. We need to study the way that information organization is being practiced in libraries, which strive to organize information resources to cater for the needs of users. The ideal approach is to appreciate and understand the way that information is logically organized in the human mind. The study of reusable knowledge components, called ontologies, is an approach of the same type. Typical applications include semantically marking up web pages using terms from an explicit ontology, and can improve retrieval and help in the integration of data from varied resources. The same concept can be applied in the case of digital libraries. However, the basic anatomy of digital libraries is different from that of traditional counterparts because of their being web based, having content in e-forms of resources, and having an online dissemination model.

Digital libraries today are deployed in domains, for different types of data and applications. The most common issues discussed in relation to digital libraries are the technical architecture, metadata standards, copyright, interoperability and so on. While concentration has mainly been on powerful machines and more disk space to accommodate more information resources, very little effort has been expended on semantic tagging and semantic retrieval of data.

Development of a sample model using the conceptual framework

Sample 1

Figure 7.9 illustrates classification of some facts related to the concept "vehicle." The simple ontological relations here represent "vehicles" as in "automobiles" and in another distinct sense as in "carrier" of disease.

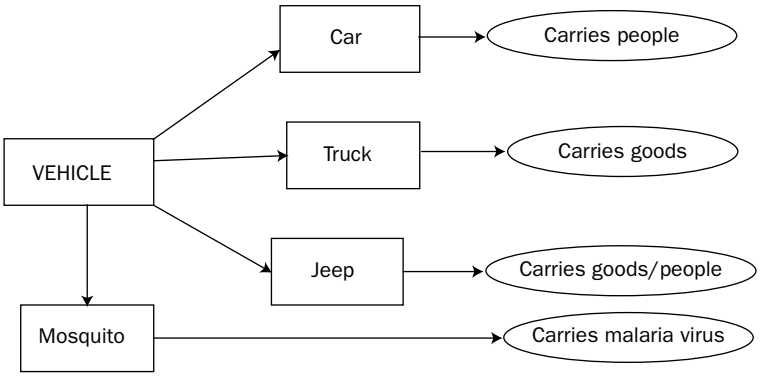

Figure 7.9 **Ontologies to classify concepts**

RDF map for concepts mapped in Figure 7.9

```
<rdf:RDF xmlns:rdf="http://www.w3.
org/1999/02/22-rdf-syntax-ns#"
xmlns:vehicle="http://www.vehicledef.com/what_
namespace">
 <rdf:Description rdf:rdf:about=" #Car">
<vehicle:purpose rdf:rdf:id="purpose">Carries
people</vehicle:purpose> </rdf:Description>

<rdf:Description rdf:rdf:about=" #Truck">
<vehicle:purpose rdf:rdf:id="purpose">Carries
goods</vehicle:purpose> </rdf:Description>

<rdf:Description rdf:rdf:about=" #Jeep">
<vehicle:purpose rdf:rdf:id="purpose">Carries
goods/people</vehicle:purpose> </rdf:
Description>

<rdf:Description rdf:rdf:about=" #Mosquito">
<vehicle:purpose rdf:rdf:id="purpose">Carries
malaria virus</vehicle:purpose> </rdf:
Description> </rdf:RDF>
```

Faceted analysis in the conceptual framework

Figure 7.10 shows an *inheritance hierarchy* – a series of classes and their subclasses, developed after following facet analysis.

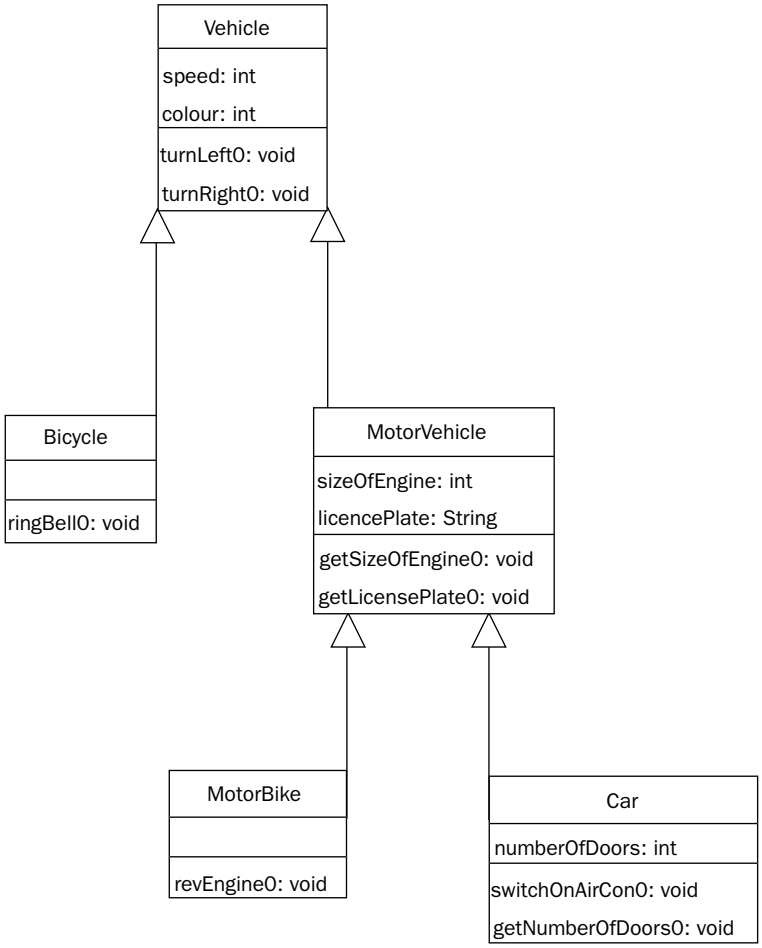

Figure 7.10 Inheritance diagram

- MotorVehicles are the ones that have engine and license plates; for these the attributes and the concerned behavior can be added accordingly.
- As for both MotorBike and Car, MotorVehicles can be the base class. These inherit the speed and color properties from Vehicle, along with the additional attributes and behavior from MotorVehicle.
- The additional attributes and behavior of both MotorBike and Car are specific to that kind of object.

Sample 2

Title of the sample document

Profile of Aparajita, working as Consultant – Knowledge Management at UNAIDS in 2008 (Figure 7.11).

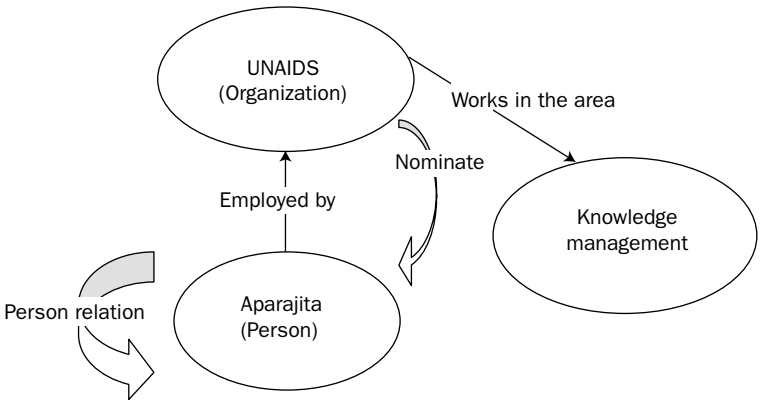

Figure 7.11 **Normal ontology model**

Faceted analysis in the conceptual framework

Facet analysis of the document

Principal entity: Aparajita/Knowledge Management

Matter (Constituent idea): Knowledge Management/ Aparajita

(Document type): Profile

Energy (Action of the domain on the constituent): Consultant

Space (Place/Location facet): UNAIDS

Time: 2008

Components of the UML model based on the facet analysis

Class: Aparajita/Knowledge Management

Responsibilities: Consultant

Composition Associates: Knowledge Management/Aparajita

Common Correlates:

Space: UNAIDS

Time: 2008

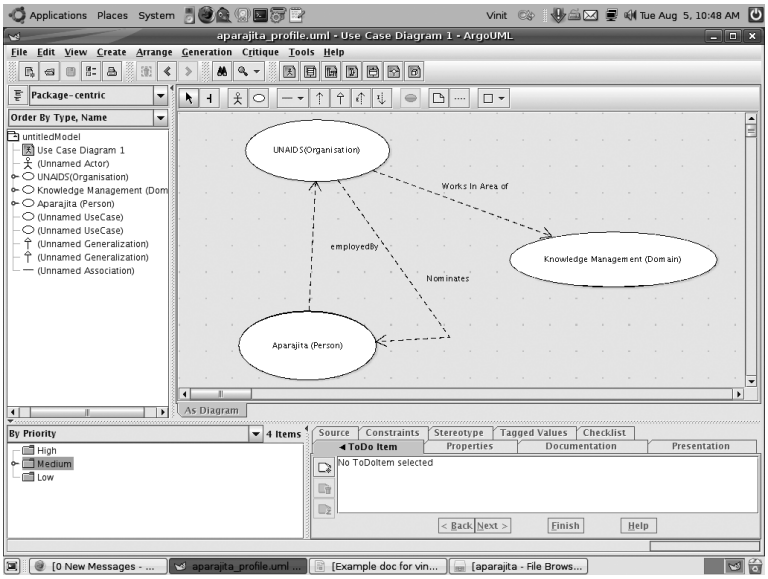

Figure 7.12 Data input screen for the faceted profile

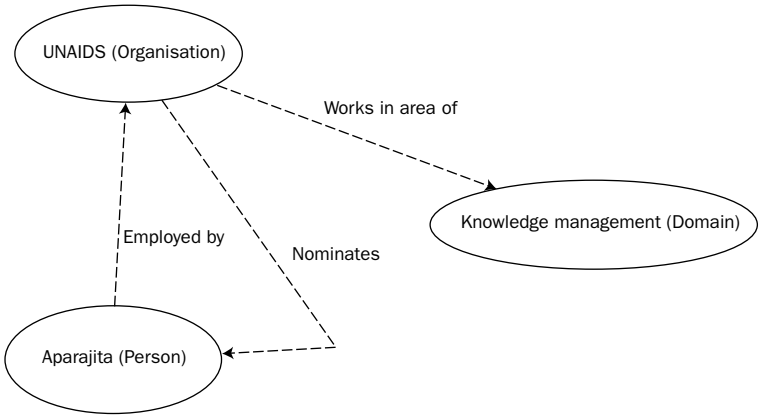

Figure 7.13 Use case diagram magnified

Input in the ArgoUML tool

XMI code for interoperability across various digital libraries in the same domain

```
<?xml version = '1.0' encoding = 'UTF-8' ?>

<XMI xmi.version = '1.2' xmlns:UML = 'org.omg.
xmi.namespace.UML' timestamp = 'Tue Aug 05
10:46:12 IST 2008'>
<XMI.header> <XMI.documentation> <XMI.
exporter>ArgoUML (using Netbeans XMI Writer
version 1.0)</XMI.exporter>

<XMI.exporterVersion>0.24(5) revised on $Date:
2006-11-06 19:55:22 +0100 (Mon, 06 Nov 2006) $
</XMI.exporterVersion> </XMI.documentation>
<XMI.metamodel xmi.name="UML" xmi.
version="1.4"/> </XMI.header>

<XMI.content> <UML:Model xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000077B'
name = 'untitledModel' isSpecification = 'false'
isRoot = 'false' isLeaf = 'false' isAbstract =
'false'> <UML:Namespace.ownedElement>

<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000077C'
name = 'UNAIDS(Organisation)' isSpecification =
'false' isRoot = 'false' isLeaf = 'false'
isAbstract = 'false'> <UML:ModelElement.
clientDependency> <UML:Dependency xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000797'/> <UML:Dependency xmi.
idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:00000000000007A4'/> </UML:ModelElement.
clientDependency> <UML:ModelElement.comment>
<UML:Comment xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000078F'/>
</UML:ModelElement.comment> </UML:UseCase>

<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000785'
```

```
name = 'Knowledge Management (Domain)'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'> <UML:Namespace.
ownedElement> <UML:Dependency xmi.id = '127-0-
1-1--4a4be9c:11b913c1d22:-8000:0000000000000797'
name = 'Works In Area of' isSpecification =
'false'> <UML:Dependency.client> <UML:UseCase
xmi.idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000077C'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000785'/> </UML:Dependency.
supplier> </UML:Dependency> </UML:Namespace.
ownedElement> </UML:UseCase>

<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000786'
name = 'Aparajita (Person)' isSpecification =
'false' isRoot = 'false' isLeaf = 'false'
isAbstract = 'false'> <UML:ModelElement.
clientDependency> <UML:Dependency xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000793'/> </UML:ModelElement.
clientDependency> <UML:GeneralizableElement.
generalization> <UML:Generalization xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:00000000000007A3'/> </UML:
GeneralizableElement.generalization> <UML:
Namespace.ownedElement> <UML:Dependency xmi.id
= '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000793' name = 'employedBy'
isSpecification = 'false'> <UML:Dependency.
client> <UML:UseCase xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000786'/>
</UML:Dependency.client> <UML:Dependency.
supplier> <UML:UseCase xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000077C'/>
</UML:Dependency.supplier> </UML:Dependency>
<UML:Dependency xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A4'
```

```
name = 'Nominates' isSpecification = 'false'>
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000077C'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000786'/> </UML:Dependency.
supplier> </UML:Dependency> </UML:Namespace.
ownedElement> </UML:UseCase>

<UML:Comment xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000078F'
isSpecification = 'false'> <UML:Comment.
annotatedElement> <UML:UseCase xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000077C'/> </UML:Comment.
annotatedElement> </UML:Comment>

<UML:Actor xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000798'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'/> <UML:Association
xmi.id = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000799' name = '' isSpecification
= 'false' isRoot = 'false' isLeaf = 'false'
isAbstract = 'false'> <UML:Association.
connection> <UML:AssociationEnd xmi.id = '127-
0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000079A' visibility = 'public'
isSpecification = 'false' isNavigable = 'true'
ordering = 'unordered' aggregation = 'none'
targetScope = 'instance' changeability =
'changeable'> UML:AssociationEnd.multiplicity>
<UML:Multiplicity xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000079B'>
<UML:Multiplicity.range> <UML:MultiplicityRange
xmi.id = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000079C' lower = '1' upper = '1'/>
</UML:Multiplicity.range> </UML:Multiplicity>
</UML:AssociationEnd.multiplicity>
```

```
<UML:AssociationEnd.participant> <UML:Actor
xmi.idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000798'/> </UML:AssociationEnd.
participant> </UML:AssociationEnd>

<UML:AssociationEnd xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000079D'
visibility = 'public' isSpecification = 'false'
isNavigable = 'true' ordering = 'unordered'
aggregation = 'none' targetScope = 'instance'
changeability = 'changeable'> <UML:
AssociationEnd.multiplicity> <UML:Multiplicity
xmi.id = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000079E'> <UML:Multiplicity.
range> <UML:MultiplicityRange xmi.id = '127-0-
1-1--4a4be9c:11b913c1d22:-8000:000000000000079F'
lower = '1' upper = '1'/> </UML:Multiplicity.
range>
</UML:Multiplicity> </UML:AssociationEnd.
multiplicity>

<UML:AssociationEnd.participant> <UML:UseCase
xmi.idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000786'/> </UML:AssociationEnd.
participant> </UML:AssociationEnd> </UML:
Association.connection> </UML:Association>

<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A0'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'> <UML:
GeneralizableElement.generalization> <UML:
Generalization xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A1'/>
</UML:GeneralizableElement.generalization>
</UML:UseCase>

<UML:Generalization xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A1'
isSpecification = 'false'> <UML:Generalization.
child> <UML:UseCase xmi.idref = '127-0-1-1--
```

```
4a4be9c:11b913c1d22:-8000:00000000000007A0'/>
</UML:Generalization.child> <UML:
Generalization.parent> <UML:UseCase xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000786'/> </UML:Generalization.
parent> </UML:Generalization>

<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A2'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'/> <UML:
Generalization xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A3'
isSpecification = 'false'> <UML:Generalization.
child> <UML:UseCase xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000786'/>
</UML:Generalization.child> <UML:
Generalization.parent> <UML:UseCase xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:00000000000007A2'/> </UML:Generalization.
parent> </UML:Generalization> </UML:Namespace.
ownedElement> </UML:Model> </XMI.content>
</XMI>
```

UML script to facilitate semantics for normal user with the graphical map and machine with the code

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<uml version="5"> <argo version="5">

<documentation> <authorname></authorname>
<authoremail></authoremail> <version>0.24</
version> <description> </description> </
documentation>

<settings> <notationlanguage>UML 1.4</
notationlanguage> <useguillemots>false</
useguillemots> <showvisibility>false</
showvisibility> <showmultiplicity>false</
```

```
showmultiplicity> <showinitialvalue>false</
showinitialvalue> <showproperties>false</
showproperties> <showtypes>true</showtypes>
<showstereotypes>false</showstereotypes> <shows
ingularmultiplicities>true</
showsingularmultiplicities>
<defaultshadowwidth>1</defaultshadowwidth>
</settings>

<searchpath href="PROJECT_DIR" /> <member
type="xmi" name="aparajita_profile.xmi" />
<member type="pgml" name="aparajita_profile_
UseCaseDiagram1.pgml"
diagramclass="org.argouml.uml.diagram.use_case.
ui.UMLUseCaseDiagram" diagramname="Use Case
Diagram 1" /> <member type="todo"
name="aparajita_profile.todo" /> <historyfile
name="" /> </argo>

<XMI xmi.version = '1.2' xmlns:UML = 'org.omg.
xmi.namespace.UML' timestamp = 'Tue Aug 05
10:48:37 IST 2008'>

<XMI.header> <XMI.documentation> <XMI.
exporter>ArgoUML (using Netbeans XMI Writer
version 1.0)</XMI.exporter> <XMI.
exporterVersion>0.24(5) revised on $Date: 2006-
11-06 19:55:22 +0100 (Mon, 06 Nov 2006) $ </
XMI.exporterVersion> </XMI.documentation> <XMI.
metamodel xmi.name="UML" xmi.version="1.4"/></
XMI.header>

<XMI.content> <UML:Model xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000077B'
name = 'untitledModel' isSpecification = 'false'
isRoot = 'false' isLeaf = 'false' isAbstract =
'false'> <UML:Namespace.ownedElement>

<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000077C'
name = 'UNAIDS(Organisation)' isSpecification =
```

```
'false' isRoot = 'false' isLeaf = 'false'
isAbstract = 'false'> <UML:ModelElement.
clientDependency> <UML:Dependency xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000797'/> <UML:Dependency xmi.
idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:00000000000007A4'/> </UML:ModelElement.
clientDependency> <UML:ModelElement.comment>
<UML:Comment xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000078F'/>
</UML:ModelElement.comment> </UML:UseCase>

<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000785'
name = 'Knowledge Management (Domain)'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'> <UML:Namespace.
ownedElement> <UML:Dependency xmi.id = '127-0-
1-1--4a4be9c:11b913c1d22:-8000:0000000000000797'
name = 'Works In Area of' isSpecification =
'false'> <UML:Dependency.client> <UML:UseCase
xmi.idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000077C'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000785'/> </UML:Dependency.
supplier> </UML:Dependency> </UML:Namespace.
ownedElement> </UML:UseCase>

<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000786'
name = 'Aparajita (Person)' isSpecification =
'false' isRoot = 'false' isLeaf = 'false'
isAbstract = 'false'> <UML:ModelElement.
clientDependency> <UML:Dependency xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000793'/> </UML:ModelElement.
clientDependency> <UML:GeneralizableElement.
generalization> <UML:Generalization xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
```

```
8000:00000000000007A3'/> </UML:
GeneralizableElement.generalization> <UML:
Namespace.ownedElement> <UML:Dependency xmi.id
= '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000793' name = 'employedBy'
isSpecification = 'false'> <UML:Dependency.
client> <UML:UseCase xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000786'/>
</UML:Dependency.client> <UML:Dependency.
supplier> <UML:UseCase xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000077C'/>
</UML:Dependency.supplier> </UML:Dependency>
<UML:Dependency xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A4'
name = 'Nominates' isSpecification = 'false'>
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000077C'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000786'/> </UML:Dependency.
supplier> </UML:Dependency> </UML:Namespace.
ownedElement> </UML:UseCase>

<UML:Comment xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000078F'
isSpecification = 'false'> <UML:Comment.
annotatedElement> <UML:UseCase xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000077C'/> </UML:Comment.
annotatedElement> </UML:Comment>

<UML:Actor xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000798'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'/> <UML:Association
xmi.id = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000799' name = '' isSpecification
= 'false' isRoot = 'false' isLeaf = 'false'
isAbstract = 'false'> <UML:Association.
```

```
connection> <UML:AssociationEnd xmi.id = '127-
0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000079A' visibility = 'public'
isSpecification = 'false' isNavigable = 'true'
ordering = 'unordered' aggregation = 'none'
targetScope = 'instance' changeability =
'changeable'> <UML:AssociationEnd.multiplicity>
<UML:Multiplicity xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000079B'>
<UML:Multiplicity.range> <UML:MultiplicityRange
xmi.id = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000079C' lower = '1' upper = '1'/>
</UML:Multiplicity.range> </UML:Multiplicity>
</UML:AssociationEnd.multiplicity> <UML:
AssociationEnd.participant> <UML:Actor xmi.
idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000798'/> </UML:AssociationEnd.
participant> </UML:AssociationEnd>

<UML:AssociationEnd xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000079D'
visibility = 'public' isSpecification = 'false'
isNavigable = 'true' ordering = 'unordered'
aggregation = 'none' targetScope = 'instance'
changeability = 'changeable'> <UML:
AssociationEnd.multiplicity> <UML:Multiplicity
xmi.id = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000079E'> <UML:Multiplicity.
range> <UML:MultiplicityRange xmi.id = '127-0-
1-1--4a4be9c:11b913c1d22:-8000:000000000000079F'
lower = '1' upper = '1'/> </UML:Multiplicity.
range>
</UML:Multiplicity> </UML:AssociationEnd.
multiplicity>

<UML:AssociationEnd.participant> <UML:UseCase
xmi.idref = '127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000786'/> </UML:AssociationEnd.
participant> </UML:AssociationEnd> </UML:
Association.connection> </UML:Association>
```

```
<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A0'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'> <UML:
GeneralizableElement.generalization> <UML:
Generalization xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A1'/>
</UML:GeneralizableElement.generalization>
</UML:UseCase>

<UML:Generalization xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A1'
isSpecification = 'false'> <UML:Generalization.
child> <UML:UseCase xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A0'/>
</UML:Generalization.child> <UML:
Generalization.parent> <UML:UseCase xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000786'/> </UML:Generalization.
parent> </UML:Generalization>

<UML:UseCase xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A2'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'/> <UML:
Generalization xmi.id = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A3'
isSpecification = 'false'> <UML:Generalization.
child> <UML:UseCase xmi.idref = '127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000786'/>
</UML:Generalization.child> <UML:
Generalization.parent> <UML:UseCase xmi.idref =
'127-0-1-1--4a4be9c:11b913c1d22:-
8000:00000000000007A2'/> </UML:Generalization.
parent> </UML:Generalization>

</UML:Namespace.ownedElement> </UML:Model>
</XMI.content> </XMI>

<pgml description="org.argouml.uml.diagram.use_
case.ui.UMLUseCaseDiagram|127-0-1-1--
```

```
4a4be9c:11b913c1d22:-8000:000000000000077B"
name="Use Case Diagram 1" >

<group name="Fig0" description="org.argouml.
uml.diagram.use_case.ui.FigUseCase[80, 8, 176,
76]pathVisible=false;extensionPointVisible=fals
e" href="127-0-1-1--4a4be9c:11b913c1d22:-
8000:000000000000077C" fill="1"
fillcolor="white" stroke="1"
strokecolor="black" >

<private> </private>

<ellipse name="Fig0.0" x="168" y="46" rx="88"
ry="38" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<ellipse name="Fig0.1" x="168" y="46" rx="88"
ry="38" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<text name="Fig0.2" x="111" y="36" fill="0"
fillcolor="white" stroke="0"
strokecolor="black" font="Dialog" textsize="10"
>UNAIDS(Organisation)</text>

<group name="Fig0.3" description="org.argouml.
uml.diagram.ui.FigStereotypesCompartment[168,
88, 0, 0]" href="127-0-1-1--
4a4be9c:11b913c1d22:-8000:000000000000077C"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" >

<private> </private>

<rectangle name="Fig0.3.0" x="168" y="88"
width="0" height="0" fill="0" fillcolor="white"
stroke="0" strokecolor="black" /> </group>

<path name="Fig0.4"
description="org.tigris.gef.presentation.
FigLine" fill="1" fillcolor="white" stroke="1"
strokecolor="black" visibility="0" > <moveto
```

```
x="80" y="54" /> <lineto x="190" y="54" />
</path>

<group name="Fig0.5" description="org.tigris.
gef.presentation.FigGroup[80, 24, 0, 0]"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" visibility="0" >

<private> </private>

<rectangle name="Fig0.5.0" x="133" y="57"
width="4" height="4" fill="0" fillcolor="white"
stroke="0" strokecolor="black" visibility="0"
/> </group> </group>

<group name="Fig1" description="org.argouml.
uml.diagram.use_case.ui.FigUseCase[400, 144,
252, 68]pathVisible=false;extensionPointVisible
=false" href="127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000785" fill="1"
fillcolor="white" stroke="1"
strokecolor="black" >

<private> </private>

<ellipse name="Fig1.0" x="526" y="178" rx="126"
ry="34" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<ellipse name="Fig1.1" x="526" y="178" rx="126"
ry="34" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<text name="Fig1.2" x="438" y="168" fill="0"
fillcolor="white" stroke="0"
strokecolor="black" font="Dialog" textsize="10"
>Knowledge Management (Domain)</text>

<group name="Fig1.3" description="org.argouml.
uml.diagram.ui.FigStereotypesCompartment[526,
216, 0, 0]" href="127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000785"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" >
```

```
<private> </private>

<rectangle name="Fig1.3.0" x="526" y="216"
width="0" height="0" fill="0" fillcolor="white"
stroke="0" strokecolor="black" /> </group>

<path name="Fig1.4"
description="org.tigris.gef.presentation.
FigLine" fill="1" fillcolor="white" stroke="1"
strokecolor="black" visibility="0" > <moveto
x="400" y="182" /> <lineto x="510" y="182" />
</path>

<group name="Fig1.5" description="org.tigris.
gef.presentation.FigGroup[400, 152, 0, 0]"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" visibility="0" >

<private> </private>

<rectangle name="Fig1.5.0" x="453" y="185"
width="4" height="4" fill="0" fillcolor="white"
stroke="0" strokecolor="black" visibility="0" />
</group> </group>

<group name="Fig2" description="org.argouml.
uml.diagram.use_case.ui.FigUseCase[72, 240,
168, 84]pathVisible=false;extensionPointVisible
=false" href="127-0-1-1--4a4be9c:11b913c1d22:-
8000:0000000000000786" fill="1"
fillcolor="white" stroke="1"
strokecolor="black" >

<private> </private>

<ellipse name="Fig2.0" x="156" y="282" rx="84"
ry="42" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<ellipse name="Fig2.1" x="156" y="282" rx="84"
ry="42" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />
```

```
<text name="Fig2.2" x="109" y="272" fill="0"
fillcolor="white" stroke="0"
strokecolor="black" font="Dialog" textsize="10"
>Aparajita (Person)</text>

<group name="Fig2.3" description="org.argouml.
uml.diagram.ui.FigStereotypesCompartment[156,
328, 0, 0]" href="127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000786"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" >

<private> </private>

<rectangle name="Fig2.3.0" x="156" y="328"
width="0" height="0" fill="0" fillcolor="white"
stroke="0" strokecolor="black" /> </group>

<path name="Fig2.4"
description="org.tigris.gef.presentation.
FigLine" fill="1" fillcolor="white" stroke="1"
strokecolor="black" visibility="0" > <moveto
x="80" y="294" /> <lineto x="190" y="294" />
</path>

<group name="Fig2.5" description="org.tigris.
gef.presentation.FigGroup[80, 264, 0, 0]"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" visibility="0" >

<private> </private>

<rectangle name="Fig2.5.0" x="133" y="297"
width="4" height="4" fill="0" fillcolor="white"
stroke="0" strokecolor="black" visibility="0"
/> </group> </group>

<group name="Fig3" description="org.argouml.
uml.diagram.ui.FigDependency" href="127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000793"
stroke="1" strokecolor="black" >
```

```
<private> sourcePortFig="Fig2.0"
destPortFig="Fig0.0" sourceFigNode="Fig2"
destFigNode="Fig0" </private>

<path name="Fig3.0"
description="org.tigris.gef.presentation.
FigPoly" fill="0" fillcolor="white" stroke="1"
strokecolor="black" > <moveto x="158" y="240"
/> <lineto x="166" y="83" /> </path> </group>

<group name="Fig4" description="org.argouml.
uml.diagram.ui.FigDependency" href="127-0-1-1--
4a4be9c:11b913c1d22:-8000:0000000000000797"
stroke="1" strokecolor="black" >

<private> sourcePortFig="Fig0.0"
destPortFig="Fig1.0" sourceFigNode="Fig0"
destFigNode="Fig1" </private>

<path name="Fig4.0"
description="org.tigris.gef.presentation.
FigPoly" fill="0" fillcolor="white" stroke="1"
strokecolor="black" > <moveto x="235" y="70" />
<lineto x="451" y="150" /> </path> </group>

<group name="Fig5" description="org.argouml.
uml.diagram.ui.FigDependency" href="127-0-1-1--
4a4be9c:11b913c1d22:-8000:00000000000007A4"
stroke="1" strokecolor="black" >

<private> sourcePortFig="Fig0.0"
destPortFig="Fig2.0" sourceFigNode="Fig0"
destFigNode="Fig2" </private>

<path name="Fig5.0"
description="org.tigris.gef.presentation.
FigPoly" fill="0" fillcolor="white" stroke="1"
strokecolor="black" > <moveto x="193" y="82" />
<lineto x="328" y="272" /> <lineto x="239"
y="277" /> </path> </group> </pgml>

<todo> <todolist> </todolist> <resolvedcritics>
</resolvedcritics> </todo> </uml>
```

Sample 3

Title of a resource

"Treatment of Respiratory Disease in US children according to the Ayurvedic System in 2006." There are many concepts from the domain of medicine, time, and systems of medicine intermingled in this topic.

Common ontology models

To represent the extension of the given topic the title is first analyzed into its parts. The parts (or the concepts that they represent) are then expressed as being related to other concepts through some relations. Typical models of the above title translate to facts as shown in Figure 7.14.

Information that is encoded is therefore:

USA is a country (Space)

2006 is a year (Time)

Respiratory disease is a disease

Children are types of humans

Children are also US citizens

The limitation of such a representation is that concepts are expressed as facts but clear context is missing. Also, it is difficult to put in the perspective of time/location frames as commons for all concepts, not only particular instances. Another problem is that of defining the core theme or the central idea that is to be represented, around which the other concepts should be relationally arranged. In the model in Figure 7.14, all concepts and their relations with others are treated equally. But the relationship model is not that simple or uni-dimensional because the interrelationships of

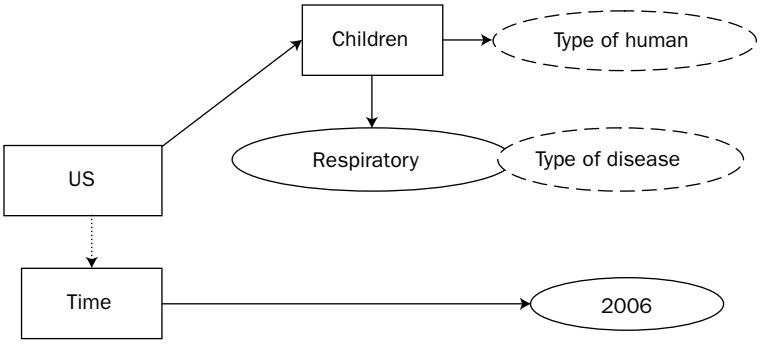

Figure 7.14 Common ontology map for the given title

facets and concepts within are complex and multi-dimensional. Further, in these models facets cannot be arranged in order of priority.

Faceted analysis in the conceptual framework

The subjects are first analyzed into personalities or entities that form the centrality of the idea in the domain being discussed. The other facets are in relation to those personalities or entities. The logic used is "disease" is a "property" of the human body and "treatment" or "therapy" is an action on entity. Hence, personality or entity is the central idea around which other concepts are arranged in their respective relations. Following this method, the facet analysis stage yields:

Main facets: Medicine, Child, Respiratory organ, Disease, Therapy, Time-Space

Basic Domain: Medicine Particular focus: Child medicine (pediatrics)

Matter/Property: Disease Particular focus: Respiratory disease

Action/Energy: Treatment normalized/standardized to "Therapy"

Space: USA

Time: 2006

Hence at the aggregation level it is the concept "Human Body and its parts" grouped under "Personality/Entities" that forms the centrality where "disease" is "carried by," "treated," or "diagnosed" that form the "Energy/Action" facets. Time and space form common isolates to set the spatial, physical, temporal constraints.

Input in the ArgoUML tool

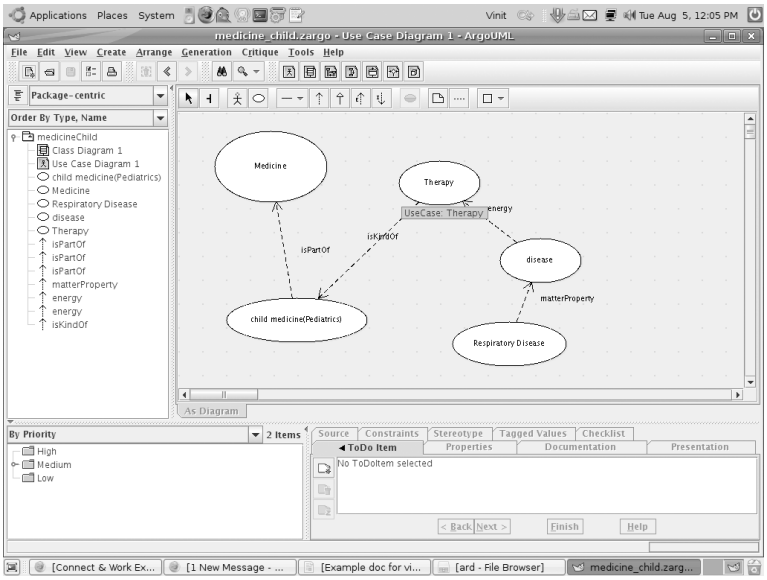

Figure 7.15 Data input screen for the faceted profile

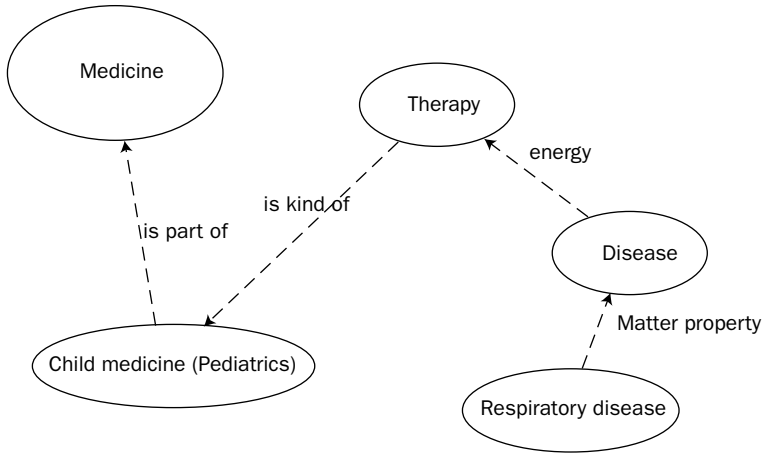

Figure 7.16 **Magnified UML map**

XMI code for interoperability across various digital libraries in the same domain

```
<?xml version = '1.0' encoding = 'UTF-8' ?> <XMI
xmi.version = '1.2' xmlns:UML = 'org.omg.xmi.
namespace.UML' timestamp = 'Tue Aug 05 12:06:31
IST 2008'>

<XMI.header> <XMI.documentation> <XMI.
exporter>ArgoUML (using Netbeans XMI Writer
version 1.0)</XMI.exporter> <XMI.
exporterVersion>0.24(5) revised on $Date: 2006-
11-06 19:55:22 +0100 (Mon, 06 Nov 2006) $ </
XMI.exporterVersion> </XMI.documentation> <XMI.
metamodel xmi.name="UML" xmi.version="1.4"/></
XMI.header>

<XMI.content> <UML:Model xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000077B'
name = 'medicineChild' isSpecification = 'false'
isRoot = 'false' isLeaf = 'false' isAbstract =
'false'> <UML:Namespace.ownedElement>
```

```
<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000782'
name = 'child medicine(Pediatrics)'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'> <UML:
ModelElement.clientDependency> <UML:Dependency
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:000000000000078F'/> </UML:ModelElement.
clientDependency> </UML:UseCase>

<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000078D'
name = 'Medicine' isSpecification = 'false'
isRoot = 'false' isLeaf = 'false' isAbstract =
'false'> <UML:ModelElement.clientDependency>
<UML:Dependency xmi.idref = '127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000078E'/>
</UML:ModelElement.clientDependency> </UML:
UseCase>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000078E'
name = 'isPartOf' isSpecification = 'false'>
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1-34bace4b:11b91863d86:-
8000:000000000000078D'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000782'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000078F'
name = 'isPartOf' isSpecification = 'false'>
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000782'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:000000000000078D'/> </UML:Dependency.
supplier> </UML:Dependency>
```

```
<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000790'
name = 'Respiratory Disease' isSpecification =
'false' isRoot = 'false' isLeaf = 'false'
isAbstract = 'false'> <UML:ModelElement.
clientDependency> <UML:Dependency xmi.idref =
'127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000791'/> <UML:Dependency xmi.
idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000793'/> </UML:ModelElement.
clientDependency> </UML:UseCase>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000791'
name = 'isPartOf' isSpecification = 'false'>
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000790'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000782'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000792'
name = 'disease' isSpecification = 'false' isRoot
= 'false' isLeaf = 'false' isAbstract = 'false'>
<UML:ModelElement.clientDependency> <UML:
Dependency xmi.idref = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000796'/>
</UML:ModelElement.clientDependency> </UML:
UseCase>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000793'
name = 'matterProperty' isSpecification =
'false'> <UML:Dependency.client> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000790'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
```

```
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000792'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000794'
name = 'Therapy' isSpecification = 'false' isRoot
= 'false' isLeaf = 'false' isAbstract = 'false'>
<UML:ModelElement.clientDependency> <UML:
Dependency xmi.idref = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000795'/>
<UML:Dependency xmi.idref = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000797'/>
</UML:ModelElement.clientDependency> </UML:
UseCase>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000795'
name = 'energy' isSpecification = 'false'> <UML:
Dependency.client> <UML:UseCase xmi.idref =
'127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000794'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000792'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000796'
name = 'energy' isSpecification = 'false'> <UML:
Dependency.client> <UML:UseCase xmi.idref =
'127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000792'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000794'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000797'
name = 'isKindOf' isSpecification = 'false'>
```

```
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000794'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000782'/> </UML:Dependency.
supplier> </UML:Dependency>

</UML:Namespace.ownedElement> </UML:Model> </
XMI.content> </XMI>
```

UML script to facilitate semantics for normal user with graphical map and machine with the code

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<uml version="5"> <argo version="5">

<documentation> <authorname></authorname>
<authoremail></authoremail> <version>0.24</
version> <description> </description> </
documentation>

<settings> <notationlanguage>UML 1.4</
notationlanguage> <useguillemots>false</
useguillemots> <showvisibility>false</
showvisibility> <showmultiplicity>false</
showmultiplicity> <showinitialvalue>false</
showinitialvalue> <showproperties>false</
showproperties> <showtypes>true</showtypes>
<showstereotypes>false</showstereotypes> <shows
ingularmultiplicities>true</
showsingularmultiplicities>
<defaultshadowwidth>1</defaultshadowwidth> </
settings>

<searchpath href="PROJECT_DIR" /> <member
type="xmi" name="medicine_child.xmi" /> <member
type="pgml" name="medicine_child_ClassDiagram1.
pgml" diagramclass="org.argouml.uml.diagram.
static_structure.ui.UMLClassDiagram"
diagramname="Class Diagram 1" /> <member
```

```
type="pgml" name="medicine_child_
UseCaseDiagram1.pgml"
diagramclass="org.argouml.uml.diagram.use_case.
ui.UMLUseCaseDiagram" diagramname="Use Case
Diagram 1" /> <member type="todo"
name="medicine_child.todo" /> <historyfile
name="" /> </argo> <XMI xmi.version = '1.2'
xmlns:UML = 'org.omg.xmi.namespace.UML'
timestamp = 'Tue Aug 05 12:07:37 IST 2008'>

<XMI.header> <XMI.documentation> <XMI.
exporter>ArgoUML (using Netbeans XMI Writer
version 1.0)</XMI.exporter> <XMI.
exporterVersion>0.24(5) revised on $Date: 2006-
11-06 19:55:22 +0100 (Mon, 06 Nov 2006) $ </
XMI.exporterVersion> </XMI.documentation> <XMI.
metamodel xmi.name="UML" xmi.version="1.4"/></
XMI.header>

<XMI.content> <UML:Model xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000077B'
name = 'medicineChild' isSpecification = 'false'
isRoot = 'false' isLeaf = 'false' isAbstract =
'false'> <UML:Namespace.ownedElement>

<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000782'
name = 'child medicine(Pediatrics)'
isSpecification = 'false' isRoot = 'false' isLeaf
= 'false' isAbstract = 'false'> <UML:
ModelElement.clientDependency> <UML:Dependency
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:000000000000078F'/> </UML:ModelElement.
clientDependency> </UML:UseCase>

<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000078D'
name = 'Medicine' isSpecification = 'false'
isRoot = 'false' isLeaf = 'false' isAbstract =
'false'> <UML:ModelElement.clientDependency>
<UML:Dependency xmi.idref = '127-0-1-1-
```

```
34bace4b:11b91863d86:-8000:000000000000078E'/>
</UML:ModelElement.clientDependency> </UML:
UseCase>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000078E'
name = 'isPartOf' isSpecification = 'false'>
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1-34bace4b:11b91863d86:-
8000:000000000000078D'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000782'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000078F'
name = 'isPartOf' isSpecification = 'false'>
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000782'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:000000000000078D'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000790'
name = 'Respiratory Disease' isSpecification =
'false' isRoot = 'false' isLeaf = 'false'
isAbstract = 'false'> <UML:ModelElement.
clientDependency> <UML:Dependency xmi.idref =
'127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000791'/> <UML:Dependency xmi.
idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000793'/> </UML:ModelElement.
clientDependency> </UML:UseCase>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000791'
name = 'isPartOf' isSpecification = 'false'>
```

```
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000790'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000782'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000792'
name = 'disease' isSpecification = 'false' isRoot
= 'false' isLeaf = 'false' isAbstract = 'false'>
<UML:ModelElement.clientDependency> <UML:
Dependency xmi.idref = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000796'/>
</UML:ModelElement.clientDependency> </UML:
UseCase>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000793'
name = 'matterProperty' isSpecification =
'false'> <UML:Dependency.client> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000790'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000792'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:UseCase xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000794'
name = 'Therapy' isSpecification = 'false' isRoot
= 'false' isLeaf = 'false' isAbstract = 'false'>
<UML:ModelElement.clientDependency> <UML:
Dependency xmi.idref = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000795'/>
<UML:Dependency xmi.idref = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000797'/>
</UML:ModelElement.clientDependency> </UML:
UseCase>
```

```
<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000795'
name = 'energy' isSpecification = 'false'> <UML:
Dependency.client> <UML:UseCase xmi.idref =
'127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000794'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000792'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000796'
name = 'energy' isSpecification = 'false'> <UML:
Dependency.client> <UML:UseCase xmi.idref =
'127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000792'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000794'/> </UML:Dependency.
supplier> </UML:Dependency>

<UML:Dependency xmi.id = '127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000797'
name = 'isKindOf' isSpecification = 'false'>
<UML:Dependency.client> <UML:UseCase xmi.idref
= '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000794'/> </UML:Dependency.
client> <UML:Dependency.supplier> <UML:UseCase
xmi.idref = '127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000782'/> </UML:Dependency.
supplier></UML:Dependency>

</UML:Namespace.ownedElement></UML:Model></XMI.
content></XMI>

<pgml description="org.argouml.uml.diagram.
static_structure.ui.UMLClassDiagram|127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000077B"
name="Class Diagram 1"></pgml>
```

```
<pgml description="org.argouml.uml.diagram.use_
case.ui.UMLUseCaseDiagram|127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000077B"
name="Use Case Diagram 1" >

<group name="Fig0" description="org.argouml.
uml.diagram.use_case.ui.FigUseCase[64, 248,
190, 60]pathVisible=false;extensionPointVisible
=false" href="127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000782" fill="1"
fillcolor="white" stroke="1"
strokecolor="black" >

<private> </private>

<ellipse name="Fig0.0" x="159" y="278" rx="95"
ry="30" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<ellipse name="Fig0.1" x="159" y="278" rx="95"
ry="30" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<text name="Fig0.2" x="93" y="268" fill="0"
fillcolor="white" stroke="0"
strokecolor="black" font="Dialog" textsize="10"
>child medicine(Pediatrics)</text>

<group name="Fig0.3" description="org.argouml.
uml.diagram.ui.FigStereotypesCompartment[159,
312, 0, 0]" href="127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000782"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" >

<private> </private>

<rectangle name="Fig0.3.0" x="159" y="312"
width="0" height="0" fill="0" fillcolor="white"
stroke="0" strokecolor="black" /> </group>

<path name="Fig0.4"
description="org.tigris.gef.presentation.
FigLine" fill="1" fillcolor="white" stroke="1"
```

```
strokecolor="black" visibility="0" > <moveto
x="64" y="278" /> <lineto x="174" y="278" /> </
path>

<group name="Fig0.5" description="org.tigris.
gef.presentation.FigGroup[64, 248, 0, 0]"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" visibility="0" >

<private> </private>

<rectangle name="Fig0.5.0" x="117" y="281"
width="4" height="4" fill="0" fillcolor="white"
stroke="0" strokecolor="black" visibility="0"
/> </group> </group>

<group name="Fig1" description="org.argouml.
uml.diagram.use_case.ui.FigUseCase[48, 24, 152,
96]pathVisible=false;extensionPointVisible=fals
e" href="127-0-1-1-34bace4b:11b91863d86:-
8000:000000000000078D" fill="1"
fillcolor="white" stroke="1"
strokecolor="black" >

<private></private>

<ellipse name="Fig1.0" x="124" y="72" rx="76"
ry="48" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<ellipse name="Fig1.1" x="124" y="72" rx="76"
ry="48" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<text name="Fig1.2" x="98" y="62" fill="0"
fillcolor="white" stroke="0"
strokecolor="black" font="Dialog" textsize="10"
>Medicine</text>

<group name="Fig1.3" description="org.argouml.
uml.diagram.ui.FigStereotypesCompartment[124,
124, 0, 0]" href="127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000078D"
```

```
fill="0" fillcolor="white" stroke="0"
strokecolor="black" >

<private></private>

<rectangle name="Fig1.3.0" x="124" y="124"
width="0" height="0" fill="0" fillcolor="white"
stroke="0" strokecolor="black" /> </group>

<path name="Fig1.4"
description="org.tigris.gef.presentation.
FigLine" fill="1" fillcolor="white" stroke="1"
strokecolor="black" visibility="0" > <moveto
x="48" y="54" /> <lineto x="158" y="54" /> </
path>

<group name="Fig1.5" description="org.tigris.
gef.presentation.FigGroup[48, 24, 0, 0]"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" visibility="0" >

<private> </private>

<rectangle name="Fig1.5.0" x="101" y="57"
width="4" height="4" fill="0" fillcolor="white"
stroke="0" strokecolor="black" visibility="0"
/> </group> </group>

<group name="Fig2" description="org.argouml.
uml.diagram.ui.FigDependency" href="127-0-1-1-
34bace4b:11b91863d86:-8000:000000000000078F"
stroke="1" strokecolor="black" >

<private> sourcePortFig="Fig0.0"
destPortFig="Fig1.0" sourceFigNode="Fig0"
destFigNode="Fig1" </private>

<path name="Fig2.0"
description="org.tigris.gef.presentation.
FigPoly" fill="0" fillcolor="white" stroke="1"
strokecolor="black" > <moveto x="153" y="248"
/> <lineto x="131" y="119" /> </path> </group>
```

```
<group name="Fig3" description="org.argouml.
uml.diagram.use_case.ui.FigUseCase[368, 280,
154, 60]pathVisible=false;extensionPointVisible
=false" href="127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000790" fill="1"
fillcolor="white" stroke="1"
strokecolor="black" >

<private></private>

<ellipse name="Fig3.0" x="445" y="310" rx="77"
ry="30" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<ellipse name="Fig3.1" x="445" y="310" rx="77"
ry="30" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<text name="Fig3.2" x="393" y="300" fill="0"
fillcolor="white" stroke="0"
strokecolor="black" font="Dialog" textsize="10"
>Respiratory Disease</text>

<group name="Fig3.3" description="org.argouml.
uml.diagram.ui.FigStereotypesCompartment[445,
344, 0, 0]" href="127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000790"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" >

<private></private>

<rectangle name="Fig3.3.0" x="445" y="344"
width="0" height="0" fill="0" fillcolor="white"
stroke="0" strokecolor="black" /> </group>

<path name="Fig3.4"
description="org.tigris.gef.presentation.
FigLine" fill="1" fillcolor="white" stroke="1"
strokecolor="black" visibility="0" > <moveto
x="368" y="310" /> <lineto x="478" y="310" />
</path>
```

```
<group name="Fig3.5" description="org.tigris.
gef.presentation.FigGroup[368, 280, 0, 0]"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" visibility="0" >

<private> </private>

<rectangle name="Fig3.5.0" x="421" y="313"
width="4" height="4" fill="0" fillcolor="white"
stroke="0" strokecolor="black" visibility="0"
/> </group> </group>

<group name="Fig4" description="org.argouml.
uml.diagram.use_case.ui.FigUseCase[432, 168,
110, 60]pathVisible=false;extensionPointVisible
=false" href="127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000792" fill="1"
fillcolor="white" stroke="1"
strokecolor="black" >

<private> </private>

<ellipse name="Fig4.0" x="487" y="198" rx="55"
ry="30" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<ellipse name="Fig4.1" x="487" y="198" rx="55"
ry="30" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<text name="Fig4.2" x="464" y="188" fill="0"
fillcolor="white" stroke="0"
strokecolor="black" font="Dialog" textsize="10"
>disease</text>

<group name="Fig4.3" description="org.argouml.
uml.diagram.ui.FigStereotypesCompartment[487,
232, 0, 0]" href="127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000792"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" >

<private> </private>
```

```
<rectangle name="Fig4.3.0" x="487" y="232"
width="0" height="0" fill="0" fillcolor="white"
stroke="0" strokecolor="black" /> </group>

<path name="Fig4.4"
description="org.tigris.gef.presentation.
FigLine" fill="1" fillcolor="white" stroke="1"
strokecolor="black" visibility="0" > <moveto
x="432" y="198" /> <lineto x="542" y="198" />
</path>

<group name="Fig4.5" description="org.tigris.
gef.presentation.FigGroup[432, 168, 0, 0]"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" visibility="0" >

<private> </private>

<rectangle name="Fig4.5.0" x="485" y="201"
width="4" height="4" fill="0" fillcolor="white"
stroke="0" strokecolor="black" visibility="0"
/> </group> </group>

<group name="Fig5" description="org.argouml.
uml.diagram.ui.FigDependency" href="127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000793"
stroke="1" strokecolor="black" >

<private> sourcePortFig="Fig3.0"
destPortFig="Fig4.0" sourceFigNode="Fig3"
destFigNode="Fig4" </private>

<path name="Fig5.0"
description="org.tigris.gef.presentation.
FigPoly" fill="0" fillcolor="white" stroke="1"
strokecolor="black" > <moveto x="455" y="280"
/> <lineto x="475" y="227" /> </path> </group>

<group name="Fig6" description="org.argouml.
uml.diagram.use_case.ui.FigUseCase[296, 64,
110, 60]pathVisible=false;extensionPointVisible
=false" href="127-0-1-1-34bace4b:11b91863d86:-
8000:0000000000000794" fill="1"
```

```
fillcolor="white" stroke="1"
strokecolor="black" >

<private> </private>

<ellipse name="Fig6.0" x="351" y="94" rx="55"
ry="30" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<ellipse name="Fig6.1" x="351" y="94" rx="55"
ry="30" fill="1" fillcolor="white" stroke="1"
strokecolor="black" />

<text name="Fig6.2" x="327" y="84" fill="0"
fillcolor="white" stroke="0"
strokecolor="black" font="Dialog" textsize="10"
>Therapy</text>

<group name="Fig6.3" description="org.argouml.
uml.diagram.ui.FigStereotypesCompartment[351,
128, 0, 0]" href="127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000794"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" >

<private> </private>

<rectangle name="Fig6.3.0" x="351" y="128"
width="0" height="0" fill="0" fillcolor="white"
stroke="0" strokecolor="black" /> </group>

<path name="Fig6.4"
description="org.tigris.gef.presentation.
FigLine" fill="1" fillcolor="white" stroke="1"
strokecolor="black" visibility="0" > <moveto
x="296" y="94" /> <lineto x="406" y="94" /> </
path>

<group name="Fig6.5" description="org.tigris.
gef.presentation.FigGroup[296, 64, 0, 0]"
fill="0" fillcolor="white" stroke="0"
strokecolor="black" visibility="0" >

<private> </private>
```

```
<rectangle name="Fig6.5.0" x="349" y="97"
width="4" height="4" fill="0" fillcolor="white"
stroke="0" strokecolor="black" visibility="0"
/> </group> </group>

<group name="Fig7" description="org.argouml.
uml.diagram.ui.FigDependency" href="127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000796"
stroke="1" strokecolor="black" >

<private> sourcePortFig="Fig4.0"
destPortFig="Fig6.0" sourceFigNode="Fig4"
destFigNode="Fig6" </private>

<path name="Fig7.0"
description="org.tigris.gef.presentation.
FigPoly" fill="0" fillcolor="white" stroke="1"
strokecolor="black" > <moveto x="454" y="173"
/> <lineto x="382" y="118" /> </path> </group>

<group name="Fig8" description="org.argouml.
uml.diagram.ui.FigDependency" href="127-0-1-1-
34bace4b:11b91863d86:-8000:0000000000000797"
stroke="1" strokecolor="black" >

<private> sourcePortFig="Fig6.0" destPort
Fig="Fig0.0" sourceFigNode="Fig6" destFigNode="
Fig0" </private>

<path name="Fig8.0"
description="org.tigris.gef.presentation.
FigPoly" fill="0" fillcolor="white" stroke="1"
strokecolor="black" > <moveto x="323" y="120"
/> <lineto x="188" y="249" />

</path> </group> </pgml>

<todo> <todolist> </todolist> <resolvedcritics>
</resolvedcritics> </todo> </uml>
```

Conclusion

Keyword-based search tools often do not yield much user satisfaction or retrieval success because queries are linear, without any semantic links or contexts. Graphical search attempts also meet with little success because it is difficult to put semantics even into graphical mode. In the present research work the framework includes a graphic interface using UML concept maps with faceted keyword analyzer in the backend. This enables domain experts, even with little coding expertise, to build ontologies without having to envisage all relations. They can simply place a new concept in the hierarchy at the level that is deemed to be right for that concept. All the relations at that level are then inherited by the new concept. UML provides a modular approach to knowledge representation and the framework in digital libraries would facilitate answering specific and context-based queries.

This model can be integrated in any Semantic Web application where the relations and correlations remain intact, irrespective of the ontology language used, and the subject expert who would build the ontologies need not understand/know the complications of the same. The whole infrastructure, while immensely improving retrieval efficiency by being context based, would be transparent to end-users.

The framework has been tested:

- to be domain specific and non-authoritarian, and able to support multiple views of the same resource in different contexts
- to be able to help in the evolution of subject-specific models, as it helps both the user and the machine to express semantics and context for each piece of information

- to enable anyone to describe anything, at any location, through its distributed architecture with facility to export in XMI code
- to be flexible
- to be conceptual.

References

1. Sullivan, D. (2002) "Death of a Meta Tag." Available at: *http://searchenginewatch.com/article/2048100/Death-Of-A-Meta-Tag* [Accessed 2 September 2013].
2. Carnot, M. J. et al. (2003) "Concept Maps vs. Web Pages for Information Searching and Browsing." Available at: *http://www.ihmc.us/users/acanas/Publications/CMapsVSWebPagesExp1/CMapsVSWebPagesExp1.htm* [Accessed 2 September 2013).
3. Pepper, S. (2000) "The TAO of Topic Maps." Available at: *http://www.ontopia.net/topicmaps/materials/tao.pdf* [Accessed 28 March 2008].
4. Van Schie, J. P. (2002) "Visualization Tools for Knowledge Management." Available at: *http://www.observatory.com/conceptmappingsvs1.htm* [Accessed 21 February 2008].
5. Fernandez, B. and Martinez-Bejar, R. (2002) "A Cooperative Framework for Integrating Ontologies", *International Journal of Human–Computer Studies*, 56: 665–720.
6. Jeromedl (2005) Social Semantic Digital Library. Available at *http://www.jeromedl.org/* [Accessed 10 January 2008].
7. Cranefield, S. et al. (2001) "UML-based Ontology Modeling for Software Agents," The Information Science Discussion Paper Series, No. 2001/07. Available at: *waitaki.otago.ac.nz/~martin/Documents/dp2001-07.pdf* [Accessed 12 January 2008].
8. Mintz, A. (2002) *Web of Deception: Misinformation on the Internet*, Medford, NJ: CyberAge Books.
9. Johannesson, P., Bowman, M., Bubenko, J. Jr., and Wangler, B. (1997) *Conceptual Modelling* (Prentice Hall).

10. Kabilan, V. and Johannesson, P. (n.d.) "UML for Ontology Modelling and Interoperability." Available at: *http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-125/paper33.pdf* [Accessed 2 September 2013].
11. Levy, D. M. and Marshall, C. C. (1995) "Going Digital: A Look at Assumptions Underlying Digital Libraries," *Communications of the ACM* 38(8): 77–84.
12. Lesk. M. (1999) "Expanding Digital Library Research: Media, Genre, Place and Subjects," in *Proceedings of the International Symposium on Digital Libraries 1999: ISDL'99, Tsukuba, Ibaraki, Japan, September 28–29*.
13. Reddy, R. and Wladawsky-Berger, I. (2001) *Digital Libraries: Universal Access to Human Knowledge*. A Report to the President. President's Information Technology Advisory Committee Panel on Digital Libraries. Available at *http://www.itrd.gov/pubs/pitac/pitac-dl-9feb01.pdf* [Accessed 2 September 2013].
14. Fox, E. A. and Marchionini, G. (1998) "Toward a Worldwide Digital Library," *Communications of the ACM* 41(4): 22–28.
15. Gladney, H., Fox, E. A., Ahmed, Z., Ashany, R., Belkin, N. J., and Zemankova M. (1994) "Digital Library: Gross Structure and Requirements: Report from a March 1994 Workshop," in *Proceedings of the 1st Annual Conference on the Theory and Practice of Digital Libraries, Texas, 1994*, 101–107.
16. Schauble, P. and Smeaton, A. F. (1998) *An International Research Agenda for Digital Libraries. Summary Report of the Series of Joint NSF-EU Working Groups on Future Directions for Digital Library Research*. Available at: *http://www.ercim.eu/publication/ws-proceedings/DELOS-B/dl_sum_report.pdf* [Accessed 2 September 2013].
17. Yang, C., and Pu, H.-T. (2003) "Enriching User-oriented Class Associations for Library Classification Schemes", *The Electronic Library* 21(2): 130–141.
18. Flamenco (2000) "The Flamenco Search Interface Project." Available at: *http://flamenco.berkeley.edu/index.html* [Accessed 22 February 2008].
19. Broughton, V. (2004) *Essential Classification* (London: Facet Publishing).

20. Cranefield, S. (2001) "Networked Knowledge Representation and Exchange Using UML and RDF," *Journal of Digital Information*, 1(8).
21. Kabilan, V. and Johannesson, P. "UML for Ontology Modelling and Interoperability," First International Workshop on Enterprise Modeling and Interoperability (INTEROP-EMOI 2004). *CEUR Workshop Proceedings*, ISSN-1613-0073, vol. 125.
22. Madalli, D. P. and Suman, A. "UML for the Conceptual Web," *Online Information Review* 32(4): 511–515.
23. ArgoUML. *http://argouml.tigris.org/* [Accessed 2 September 2013].

8

Conclusion

Abstract: This chapter reviews the observations and suggestions made in the preceding chapters. It makes suggestions for future research and its application to digital libraries in particular and information systems in general. The framework suggested in this book is not a one-off solution for creating models for all subject domains; rather, it is a research attempt at designing a template with generic guidelines that can be used to develop digital library models for any subject using the analytico-faceted approach.

Keywords: Frameworks, domain model, facetization.

The conceptual framework

The research presented in this book started from the premise that although there have been attempts to classify/categorize information on the web, there is an unaddressed gap in the area of defined standards and the existing models do not follow the same pattern of cognizance that the human mind does to understand associated concepts and context. Current faceted search engines/portals/web directories claim to be following faceted knowledge organization methodology, but generally they either list/classify based on the apparent attributes of entities, or cluster them on the application of pragmatic methods. Thus, what the user gets is a heap of information, with very little flexibility to put the query into

different perspectives and context; it ends up as another classification attempt using something similar to enumerated classification schema.

LIS has a well-tested theory behind the classification schema that has been successful in organizing the universe of knowledge on the library shelves. In particular, the methodology of faceted classification provides well-defined principles for the organization of concepts into categories and is able to add a linear dimension to the multiple dimensions of a subject. The schema is sturdy enough to handle complex/compound subjects at all the levels of complexity.

As faceted classification is backed by a solid internal logic of analytico-synthetic classification theory, it appears to be the most appropriate schema to use for the development of a conceptual framework for digital libraries (an automated system).

The models prepared using this framework enable contextual navigation because the elements correspond to the contextual markup of the information. As it has been developed on the foundation of S. R. Ranganathan's theory of facetization, so it even takes care of the browsing of associated concepts, thus addressing the multi-dimensionality of information resources. The existing keyword-based knowledge organization has been a total failure because it is linear and semantics are completely missing. Clustering engines tried to address the problem by providing graphical interfaces to help users to understand the semantics, but this attempt was basically demonstrative, with no real semantic architecture in the background. The conceptual framework can be successful because it has tried to embed semantics in the background (faceted ontologies) while providing a graphical interface for the human user, thus making it easy to comprehend.

This contextual navigation becomes significant for constructing, exploring, and presenting structured information.

- Embedded metadata models: Using the embedded metadata in the form of facets of the information resource, one can study the labeling of the resource components.
- It facilitates the reuse of concepts and their relations, set in different contexts, which can eventually be extrapolated to very large sets of content components.
- The context-dependent presentation of content can be made possible as one can narrow the scope/drill down content components of a given concept.
- Concept graph for the uninitiated user: While browsing, users generally do not follow a logical path. They start with a particular concept, which is followed for a couple of links, but may get carried away by something of peripheral significance. The conceptual model allows people to identify the relevant resources in the beginning because they are able to put their queries in context. It also becomes useful for tracking the contextual path when the user gets lost in the maze of information.

Mechanism to evaluate completeness of the newly developed model[1]

- One should check if the class hierarchy is complete. Errors appear in cases where the superclasses are imprecise. They appear also when the superclasses are over-specified. Further, if information is missing about those subclasses that are either a subclass partition or about exhaustive subclass partitions, even then errors can occur.

- One should check the domains for completeness and the associated functions and relations to trace whether, for each argument of each function, the domain and range delimit the appropriate classes. Here, errors appear for imprecise or over-specified domains and ranges.
- To ascertain if the required information is in the class, one should check the completeness of the classes. If in the definition of a class the properties are not defined, errors appear. Even if different classes are defined similarly or the class does not include imaginary properties, errors appear.
- One needs to check for the precision of information in the ontology to establish its *conciseness*. Further, it should not store unnecessary definitions.
- The ability to append new definitions to an ontology and more attributes to its definitions determines its expandability. Further, one should be able to do so without changing the already defined set of properties.
- *Sensitiveness* determines how minute changes in a definition change the set of guaranteed and well-defined properties.

Limitations

The framework suggested in this book is not a one-off solution for creating models for all subject domains; rather, it is a research attempt at designing a template with generic guidelines that can be used to develop digital library models for any subject using the analytico-faceted approach. It allows one to embed relationships between concepts and thus put them in context, both at the user query level and for the subject expert to deposit information resources. The relationships could be both intra-facet

(semantic) and inter-facet (syntactic relationship), thus providing a three-dimensional representation of thoughts on the ideas plane.

There are certain limitations in the application of faceted formulae for some subject domains. In the social sciences and humanities it is not easy to match the primary facet with a tangible concept and then derive its associated parts and properties. So it becomes difficult to decide a standard citation order and thus make rules in the conceptual model. Similarly, in fine arts, place and time hold a more important order than that suggested by standard citation orders.

As discussed above, the best rule will be to follow the subject, its users, and subject experts; the framework being flexible allows one to define one's own rules by following the faceted approach. The most important thing is that the same citation order should be made standard for that particular domain so as to maintain uniformity and control chaos.

Distinct advantages of the framework in information retrieval

When embedded in information retrieval systems, the ontology models developed using this framework contribute greatly to the Semantic Web by making context-based information search possible. Today if a user searches for a piece of information s/he is not able to put the context around the query, and the search engine is also unable to retrieve context around the resource with the existing ontologies. In the faceted ontologies, the context is always wrapped around the piece of information in terms of defined facets which can be mixed and matched, based on the defined citation order for that particular domain. So, when the user queries the web/database, the search algorithm can easily translate the query

to the principal facet and associated concepts and thus retrieve the appropriate and relevant information only. The following example explains this better:

Query: Address of Aparajita Suman, employee of UNODC, zodiac sign Scorpio, friends call Appu.
Search results: When the resource has normal ontology:

- Aparajita Sen, stays in India, address at Mumbai: ……..
- Suman Aarya, employee of NASA
- Home page: UNODC. *http://www.unodc.org*
- Friends circle. Join the mailing list to meet friends ……

Search results: When the resource has faceted ontology:

- Employee Profile: UNODC, Aparajita Suman (Designation, Department, Contact information)
- Associated concept
  - Personal page: Aparajita Suman,
  - Date of birth: 3 November
  - Zodiac sign: Scorpio
  - Nickname: Appu

Further developments: design of domain-specific models from the framework

Fundamental categories in web context

For simplicity's sake, S. R. Ranganathan's PMEST has been taken as the foundation for developing sample models in this

framework, but the extensible list can always be expanded to accommodate the needs of different domains. E.g. the following list of categories has been suggested in Bliss Bibliographic Classification[2,3] (BC2):

- thing/entity
- space
- part
- agent
- time
- process
- patient
- product
- by-product.

Vanda Broughton, the editor of BC2, has said that "These fundamental thirteen categories have been found to be sufficient for the analysis of vocabulary in almost all areas of knowledge. It is however quite likely that other general categories exist; it is certainly the case that there are some domain specific categories, such as those of form and genre in the field of literature."[2]

Domain-specific models

The following generic steps can be taken to derive a domain-specific model from the conceptual framework.

1. *Study of subject*: One needs to thoroughly study the subject domain by getting sufficient information samples, locating creditworthy resources, following the evolution of the subject, and, the most important, consulting the subject experts – the human resource.

2. *Conceptualization of the domain ramifications*: Isolating the basic concepts and their branches is the primary exercise in the process of facet analysis.
3. *Facet creation*: Examination of the resulting concept maps and relationship graphs gives a fair insight into the exclusivity of certain domain sects and their interplay.

   When applying facet analysis to the subject resource, the first step is to examine some representative samples and to try to enumerate the subject, standard terms, interrelationship with other subjects, etc. In applied subjects, this becomes all the more important because the analysis has to be operational as well.

   Here, one cannot undermine the users' needs and their way of approaching the information. E.g., the approach of a medical practitioner will be different from that of an automobile engineer.
4. *Facet arrangement*: Here, the ordering of the facets in terms of their scope and relevance can be easily done by following S. R. Ranganathan's principles for helpful sequence and principles for facet sequence.
5. *Citation order*: The citation order has to be decided based on the subject need and the focus of the facets. It will be very subject specific and there cannot be a generic formula for it. The rule of thumb is that the citation order should always be decided in the way in which the user will expect to find the subject organized.
6. *Incorporation in the framework*: Once the citation order has been decided, then the schema can be finalized and incorporated into the model for the particular subject domain, which can then be used to develop UML maps for the information resources in the digital library

7. *Revision, testing, and maintenance*: Of course, one cannot underestimate the significance of revisions of the schema based on input from users and the subsequent growth and evolution of the subject.

References

1. Gómez-Pérez, A. (2001) "Evaluation of Ontologies," *International Journal of Intelligent Systems* 16(3): 391–409.
2. Broughton, V. (2001) "Faceted Classification as a Basis for Knowledge Organization in a Digital Environment; the Bliss Bibliographic Classification as a Model for Vocabulary Management and the Creation of Multi-dimensional Knowledge Structures," *The New Review of Hypermedia and Multimedia* 7(1): 67–102.
3. Miskatonic University Press (2009) "How to Make a Faceted Classification and Put It On the Web." Available at: *http://www.miskatonic.org/library/facet-web-howto.html* [Accessed 10 August 2008].

Bibliography

Abrahamsen, K. T. (2003) "Indexing of Musical Genres. An Epistemological Perspective," *Knowledge Organization*, 30(3/4): 144–169.

Adkisson, H. P. (2005) "Use of Faceted Classification," Web Design Practices, *www.webdesignpractices.com/navigation/facets.html* [Accessed 2 September 2013].

AKT Reference Ontology. *http://www.aktors.org/publications/ontology/* [Accessed 2 September 2013].

Anderson, J. D. (2003) "Organization of Knowledge," in J. Feather and P. Sturges (eds) *International Encyclopedia of Information and Library Science* (London: Routledge), 471–490.

Antoniou, G. and van Harmelen, F. A. (2003) "Web Ontology Language: OWL," in S. Staab and R. Studer (eds), *Handbook on Ontologies in Information Systems*, (Springer).

Antoniou, G. and van Harmelen, F. A. (2004) *Semantic Web Primer*, Cambridge, MA: MIT Press.

Baader, F. et al. (2003) *Description Logic Handbook: Theory, Implementation and Applications* (Cambridge: Cambridge University Press).

Berners-Lee, T., Hendler, J. and Lassila, O. (2001) "The Semantic Web," *Scientific American* 284(5): 34–43.

Bliss, H. E. (1929) *The Organization of Knowledge and the System of the Sciences*, with an introduction by John Dewey (New York: Henry Holt and Co.).

Broughton, V. (1999) "Notational Expressivity; the Case for and against the Representation of Internal Subject Structure in Notational Coding," *Knowledge Organization* 26(3): 140–148.

Broughton, V. (2001) "Faceted Classification as a Basis for Knowledge Organization in a Digital Environment: The Bliss Bibliographic Classification as a Model for Vocabulary Management and the Creation of Multidimensional Knowledge Structures," *The New Review of Hypermedia and Multimedia* 7(1): 67–102.

Broughton, V. and Heather, L. (2000) "Classification Schemes Revisited: Applications to Web Indexing and Searching," *Journal of Internet Cataloguing* 2(3/4): 143–155.

Chan, L. M. et al. (2001) "A Faceted Approach to Subject Data in the Dublin Core Metadata Record," *Journal of Internet Cataloging* 4(1/2): 35–47.

Cimiano, P. and Voelker, J. (n.d.) "Text2Onto. A Framework for Ontology Learning and Data-driven Change Discovery." Available at: *http://sourceforge.net/projects/texttoonto/* [Accessed 2 September 2013].

CMS Review. *http://www.cmsreview.com/Directory.html* [Accessed 2 September 2013].

Cranefield, S. and Purvis, M. (1999) "UML as an Ontology Modelling Language," *Proceedings of the IJCAI-99 Workshop on Intelligent Information Integration*.

Dahlström, M. and Gunnarsson, M. (2000) "Document Architecture Draws a Circle: On Document Architecture and Its Relation to Library and Information Science and Research," *Information Research* 5(2). Available at:

http://informationr.net/ir/5-2/paper70.html [Accessed 2 September 2013].

Devadason, F. J. et al. (2002) "Faceted Indexing Based System for Organizing and Accessing Internet Resources," *Knowledge Organization* 29(2): 65–77.

Dewey, M. (1979) *Dewey Decimal Classification and Relative Index*, 19th edn, vol. 1 (Albany, NY: Forest Press).

Doyle, B. (2003) "The Classification and Evaluation of Content Management Systems," *The Gilbane Report* 11(2): 2–13. Available at: *http://www.gilbane.com/artpdf/GR11.2.pdf* [Accessed 13 September 2013].

Duncan, E. B. (1989) "A Faceted Approach to Hypertext," in Ray McAleese (ed.) *Hypertext: Theory into Practice*, (Oxford: Intellect, 1989): 157–163.

Duncan, E. B. (1989) "Structuring Knowledge Bases for Designers of Learning Materials," *Hypermedia* 1(1): 20–33.

Duncan, E. B. (1990) "A Concept-map Thesaurus as a Knowledge-Based Hypertext Interface to a Bibliographic Database," in K. P. Jones (ed.) *Informatics 10: Prospects for Intelligent Retrieval* (London: Aslib), 43–52.

Dzbor, M., Motta, E., and Domingue, J. B. (2004) "Opening Up Magpie via Semantic Services," in McIlraith et al. (eds), *The Semantic Web – ISWC 2004, Third International Semantic Web Conference. Hiroshima, Japan, November 2004.* Lecture Notes in Computer Science, 3298 (Springer-Verlag).

Ellis, D. and Vasconcelos, A. (1999) "Ranganathan and the Net: Using Facet Analysis to Search and Organise the World Wide Web," *Aslib Proceedings* 51(1): 3–10.

Ellis, D. and Vasconcelos, A. (2009) "The Relevance of Facet Analysis for World Wide Web Subject Organization and Searching," *Journal of Internet Cataloging* 2(3/4): 97–114.

Ereshefsky, M. (2000) *The Poverty of the Linnaean Hierarchy: A Philosophical Study of Biological Taxonomy* (Cambridge: Cambridge University Press).

Fast, K., Leise F., and Steckel, M. (n.d.) "Facets and Controlled Vocabularies: An Annotated Bibliography," *Boxes and Arrows*. Available at: *http://www.boxesandarrows.com/files/banda/Bibliography.htm* [Accessed 2 September 2013].

Fensel, D. et al. (2001) "OIL: An Ontology Infrastructure for the Semantic Web," *IEEE Intelligent Systems* 16(2): 38–44.

Fidel, R. and Pejtersen, A. M. (2004) "From Information Behaviour Research to the Design of Information Systems: The Cognitive Work Analysis framework," *Information Research*, 10(1) paper 210. Available at: *http://InformationR.net/ir/10-1/paper210.html* [Accessed 2 September 2013].

Fjordback Søndergaard, T., Andersen, J., and Hjørland, B. (2003) "Documents and the Communication of Scientific and Scholarly Information. Revising and Updating the UNISIST model," *Journal of Documentation*, 59(3): 278–320.

Foskett, D. J. (1992) "Ranganathan and 'User-Friendliness'," *Libri* 42(3): 227–234.

Francke, H. (2005) "What's in a Name? Contextualizing the Document Concept," *Literary and Linguistic Computing* 20(1): 61–69.

Frohmann, B. (1990) "Rules of Indexing: A Critique of Mentalism in Information Retrieval Theory," *Journal of Documentation* 46(2): 81–101.

Frohmann, B. (2004) *Deflating Information. From Science Studies to Documentation* (Toronto: University of Toronto Press).

Frohmann, B. (2004) "Documentation Redux: Prolegomena to (Another) Philosophy of Information," *Library Trends*, 52(3): 387–407.

Garshoel, L. M. (2002) "What Are Topic Maps?" Available at: *http://www.xml.com/pub/a/2002/09/11/topicmaps.html* [Accessed 2 September 2013].

Gómez-Pérez, A. and Corcho, O. (2002) "Ontology Languages for the Semantic Web," *IEEE Intelligent Systems 2002*, 54–60.

Guha, R. and McCool, R. (2003) "Tap: A Semantic Web Platform," *Computer Networks*, 42(5): 557–577.

Hansson, J. (1997) "Why Public Libraries in Sweden Did Not Choose Dewey," *Knowledge Organization*, 24(3): 145–153.

Hansson, J. (1999) *Klassifikation, bibliotek och samhälle: en kritisk hermeneutisk studie av "Klassifikationssystem för svenska bibliotek"* (Borås: Valfrid, Skrifter från Valfrid 19).

Hansson, J. (2004) "The Social Legitimacy of Library and Information Studies: Reconsidering the Institutional Paradigm," in B. Rayward (ed.) *Aware and Responsible: Papers of the Nordic-International Colloquium on Social and Cultural Awareness and Responsibility in Library, Information and Documentation Studies* (Lanham, MD: Scarecrow Press), 49–69.

Hjørland, B. (ed.) (2005) *Lifeboat for Knowledge Organization* (Copenhagen: Danmarks Biblioteksforening).

Hjørland, B. and Nicolaisen, J. (eds) (2005) *The Epistemological Lifeboat. Epistemology and Philosophy of Science for Information Scientists* (Copenhagen: Danmarks Biblioteksforening).

Huynh, D., Mazzocchi, S. and Karger, D. (2005) "Piggy Bank: Experience the Semantic Web inside Your Web Browser," in Gil et al. (eds) *The Semantic Web – ISWC*

2005, 4th International Next Generation Semantic Web Applications 6 Semantic Web Conference, ISWC 2005. Galway, Ireland, November 6–10, 2005. Lecture Notes in Computer Science, 3729 (Springer-Verlag).

Hyvönen, E., Mäkelä, E., Salminen, M., Valo, A., Viljanen, K., Saarela, S., Junnila, M. and Kettula, S. (2005) "MuseumFinland – Finnish Museums on the Semantic Web," *Journal of Web Semantics* 3(2): 25.

Kabilan, V., Johannesson, P. and Rugaimukammu, D. (2003) "An Ontological Approach to Unified Contract Management," in *Proceedings of 13th European Japanese Conference on Information Modeling and Knowledge Bases, June 6–7, 2003, Kitakyushu, Japan*.

Kent, R. "Conceptual Knowledge Markup Language." Available at: *http://arxiv.org/ftp/arxiv/papers/1109/1109.1525.pdf* [Accessed 22 September 2013].

Kessler, M. M. (1965) "Comparison of the Results of Bibliographic Coupling and Analytic Subject Indexing," *American Documentation* 16(3): 223–233.

Kwasnick, B. H. (1999) "The Role of Classification in Knowledge Representation and Discovery," *Library Trends* 48(1): 22–47.

Lee, J. and Goodwin, R. (2005) "The Semantic Webscape: A View of the Semantic Web," in *WWW '05: Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, New York, 2005 (ACM Press), 1154–1155.

Leise, F. and Fast, K. (2002, 9 December) "All About Facets and Controlled Vocabularies," *Boxes and Arrows*. Available at: *http://www.boxesandarrows.com//all-about-facets-controlled-vocabularies* [Accessed 2 September 2013].

Leise, F. and Fast, K. (2002, 16 December) "What Is a Controlled Vocabulary?" *Boxes and Arrows*. Available

at: *http://boxesandarrows.com/what-is-a-controlled-vocabulary/* [Accessed 2 September 2013].

Leise, F. and Fast, K. (2003, 7 April) "Creating a Controlled Vocabulary," *Boxes and Arrows*. Available at: *http://www.boxesandarrows.com/creating-a-controlled-vocabulary* [Accessed 2 September 2013].

Leise, F. and Fast, K. (2003, 26 August) "Synonym Rings and Authority Files," *Boxes and Arrows*. Available at: *http://boxesandarrows.com/synonym-rings-and-authority-files/* [Accessed 2 September 2013].

Leise, F. and Fast, K. (2003, 17 October) "Controlled Vocabularies: A Glosso-Thesaurus," *Boxes and Arrows*. Available at: *http://www.boxesandarrows.com/controlled-vocabularies-a-glosso-thesaurus* [Accessed 2 September 2013].

Line, M. B. (2005) Editorial: "Librarianship as It Is Practised: A Failure of Intellect, Imagination and Initiative," *Interlending & Document Supply*, 33(2): 109–113.

Lopez, V. and Motta, E. (2004) "Ontology Driven Question Answering in AquaLog," in *9th International Conference on Applications of Natural Language to Information Systems (NLDB 2004)*.

Lopez, V., Motta, E. and Uren, V. (2006) "PowerAqua: Fishing the Semantic Web," in Y. Sure and J. Domingue (eds) *The Semantic Web: Research and Applications, 3rd European Semantic Web Conference, ESWC 2006, Budva, Montenegro, June 11–14, 2006*. Lecture Notes in Computer Science 4011 (Springer, ISBN 3-540-34544-2).

Lopez-Huertas, M. J. and Contreras, E. J. (2004) "Spanish Research in Knowledge Organization (1992–2001)," *Knowledge Organization* 31(3): 136–150.

Louie, A. J. et al. "Using Faceted Classification to Provide Structure for Information Architecture." Paper presented at the ASIS&T 2003 Information Architecture Summit,

Portland, OR, 21–23 March 2003. Available at: *http://depts.washington.edu/pettt/presentations/conf_2003/IASummit.pdf* [Accessed 2 September 2013].

Luke, S. and Hefflin, J. "SHOE 1.01 Proposal Specification." Available at: *http://www.cs.umd.edu/projects/plus/SHOE* [Accessed 2 September 2013].

Manola, F. and Miller, E. *RDF Primer*. Available at: *http://www.w3.org/TR/rdf-primer/* [Accessed 2 September 2013].

McGuinness, D. L. and van Harmelen, F. (2004) "OWL Web Ontology Language Overview." Available at: *http://www.w3.org/TR/owl-features/* [Accessed 2 September 2013].

Mika, P. (2005) "Flink: SemanticWeb Technology for the Extraction and Analysis of Social Networks," *Journal of Web Semantics* 3(2).

Miksa, F. L. (1998) *The DDC, the Universe of Knowledge, and the Postmodern Library* (Albany, NY: Forest Press).

Mills, J. (2004) "Faceted Classification and Logical Division in Information Retrieval," *Library Trends* 52(3): 541–570.

Milstead, J. L. (1998) "Use of Thesauri in the Full-Text Environment." Based on a paper presented at the 34th Clinic on Library Applications of Data Processing. Available at: *http://www.bayside-indexing.com/Milstead/useof.htm* [Accessed 2 September 2013].

Noy, N. F. and McGuinness, D. L. (n.d.) "Ontology Development 101: A Guide to Creating Your First Ontology." Available at: *http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html* [Accessed 2 September 2013].

Opdahl, A. L. and Henderson-Sellers, B. (2001) "Grounding the UML Metamodel in Ontology," *Journal of Systems and Software* 57(2): 119–143.

Opdahl, A. L. and Sindre G. (1994) "A Taxonomy for Real-world Modeling Concepts," *Information Systems* 19(3): 229–241.

Ørom, A. (2003) "Knowledge Organization in the Domain of Art Studies History, Transition and Conceptual Changes," *Knowledge Organization* 30(3/4): 128–143.

Pan, J. and Horrocks, I. (2003) "(FA) and RDF MT: Two Semantics for RDFS," in *Proceedings of the 2003 International Semantic Web Conference (ISWC 2003)*, Lecture Notes in Computer Science, 2870 (Springer), 30–46.

Pao, M. L. (1993) "Term and Citation Retrieval: A Field Study," *Information Processing & Management*, 29(1), 95–112.

Pao, M. L. and Worthen, D. B. (1989) "Retrieval Effectiveness by Semantic and Pragmatic Relevance," *Journal of the American Society for Information Science* 40(4), 226–235.

Patel-Schneider, P. F., Hayes, P. and Horrocks, I. (2004) "OWL Web Ontology Language Semantics and Abstract Syntax." Available at: *http://www.w3.org/TR/owl-semantics/* [Accessed 2 September 2013].

Patel-Schneider, P., Horrocks, I., and van Harmelen F. (2002) "Reviewing the Design of DAML+OIL: An Ontology Language for the Semantic Web," in *Proceedings of the Eighteenth National Conference on Artificial Intelligence, AAAI*.

Pejtersen, A. M. (1989) *The Book House: Modeling User Needs and Search Strategies as a Basis for System Design*, Risø report M-2794 (Roskilde: Risø National Laboratory).

Pejtersen, A. M. (1989) "A Library System for Information Retrieval Based on a Cognitive Task Analysis and Supported by an Icon-Interface," *ACM SIGIR Forum*, 40–47.

Pejtersen, A. M. (1992) "The Book House. An Icon Based Database System for Fiction Retrieval in Public Libraries," in B. Cronin, (ed.) *The Marketing of Library and Information Services 2* (London: Aslib), 572–591.

Pilgrim, M. (2002) *Dive Into Accessibility*. Available at: *http://www.diveintoaccessibility.info/* [Accessed 2 September 2013].

Popov, B., Kiryakov, A., Kirilov, A., Manov, D., Ognyanoff, D., and Goranov, M. (2003) "KIM – A Semantic Annotation Platform," in D. Fensel, K. Sycara, and J. Mylopoulos (eds) *The Semantic Web – ISWC 2003, Second International Semantic Web Conference*. Lecture Notes in Computer Science, 2870 (Springer-Verlag).

Priss, U., and Jacob, E. (1999) "Utilizing Faceted Structures for Information Systems Design," in L. Woods (ed.) *ASIS '99: Proceedings of the 62nd ASIS Annual Meeting, Washington, D.C., October 31–November 4, 1999: Knowledge: Creation, Organization, and Use* (Medford, NJ: ASIS), 203–212. Available at: *http://www.upriss.org.uk/papers/asis99.pdf* [Accessed 2 September 2013].

Ranganathan, S. R. (1962) *Elements of Library Classification*, 3rd edn (New York: Asia Publishing House).

Rees-Potter, L. K. (1989) "Dynamic Thesaural Systems: A Bibliometric Study of Terminological and Conceptual Change in Sociology and Economics with Application to the Design of Dynamic Thesaural Systems," *Information Processing & Management* 25(6): 677–691.

Rees-Potter, L. K. (1991) "Dynamic Thesauri: The Cognitive Function. Tools for Knowledge Organisation and the Human Interface," *Proceedings of the 1st International ISKO Conference, Darmstadt, 14–17 August 1990. Part 2*, 145–150.

Salton, G. (1968) *Automatic Information Organization and Retrieval* (New York: McGraw-Hill).

Salton, G. (1971) "Automatic Indexing Using Bibliographic Citations," *Journal of Documentation* 27(2): 98–110.

Schneider, J. W. (2004) "Verification of Bibliometric Methods' Applicability for Thesaurus Construction," PhD dissertation (Aalborg: Royal School of Library and Information Science).

Schraefel, M. C., Shadbolt, N. R., Gibbins, N., Glaser, H. and Harris, S. (2004) "CS AKTive Space: Representing Computer Science in the Semantic Web," in *Proceedings of the 13th International World Wide Web Conference (WWW2004) May 17–22, New York*. Available at: *http://eprints.soton.ac.uk/259084/1/p276-schraefel.pdf* [Accessed 2 September 2013].

Schreiber, G. (2002) "The Web is Not Well-formed," *IEEE Intelligent Systems* 17(2).

Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W., and Wielinga, B. (1999) *Knowledge Engineering and Management – The CommonKADS Methodology* (Cambridge, MA: MIT Press).

Shorten, J., Seikel, M., and Ahrberg, J. H. (2005) "Why Do You Still Use Dewey? Academic Libraries that Continue with the Dewey Decimal Classification," *Library Resources & Technical Services*, 49(2): 123–133.

Small, H. (1973) "Co-citation in the Relationship between Two Documents," *Journal of the American Society for Information Science* 24: 256–269.

Smiraglia, R. P. (2001) *The Nature of a Work. Implications for the Organization of Knowledge* (Lanham, MD: Scarecrow Press).

Smith, M. K., Welty, C. and McGuinness, D. L. (2004) "OWL Web Ontology Language Guide." Available at: *http://www.w3.org/TR/owl-guide/* [Accessed 2 September 2013].

Spiteri, L. (1998) "A Simplified Model for Facet Analysis: Ranganathan 101," *Canadian Journal of Information and Library Science* 23(1/2): 1–30. Available at: *http://aifia.org/pg/a_simplified_model_for_facet_analysis.php* [Accessed 2 September 2013].

Steckel, M. (2002, 7 October) "Ranganathan for IAs," *Boxes and Arrows*. Available at: *http://www.boxesandarrows.com/Ranganathan-for-ias/* [Accessed 2 September 2013].

Stichweh, R. (2001) "Scientific Disciplines, History of," in N. J. Smelser and P. B. Baltes (eds) *International Encyclopedia of the Social and Behavioral Sciences* (Oxford: Elsevier Science), 13727–13731.

Svenonius, E. (2000) *The Intellectual Foundation of Information Organization* (Cambridge, MA: The MIT Press).

Taylor, A. G. (1999) *The Organization of Information* (Englewood, CO: Libraries Unlimited).

The Faceted Classification Discussion (FCD) Mailing List. *http://poorbuthappy.com/fcd/*.

The Knowledge Management Connection, "Faceted Classification of Information."

Thomere, J., Chaudri, V. and Karp, P. (2000) "An XML-Based Ontology Exchange Language." Available at: *http://www.sri.com/work/publications/xol-xml-basedeontology-exchange-language* [Accessed 2 September 2013].

Tzitzikas, Y. (2002) "Extended Faceted Taxonomies for Web Catalogs." Paper presented at the *Third International Conference on Web Information Systems Engineering, WISE 2002, Singapore, December*.

Tzitzikas, Y. et al. (2003) "An Algebraic Approach for Specifying Compound Terms in Faceted Taxonomies," in *Proceedings of the 13th European-Japanese Conference*

on Information Modelling and Knowledge Bases, EJC2003 (Kitakyushu, Japan).

van Dijck, P. (2003) "Introduction to XFML," *xml.com*. Available at: *http://www.xml.com/lpt/a/2003/01/22/xfml.html* [Accessed 2 September 2013].

van Harmelen, F. and Fensel, D. (1999) "Practical Knowledge Representation for the Web," in *Proceedings of the IJCAI'99 Workshop on Intelligent Information Integration*.

Vickery, B. C. (1960) *Faceted Classification: A Guide to Construction and Use of Special Schemes* (London: Aslib).

W3C (2002, 31 January) "XHTML+SMIL Profile," W3C Note. Available at: *http://www.w3.org/TR/XHTMLplusSMIL/* [Accessed 2 September 2013].

W3C (2004) "Web Ontology Language OWL: W3C Semantic Web Activity – links. Available at: *http://www.w3.org/2004/OWL/* [Accessed 2 September 2013].

W3C (2008) "OWL 1.1 Web Ontology Language: Model-Theoretic Semantics," W3C Working Draft, 8 January. Available at: *http://www.w3.org/TR/2008/WD-owl11-semantics-20080108/* [Accessed 2 September 2013].

Wand, Y. (1989) "An Ontological Foundation for Information Systems Design Theory," in B. Pernici and A. A. Verrijn-Stuart (eds) *Office Information Systems: The Design Process* (Amsterdam: North-Holland) 201–221.

Wand, Y. and Weber, R. (1989) "An Ontological Analysis of Systems Analysis and Design Methods," in *Proceedings of the IFIP WG 8.1 Working Conference on Information Systems Concepts: An In-Depth Analysis* (Amsterdam: North-Holland).

Wand, Y. and Weber, R. (1990) "An Ontological Model of an Information System," *IEEE Transactions on Software Engineering* 16(11): 1282–1292.

Weber, R. (1997) *Ontological Foundations of Information Systems*, Coopers & Lybrand Research Methodology Monograph No. 4 (Melbourne: Coopers & Lybrand).

Wellisch, H. H. (2000) *Glossary of Terminology in Abstracting, Classification, Indexing, and Thesaurus Construction*, 2nd edn (Medford: Information Today, Inc.).

White, H. D. and McCain, K. W. (1998) "Visualizing a Discipline: An Author Co-citation Analysis of Information Science, 1972–1995," *Journal of the American Society for Information Science* 49(4), 327–355.

Winfried, G. T. (1991) "Facet Classification in Online Retrieval," *International Classification* 18(2): 98–109.

XFML (a mailing list about XFML). *http://groups.yahoo.com/group/xfml/*.

XFML Core – eXchangeable Faceted Metadata Language. 2002. Available at: *http://xml.coverpages.org/ni2002-10-08-c.html* [Accessed 22 September 2013].

Index